The Economics of Information and Communications Technology

by

John P. Conley[1]
Vanderbilt University

January 2024

Draft

Not for General Distribution

[1] Disclaimer: The author spent the 2016-2017 academic year as a visiting researcher at Microsoft Research in Redmond, Washington, and parts of Chapter 6 derive from work partly sponsored by Microsoft. The author has also served as a consultant for the World Bank, and parts of Chapter 10 derive from work partly sponsored by the Bank. In addition, the author has served as a consultant to several blockchain projects and is one of the founders of the Geeq Project (geeq.io) a base-level blockchain platform, which certainly have had an impact on the authors understanding of DLT and Chapter 11 through 14 and in particular. The views expressed in this book, however, are the responsibility of the author and are not necessarily shared by any of the organizations above.

Table of Contents

Chapter 1. Introduction

Section 1.1. Information

Information is all around us. In fact, physicist have suggested that information is all there is. The quantum states, mass, velocity, etc. of all particles in existence may not only describe universe, but, in fact, be the universe. While this universe of information is in principle knowable, it is far beyond the limits of human cognition. At a minimum, the information must first become knowledge to be usable by humans. The value of knowledge then depends upon the set of interrelated biological, social, and technolgical, tools available to process and utilize it.

As an example, there are many undiscovered plant species in the amazon. No doubt some of them produce substances that have important medical applications. Suppose that one of these plants is discovered by an indigenous shaman. Even if he keeps this K**nowledge** to himself, he could still use it for the benefit to his tribe, If he chooses to **Communicate** his knowledge to his tribe, they could build on his finding by experimenting with similar plants they might have seen. If he finds a way to record and **Store** his knowledge, it could be passed on to future generations after his death. If is able to **Distribute** his records, other people he does not know, and will never meet can enjoy the benefits of his discovery.

There is an essential nexus between information itself, and the technology that allows it to be stored, and communicated, and interpreted. Information can be stored in writing, in pictures, in various electronic forms, and even in human memories. Information can be communicated through speech, writing, images, art, either directly, or using available technologies. Information can be interpreted with the help of computers, complicated software systems, educational and cultural institutions, groups of people to whom the information has been communicated, and even by individual human brains.

Section 1.2. Technology

Technological advances have improved the human condition enormously by almost any measure. Worldwide life-expectancy seldom exceed 30 year from the paleolithic era al the way into the late middle-ages. Ir remained below 40 up to the start of the 20th century. Life-expectancy now is universally over 60, in the vast majority of nations it exceeds 70, and in many places it now approaches 80. It is estimated that 80% of the world's population lived in poverty in 1800, compared to about 10% today. Similar improvements are seen in literacy, hours worked, and the list goes on.

Information and Communication Technologies have always played pivotal roles in human history and development. The rate of change in the last few decades has been astounding. Information goods and services have become the central factor of economic growth in more advanced economies, and the driver of social change everywhere. In a real sense, our societies are becoming information.

Fortunately, humans are adaptable. The question is, are we adaptable enough. At a genetic level, adaptation is at minimum a multi-generational affair. More fundamental changes can take thousands of years. Social and societal adaptions take place more quickly, but the scale is still measured in years or decades. We delegate the difficult task keeping up with large scale social, political, and technological challenges to governments, regulators, and experts in hopes that they will be more agile and decisive. When change its too fast, or the problems grow too complex, then this mechanism fails as well.

Arthur C. Clarke famously said: "**Any sufficiently advanced technology is indistinguishable from magic**." Herein lies the essential dilemma. The basic machines and tools that we now rely on, not only for out way of life, but our literal survival, are beyond understanding of ordinary people. Even the smartest, most clear-sighted, and best-intentioned, policy-maker is destined to be fighting the last war when the battleground changes in fundamental ways several times a year.

ICT is the ocean in which we all swim. Our work, our play, our institutions, our potentials, and our way of life, are anchored in this medium. To treat it as immutable, incomprehensible, magical thing, is to surrender agency. No man can hold back the sea, but we can, perhaps, learn how to navigate around the most dangerous shoals.

Section 1.3. Purpose

The purpose of this book is to provide a tool kit to help us live rationally in a world beyond our understanding. We focus on four main elements:

<u>Technological Basics</u>: While it is impossible to understand every new device, system, or platform, there are few technologies and principles that are foundational. New applications and devices must incorporate them to one degree or another. Understanding these basics allows us to see how new technologies work and fit together despite their novelty.

<u>Economic Principals</u>: Technology changes quickly, but humans changer more slowly. The same analytical tools that economists have always used continue to be to give useful and verifiable predictions. What has changed is exactly how preferences are formed, what constraints we face, and especially, how we assess the state of the world. To the extent we can assess how new technologies refine the problem we must solve, existing economics tools continue to provide us with understanding.

Practical Value: Most of us will not be asked to make social policy. All of us have to live with the consequences of whatever evolves because of, or despite social policy. The more practical value of learning the basics and principles outlined above is that they will allow you to make much more informed assessments of how you want to behave as a consumer, how you should interact with new technologies, what the risks and tradeoffs are. The future is a blind alley down which we all most walk. Having a look down this ally is a far better plan charging down it, hoping for the best.

Comparative Advantage. Some of you will have jobs one day. Whether you start your own company, or work for someone else, ICT will play a key role in your success. Most of your colleagues will know little to nothing about technology. Even specialists are, well, specialists. They often know a great deal about something. but few know a little about everything. Having generalist knowledge in this area will give you a significant advantage over your collogues to whom this is all magic.

Technical Boxes

Throughout the book we will include green technical boxes, like the ones above, that define concepts, or explain technical details.

Section 1.4. An Overview

It is traditional to distinguish between **Person-to-Person** and **Broadcast Communications.** A message meant to go from one individual to another raises questions of privacy. A broadcast message such as a speech, a pamphlet, or a radio transmission, on the other hand, raises questions about the credibility of the source, and the ability to gain the attention of an audience.

More recently, it has become important to distinguish between **Wired** and **Wireless** communications as well. Wired communications, such as the telegraph, telephone, cable television, and broadband Internet, tend to be more secure and use only privately owned resources. Wireless communications such as radio, broadcast television, and cell phones, on the other hand, are more difficult to make secure and generally use public bandwidth, which is a commonly held resource.

Wireless communications are often subject to **Congestion**. There is only so much spectrum available for use. If too many people try to use part of the spectrum at once, messages get lost in the static. For example, if a 50,000 watt radio station broadcasts on the 530 kHz band, its signal can carry for thousands of miles. This makes it impossible for local radio stations to broadcast a clear signal on the same frequency. As a result, the use of bandwidth is tightly regulated and controlled by governments. In economics, we call these congestion effects **Negative Externalities**.

Wired communications are also subject to congestion, though to a much smaller degree in general. In fact, wired communications systems often exhibit a **Positive Externality** associated with the number of subscribers. We say that an industry or product has **Network Externalities** if its value to consumers increases with the number of people who use it (or join the network).

Externality: The effect of any action, other than buying or selling commodities, that affects the welfare of any other agent. Since these actions are not incentivized by prices, they lead to market failure and the need for second-best solutions, such as regulation.

Network Externality: A situation in which the value of a product to each individual consumer increases with the number of other agents who consume the good (and thus, join the network). Note that the benefits that joining a network convey to existing network members is not priced in, and so, this positive externality is ignored is decision-making.

For example, suppose it is 1876, and Alexander Graham Bell knocks on your door. He tries to sell you his amazing new communications device called the “telephone”. You ask him who you would be able reach with this device. He responds that you are the first door he has knocked on. At this point, the network would contain only you, and so the telephone is useless.

On the other hand, if several local businesses had already signed up, you might be able to order a pizza; if the whole town had signed up, it would be even more useful; if the whole country or world had already joined the network, the value of having a telephone would be very high indeed. Network externalities can be seen in many situations, for example, software, social networks, and technological standards such as MP3, PDF, HTTP, or the metric system.

Recent advances in information technology have made it cheaper than ever to distribute music, books, information services, and almost every other kind of content. There are still large **First Copy Costs** associated with producing content or software systems, but it is close to free to replicate and distribute any number of copies. As a result, many of the companies we interact with are what economists call **Natural Monopolies**. Natural monopolies are characterized by high fixed costs and low marginal costs of production.

A classic example is an electric utility company. It is very expensive to build power plants and to run wires to every consumers' home. It is relatively cheap, however, to supply an additional kilowatt-hour to the electric grid once this has been done. Thus, the average cost of production is decreasing as a function of quantity over a wide range of output levels since the fixed cost can be spread over a larger number of units.

In general, monopolies exploit their **Market Power** to raise prices by restricting output. Unfortunately, the obvious solution of breaking up such monopoly to force competition would result in a higher average cost of production than allowing one firm to continue to produce twice the output. This is because both firms would need to make same large fixed investment. (What sense would it make to run two sets of power lines to each house and have two electricity companies standing ready to supply power to the whole market, if required?) In the end, it is unclear if consumers benefit more from lower production costs than they are harmed by monopoly pricing.

Natural monopolies such as social media platforms, operating systems and application software, and broadband providers, are both a curse and a blessing. Regulating them can be a cure worse than the disease. What to do about powerful technology monopolies is one of the many difficult policy questions in ICT economics.

Monopoly: A market in which a single firm supplies all the output to the market.

Natural Monopoly: A firm is said to be a natural monopoly if the minimum point of its average cost curve is above the market demand curve. This is typical of firms producing products with first copy costs, and low marginal distribution or production costs.

Market Power: A more general situation in which a firm is able to affect the market price. This may because a small number of firms are supplying the market (an Oligopoly), or it may be that a firm is a local monopoly spatially (other firms are far away and so inconvenient to consumers), or in product characteristics (other firms make substitutes, but imperfect or differentiated ones).

Section 1.5. Economics

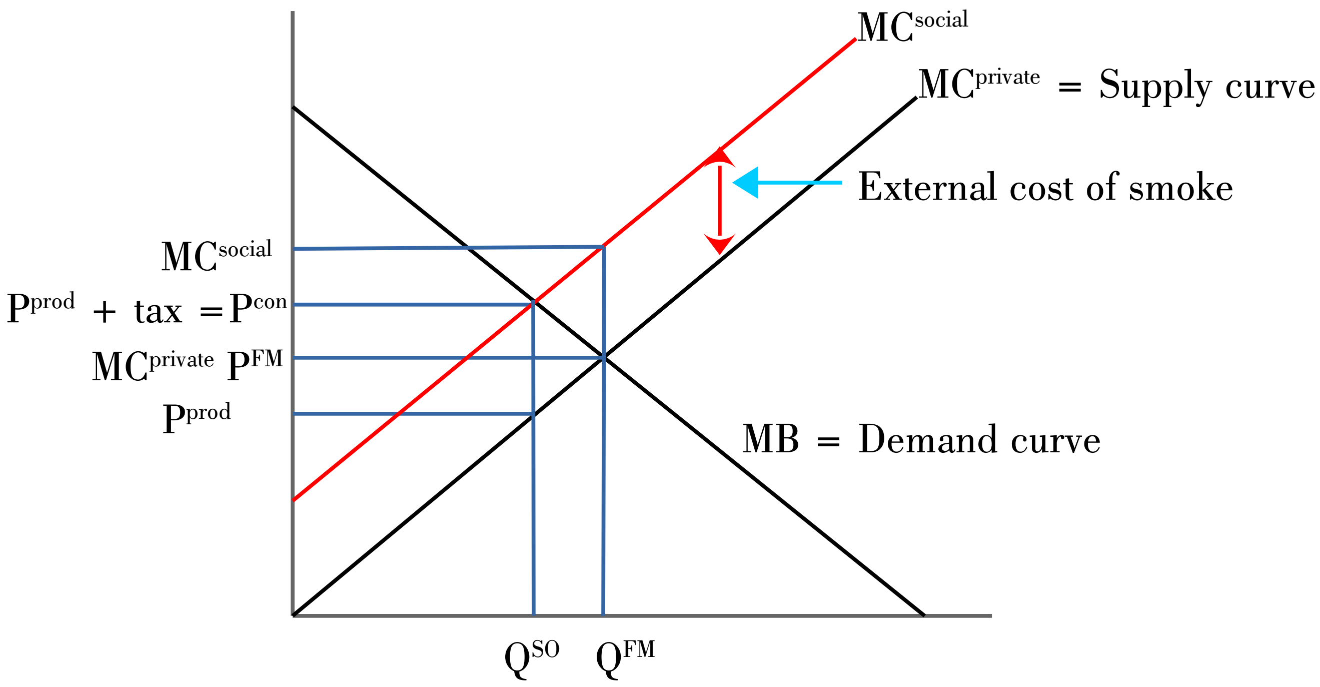
Most chapters will include a Section like this in which purely economic concepts are explained in more detail.

Subsection 1.5.1. Externalities

Externalities come in many forms, positive, negative, and even mixed, and from consumption, production, and many other choices that agents make. The key is that welfare of one set of agents is affected by the actions of another set of agents, other than through the price system.

The simplest example is a production externality, such as smoke or water pollution, that imposes costs on people or firms living near-by. We can model this using supply and demand curves.

In the figure above, competitive firms supply along their marginal cost curves, and consumers choose to purchase goods if the price is below their marginal benefit. Thus, the black supply and demand curves are the sums of the MC and MB curves of the firms and consumers, respectively, in the market, Suppose that the free market price of output is \$10 per unit. Then the **Private Marginal Cost** of production is \$10. That is to say, that if a firm wishes to produce another unit of output, it must spend \$10 to obtain the required inputs.



The firm also uses up clean air when it produces, but it does not have to pay for this input. The firm views clean air an input with zero cost it can simply appropriate. Since this imposes \$2 worth of cost on others, however, the full **Social Marginal Cost** of production is \$12. In fact, the people who are being damages would be willing to pay the firm \$2 for each unit of reduction in output.

The classic solution to this problem is to impose a tax equal to the external cost. This causes the firm to artificially **Internalize** the externality. Firms do not pay for clear air, but they do pay a unit cost equal to social value of the clean air it consumes. The result is that production goes down to the socially optimal level where marginal social cost equals marginal social benefit. Markets with positive externalities, on the other hand should be subsidized to increase output to the point where marginal social cost equals marginal social benefit.

A more general case of a consumption externality might be described as follows: Suppose that Alice and Bob go to a cigar bar for some scotch and stogies. They each are affected not only by their own consumption, but by the consumption of their companions. For example, Alice may enjoy cigars herself, but hate the smoke from the cheap cigars that Bob chooses. Bob enjoys his own scotch, and also enjoys it when Alice drinks a bit and starts to tell funny stories. Bob's consumption of both goods are therefore arguments in Alice's utility function, and visa versa:

$$U_A(CgA, ScA, CgB, ScB) \text{ and } U_B(CgA, ScA, CgB, ScB)$$

where CgA, CgB, ScA, and ScB are Alice's and Bob's consumption of cigars and scotch.

To make things simpler, let's assume that the utility of each agent is measured in terms of dollars. This means that the marginal utility of cigars and scotch for each agent equals the incremental amount of money they would exchange for an incremental unit of each good. Given this, we can calculate the private and social marginal benefit of each agent consuming these two goods as follows:

$$MB_{i,n}^{pri}(n) = \frac{\partial U_i}{\partial n} \text{ and } MB_n^{soc}(n) = \frac{\partial U_A}{\partial n} + \frac{\partial U_B}{\partial n}.$$

The socially optimal number of cigars for Bob to smoke (CgB^{soc}) is found by equating the social marginal benefit with the social marginal cost. However, Bob's incentives are to equate the private marginal benefit with the marginal cost, and smoke CgB^{pri} cigars:

$$MB_{B,CgB}^{pri}(CgB^{pri}) = \frac{\partial U_B(CgB^{pri})}{\partial CgB} = MC$$

$$MB_{CgB}^{soc}(CgB^{soc}) = \frac{\partial U_A(CgB^{soc})}{\partial CgB} + \frac{\partial U_B(CgB^{soc})}{\partial CgB} = MC.$$

Since $\partial U_A(CgB^{soc})/\partial CgB < 0$, it must be that $\partial U_B(CgB^{pri})/\partial CgB < \partial U_B(CgB^{soc})/\partial CgB$ for both equations to be true. Assuming diminishing marginal utility, we conclude that $CgB^{pri} < CgB^{soc}$. That is, it is social optimal for Bob to smoke fewer cigars that he chooses.

If Bob were forced to pay a tax of $t = -\partial U_A(CgB^{soc})/\partial CgB$ for each cigar. he would equate his private marginal benefit to the marginal cost plus this tax ,and choose to consume CgB^{soc} cigars:

$$MB_{B,CgB}^{pri}(CgB^{soc}) = \frac{\partial U_B(CgB^{soc})}{\partial CgB} - t = MB_{CgB}^{soc}(CgB^{soc}) = \frac{\partial U_A(CgB^{soc})}{\partial CgB} + \frac{\partial U_B(CgB^{soc})}{\partial CgB}.$$

In other words, the setting the tax equal to the externality he imposes cases Bob to behave as if he experiences the negative externalities he imposes personally.

Using the same logic, Alice chooses to consume less than the socially optimal amount of scotch. She should therefore receive a subsidy equal to the external benefits her drinking generates fob Bob. There are no externality associated with Alice's cigars, or Bob's scotch, and so no tax or subsidy is needed.

Of course there are a great many problems implementing these solutions in the real-world. For one thing, it is very difficult the measure the social cost or benefit of externalities. The real lesson here is that externalities prevent markets from working at full efficiency. Sometimes the failure to internalize externalities has catastrophic social costs, or cause complete market failure. Government interventions are often the only instrument available to make things better, but sometimes the cure is worse than the disease. Nothing is easy.

Subsection 1.5.2. Network Externalities

Network externalities are a special case of externalities that are usually positive. We can model them very simply as follows:

$$U(X,N)$$

where:

X : the quantity, or perhaps a vector of quality measures

N : the number of people that are consuming the good with you

and since there is a network externality:

$$\frac{\partial U}{\partial X}>0 \ , \ \frac{\partial U}{\partial N}>0.$$

For example, X might be the amount spent developing a platform, describe the features of social network, or the technical aspects of an operating system, while N is the number of subscribers or users. Obviously, agents care about the features of a platform. Some people like Facebook or Apple OS, and others prefer Instagram or Windows. Nevertheless, if a platform has too few users, it loses much of its value, even if it has a good feature set. It is no fun being the last person on Facebook, and it is easier to work with a well-supported, and widely-used operating system.

Of course, network externalities can be negative, in which case we can think of them as congestion. They may even start out positive, and turn negative. For example, a party can have too few, or too many, people in attendance. Even more complicated situations can be imagined. See the discussion of semirival goods in Chapter 6.

Chapter 2. The History and Future of ICT

The history of ICT is a fascinating topic by itself. This single chapter cannot do it justice. There are three main points we hope will that this outline will make.

First, how profound the effects on human culture advances in ICT have been. Consider how things such as writing, the printing press, the telegraph, or radio must have changed the world.

Second, that the time between paradigm shifting advances is getting shorter and shorter. There were 100,000 years or more between the development of language and writing, but only 4000 years or so between writing and the printing press. About 400 years after that, the telegraph, telephone, and radio were invented, and about 50 years after radio, we had the transistor, the integrated circuit, and the electronic computer,

Only 20 years passed between the production of the first commercially available mainframe computer and the beginnings of computer networking and the personal computer. Then, 10 years after that, we had the first Internet browser, which quickly led to widespread electronic commerce, Web 2.0, the mobile web, and the cloud.

Third, the incredible newness of the technologies that most readers of this book will take for granted as part of the landscape of their world. To choose some examples: the iPad (2010), Bitcoin (2009), Android (2009), the iPhone (2007), the cloud (2006), YouTube (2005), Facebook (2004), Google (1998), Amazon (1995), the World Wide Web (1991), the Apple Macintosh (1984), and the Audio CD (1980).

To put this another way, 30 years ago, the Internet and World Wide Web just barely existed, almost no one had a cell phone, personal computers were just starting to become common, and most people got their news from the nightly broadcast of one of the big three television networks or from newspapers.

Only 50 years ago that mainframe computers were just starting to become common, the Internet was a toy available only to a tiny number of researchers, integrated circuits were a new thing, and typewriters, Xerox machines, vinyl records, and cassette tapes, were the order of the day.

Now, artificial intelligence, machine learning, connected devices, blockchain, and smart everything, are transforming how we experience the world at pace too fast for governments, regulators, enterprises, and everyone else, to really understand.

Section 2.1. The Traditional Ages of Communication

The development of information and communications technology can be divided into four broad periods.

The Premechanical Age (? – 1450 AD): Human communication began with the development of spoken language. There is little direct evidence to tell us precisely how when this occurred. Inferences about the size and morphology of hominid brains from the fossil record suggest a range of between 100,000 and 350,000 years ago.

Cave paintings and stone carvings as old as 40,000 years have been found. These may be the earliest examples of humans storing ideas in physical form for transmission to future generations.

Written languages began to appear around 3000 BC. The Egyptians developed a system of hieroglyphs representing specific people, offices, and objects, for the most part. There were no phonetic elements. At about the same time, the Sumerians developed cuneiform, a kind of writing using a wedge-shaped stylus to make impressions in wet clay tablets.

Early cuneiform recorded only numbers, but latter included pictograms to indicate what was being counted. This eventually evolved in to a phonetic alphabet, but one which represented only consonants. The Greeks adopted these symbols and added vowels around 1500 BC. Written language also appeared in a few other civilizations in the fertile crescent and Mediterranean at around the same time. The earliest examples of writing in Asia date from about 1200 BC in China, and in the Americas, Mayan inscriptions dating from about 500 BC have been found.

Early writing was painted on walls or objects, carved in stone, or inscribed on clay tablets. Around 2500 BC, however, the Egyptians started recording information on scrolls of paper made from the papyrus plant, the Greeks began to cut paper into sheets and bind them into books in about 600 BC, and the Chinese developed a paper made from rags in about 100 AD. These innovations made it cheaper and easier to store and distribute information.

The Mechanical Age (1450 – 1840): This period begins with the invention of the printing press using movable metal type by Johann Gutenberg. His press dramatically reduced to cost of making a copy of a book. While there were still large **First Copy Costs**, once the type was set, additional copies could be produced for not much more than costs of the materials. This greatly facilitated and broadened access to knowledge and ideas of all kinds and challenged the effective monopoly that the church had on producing and disseminating knowledge. In the latter part of this era, scientists such as Pascal and Babbage invented mechanical calculators and computers of various kinds. These machines were analog devices that assisted in doing complicated calculations.

The Electromechanical Age (1840 – 1940):This period begins with the invention and spread of the telegraph. Wires carried pulses of electricity over long distances and activated a mechanical key that tapped out messages. Previously, news and information might take many months to travel by land or water to reach its intended recipient. It was not uncommon for battles to be fought between ships or at distant outposts long after a war was over simply because the news had not yet reached them. The telegraph effectively made the world a smaller place. In 1858, a cable was laid under the Atlantic Ocean connecting Europe and America with almost instantaneous communication. The telephone was invented in 1876, and the radio followed in 1894. These inventions radically transformed the scope and cost of both person-to-person, and broadcast communications. Electricity made these innovations possible, and was also used in new versions of the mechanical computer to power relays, gears, and mechanical disks.

The Electronic Age (1940 – ?): The Second World War sparked interest in computers to help in code breaking, atomic weapon development, and the calculation of artillery ballistic tables. Huge, fully electronic computers using vacuum tubes instead of relays and gears were built to help. These machines weighed many tons, had hundreds of thousands of components, and consumed a great deal of power. They were custom-built, programmed by hand, and very labor-intensive to use.

First Copy Costs: The fixed cost of producing the first unit of output. For example, the cost of writing a book, setting it in type, and printing the first copy, or making a movie, or writing software. Typically, the first copy costs are high, but the marginal costs of producing and distributing the second, and subsequent copies are very low, even close to zero. The average cost curve may always be above the marginal cost.

Section 2.2. Electronics of the 1950s and 1960s

Computers began to be commercially available in the early 1950s, The UNIVAC I and the IBM 701 are early examples. Both used vacuum tubes and were very expensive. Perhaps as a result, the market was not large. A total of 46 UNIVACs and 19 701s were eventually built before the designs were retired.

The solid state **Transistor** was invented in 1947, and the **Integrated Circuit**, which allowed transistors and other electronic components to be etched directly onto semiconducting material, was invented in 1958. These two technologies made it possible replace vacuum tube with smaller, cheaper, more reliable, less power hungry, components.

The **Mainframe** computers of this period gradually grew more powerful, but they remained expensive and rare. Data and programs were stored on punch cards, or reel-to-real tapes. Operating systems, and applications we usually built for, and packaged with, the hardware. Mainframes were generally accessed via local terminals, and were not connected to other computers.

These early mainframes were often custom-built and configured with the needs of purchaser in mind. There was very little thought given to compatibility or interoperability between systems. The idea of creating a network to allow computers to communicate with each other originated in the early 1960s at **ARPA (Advanced Research Projects Agency)**. The motivation was to allow researchers to efficiently share the use of computational resources even if they, or their institution, did not own the equipment.

The breakthrough that enabled networking was the development of **Routers**. Routers are small computers that stand in front of internal networks, and serve as **Gateways** to external networks. The broader network and its protocols were not directly visible to the interior network. Interior networks could behave in any way they wished, and only had to communicate with their own local gateway.

Routers break messages down into smaller **Data Packets**, add descriptive headers, and sent them out over the external network to other routers. The receiving router had the job of verifying that all the packets making up the message had arrived, reassembling them in the right order, and then delivering the message to the destination host in their local network in format it could comprehend.

Routers constantly updated one another about the state of the network, and had the task of choosing the best route for a packet to take. Different parts of a message might get to a destination node via different routes. and arrive out of order.

By the end of this period, the basics of semiconductors, integrated circuits, how to use them to build computer hardware, and fundamentals networking were worked out. How to do these things cheaply, and at scale, however, was not. These technologies were almost exclusively used by government agencies, research institutions, and few large companies.

Section 2.3. The IT Revolution of the 1970s and 1980s

As the 1970s progressed, large mainframe computers steadily dropped in price. Mainframes were used for applications such as processing large databases, keeping track of records, and executing complicated statistical and mathematical programs.

Examples include processing tax returns, storing social security and other government data in useful and accessible ways, keeping track of insurance, sales, and inventory records, hosting various accounting systems, doing statistical estimations based on scientific and other data, modeling weather systems and atomic explosions, and running aerodynamic and other simulations. Image processing and graphical applications also began in this period, and advanced significantly in the 1980s.

The biggest impact of mainframes was to automate wide range of clerical and office work dealing with information and communications. Significant capital investments were required to take advantage of these new possibilities. This favored large, well-funded companies with the expertise to deploy these new devices. It was difficult, especially in the earlier part of this period, for small businesses to justify the expense of modernizing their information systems, or to find employees who could use it effectively.

In the 1970s and before, the two most advanced pieces of office equipment in general use were the **IBM Selectric Typewriter**, and the **Xerox Copy Machine**. IBM introduced its version of **Personal Computers** in 1981, and desktop computers became a normal piece of office equipment by 1990. This significantly accelerated the pace of office automation and transformed the workplace. Word processing and spreadsheets were especially important, and greatly added to what office workers could do in-house, at low cost.

Perhaps surprisingly, office automation did not result in a decrease in the number of clerical employees in general. The new systems made it possible to do more with information than ever before. As a result, clerical workers changed their jobs and skill sets, but over all, were more in demand than ever. It is interesting to note that it is not always the case that technological advance throws people out of work.

Section 2.4. The Internet Revolution of the 1990s and 2000s

The experiments in the 1970s and 1980s, resulted in a number of networks developed independently, and with different priorities:

- ARPANET and NSFnet were focused on research and experimentation. They transmitted data via **Landline** and used point-to-point connections between hosts and nodes.
- ALOHAnet and PRNET were packet radio networks used to access remote locations. The quality and dependability of transmission was low, and so error control and correction was very critical.
- SATNET employed satellites that had higher bandwidth, but were more expensive to use.
- MILNET was established by the military, and encryption and security were of paramount importance.

These networks each used protocols and mechanisms optimized for their specific environments. The result was a set of incompatible networks that could not communicate with one another. The solution required Three things:

Data Packets each contain a small section of a larger binary file, along with a header that contains information about the sender and receiver, its sequence number (for example, a packet header might say that it is the 1203rd out of 39,400 total packets in the file), and other technical and routing information.

TCP (Transfer Control Protocol) standard was developed in 1974 and focused on the **Transport Layer**. Mores specifically, TCP breaks messages into a series of packets, adds headers, and passes them to the network layer for delivery. TCP also receives packages from the network layer reassembles them in the right order, and requests the missing or damaged packages be re-transmitted.

Routing is the job of **IP (Internet Protocol)** which was developed around 1978, and focused the **Network Layer**. IP finds addresses (32 bit IP addresses in the case of IPv4) to which packets and other data should be delivered, and sends them out through the local gateway.

These two were combined by DARPA in 1981, and become the official standard for the Inter net in 1983. **TCP/IP** continues to serve as a common protocol that allows arbitrary networks to communicate regardless of how these networks behave internally.

TCP/IP WAS THE BEGINNING OF THE INTERNET: A NETWORK OF NETWORKS

Although TCP/IP as an enabling technology was available in the 1980s, it was slow to have an impact. While application layer protocols such as **SMTP** (**Simple Mail Transport Protoco**l) for email, and **FTP (File Transfer Protocol)** for file exchanges, used TCP/IP as their transport and network layers, traffic was largely confined to universities and research institutes. The general public had little need for file transfer, and used other more familiar ways to send messages and communicate.

At the beginning of the 1990s, personal computers are fairly wide-spread (about 15% of US households owned a computer in 1990), and the TCP/IP made it technically possible for all these machines to communicate in a common network. Two more elements had to come together before the Internet would be widely used.

The first was general access to the network. Dial-up access only became possible in 1989 and AOL (America on Line) started supporting email for DOS in 1991. It was not until 1992, however, that Delphi emerged as a nation-wide provider of dial-up Internet services.

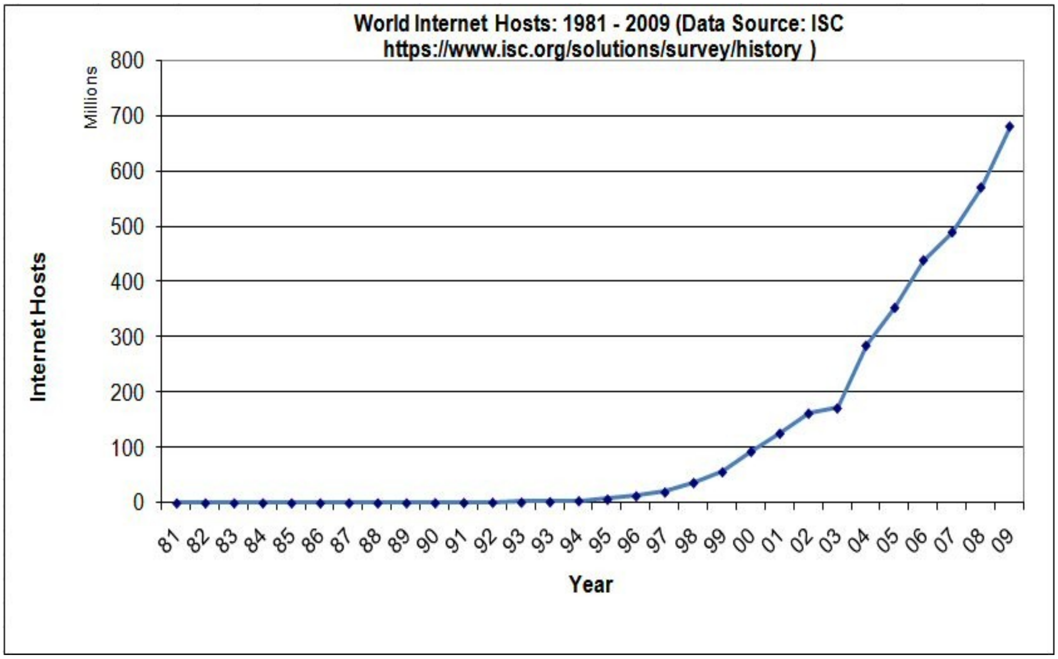
The second was a killer app. That is, there had to be a compelling reason to go to the trouble of connecting your PC to the internet. This turned about to be another application layer protocol called **HyperText Transfer Protocol** (**HTTP**).

Tim Berners-Lee's deployment of HTTP in 1991, marked the beginning of **World Wide Web**.[2] Berners-Lee and his team at CERN developed **HyperText Markup Language (HTML)** at the same time as a kind of language to describe (markup) how diffident elements of a **Webpage** should be rendered on a screen. It also described how images and other elements should be included and displayed. Most importantly, it defined the idea of **Hyperlinks**, which were references to content on other webpages that could be understood by HTTP, and allowed users to jump to other documents, and for documents to include content from outside sources.

The World Wide Web made it possible for people to sit at their desks, and access a huge array of text, music, pictures, video, and data, as well as make their own content available to others. Both broadcast and person-to-person communications became essentially instantaneous and free. The world became a smaller place than it had ever been before.

Windows 3.1 was released in 1992, and freed users from the command line, allowing them use a mouse to point and click at what they wanted the computer to do. Netscape Navigator, released in 1994 brought the World Wide Web into this new point and click world. With all of these key components in place, the Internet was at last ready to become useful to the average person.

The graph below shows how Internet, including some precursor networks, has grown over the years as measured by the number of hosts (that is, places that can be reached using an externally visible Internet address). You can see that not much happened before 1994, but since then, the Internet has seen an astonishing increase in size. As of 2019, there are approximately one billion internet hosts. By 2021, about 63% of the population of the planet has become connected. Europe is the most connected (90%), while Africa is the least connected (40%).



[2] Note that we are making a distinction between the Internet and the World Wide Web. The Internet is network of networks that moves files, emails, data, webpages, and other content using TCP/IP. The World Wide Web, on the other hand, is the part of the Internet that gets transmitted using HTTP.

As the number of users, the speed, and the quality, of the Internet increased, entrepreneurs began to develop a variety of services. The 1994 invention of **SSL (Secure Socket Layer)** made it reasonably safe to use credit cards on the web and both eBay and Amazon came onto the scene to take advantage. Expedia and PriceLine followed in 1996 and 1998, respectively. These companies offered early challenges to the retail and travel industries.

Entertainment services started later, mainly because of limited bandwidth. Commercial downloads of music were offered as early as 1994, but were not very successful. In 1999, Napster used **P2P (Peer-to-Peer)** technology to facilitate the sharing of files among Internet users. In large part, the files shared through Napster turned out to be copyrighted music and video files. The service was shutdown after an unfavorable court ruling in 2002.

On commercial side, Apple opened its iTunes music store in 2003, and was followed in 2006 by Microsoft's Zune Marketplace (which officially closed in 2013). YouTube was also founded in 2006, and in the same year, Netflix first offered video streaming service. Hulu quickly followed in 2007. This period also saw the launch of Skype and Wikipedia.

As the web grew, being able to find the content one wanted became more difficult. In 1991, a project called **Gopher** began with the aim of making a comprehensive index the web. Yahoo! entered the market in 1994 with a much more sophisticated approach using an algorithm to suggest search results based upon measures of popularity and linkages to other sites. Google and Microsoft's MSN Search both joined the search engine market in 1998. Search engines soon became central to the web experience. These companies took advantage of their gatekeeper role by selling search terms and advertising to companies hoping to gain the attention of web-users.

In the early days of the Internet, most webpages were static did not allow much interaction or input from the user. For the most part, users would go to a site and view, listen to, or download content. At best, sites facilitated person-to-person communications (Skype and email, for example) or pointed users in the direction of other fixed content or services (search engines, for example).

The founding of Facebook, Flickr, and World of Warcraft, in 2004 marked the beginning of what was termed **Web 2.0**. The appearance of YouTube (2005) and Twitter (2006) broadened this trend. The web started to become a kind of public space to meet and socialize, and to share opinions, art, and knowledge, with groups of followers, or the world at large. The days of the web as a kind of glorified library reference desk were over.

One of the consequences of Web 2.0 was that users began to voluntarily give up all kinds of private and personal information about themselves in order to participate. Internet companies, in turn, started to store and analyze information about users' search and purchasing habits, and even read and indexed their email and contact lists. The cost of participating in this new world has been the loss of a great deal of privacy.

Ironically, while the Web 2.0 brings us together in many ways, it also splinters us. We now have access to billions of people instead of just our families, friends, coworkers, and neighbors. It is possible to find people who share our exact interests and worldview, and then insulate ourselves from anyone who happens to be different. We can live in an echo chamber with others who agree with our political, religious, or lifestyle choices, however unusual, or mainstream, they happen to be. We no longer need fear being ostracized by those around us if we fail to accommodate ourselves to prevailing social norms or rules of civility. Web 2.0 can provide a community for anyone.

More recently, large technology companies have begun to impose their own views of what acceptable social discourse, and even viewpoints, should be. Most people would probably agree that threats of violence, distributing bomb-making instructions, certain types of pornography, slander, organizing criminal enterprises or behavior, should not be allowed. Of course, we already have laws against these things that can be enforced by existing, elected, civil authorities.

Whether, and to what extent, private companies should be allowed to, or be required to, enforce limitations of otherwise legal speech and communication is another one of those difficult policy questions such as what to do about natural monopoly. There is no perfect answer, and no solution that does not bring with it problems of its own.

Web 3.0 is supposed to be just around the corner. What this will turn out to be is anyone's guess at this point.

Section 2.5. The Cloud Revolution of the 2010s

The Cloud refers to large pools of easily usable and accessible virtualized resources such as hardware, development platforms and/or services. These resources can be dynamically reconfigured to adjust to variable loads (that is, they are scalable), allowing for optimum resource utilization.

This pool of resources is typically exploited under a pay-per-use model in which guarantees of service quality and access to computational resources are encoded in customized **SLA (Service Level Agreements)** offered by the infrastructure providers. We explore in more detail below exactly what cloud computing is, and why it is so revolutionary.

Subsection 2.5.1. Service Models

The term **Everything as a Service** (**XaaS**) is applied to the core services offered by cloud providers. There are three primary **Service Models** in **Cloud Computing**:

SaaS (Software as a Service): This service model abstracts from components at the lower layers such as programming languages, operating systems, networks, servers, and storage, and focuses on providing services directly to end users. Examples include Gmail and other Google Apps, Microsoft 365, Concur, Salesforce.com, social media, and content delivery services.

PaaS (Platform as a Service): This service model abstracts from the hardware layer, but gives users a flexible development or runtime environment that can be provisioned with various programming languages, **APIs (Application Programming Interfaces)**, web services, databases, and so on. User do not manage or control the underlying cloud infrastructure such as the network, computer hardware, operating systems, and storage. PaaS is primarily targeted at enterprise and business users building applications or ICT infrastructure for their own purposes. Examples of these scalable services include Google App Engine and Microsoft Azure.

IaaS (Infrastructure as a Service): This service model allows users to put together the processing, storage, data transfer, and other fundamental computing resources they need to build platforms. Uses can run arbitrary software, operating systems, and applications without needing to manage or control the underlying cloud hardware. Typically, resources are dynamically provisioned to meet the users needs at each moment. Amazon Web Services (AWS), Rackspace, Google Compute Engine, and some elements of Microsoft Azure, are examples.

Cloud services are becoming more disaggregated, and specific, and now include communication, data storage, blockchain, and even robots as a service.

Subsection 2.5.2. Why does the Cloud Matter?

Cloud computing is a relatively new phenomenon. Although one might argue that we saw a leading edge in time-shared computers as far back as the 1970s, and that Oracle and Salesforce.com began to offer SaaS applications in the 1990s and early 2000s, respectively. However, these were specialized services with a limited impact.

The credit for first making cloud services available to a wide audience probably has to go to Amazon. In 2006, this company launched their S3 simple storage solution and EC2 Beta (the latter became a full-blown commercial service in 2008). Google App Engine and MS Windows Azure started in 2008 as well.

On the face of it, cloud computing does not sound that revolutionary. After all, almost everything that is done on cloud services was already being done before using other means. So what makes this a transformatative technology? There are three main factors:

Cost: Cloud computing services are much cheaper than the equivalent ICT services provided in-house. This is for a number of reasons:

Usage Rates: If a company keeps all of its servers in-house, it must build enough capacity to handle its peak-load. In practice, most private servers run at an average capacity of about 20%. Cloud providers typically run their servers at about 80% of capacity. The ability to provide the same server capacity using a quarter of the physical resources is perhaps the most important cost advantage of cloud services.

Location: Private server rooms are typically colocated, and occupied expensive real estate in big cities, Cloud server farms, on the other hand, are located out in the country where land and electricity are cheap.

Scale Efficiencies: Private servers are bought as needed, are heterogeneous, and typically require one technician per 50 machines. Cloud servers are deployed, provisioned and updated, at the same time, are homogeneous, and typically require one technician per 1000 machines.

Security: ICT brings with it a number of different security concerns.

Physical Security: Storms, floods, and terrorist attacks, can literately destroy servers and their data with them. Server rooms in heavily trafficked buildings are difficult to make secure from employees and other who may steal or destroy data, or even the servers themselves. Server farms are built in rural locations, are easier to secure, can be specifically hardened to withstand natural disasters and human assaults, and spread the costs of security more broadly.

Hacking and Virtual Security: Maintaining and updating firewalls, security protocols, blocking newly discovered exploits, and making sure encryption and routing is secure, are complicated tasks that require specialized skills. Not every company can find or afford the expertise to make sure that their data is safe. Security experts at server farms provide this service for thousands of users at once. This is far cheaper, and more dependable.

Robustness and Redundancy: Systems fail from time to time, and the cloud approach offers some distinct advantages in dealing with this risk.

Bandwidth and Power: Servers in cities typically have one choice of where to connect to the backbone, and where to get power. Cloud providers build farms where they can access two or more fiber lines, and on the border between power suppliers.

Multiple Data Centers: Cloud providers can easily keep synchronized copies of data in physically separate locations. If one facility drops off the network, goes down, or is destroyed, the data is still safe and available.

Distributed Caching: Latency refers to the time between a request from a client, and arrival of the data from the server. If latency is too high, then content seems to trickle in, interactions with webpages are slow and laggy, and in particular, streaming content is glitchy because of delays in the arrival of necessary packets. This can make interactive gaming, and music and video streaming services unusable.

Google, Netflix, and many other companies, use a cloud-based solution called **Distributed Local Data Cacheing** in which multiple servers holding the most commonly requested content are set up closer to each set of users. This improves the average user's experiences and also reduces overall traffic on the web by shortening delivery routes.

Flexibility and Scalability: It takes considerable time, and a commitment of significant capital, to buy, locate, and deploy, private servers, and hire the technicians needed to support them. Online companies must anticipate the server capacity they will need well in advance of actually acquiring customers. Deploying too much, or too little capacity are both disasters. Depending on in-house servers is risky, capital intensive, and limits flexibility. AWS, Microsoft, and other cloud providers, use virtualization to create any number virtual servers a client might need on the fly. Companies are never caught flatfooted when demand grows or sales spike, and are not left with unneeded capacity if demand does not meet expectations.

Cloud computing and virtualization, turn the fixed cost of buying servers, into the variable cost of renting them as needed. Note that this is exactly the opposite of what happened in the IT revolution of the 1970s and 1980s where taking advantage of new technologies required buying equipment and training employees. This increased the fixed capital costs of being in business, and also committed companies to difficult-to-reverse investment decisions. These IT advances worked in favor of large, well-capitalized, incumbent companies.

Cloud services allow anyone with a good idea to get it into the marketplace with little investment or risk. If demand turns out to be large, then the revenues will cover the bill for cloud services. If the demand turns out to be small, the bill for cloud services will be small as well. This democratizes entrepreneurship by lowering barriers to entry and making it easier for new companies to challenge incumbents.

Other interesting things about the cloud:

- Overall, the cloud services industry’s revenues for 2022 are estimated to be $408.6 billion. By way of comparison, 2022 global industry revenue for recorded music was about $25.9 billion, online streaming and subscription video, about $82 billion, and online gaming about $249.5 billion. The overall US GDP in 2022 is estimated at $25,462 billion.)

- It is estimated that there were 6.8 ZB of total cloud storage in 2020 (a Zettabyte is 10^{21} bytes. or one trillion Gigabytes).

- Market share estimates:
  - Amazon Web Services 32%
  - Microsoft Azure second with 16.8%
  - Google Cloud 8.5%.

Some takeaways:

- Costs, flexibility, and security are generally better with cloud solutions compared to running private servers. There is an overwhelming business case to depend on cloud providers.
- On the other hand, by putting your data and applications in cloud, you are using machines that are not under your direct control. A cloud provider can:
  - Prevent you from accessing to your data.
  - De-platform your or your company.
  - Raise the cost of the services you use
  - Change your data.
  - Read your data, or violate your privacy.
  - Allow its business partners, or the government to access your data.
- In general, extreme concentration in the cloud industry, and the existence of communications choke points, make internet must less decentralized, redundant, uncensorable, robust, and free, than was originally envisioned.

Section 2.6. Emerging Technologies

In this Section we discuss several critical emerging technologies, and their potential implications for the future.

Subsection 2.6.1. Blockchain

Most data is under the control of one company or another. Visa, Mastercard, and our banks maintain and control our financial records, the government, our tax and social security records, Amazon, our purchase history, Google, our search and YouTube preferences, etc. These entries are often referred to as trusted data intermediaries.

Trusted Data Intermediaries (TDI): A company or other organization that aggregates, stores, and/or distributes data contributed by users under some agreed upon terms and conditions.

TDIs are central to almost every aspect of the cloud revolution. Network externalities make it impractical for most people, and even most corporations and government agencies, to avoid using their services.

One concern this raises is that users can never really know what TDIs do with their data. Some TDIs are quite explicit that they mine user data for profit. Google uses AI/ML to read the emails that go through its systems, and Microsoft does the same for all the documents it stores for individuals and businesses.

Of even more concern is that users have an extremely limited ability to know, much less prove, if their data is being kept correctly. It is easy for a TDI add, modify, or delete any record they control. There is very little a user can do to prevent this, even if he can detect such manipulations.

Perhaps the greatest concern is that TDIs could choose, or be forced, to deny users access to their data, and the services that they support. Imagine if everything you had stored in the cloud, your photos, your documents, your social media postings, and so on, were deleted. What would you do if your access to email, messaging, and even mobile phone services, were cut off?

Most frightening of all is the possibility of being denied access to your bank and credit card accounts. How would you survive if you literally had no access to electronic money? You could neither work, nor buy food, shelter, or any other necessity for living.

We are rightly concerned about relying on the competency, ethics, and honesty of large financial institutions, technology monopolies, and government agencies. But what is the alternative? In particular, how is it possible to build a consistent, non-manipulable, data system to coordinate the activities of agents who neither know, nor trust each other, without a TDI?

You have may have heard of **Bitcoin**, which is often described as a **Decentralized Currency**.[3] Bitcoin is one of several thousand **Cryptocurrencies** that have appeared in the last decade (Ethereum, XRP, USDT, and Dogecoin, to name a few prominent examples) built on blockchains.

Blockchain: A database mechanism using **Distributed Ledger Technology** (**DLT**) that allows transparent information sharing and coordination between mutually untrusting parties. The ledger is maintained by a network of nodes that choose and verify transactions submitted by users. Valid transactions are collected into blocks, and if a qualified majority of the nodes reach a consensus that a block is correct, it is cryptographically appended to the previous block, and used to update the ledger state.

Cryptocurrency is only one of many uses for blockchains, and arguably one of the less interesting ones. Blockchains provide a distributed source of consistency for any kind of data without the need for trust between parties, a TDI, or central point of failure.

More interesting applications include: Non-Fungible Tokens (NFT), tokenization of financial and real assets, decentralized finance, decentralized currency exchanges, logistics, provenance, chain of custody, notary, and attestation services, to name a few.

Blockchain is the only emerging technology that supports decentralization instead of monopoly and other forms of concentrated of power and control. It is possible to build an ecosystem of companies and services that compete with banks, Google, and Amazon, and allow people to interact and exchange without the permission of large central players. How much of this vision will ulti-

[3] Technically, "coin" refers to the single cryptocurrency that is native to the platform, bitcoins on the Bitcoin chain, or ethers on the Ethereum chain, for example. Cryptocurrencies that are created on top of a blockchain platform, usually through a smart contract are referred to as "tokens", Tethers (USDT) Basic Attention Tokens (BAT), and Chainlink (LINK). We will both interchangeably below.

mately be realized, remains to be seen. We will discuss blockchain in greater detail in later chapters.

Subsection 2.6.2. Quantum Computing

The digital age is more accurately called the binary age. Essentially all modern information and communications technology is based on binary encoding. Analog information of all types must be reduced sequence of binary **Bits**, that have a value or either zero or one, before it can be processed by computers and related hardware.

Quantum Computing uses something call **Qubits** that take advantage of the superposition of quanta to gain computational advantage compared to standard digital computing methods. Qubits have the strange property (which Einstein called “weird”) of having more than a single value at once.

A major downside of qubits is that they must be kept extremely cold to be useful (0.015 degrees Kelvin). For reference, the background temperature of interstellar space is 3 degrees Kelvin. Several stages of cooling involving liquid nitrogen, hydrogen, and helium, in succession, are required. The apparatus that supports each qubit weighs hundreds of pounds, and requires a large building with heavy industrial equipment. Quantum computing is a massively expensive undertaking, and is likely to remain so.

Currently, there are two main approaches to quantum computing:

Quantum Annealer: This approach is similar a mathematical optimization algorithm called **Simulated Annealing**. A company call D-Wave has been working on this since 1999, and Google has now taken an interest. The upside to this approach is that its qubits are *relatively* easy to make and use, and systems with 1000 qubits or more seem feasible. The downside is that quantum annealing can only the address a limited class of problems. In particular, it is unable to run **Shor's** or **Grover's Algorithms** to attack asymmetric and symmetric cryptography. It is unclear how much of a general computational advantage quantum annealers offer over digital computation beyond of class of specialized problems they are designed for.

Analog Quantum Computing: This approach is more general and uses qubits as a type of logic gate. Computers of this type can run Shor’s and Grover’s algorithms, and confer a **Quadratic Advantage** is breaking AES and most existing forms of symmetric encryption. That is, if it would take N guesses to crack a given key using a digital computer, it would take only $\sqrt[4]{N}$ guesses using an analog quantum computer (or so it is thought). The downside is that analog systems are difficult to build, especially at scale. Their qubits seem to suffer from cross-talk, and interference from external energy sources (electromagnetic emissions, vibrations, etc.)

The current target is something called **Universal Quantum Computing** which will give generalized computational advantage over digital computers for all classes of problems. It is unclear at this point how this will be accomplished, or how soon.

If and when quantum computing becomes truly practical, it is likely to be expensive and available in limited quantities at first. At best, it will probably not be widespread until 2030 or 2040. It is unknown if quantum computing will have transformative application to artificial intelligence. At this point, the main applications seem to be in cryptography.

The most serious threat of quantum computing is to legacy encryption schema. In particular static records, communications, and other data that are encrypted with currently standard methods are vulnerable to quantum attack.

For example, if a bad actor hacked, and captured, sensitive, but encrypted data, it would only be a matter of type until quantum technology would advance to the point that it would practical to break its encryption. Similarly, blockchain accounts are secured by public/private key pairs in a schema that is fixed by protocol will also become vulnerable as time goes on.

The solution is to move to quantum resistant encryption schema, where possible. Basically, this requires that much larger keys be used for both symmetric and asymmetric encryption. There is a trade-off between using longer keys with less computation (which takes up storage and bandwidth) and relatively shorter keys with more computation (which costs time and electricity). In all cases, quantum resistant keys are at least several times as long as what is standard at present.

The conclusion is that quantum computing should be seen is as a potentially large, but ultimately incremental, computational advance. It will eventually be able to solve some types of problems that are computationally impractical for digital computers, although at high cost. Fortunately, we already have available equivalent problems that are computational impractical even for quantum computers.

Subsection 2.6.3. Artificial Intelligence

Artificial Intelligence is a broad term for the ability of machines to imitate human behavior or perform tasks that require human intelligence. There are many possible approaches including direct programing, machine learning, and neural networks. Alan Turing proposed a test for AI in 1950 that involves a conversation using text between an evaluator and either a human or a machine. If the evaluator can not reliably identify when he was talking with a machine, then the machine passes the **Turing Test**, and could be considered an artificial intelligence.

Before giving definitions, consider the following example for context. Suppose that I had a bunch of data on my customers including names and addresses, purchasing history, click steams, and perhaps information collected through other platforms. I would like to see if I could somehow up-sell my customers to more expensive items by advertising to them, offering options, displaying the page differently, or biasing search results.

I could run experiments of various kinds (sometimes called **A/B Testing**) and record the outcomes. Next, I would randomly divide this data into a **Training Set** and a **Test Set**. I would give the training set to the machine and include the outcomes. The machine would use various mathematical and statistical approaches to make best model it could to predict how customer data was related to the outcomes.

The last step is to use the test set to validate the ML model just created. The machine is given the test set data without the outcomes, and uses the model it created to predict customer response. The trainer knows the outcomes, and so can compare the machine's predictions with what really happened. Essentially, the model attempts to categorize customers into types, for example customers who respond to "special offers!", claims of high quality, or the cool factor, and those who are unmoved by anything.

Machine Learning is the most widely used approach to AI. Google, Amazon, and other companies with huge data sets use ML for analytics. Machine learning begins with a set of observations of structured and unstructured data that are associated with one or more outcomes. Mathematical models that attempt to identify patterns in these observations that are associated with each of these outcomes. These models are calibrated or "trained" using a sample of the data, and them used to predict the outcomes associated with new observations. In essence, machine learning is the process of optimizing a predictive model to produce a **Classification Engine**.

Neural Networks: This is a special approach to ML that uses the human brain as a model. A system of interconnected artificial neurons is constructed on a computer and these connections modified and reorganized in response to external inputs such as data sets or sensor information. The machine needs to be trained in some way in order to understand how to improve the organization of its neural net. This is similar to how humans learn that fire is hot and chocolate is yummy.

Deep Learning: This is a neural network on steroids. Deep learning uses large multilayered networks and huge amounts of processing power. Important applications include image and speech recognition, natural language, and translation services.

Artificial intelligence is making great leaps forward with **Large Language Models (LLM)** and **Generative AI**. Such systems are largely self-training, unlike traditional machine learning approaches. It seems like magic in some ways. AI is able to extract something meaningful from natural language, and correlate it with its knowledge base to generate something like a meaningful response.

Generative AI can be thought of as a kind of search engine that can create responses by integrating data, rather than simply pointing to some piece of existing content. These systems process truly vast amounts of data to extract salient characteristics, and note similarities and underlying structure. The resulting models can then be used to create new content that resembles data that the system classifies as examples of what the user is requesting.

In effect, Generative AI returns a kind of weighted average of information it finds likely to be responsive to queries. In this sense, it is a bit dumb. On the other hand, it is basing these responses on a vastly larger store of information than any human could every hope to integrate. In this sense, they are very smart, or at least knowledgeable.

Generative AI is also known to make things up. Averaging over a scientific literature, for example, can yield papers that could, or should, have written on certain topics by certain authors. The fact the such papers were never actually written is not disqualifying for generative AI. Of course, people, especially economists, make things up (or sometime misremember them) as well.

Many people wonder if these technologies can be racist, sexist, or have other undesirable biases. The answer is that it depends upon what you mean by bias. In all cases, the choice of training data, structural parameters built into ML algorithms, and the definition of what a correct classification or response is, drive the results ML provides. Systems simply optimize algorithms trained on whatever data its is shown. Data can be biased, non-representative, or incorrect. The questions asked, and how outcomes are scored, can also reflect the basis of the developer. The algorithms themselves, however, are just math instantiated in computer programs. They don’t care.

A better question is whether it is ethical to use certain results, or ask questions in certain ways. For example, AI will probably show that income predicts health, and therefore the costs of providing a person medical insurance. Race is correlated to income, and may have other independent predictive value. Assuming these findings are correct, should we allow insurance rates to be higher for poor people, or let race play a role in setting premiums? These are social and ethical questions that are independent of any analytical finding, regardless of source. Predictive engines gotta predict. It is up to us what we do the results.

Section 2.7. Social Implications of Technological Convergence

Each of these emerging technologies brings with them an array of benefits, as well as possible harms. Any powerful technology can be used for good or ill. There is confluence of three technologies, however, which taken together, create the potential to change society in fundamental, and concerning ways.

Universal Connectivity: Cell phones, tablets, and computers, are devices specifically designed to allow humans to make connections to one another. The broader array of devices that communicate through **Near Field Radio** (**NFR**), Bluetooth, Wi-Fi, cellular **Personal Communication Service** (**PCS**), and even wired, connections form what is called the **Internet of Things** (**IoT**). This includes “smart” TV’s, appliances, thermostats, speakers, light bulbs, locks, security cameras, robot vacuums, cars, industrial and agricultural equipment, medical monitors and devices, traffic signals, parking meters, security cameras, and the list goes on. All of these devices send user created input, data gathered by their sensors, and other types of telemetry, to Google, Microsoft, the government, and a huge variety of others. A large and rapidly growing fraction of everything we do is recorded by some device, and transmitted to some central location. Surveillance is all but impossible to escape.

Cloud Computing: All this information has to go somewhere. The cloud provides an inexpensive way to store all the data that these billions of devices send about their users forever.

Artificial Intelligence: The amount of data that devices collect, and the cloud stores, is huge. It is also messy, consisting of documents, postings, click streams, location tracking, browser histories, sound and video recordings, and so on. It would have been impossible to make any kind of sense such a massive amount of unsorted, unstructured, data until quite recently. Now artificial intelligence can use tools like voice and image recognition to sort data by user, and them analyze it for any number of purposes.

Let's imagine for a minute what this makes possible.

- Any sort of information or viewpoint can be automatically filtered and suppressed. Google, content providers, media platforms, your ISP, etc. can simply choose not to show you anything they feel is undesirable or dangerous. The government could easily force them to so as well. Even if information is not suppressed, the fact you choose to access it can be recorded and added to your social profile.
- Your smart speaker, mobile device, TV, or computer, can turn on its microphone and camera without you being aware of it. Natural language engines are able to pars your words into text, and AI can interpret the images. Have you every said something in you home you would not want to be made public? Have you ever done things, legal or illegal, that you would not want filmed? Are your eating properly, drinking too much? Even if some government agency, or your boss, does not use this information directly to punish or control you, such surveillance data combined with analytics would establish a very complete picture of your tastes interests, physical habits, and even your private thoughts, should you choose to express them where a device might hear. But have nothing to hide, right? Why should your worry.
- Your GPS enabled, connected, car can tell the police in real time if you are speeding, potentially drunk, driving without insurance, or driving to a church, a protest, or some other place you shouldn't. The police, a hacker, or even the car manufacturer, could turn your car off, or make it crash.
- Any GPS enabled device (a car, phone, a tablet, or laptop, for example) can create and report a complete record of your movements and transmit your current location. Who you visit, associate with, who is beside you in a bar, or in your bedroom, can be detected by noticing that their cellphone is close to yours. Did you witness or commit a crime, did you attend a political meeting or a demonstration? Are you where your boss thinks you should be?
- Do you fit the profile of someone who is likely to have committed a crime in the past, or is about to commit one in the immediate future? In the first case, authorities might go on a fishing expedition, despite having no actual evidence. In the latter case, a predictive policing policy might lead to your being arrested, or closely surveilled, without your ever having done anything wrong.

You may think this list is far-fetched. I hope you are correct. These three technologies will only get better, and cheaper, in the future. The temptation to use them in big ways and small will be overwhelming. Perhaps we will refrain from using these techniques except in cases of terrorism, or child trafficking, or maybe racism. It is s slippery slope, and it is impossible to really know the degree to which governments, corporations, and other large actors, are using these technologies. Even if our government continues to respect the rights of its citizens, totalitarian nations like China and Canada are likely not to.

Section 2.8. Economics

Subsection 2.8.1. Normative and Positive Economics

The models and analysis we develop in this book are aimed at addressing questions from a positive rather than a normative standpoint. By this we mean:

Positive Economics: Economic statements about what is. No opinions are offered, just facts and analysis that follow from assumptions.

Normative Economics: Economic statements about what should be. These involve value judgments based on political, religious, philosophical, and ethical beliefs.

This is why we talk about economics as a science. Just like a doctor or a physicist, our job is to give the best possible prediction of how cause leads to effect. It is not the doctor's job to tell you that you should exercise, just that you are likely to die sooner if you do not. It is not the physicists job to tell you that you should not drop bricks off of highway overpasses, but to tell you the consequences if you choose to do so.

In the same way, it is not the job of economists to tell you to support lower taxes, free trade, or welfare reform. Our job is to use analytical tools, like the ones we develop in this course, to understand the implications of such choices and then allow others (and ourselves) acting as citizens to decide which outcomes we prefer as individuals or a society.

Subsection 2.8.2. Economics of Technology Diffusion

There is a large economic literature on innovation, the impact of new technologies, and many related topics. On especially interesting finding is that rate at which technological knowledge diffuses from its source has increased with improved communications.

Adoption lags between the invention of new technologies, and their adoption in other countries have narrowed. This should have implications for speeding the convergence of incomes across countries. An interesting question is whether the first mover advantage that new technologies give to innovative societies will continue to outweigh the windfall of freeriding on these innovative efforts gained by later adopters. The table below gives several historical examples showing how diffusion has changed over history:[4]

Historical Rates of Technology Diffusion		
Technology	Date of Invention	Average Adoption Lag
Steam and Motor Ships	1788	121
Railways	1825	72
Telegraph	1835	45
Automobiles	1885	39
Aviation	1903	28
Synthetic Fiber	1924	28
Liver Transplant	1968	18
PCS/Cellphones	1973	14
Internet	1983	7

Subsection 2.8.3. Monopoly

In all markets, profits (π) of a firm equal the Total Revenue (TR) minus Total Cost (TC).

$$\pi(Q) = TR(Q) - TC(Q).$$

We can find the profit maximizing output level by taking the derivative and setting it equal to zero.

$$\frac{\partial \pi}{\partial Q} = \frac{\partial TR(Q)}{\partial Q} - \frac{\partial TC(Q)}{\partial Q} = 0 \;\; \Rightarrow \; MR(Q) = MC(Q).$$

In competitive markets, firms are price takers, and so TR = $Q\overline{P}$. That is, from a firm's perspective, the price is fixed and does not change regardless of the quantity a firm produces. This implies that MR(Q) = $\overline{P}$ for competitive firms.

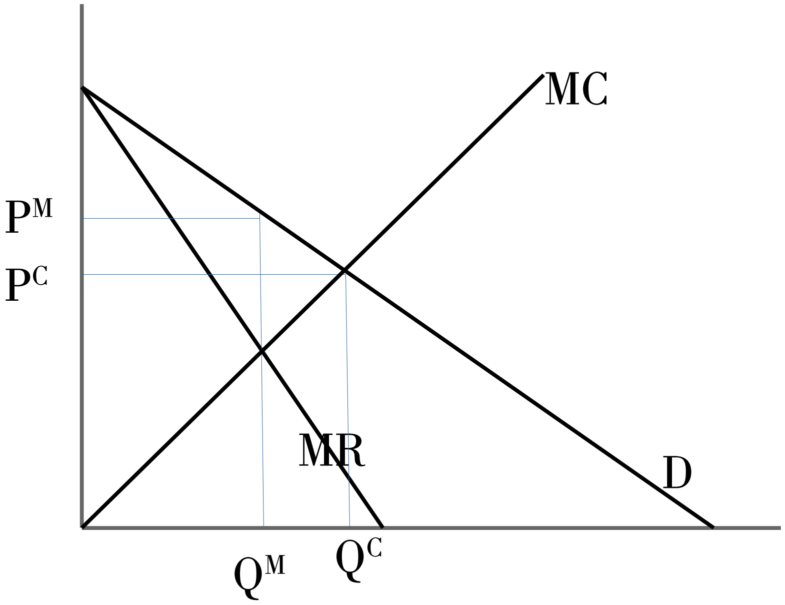
Monopolies are price makers. Their market price is a function of the quantity they produce. The dilemma faced by a monopolist is that to sell more output, it has to lower price not only on the last item sold, *but on all the previous units of output produced as well*. Thus, TR = QP(Q). This implies that:

$$MR(Q) = P(Q) + Q\frac{\partial P(Q)}{\partial Q}$$

[4] Table abstracted from: Comin, Diego, and Martí Mestieri. "If technology has arrived everywhere, why has income diverged?." *American Economic Journal: Macroeconomics* 10, no. 3 (2018): 137-78.

and so the marginal revenue curve is downward sloping and steeper than the demand curve.

For linear demand curves, it is easy to show that marginal revenue curve has twice the slope of the demand curve. We can use this to compare the equilibrium price and quantity outcomes for monopoly industries compared to competitive ones.



Subsection 2.8.4. Natural Monopoly

Natural monopolies occur when an industry is characterized by high costs of initiating production, but lower incremental costs for producing subsequent units of output. Formally:

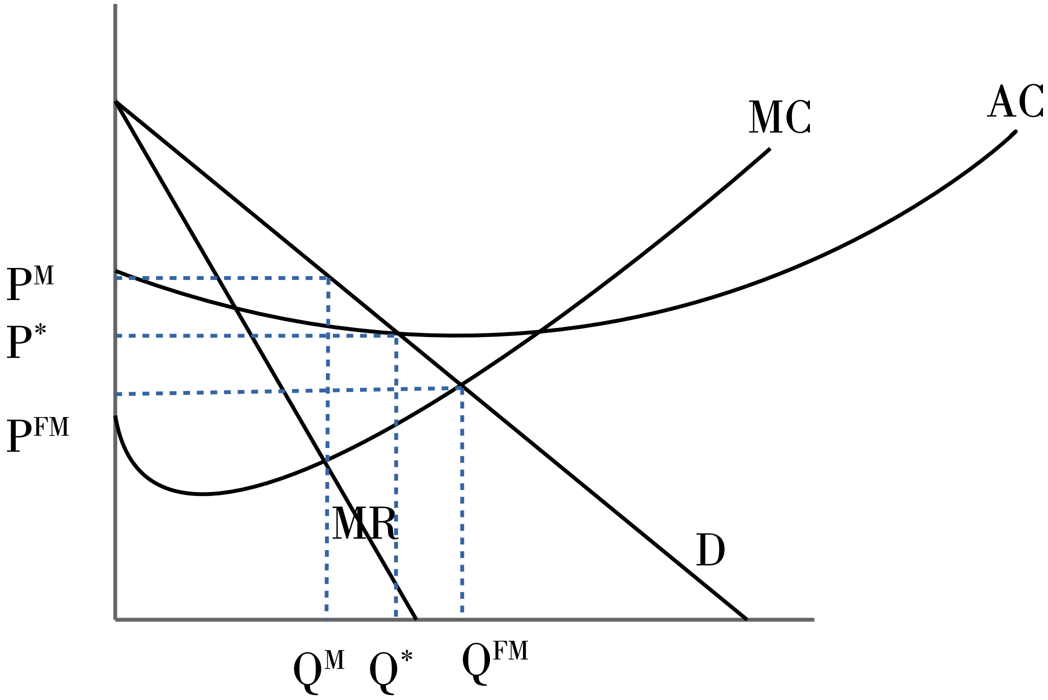
Natural monopoly: A firm is said to be a natural monopoly if the minimum of its *AC* curve is above the market demand curve.

For example, to build even a single new passenger aircraft, Boeing has to design it, test the model, get regulatory approval, build an assembly line, retool its machines, train its workers in the new process, and so on. Thus, if it made only one copy of the new design, it would be extremely expensive. The second unit, however, only requires the additional time and materials that go into actually producing the aircraft, so the incremental cost of the second unit is much smaller than the first. In other words, the *AC* of two aircraft is smaller than the *AC* of one aircraft, and that AC continues to decrease for a long time (maybe forever).

In the ICT space, businesses characterized by high first copy costs are often natural monopolies. Products like pharmaceuticals that involve significant research and development, or complex computer chips and devices that require extensive design and testing, also fall into this category. More generally, industries with significant economies of scale or scope, or with large “fixed costs” of beginning production are likely to be natural monopolies.

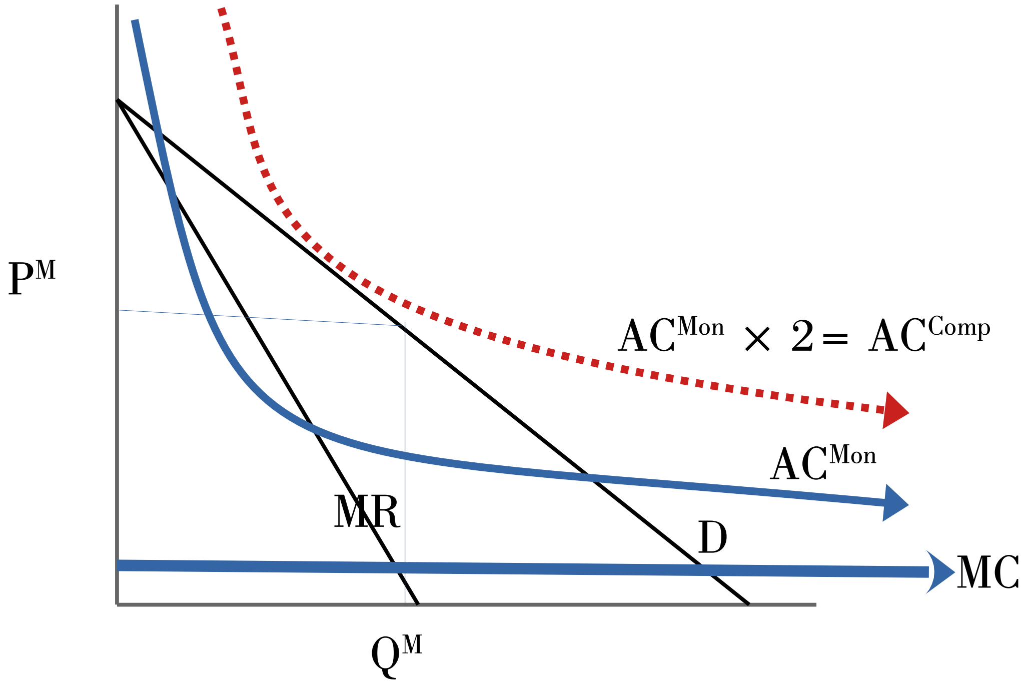
An example can be seen in the figure below. Notice that if the firm was forced to behave like a competitor, it would set choose price and quantity where marginal cost curve crossed the demand curve. This is unsustainable, however, because this "free market" price is below average cost. Thus, average cost of production less than average revenue (P^{FM}), implying the firm loses money on every unit of output. It would have to go bankrupt since profits would be negative.

On the other hand, allowing the firm to set the monopoly price results in consumers paying a higher price, for a smaller quantity. If the price and quantity were set by regulation at P^*, where $AC = D$, consumers would be better off with a lower price for a larger quantity, while the firm would just cover costs.



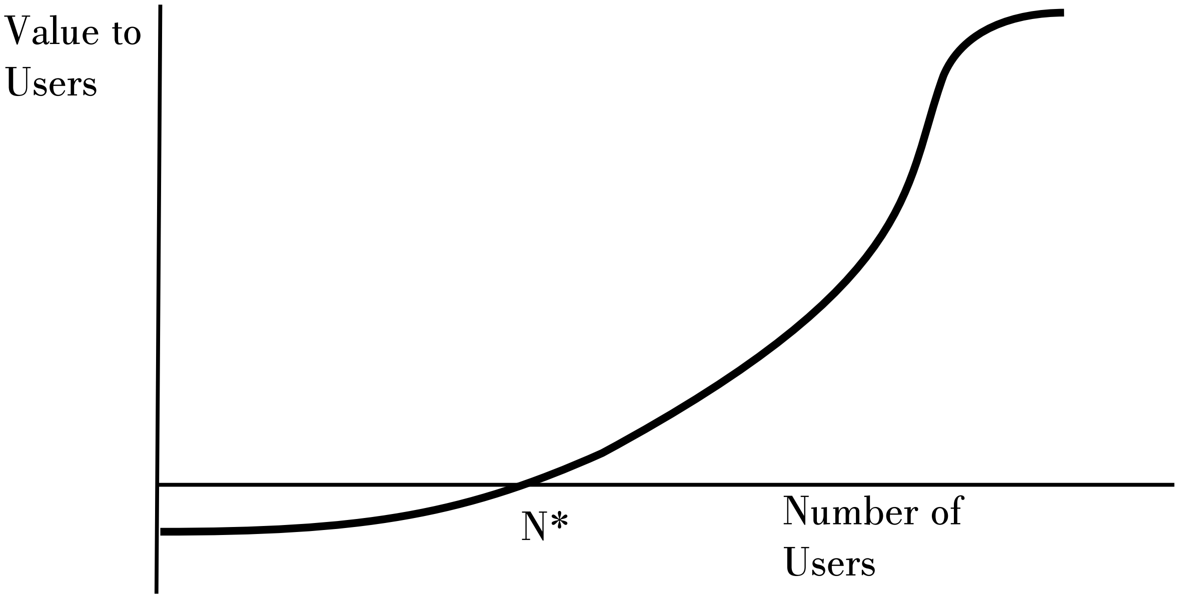
In the illustration below shows the extreme, but common, case of large first copy costs and low marginal costs. We see that the AC will above the MC even at high production volumes The blue curves show the monopoly outcome.

If we broke this natural monopoly into two firms that split the market between them, then the average cost would approximately double (since twice the fixed first copy costs would have to be divided into the same total output). In the example, we see the result is that the AC is above the demand curve at every quantity. In turn, this implies that there is no quantity where the competitive price could cover the average costs of production, and so the two firms would necessarily make negative profits and have to shut now. Clearly the monopoly outcome is better than zero output. In all cases, it is unclear whether consumers or society benefit from breaking up natural monopolies.



Subsection 2.8.5. Agglomeration and Network Product Differentiation

Consider the incentives of an up-and-coming social media platform, or any business with network externalities. The platform spends money developing, testing, and refining, its product, all of which becomes sunk costs. The value of the platform increases with number of users. The platform may even have negative value initially (as shown if the diagram below) if the user-base is too small.



How does a platform increase its user-base? Early adopters may have to be paid or bribed to join. Influencers, or advertising campaigns can be employed to make people think that the platform has more value that it currently does, and so join to check it out. The rate of customer acquisition depends positively on the amount spent to acquire users, the innate value the platform offers (which is often related to development costs, or customer support), and the number of current users. The higher the price that the platform charges it users, however, the smaller the net value to users, and the slower the rate of customer acquisition.

At first, a platform may be willing to lose money to get to a critical mass of users. As more join, the value of the platform increases, and the rate of acquisitions accelerates. This creates a **First-mover Advantage**. The first to market (the incumbent) will have had a chance to acquire a core of users that makes its platform more valuable, all else equal, than anything offered by a latter entrant with zero users.

If the incumbent is able to pull far enough ahead, it can become impossible, or at least too costly, for any entrant to displace them. Thus, markets with strong network externalities tend to have **Winner-take-all**, or **Winner-take-most**, market outcomes.

Once market domination is achieved, the incumbent can turn its attention away from customer acquisition, and towards making a profit. Mature platforms raise prices, change the terms of service to monetize their users, become less responsive to customer desires, offer less support, and so on. Smart platforms also build in features, such making it difficult or impossible for users to move their data to competitors, or encouraging users to build specific capital (such as large numbers of followers, reputations, or connections to other users) in order to create user lock-in.

Several related factors mitigate this result:

- The first-mover may gain market dominance, but also be the creator of a new product category. This creates generalized product awareness and a more fertile market for entrants.
- Consumers are not undifferentiated sheep. The first-mover may have established the product category, but it also had to guess about what the best feature set might be. It is costly to change once critical mass is achieved, and risks alienating existing customers. An entrant can therefore create a somewhat differentiated platform if the incumbent missed the market sweet spot. In turn, this may allow two, or more, different, but related platforms to exist in equilibrium.
- Similarly, tastes may change over time, with new generations of consumers, new technologies, or general social shifts. Again, entrants may be able to swoop in with a better targeted platform, and either coexist, or eclipse, the incumbent.

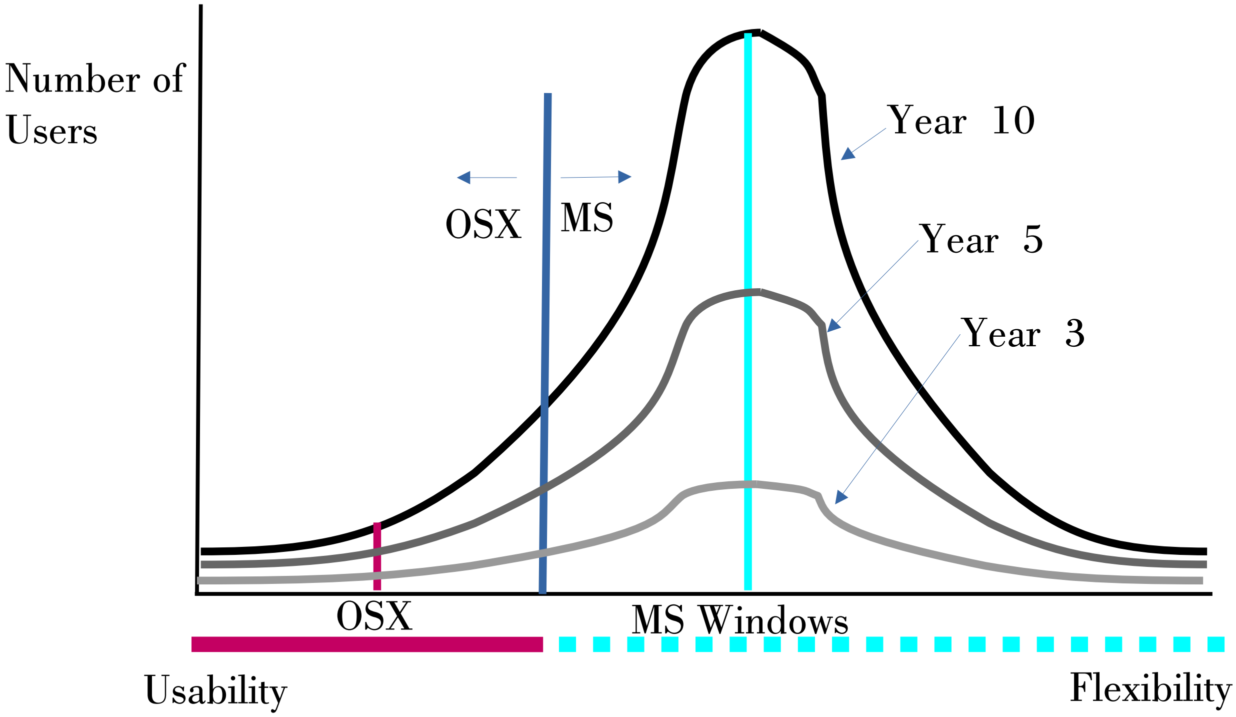
Consider the figure below. Suppose that operating systems must choose between flexibly, and easy of use. Each user has his or her own most preferred compromise between these two qualities. The gray and black curves show the distribution of users with any given ideal point as the product category matures.

In the figure, Microsoft got lucky and chose the feature set preferred by the largest number of users. The number of computer owners grows over time, and so the user preference distribution curve grows taller each year.

What if Apple wanted to enter the OS market? If it chose the same feature set as Microsoft why would anyone switch, given that the current Microsoft user-base creates a network externality? Instead, Apple could choose to offer an OS that skews towards usability. Ignoring network externalities for the moment, anyone with a preference to the left of the blue line would find that Apple comes closer to his ideal OS than Microsoft, and would switch. Those on the right would stay with Microsoft.

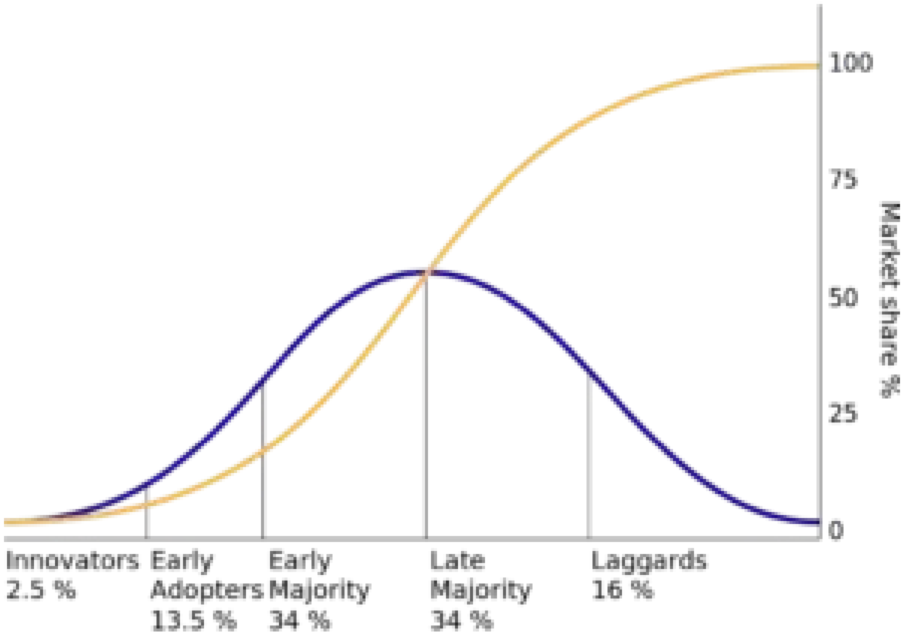
Whether or not Apple could successfully enter the OS market depends in part on the overall market size. In year 3, the number of OS users is smaller, and may not be numerous enough to support an entrant. By year 10, the opposite may be the case.

Finally, what if tastes changed over time? If user preferences drifted towards usability over the years, then the opening for an entrant focused on usability would be that much wider. Microsoft might even end up as a legacy OS that eventually could not attract enough users to make it viable. This is what happened to Blackberry (a Canadian non-smart, mobile-phone maker) and may be happening to Facebook now.



This dynamic can also lead to a **Second Mover Advantage**. The first mover must guess what the best set of product features might be for a technology that no one has ever seen before. It is costly to make dramatic changes once a product is launched, and effort has been invested in cultivating a market for it. A second mover is able to learn from any mis-targeting by the first mover, and also benefits from any marketing effort that makes consumers familiar with the product category.

A related idea is the **Technology Adoption Life Cycle**. The notion is that information is slow to defuse. Even in the absence of competition or differentiated tastes, it can be worthwhile to lose money early on in order to speed growth towards earlier profitability. Amazon famously did not make a profit for the first nine years it was in business. Uber has lost $31 billion since its founding in 2008 but nevertheless has a market cap of $130 billion as of January 2024. Investors are betting that we are still early enough in the adoption cycle that these investments in growing marketshare will ultimately payoff.



Section 2.9. Appendix – A Timeline of Information and Communications Technology

We will discuss these technologies and their implications in more detail in subsequent sections. To get a broad overview of how ICT has developed, however, what follows is a timeline giving some of the most significant developments.

BC

100,000?	Spoken language is first developed.
40,000	Cave paintings begins.
3000	Cuneiform, writing on clay tablets with a stylus, is developed by the Sumerians.
2500	Paper from the papyrus plant is developed by the Egyptians.
2000	An alphabet having only consonants is developed by the Phoenicians.
1500	A phonetic alphabet using Phonetician letters with the addition of vowels is developed by the Greeks.
1200	The earliest Chinese script is developed.
600	Books of papyrus paper are made in Greece to replace papyrus scrolls.
550	A postal service is instituted by Cyrus the Great of Persia for the first time.

AD Before 1800

100	Paper made from rags is developed by the Chinese.
900	Zero, as a numerical concept, is developed in India.
1040	Movable type for printing made from ceramic is invented by Bi Sheng in China.
1234	Mongols set us the “yam” horse-relay postal system with hundreds of stations extending from Eastern Europe to the Pacific Ocean.
1450	The mechanical printing press using movable metal type is invented by Johann Gutenberg, making the production of books radically cheaper.
1604	An adding machine powered by hand is invented by Blaise Pascal.

1800s

1800 Alessandro Volta produces a continuous electrical power from a stack of silver and zinc plates.

1801 A water-powered loom controlled by wooden punch cards for producing complicated patterns in cloth is invented by Joseph-Marie Jacquard.

1822 An analog difference engine using punch cards and designed to mechanically tabulate polynomial functions is invented by Charles Babbage.

1834 – 1836 Thomas Davenport, Johann, Moritz Jacob, Philipp Wagner, Sibrandus Stratingh, and Christopher Becker, and others, build various different types of the electric motor. Industrial adoption was slow, and use of electric motors at scale began in the 1870s and 1880s.

1835 Morse code, a kind of binary system for sending messages, is invented by Samuel Morse.

1836 The telegraph is invented by William Cooke and Charles Wheatstone. Morse code is not used until the mid 1840s.

1839 The Daguerreotype photographic process, the first system to gain wide use, is invented by Louis Daguerre.

1847 The rotary press is patented.

1858 The first transatlantic telegraph cable is laid.

1873 Zénobe Gramm realizes that DC electric motors that convert electricity into motion, could be used in reverse to convert motion into electricity. That is, motors are also generators.

1876 The telephone is invented by Alexander Graham Bell.

1877 The phonograph is invented by Thomas Edison.

1878 Thomas Edison patents the first practical electric light bulb. Edison merged several of his companies in 1889 to form the General Electric Company.

1890 Herman Hollerith of MIT (Massachusetts Institute of Technology) develops an electric machine using punch cards to tabulate the 1890 census. He goes on to found a company that evolves into IBM (International Business Machines).

1892 A hand-powered, printing calculator is developed by William Burroughs.

1894 Radio is invented by Guglielmo Marconi.

1895 The Lumière brothers patented the cinematographe and produce and showed the first projected movie in Paris.

1900-1940

1905 The vacuum tube is invented.

1918 Enigma, an electromechanical coding and decoding machine used extensively in the Second World War by the German military, is invented by Arthur Scherbius.

1925 The first television signal is broadcast by John Baird.

1927 Vannevar Bush at MIT builds a mechanical analog computer using wheels, disks, and gears, to solve differential equations.

1928 Magnetic tape is patented.

1940s

1944 The ASCC (Automatic Sequence Controlled Calculator) Mark I computer weighing five tons is built. This is an electromechanical elaboration of Babbage's difference engine and is used by John von Neumann on the Manhattan Project to develop the first atomic bomb.

1945 Grace Hooper finds a dead moth stuck in one of the relays of the ASCC, which she fixes by "debugging" it.

1946 The ENIAC (Electronic Numerical Integrator and Computer), a digital computer using 17,000 vacuum tubes and punch cards, is built.

1947 The solid state transistor is invented at AT&T (American Telephone and Telegraph) Bell Labs.

1947 The first commercial television broadcast is made.

1948 Claude Shannon at Bell Labs publishes "A Mathematical Theory of Communication" founding the field of Information theory.

1948 The Manchester Mark I is constructed. This is the first stored-program computer. Previously, computers were programmed with patch cables.

1948 The IBM SSEC (Selective Sequence Electronic Calculator) is constructed using both vacuum tubes and mechanical relays.

1950s

1951 The UNIVAC I (UNIVersal Automatic Computer I) is first produced. This is the second, but most widely used, commercial computer at the time. It had 5200 vacuum tubes and weighed 13 tons. A total of 46 were built between 1951 and 1954.

1952 The IBM 701, IBM's first commercial scientific computer is produced. A total of 19 were built.

1957 Sputnik, the first artificial earth satellite, is launched by the USSR (Union of Soviet Socialist Republics) starting the space race.

1957 The IBM 350 disk storage unit, the first hard disk holding 3.75 MB (MegaBytes) of data, is released.

1958 ARPA (Advanced Research Projects Agency) is created. It became DARPA in 1972, ARPA again in 1993 and DARPA yet again in 1996 (the D stands for Defense).

1958 The IC (Integrated Circuit) is invented at TI (Texas Instruments).

1959 The electrostatic copy machine is introduced by Xerox. Previously, carbon covered paper was layered between sheets of typing paper which allowed the production of up to eight copies of a document at once if a typist hit the typewriter keys with enough force. This is the origin of "CC" or "carbon copy" used in email headers.

1960s

1961 The IBM Selectric Typewriter is released.

1962 SpaceWar, the first interactive computer game, is written by MIT students. The output is displayed on a screen and the action controlled with an early form of the joystick.

1963 The compact audio cassette (the cassette tape) is released by Philips.

1966 DRAM (Dynamic Random Access Memory) is invented.

1964 The IBM System/360, a family of commercial computers using transistors and a primitive type of integrated circuit, is introduced.

1969 The Internet is invented. The first packet data is exchanged between Stanford and UCLA over APRANET.

1969 Unix, a flexible and adaptable operating systems for computers, is created at AT&T Bell Labs.

1970s

1970 The resistive touch screen is invented by Samuel Hurst.

1971 The first email message is sent over ARPANET, which had 15 nodes and 23 hosts at the time.

1971 The first pocket calculator, the Texas Instruments TI-58 weighing 2.5 pounds, is released.

1971 The Intel 4004, the first 4-bit IC chip CPU (Central Processing Unit), is released.

1971 The 8" floppy disk is released.

1972 The mouse (an input device) is invented at SRI (Stanford Research Institute).

1972 The Intel 8008, the first 8-bit IC chip CPU, is released.

1972 The INWG (Internet Working Group) created to address the need to establish agreed upon protocols.

1973 The Ethernet standard and FTP (File Transfer Protocol) are specified.

1974 TCP (Transmission Control Protocol) is specified.

1974 Telenet, a commercial version of ARPANET, is launched.

1975 The first laptop, the IBM SCAMP, is released.

1975 Microsoft is launched with an implementation of the BASIC programming language as its first product.

1976 The Apple I is released. A total of 200 are built.

1976 The 5¼" floppy disk is released.

1977 The Apple II is released. This version did not have a GUI (Graphical User Interface).

1977 The PC modem is released.

1972 The Intel 8086, the first 16-bit IC chip CPU, is released.

1978 The first email spam message is sent to 600 California ARPANET users by Gary Thuerk.

1979 The Atari 8-bit home computer is released.

1979 USENET (News Groups) is established using UUCP (Unix-to-Unix Copy Program).

1979 The first MUD (Multiuser Dungeon) is released.

1980s

1980 BELLMAC-32A, the first 32-bit CPU is released by AT&T's Bell Labs (The team was headed by Hing So, my father-in-law.) It had 150,000 transistors, used a 35 micron die, and ran at 8MHz. TSMC is now able to make chips with 3 nanometer dies (10,000 times smaller) with as many as 134 billion transistors (1,000,000 times more), that can run at 8GHz (1000 times as fast).

1980 The CD (Compact Disc) audio is released.

1981 The 3½" floppy disk is released.

1981 DOS (Disk Operating System) 1.0 is released by Microsoft.

1981 The IBM PC (Personal Computer) is released.

1981 BITNET, (Because It's Time NETwork) is started at the City University of New York with the first connection going to Yale.

1983 MS Windows 1.0 is released. This is Microsoft's first effort to produce a GUI for its DOS operating system.

1983 TCP/IP (Transmission Control Protocol/Internet Protocol) becomes the official protocol for the Internet. This made it possible to connect networks of dissimilar computers together. Thus, the Internet as a "Network of Networks" is born.

1984 The CD-ROM (Compact Disk Read-Only Memory) allowing programs and other data to be read from CDs is released.

1984 The Apple Macintosh is released and included a GUI.

1984 The DNS (Domain Name System) is established.

1984 The term "Cyberspace" is coined by writer William Gibson.

1985 Quantum Computer Services, later to become AOL (America Online) is launched,

1985 The 5¼-inch HDD (Hard Disk Drive), with capacities of 20, 30, and 44 MB, is released by IBM

1987 The Internet grows to 30,000 hosts.

1988 The first computer virus is created by Robert Morris.

1989 "Proposal for the World Wide Web" is published in the magazine, MacWorld, by Tim Berners-Lee.

1989 The number of hosts breaks 100,000.

1990s

1990 The USB (Universal Serial Bus) flash drive is commercially released.

1991 The World-Wide Web is implemented by Tim Berners-Lee at CERN (Conseil Européen pour la Recherché Nucléaire) using HTTP.

1991 The first web page is created.

1991 The first webcam is deployed at a Cambridge University computer lab with the sole purpose of monitoring a coffee maker so that lab users could avoid wasted trips to an empty coffee pot.

1991 AOL email for DOS is released.

1992 The Linux kernel based on Minix (a Unix derivative) is released by Linus Torvalds.

1992 The Linux kernel is combined with Richard Stallman's GNU (Gnu is Not Unix) project and released as a complete free and open source operating system.

1992 Windows 3.1 is released by Microsoft.

1992 Delphi, the first national email and Internet provider, is launched.

1993 AOL email for Windows 3.1 is released.

1993 Mosaic, the first Internet browser, is developed by Marc Andreessen of the NCSA (National Center for Supercomputing Applications) at UIUC (University of Illinois at Urbana–Champaign).

1994 Netscape Navigator, a commercial implementation of the Mosaic browser, is released by Marc Andreessen and Jim Clark through Netscape Communications.

1994 Yahoo! is launched.

1994 SSL (Secure Sockets Layer) encryption protocol is developed by Netscape, making online credit card payments secure. TLS (Transport Layer Security), fixed security flaws and replaced SSL in 1999.

1995 The DVD (Digital Versatile Disc) is released.

1995 CompuServe, AOL, and Prodigy start providing dial-up Internet access.

1995 The Java Internet programming language released by Sun Microsystems.

1995 Echo Bay, latter to become eBay, is launched.

1995 Amazon is launched.

1996	The Browser Wars between Netscape Navigator and Microsoft Internet Explorer begins.
1996	Hotmail, the first webmail service, is launched.
1997	Netflix is launched.
1998	Google is launched.
1998	MSN (Microsoft Network) Search is launched (latter to become Bing in 2008).
1999	Napster is launched.
1999	Salesforce.com enterprise application software in the cloud is launched.
1999	The SETI@home grid computing project is launched. SETI stands for the Search for Extraterrestrial Intelligence and analyzes deep space radio emissions in hopes of finding patterns suggesting something other than a natural source.
1999	The SD card (Secure Digital) flash memory card is released.

2000s

2000	The Internet stock market bubble bursts.
2001	Wikipedia is launched.
2002	Amazon Web Services is launched.
2002	Napster goes bankrupt due to court rulings.
2003	The Blu-ray disc is released.
2003	Skype is launched.
2003	MySpace is launched.
2003	The iTunes Music Store is launched.
2004	HD-DVD (High-Density Digital Versatile Disc) is released.
2004	An Internet worm called MyDoom or Novarg infects about 1 in 12 email messages.
2004	World of Warcraft is launched.
2004	Facebook is launched.
2004	Web 2.0 begins. Gaming, blogging, social networking, and similar platform, allow users to interact with webpages, instead of just passively looking at their content.

2005 YouTube is launched.

2005 AMD (Advanced Micro Devices) Athlon X2, the first native dual-core processor is released.

2006 Twitter is launched.

2006 Amazon S3 (Simple Storage Solutions) is launched. This offered data storage and access as a web service.

2006 Amazon EC2 Beta (Elastic Compute Cloud) is launched. This provided a commercial web service that allowed small companies and individuals to rent computers on which to run their own computer applications. EC2 left beta status in 2008.

2007 The iPhone is released marking the start of the mobile web.

2007 Hulu is launched.

2007 Android OS, a derivative of Linux specialized to touch-screens and mobile devices, is released.

2008 Airbnb is launched.

2008 Google App Engine cloud service is launched.

2009 Chrome OS, a derivative off Linux specialized to cloud services, is released by Google.

2009 Uber is launched.

2009 Windows Azure cloud service is launched.

2009 Kickstarter is launched.

2008 "Satoshi Nakamoto" (a pseudonym) writes "Bitcoin: A Peer-to-Peer Electronic Cash System" which outlined a Proof of Work" and Merkle Tree protocol for blockchain validation and immutability

2009 Bitcoin is launched using an open-source code base. By April 2010, bitcoin had a value of $.003.

2010 OpenStack, an open-source cloud-software initiative, is launched by Rackspace and NASA

2010 The iPad is first released.

2010 Ripple is launched, a private blockchain used for international settlements between banks.

2011 Zoom Launched

2012 Lyft is launched.

2012 The Google Fiber Project is launched to compete with incumbent broadband providers

2012 CRISPR gene-editing tool discovered independently by researchers from Harvard University, the University of California at Berkeley, and the Broad Institute.

2014 The Darkweb TOR site “Silk Road” is shut down by the FBI and 26,000 Bitcoins are seized (worth about $150 each).

2014 Vitalik Buterin releases the Ethereum white paper outlining a new approach to cryptocurrencies which included “smart contracts”.

2014 Mt. Gox, a Bitcoin exchange has 744,000 bitcoins stolen by hackers.

2014 Amazon Echo, the first smart speaker is launched.

2015 The Ethereum Blockchain is launched. The ERC20 token standard which is based on an Ethereum smart contract allows thousands of Blockchain startups to emerge with their own cryptotokens.

2015 Microsoft starts to accept Bitcoins as payment for its services.

2015 Amazon Echo released.

2017 D-Wave Systems Inc. announces the D-Wave 2000Q quantum annealer, with 2000 qubits.

2022 OpenAI and launches ChatGPT.

Note that this timeline gets thinner for more recent years. This is because we just don’t know what will really turn out to be important in the future yet. If I did, I would already have invested in it and would not be writing this book.

Chapter 3. Information Creation, Marketing, and Protection

This chapter gives an outline of some of the institutions relating to the creation, selling, and protection of intellectual property. Special attention will be paid to the law and the formal and informal institutions involved in creating new things and ideas. This will serve as a basis for a discussion of the many associated economic and policy questions to take place in subsequent chapters.

Section 3.1. Content and Preservation

Analog content comes in many forms. In the pre-Internet days, books, newspapers, official documents, and so on, were easily lost to fire or destroyed by natural or man-made disasters. Public and private records on paper take up a great deal of storage space and are often poorly indexed and difficult to access. Many such records have been lost or discarded over the years, and the cost of digitizing what remains is prohibitive.

Much of early television and radio was broadcast live. Recording technologies were expensive and not very durable. Only a small fraction of these broadcasts is still available. Wax and vinyl sound recording are easily damaged, and so what remains is often of poor quality. Films and photographs fade, become brittle, and are highly flammable. Audio tapes degrade with time and take up a great deal of space.

These problems may have been even worse in the early parts of digital age. It is difficult enough for librarians and archivists to find and preserve traditional ephemera, such as hand-bills, fliers, posters, tickets, and similarity items intended to be of only immediate interest. Digital ephemera such as text messages, Facebook posts, tweets, ads, spam and so on, might be of great interest to future historians, but are almost impossible to save.

The Internet is a very dynamic place. Webpages are posted, edited, and taken down, continuously. It would be impractical to take a snapshot of the Internet every day, and much of what appears on the Internet quickly disappears forever. There is a project called the Wayback Machine that does archive some of the historical Internet (what Google looked like in 2006, for example), but the task is too big to be done comprehensively.

Digital content is produced in a wide variety of formats which become obsolete and unreadable. WordStar documents and Lotus 1-2-3 spreadsheets are in formats that have not been used for since 2000, and the software to read them is no longer widely available. Programs written for DOS, CP/M, or Windows 95, are either lost, or soon will be, from a functional standpoint. Preserving digital documents requires forward migrating these files to newer formats as technology changes. There are too many files, in too many places, to do this in a comprehensive or systematic way.

An even more difficult problem is that early digital content was often stored on obsolete media, such as reel-to-reel data tapes, DAT cassettes, 8" and 5 ½" floppy disks, and zip disks. Hardware to read these media is no longer manufactured. When the existing machines break, the information these media contain will be lost forever. It is interesting to note that paper documents from 2000 years ago can still be read if they survived. Much of our digital data may not survive even 20 years.

There are, however, a number of efforts to digitize and preserve physical works. Google has an ongoing project to scan every book it can get its hands on. Project Gutenberg began in 1971 with the objective of convert books in the public domain to searchable text and to present the result if an appealing and easily updateable format.

We can hope that the availability of the cloud may allow us to turn this around. Properly archived digital, and digitized physical content, can be preserved in the cloud at low cost, and migrated to usable formats as time moves forward. A huge amount of early digital and historical analog content has been lost forever. but the cloud should make it much easier to save, and also make available, what remains.

Section 3.2. Intellectual Property and Creative Works

Creative Works are the fruits of creative efforts of all kinds on the part of individuals, companies and other groups. These works invariably build on the cultural, artistic, and scientific, knowledge-base of the society in which they arise. Governments have a clear interest in allowing the greatest possible access to this shared cultural inheritance which is often refereed to as the **Creative Commons**.

Although writers, artists, and inventors would not be able to carry out their work without the creative commons, their efforts activities have an opportunity cost. If creators could not profit from their work somehow, less creative effort would be expended. Writers have to eat too.

The solution is to convert creative works into **Intellectual Property** owned exclusively by the creator for a set period of time through patents, copyrights, and to a lesser extent, trademarks. All three of these institutions amount to monopolies established and enforced by law in order to give creators incentives and compensation for their efforts.

The US Constitution grants the federal government the following authority:

> *To promote the progress of science and useful arts, by securing for limited times to authors and inventors the exclusive right to their respective writings and discoveries.*

Note that the goal is to add to society's shared stock of art and science, and in exchange for their contributions, authors, and inventors are granted a limited monopoly. It is explicitly not the case that the founders believed that an inventor or writer had a moral right to own his creations. It was recognized that the creator owed a debt to the creative commons. The grant of monopoly was therefore limited so that the commons would be enriched and renewed when it lapsed.

Section 3.3. Patents

At the nation's founding patents lasted for a term of 14 years. Currently, there are three forms of patents: **Utility Patents,** which last 14 years from the date they are granted, and **Design Patents** and **Plant Patents**, both of which last for 20 years from the date they are filed

Utility Patents: Patents for processes, machines, articles of manufacture, compositions of matter, and improvements to any of these.

Design Patents: Patents for ornamental design of an article of manufacture.

Plant Patents: Patents for asexually reproduced plant varieties.

For anything to be patentable, it must be *novel*, *useful*, *nonobvious*, and *clearly described* in the patent application. In particular, it must not be **Prior Art** in the sense that it falls into the set of common ideas and techniques that are generally known by those "practiced in the art". Laws of nature, physical phenomena, abstract ideas, and morally offensive inventions, are never patentable.

Software and business methods are patentable to a limited degree. The chief problem is that abstract ideas are not patentable, and so the code or method to be patented must be fairly specific. In addition, for anything to be patentable, it must have a tangible form. This can be a bit tricky with code or methods.

Medical procedures fall into a sort of no-man's zone. They are patentable, however, major medical organizations such as the AMA consider it unethical to do so. In addition, doctors cannot be sued for using a patented procedure on a patient, even if he does not have permission from, and has not made payment to, the patent holder.

Patents are granted only by national governments, and each nation has its own rules and standards. There is no such thing as an international patent. Filing a patent application is expensive, about $10,000 to $15,000 including fees and legal costs. This adds up quickly if an inventor wants to patent his device in several jurisdictions. It usually takes between one and three years to receive a decision from the United States Patent and Trademark Office.

Subsection 3.3.1. Copyrights

At the nation's founding, copyrights also lasted for 14 years, but could be renewed once by a living author for another 14 years.

Copyright terms now are much longer, and have complicated legislative history:

- Anything published before 1928 is in the **Public Domain** and can be freely used without permission from, or compensation to, the creator. In other words, it is part of the creative commons.
- For most works created after works created after 1978, copyright lasts 70 years after the death of author, or if a work of corporate authorship, 95 years from publication or 120 years from creation, whichever expires first.

Most copyrighted works generate no revenue for their creators. A small number produce revenues immediately after the work is created, which quickly diminished to little or nothing for the reminder of the copyright term. A vanishingly small number of creative works continue to produce substantial revenue for the life of their copyright term. See the discussion of **Present Value**, below.

Copyrights are meant to protect original works of authorship which are expressed in fixed and tangible form. This includes literary, musical, dramatic, choreographic, pictorial, graphic, sculptural, architectural, audiovisual, and audio works. Titles, names, short phrases, slogans, familiar symbols or designs, and works consisting entirely of information that is common property and containing no original authorship (a tape measure or metric to standard conversion table, for example), are not eligible for copyright protection.

Copyright protection does not cover ideas, procedures, recipes, methods, systems, processes, concepts, principles, discoveries, or devices themselves, but only their specific description, explanation, illustration, or expression.

Thus, “King John rules the world” could not be copyrighted since it is a slogan or short phrase. If the slogan were to be printed on a T-shirt with an original illustration, the T-shirt itself could be copyrighted as a tangible, specific expression. The idea that King John should rule the world, and any diabolical plan to make this happen, could not be copyrighted, nor could an unrecorded speech or interpretive dance on the topic. However, a book, sound recording, or video of any of these things could be copyrighted.

The broader idea here is that it is the specific **Expression** of a creative work that is protected. The **Look and Feel**, and intangible elements, of a creative work are not copyrightable.

Authors are not required to register or take any other action to secure their copyrights in the US. The Berne Convention for the Protection of Literary and Artistic Works of 1886 is a treaty signed by 181 states out of the 195 countries in the world, and sets out three basic rules for intentional copyrights:

- The creator of the work does not have to register it with any national or international authority to gain protection.
- The creator must place as notice of his claim the circle C, the date that the work was produced, and the name of the author or owner of the work, in a visible place on the work to be protected:

  ©2015 *Jeremiah Rightsholder* .

- Signatory countries must then give copyright protections that meet a minimum standard. The most import of these is that copyright extends to 50 years after the author's death in most cases.

A significant limitation on copyrights is called the **Fair Use Doctrine**. This allows parts of copyrighted works to be quoted verbatim for purposes such as criticism, news reporting, teaching, and research, without permission from, or payment to, the copyright holder. There are four factors that determine what constitutes fair use:

- **The Purpose and Character of Use:** If the use is transformative in the sense that the work is given a new and original expression, then it is more likely to be considered fair. In addition, if the use is for a purpose that is different from that of the creator (criticism or satire, for example), then the use is more likely to be considered fair.
- **The Nature of the Copyrighted Work**: It is more permissible to use published works than unpublished works, and to use works with factual, rather than fictional, content.
- **The Amount of Material Used:** The more of a work that is used, and the more central the part that is used is to work as a whole, the less likely the use is to be considered fair.
- **The Effect on the Potential Market:** The more likely the use of a work will affect the author's income, the less likely the use is to be considered fair.

Subsection 3.3.2. Trade Secrets and Trademarks

Trade Secrets are another type of intellectual property with different protections than patents. They cover formulas, practices, processes, designs, instruments, patterns, that may not be patentable, but have economic value as long as their use is under the control of the inventor.

It is illegal to steal trade secrets, but if you figure them out yourself, or come across them in some legal way, you are entitled to use them. The most famous example of a trade secret is the recipe for Coca-Cola. Patenting it would make it public and easy to copy or approximate. Patents also expire after a period. Keeping the formula secret allows Coke to profit from its recipe indefinitely.

Trademarks consist of words, short phrases, or symbols meant to represent a company, brand, or product. They must be registered with the government, cost between $250 and $750, and must be renewed every ten years. There is no limit on the number of renewals.

The grant to use a trademark is limited in a number of ways. The trademark must be in *continuous use* and only applies in whatever *region* and *product-line* the company operates. Thus, if I start Peter's Pizza Place in Pittsburgh, I cannot sue for trademark infringement if someone else starts Peter's Pizza Place in Peoria. I would also be unable to sue if someone started a Post-Industrial Speedcore Eurobeat collective called Peter's Pizza Place.

The motivation for trademarks is to give companies a credible way to convey the nature of the product they are providing to consumers. When I buy a Coke, I know what I am getting. I am not concerned that it might be tainted or adulterated. This is because the makers of Coke profit if they produce a consistent and safe product. If the company did not maintain quality control, I would not seek Coke out as a preferred beverage since I would not know how my next can might taste, or if it might poison me.

If Coke could not prevent others from selling under their trademark, I could never know if the product I was buying was genuine. Thus, there would be no reason for Coke to make a consistent, high-quality product when consumers could have no idea what they were getting (and might be willing to pay a premium for). Thus, trademarks are granted to help provide more complete information to the market, and thereby reduce market inefficiency and failure.

Subsection 3.3.3. Institutions that Protect Intellectual Property

Designing and operating an intellectual property system to maximize society's interests is difficult. It requires balancing the negative of granting monopoly ownership of a creative work or invention, with the positive of encouraging innovation. Two major parameters that must be calibrated correctly are the following.

Term Length: Too long, and you reduce the flow of fresh ideas to the creative commons. Too short, and you reduce the creation fresh ideas by under-compensating creators.

One size does not fit all. It takes a lot more potential profit to get George Lucas to produce Star Wars, than it does to get a lonely teenager to write a poem. Some innovations are easy and would be produced even if the rewards were small. Other innovations are very difficult. It may be that certain areas of research are so speculative that they would not be worth the cost of investigating even with 50 years of patent protection.

Breadth: Too broad, and you close off large areas of scientific, technolgical, artistic, or commercial exploration to other innovators. Too narrow, and you create the potential for workarounds inspired by the patent, which devalues the patent-holder's work, and discourages innovation.

Suppose I invented the blow-dryer. Should I get a patent on all hot air drying technologies, or just the specific blue plastic blow-drying machine I developed?

The former would give me control of everything from clothes dryers, to technologies used in cement manufacturing. My royalties would be a burden that discourages the use of such devices, and causes people to spend effort looking for second-best solutions.

The latter would allow a competitor could get around my patent by producing essentially the same machine in green plastic. My competitor would get all the benefit of my investment in research and development without having to pay anything for it. Knowing that I will face price competition if I invent and start producing blow dryers, I may find that innovating is not worth the effort. Again, one size does fit all in this regard.

Patent Races: A situation where two (or more) firms pursue the invention of the same the new product or process. The first to succeed gets the patent, and all the rewards. Patent races result in redundant, and inefficiently accelerated, R&D investments. Not only does this cost money, but the opportunity cost to society is that researchers are engaged is duplicative efforts and so cannot invent other things. Patent races are more likely when patents are overly broad or long lasting.

Two other real-world problems with intellectual property protect are not as easily calibrated:.

Quality of Examiners, and Consistency of Process: It is difficult for patent examiners and judges involved in intellectual property cases to determine what constitutes "prior art", what is "useful" and what is "nonobvious". This is especially true in highly technical areas such as genetics, electronics, computer programing, and material science. This lack of expertise can result in patents being granted for things that are well-known, or straight-forward, to people in the field. Not only does the patent system fail to support the creation of new knowledge in such cases, it also locks-up old knowledge in a monopoly for many years and makes it difficult for others to continue to work on related ideas.

On the other hand, if patent examiners fail to understand when an invention is truly innovative, and instead, falsely sees it as a minor variation on prior art, further innovation can be discouraged. The outcome may depend on who happens to examine the patent application. A lack of predictability in the patent process is especially discouraging to small companies whose business plans may live or die on a single such decision.

Preventing Corruption and Undue Influence: Congress and the executive branch make the rules, and choose the leadership for the patent office. Convincing the government to alter policies, or to appoint sympathetic management and policy setters to the patent office, is very important to large companies who make money from IP. How broadly a given patent extends, and deciding to allow or disallow patent protection for certain types of computer code, business methods, or genetic innovation, can be worth billions of dollars to Apple, Google, Samsung, Microsoft, and similar companies. Some of this money will surely find its way into the campaign chests of helpful politicians.

At a lower level, patent examiners, judges, and agency managers may feel pressure from above to grant dubious patents, or be promised jobs when they leave government service if they are cooperative.

In it extreme forms, this process can lead to **Regulatory Capture**. As an example, movie studios, music labels, and other content providers, pushed Congress to pass both the Sonny Bono **CETA (Copyright Term Extension Act)** and the **DMCA (Digital Millennium Copyright Act)** which extended copyright terms and increased penalties for copyright infringement. It also made it a serious crime to try to subvert, break, or bypass, any **Digital Rights Management (DRM)** technology.

John Deer, Apple, and others, have tried to use this last provision of the DMCA to extend their control to the physical products they sell. Tractors, iPhones, and most technolgical goods today, come loaded with software and firmware that is necessary to make them work. The companies argue that they therefore have the right to prevent customers from repairing their own property. Instead, they require that only licensed dealers be used, and use DRM protected software to enforce this.

This argument can be extended to control how your use your property, and allow companies to brick their products if they decide you have violated their terms of service. This is in direct conflict with the first sale doctrine, discussed in the latter in this chapter. Fortunately, various state legislatures are beginning to respond with “right to repair” laws.

Regulatory Capture: The co-opting of a regulatory agency or policy maker who has a duty to protect the public's interest by narrower commercial, ideological, or political interests. Especially, regulators making rules or policy that benefit the regulated, rather than the public. Capture can result from:

- direct bribery
- political contributions to those who appoint or control the regulators
- "revolving doors" where regulators or their friends expect to be employed in the future by the firms they regulate today
- production and control by the regulated of the expert advice available to the regulator
- a lack of countervailing pressure from members of the public who may not even know what sorts or decisions are being made, and how they might impact them personally

In general, the regulated have concentrated interests, while interests of the public at large are diffuse. As a result, firms typically put a great deal more effort into shaping regulation than the average citizen, who simply free-rides, and hopes for the best.

Section 3.4. Open and Closed Source

Essentially all software, including operating systems, is written in high-level programming languages such as "C".

High-level Languages: Programing languages designed to abstract from the underlying operations of the computer and allow programmers to express their wishes in more direct, natural, and human-centered, ways.

Low-level Languages: Programing languages such as assembly language that use basic microprocessor commands, and do not abstract from the mechanics of the system. These languages are difficult to use since they require the programmer to express his wishes in a detailed, and unnatural, way, at least from a human standpoint. The benefit of code written in a low-level language is that it runs more efficiently than code written using a high-level language.

Regardless of the type of language used, what the programmer writes is called **Source Code**. High-level code is easier to read than low-level code, but both can be understood and interpreted by a competent programmer.

Once the source code of a program is complete, it is **Compiled** and ends up as a **Binary File** in what is called **Machine Code**, which can be only be understood and used by the CPU.

The process of compiling source into machine code involves removing white space, comments, names of object classes, rewriting the structure to remove repetitive or unnecessarily verbose parts of the source code, reorienting the code to be linear rather than modular, and in general, optimizing and compacting the program.

The objective is to generate a binary file that is small and can be run as efficiently as possible. As a result, a great deal of information contained in the source code is omitted, transformed, and compressed, and is therefore unrecoverable. In other words, it is impossible to “decompile” a binary file and get back to the underlying source code.

If programmers choose to release the source code to the public, the project is said to be **Open Source.** If programmers chose to release only the binary file, and keep the source code secret, the project is to be **Closed Source**.

The main reason to open source a project it allows users to know exactly what a program does. If a program is designed to spy on your browser history, use your computer to send or relay spam, run a background programs for its own, possibly commercial, reasons, the evidence will be clear in the source code.

Implementations of blockchain protocols, libraries that instantiate encryption algorithms, programs to run medical devices and critical infrastructure, smart contracts, and many operating systems are all open source. Making them closed source would mean that any potential user would simply have to trust that the programmer is correctly representing what his code does. Honest programmers and projects should have no reason to hide behind a binary. Moreover, no sensible person would use a blockchain, or an encryption library whose functions had not been independently verified.

On the other hand, most commercial software and operating systems are closed source. No one really knows what Windows does, or what it might do at the prompting of Microsoft. Windows 11 is estimated to have up to 100 million lines of code, much of it from legacy versions of Windows. Even Microsoft does not know exactly what Windows does. We do know it sends thousands of messages to Microsoft an hour containing telemetry and user data, but we don’t know exactly what is sent, or how it is used. You just have to trust that Microsoft is on the side of the angels.

Other reasons to go open source include:

Collaboration: Open source makes voluntary collaborations easy. Closed source requires that someone with access to the source code chose to make available the relevant parts to developers.

Reputation: Open source allows everyone to see each developer’s contributions to the code. Writing an elegant piece of code adds to a developer’s reputation, and gives him an incentive to contribute. Developer contributions to closed source projects are only seen by the higher ups that assigned them. This may lead to bonus or recognition, but a developer will never be able to use them to prove his quality to the outside world.

Linus's Law: "*Given enough eyeballs, all bugs are shallow.*" Linus Torvalds, the original developer of Linux, argues that this is one of the main strengths of open source software. Users must depend on the companies that produce closed source software to catch and fix bugs, and those companies lose the potential help of thousands of users that might have pointed them out.

The main reason to close source a project is that is give the developers much tighter commercial control over their product.

If only the compiled code is released then:

- No one can copy or imitate any clever programing approaches or subsystems contained in a given piece of software. Any trade secrets are protected.
- No one can make derivative products or customizations. Customer must either take the standard binary or pay the developer to modify the source code and provide a new binary.
- The developer can control how many, and what kinds, of versions of a program are offered. This facilitates price discrimination and increases profits.[5]
- It prevents users from finding out that a program may be accessing systems, capturing data, or using resources, is undisclosed ways, and contrary to the interests of the user. It also covers up any bugs or security exploits that the developer happened to have missed.

Section 3.5. The Digital Market Place

Subsection 3.5.1. Physical Sales

Copyright law has long included something called the **First Sale Doctrine** which says that if a copyrighted product is sold by a rights-holder, then ownership of the copy passes to the purchaser and seller's rights in the copy are **Exhausted**.

In practice, this means that if I legally buy a book, CD, DVD, Ninja Turtles lunchbox, or Chicago Cubs hoodie, I can lend it, rent it, sell it, or give it to whom ever I wish, and can even alter or destroy it if I choose. On the other hand, I cannot make copies to sell, nor can I use it for certain kinds of public performances. I can put it on display, however. The first sale doctrine is the legal foundation for the existence of libraries, used book stores, video rental stores, and museums.

Subsection 3.5.2. Digital Sales

[5] See Chapter 7 for a more detailed discussion of price discrimination.

At least since Bill Gates' 1976 open letter to hobbyists asking them not to use unauthorized copies of his BASIC compiler, commercial publishers of electronic goods have experimented with many strategies to prevent what is called **Pirating**.

Photocopying books, and making copies of analog tapes and vinyl records, are violations of copyrights. However, these violations are in a sense, self-limiting. Copies are not the same as the original, and each successive copy is lower quality than the original. In addition, it is not free to copy 400 pages, or obtain a blank tape. Copying is still illegal, but it was seen as a secondary issue.

Digital goods, such as software and media files, are just long binary strings. They can be copied without any loss or degradation, and essentially for free.

When download an app or MP3 from a seller, the seller still has the same file he started with. Thus, it is not as clear that the seller has necessarily sold you anything. The same thing is true when you give a friend a copy of an MP3 you purchased. In contrast, if you sell or give away a book or CD, you no longer have the original, and so nothing has been copied.

As a result of all this, courts have ruled that the “sale” of a digital object is not truly as sale, and the first sale doctrine does not apply. Instead, the “sale” of a digital good is actually a conveyance of a license to use the object on whatever terms are mutually agreed upon.

Subsection 3.5.3. Licensing

Almost all commercial digital goods are licensed to users under conditions contained in what are called **EULAs (End-User License Agreements)**. There are few restrictions on the conditions that sellers can put into an EULA, but most include versions of the following:

- **No Unlicensed Use:** Restrictions on the types and manner of use of the product. For example, the license may be for one year, for one machine, for one user, for several users, or for one user at a time. It may prohibit commercial use, or use on any platform not approved by the rights-holder.
- **No Warranty:** A statement that the software is sold without warranty or guarantee of suitability for any specific purpose. Suppose you bought a word processing program intending to write a letter, but you found that the program had a bug and would not run, would only use Greek letters, or decided to erase your hard drive the first time you used it. Having accepted the EULA, you would have no basis for any legal action.
- **No Copying:** Prohibitions against making copies, and using the content for public performance.
- **No Transfer:** Prohibitions against lending, giving, renting, or selling your copy.
- **No Modification:** Prohibitions against reverse engineering the software, and disabling or evading any copy protection or DRM features.

- **No Liability:** Limitations on, and apportioning of, liability between the buyer and seller, and various indemnity clauses.
- **Not a Sale:** A statement that the copyright holder has not sold the digital object to the buyer at all, but has only conveyed a limited license under the conditions contained in the EULA.

Buyers signify their acceptance of the EULA by clicking a box stating they agree as part of the installation process, or by breaking shrink wrap, tape, or seals as they open the package containing the software, CD, DVD, etc.

"Purchasing" an MP3 or other digital object really means that the EULA in some sense gives you some form of perpetual rights to use the object, although it may impose restrictive conditions in other dimensions.

Free and Open Source Software (FOSS), and well as a great many digital creative works, are distributed are also distributed under licenses. These fall into two broad categories.

Permissive Licenses: Licenses that give users essentially unconstrained permission to use, study, and modify the software or creative work. Most importantly, permissive licenses allow programmers to use the software or content in new projects even if they are commercial, closed source, and distributed under proprietary licenses. Examples of permissive free software licenses include the BSD, PHP and MIT licenses.

Copyleft Licenses: Licenses that give users permission to use, study, and modify, the software and content. However, they impose conditions that require users to publish their modifications in open source under the same license as the original. An example of a copyleft free software license is the **GNU GPL (General Public License).**

The motivation behind GPL is to maximize the creative commons, to prevent new ideas from being locked behind patent or copyright barriers, and to prevent commercial motivations from hindering intellectual and technological advance.

Unfortunately, it also effectively makes covered content and software off-limits to commercial users. This reduces the size of the user-base, and makes learning to use GPL software of less commercial value to developer. Permissive licenses sacrifice possible additions to creative comments in exchange commercial relevance, and a wider user-base.

Subsection 3.5.4. Business Models for Digital Goods

Early computers did not really have software. They were programmed by hand using patch cords. It was not until the first commercial minicomputers in the mid-1960s that electronic storage of programs became practical. Computers were still rare, and both operating systems and software applications were generally bundled and sold together with the physical machine. This created some legal issues for computer makers since this kind of **Bundling Strategy** could be construed as an illegal **Tying** of the software to the hardware product.

Bundling: A marketing strategy for firms in which specific amounts of one or more goods are sold in a package at a fixed price. This contrasts with the more conventional **per unit** pricing which allows consumers to choose how much of each good to buy at a stated price per unit. **Versioning** and quantity discounts are variations of bundling strategies, which in turn are a form of **Second Degree Price Discrimination.**

Tying: Product tying is the practice of making the purchase of one product a mandatory condition of purchasing a second product. Tying can be contractual such as requiring that all service or part be purchased from the manufacturer, or de facto, such as when the original manufacturer has a monopoly over parts or consumables needed for a product to operate.

Before the 1980s, most users worked at universities, research arms of industry and government, or were hobbyists. As a result, software was seen as something to be studied and shared. Most software was distributed for free.

Freeware and **Shareware** were popular in the late 1990s and early 2000s. These programs were generally simple and specialized (we might call them “apps” today). They were usually written by one person and the source code was typically closed. Even if the source code was open, the programmer who wrote the software retained the copyrights. Linux, Apache, LibreOffice, and other FOSS projects, are modern examples freeware.

Freeware was simply given away, probably because the writer did not think the market was big enough to make commercialization worthwhile. Shareware programmers, on the other hand, requested that users voluntarily send them a small fee (usually $5 or $10) if they found the program to be of value. This might even be a condition of the license, but users were on their honor to comply.

Many academic groups and societies have started their own open access journals, archives, and other places to publish. **Open Access** content is open for use, and can be accessed without charge. The motivation is a commitment to the idea that knowledge, especially scientific knowledge, should be free. Examples include the journals published by **PLOS (the Public Library of Science),** arXiv.org, and the *Economics Bulletin*. These free alternatives to commercial offerings are only possible because of the Internet and other recent advances in ICT.

Supporting platforms though **Advertising** is a model that goes back to newspapers and fliers. Most over-the-air television and radio broadcasting, and even cable channels, still use this model. It continues in more modern from on platforms such as Google, YouTube, Spotify, Twitter, (I will never call it “X”), as well as a wide variety of online news, content, and gaming sites. Google, in particular makes many billions of dollar supporting its search services through advertising.

Freemium is a business model in which a basic version of a digital service, or a collection of content, is provided free of charge. More complete versions, with a full set of features, or broader access are offered through paid subscriptions. The idea is to let users get to see the value of a product for without risk in hopes that a small percentage of them will be willing to pay enough to make the platform financially viable.

There are several variations on **Subscription-Based** business models. Many news and information sites, the New York Times and Statistica, for example, require that you pay a monthly fee to access their content. This is often called a **Paywall**, and is very unpopular with consumers. It is at best, marginally successful for content providers.

One variation on Paywalls might be termed **Pay-per-View**. Users are asked to pay a one time fee to access a specific piece of content, or a defined set of content. For example, a season of games for a particular team, or a single boxing match or live performance. Academic publishers use the same approach to sell access to specific journal articles, or all the articles published in a given year by a journal.

Streaming video and music platforms also use a monthly subscription model that allow consumers to access their full collections of content on demand, as often as they like. This is somewhat more successful.

Gaming platforms, operating systems like Windows, PaaS providers such as AWS, and Microsoft Azure, SaaS providers such as web hosting services, collaboration tools like Jira and Google Teams, and more recently, generative AI, all sell their services through subscriptions with prices that depend on exactly what services are chosen. This seems to be a sustainable business model if useful services are competitively priced.

Finally, some digital goods are "sold" through perpetual licenses, under some terms.

Subsection 3.5.5. Digital Rights Management

Digital Right Management (DRM) is a catchall term for enforcement of the terms under with digital goods are licensed. Many approaches have been used over the years, including requiring users to do certain things to activate software, verify that they are legitimate purchaser, and various types of hardware and software based encryption,

DRM has become much tighter with the advent of universal connectively and the cloud. Microsoft computers phone home on a regular basis to verify that they are using a correctly licensed version of Window. Apple forces updates, and limits functionally if device configurations are not satisfactory. Some MP3s, videos, and other content purchased through Apple or Amazon are encrypted, and devices must contact a DRM server periodically to get updated keys.

In effect, users must have continuous and ongoing permission from operating system, software, and content providers, to use digital items they have purchased. In some cases, the physical devices a user has purchased can be bricked if a software provider is not convinced that a proper license is in place. Books, music, and other content that a user has legally purchased can be, and has been, removed from a user's device because the user crossed into a country where then content was not licensed, was forbidden by the local government, or because the content provider simply decided the content no longer met community standards.

There is a significant movement within the ICT community that asserts the DRM protected content is **Broken by Design:**

- **Inconvenience:** DRM schemes require that users have an Internet connection, at least periodically.
- **Lack of Control:** DRM makes restrictive licenses enforceable. Rights-holders can prevent the use of content or even delete it from a device at will.
- **Loss of Privacy:** DRM systems tells rights-holders about your usage pasterns when it phones home.
- **Lack of Durability:** How likely is it that Apple or Amazon will still be in business in ten or fifty years? Will they be running DRM key servers if they are? Will you be able to move your digital collection to new, and yet to be invented, platforms or formats? Can you leave your content to your heirs?

By way of comparison, if you own a book or CD, it can always be used even when offline. You have an irrevocable right to not only use the content it contains, but to sell it, rent it, lend it, or give it away. The rights-holder can never tell if you are using the content, and both the form and format are durable. Books have existed for 2500 years, and many books published today will last for hundreds of years. A DRM-free CD can be ripped, and the files converted to new formats as technology changes. It is possible that such digital files could last forever.

Open or pirated versions of almost any copyrighted work can be found somewhere on the Internet if a user is willing to take the trouble. Rights-holders often claim that piracy has cost them hundreds of billions of dollars in lost revenue. They arrive st this number by estimating the number of movies, MP3s, and so on, that are illegally downloaded every year, and multiplying by their retail price.

Of course, this method is open to criticism. It is surely not the case that all, or even most, of the people who pirated this content would ever have purchased it. Moreover, it is hard to believe that industries that have never had more than 30 billion dollars in annual revenue could somehow be missing out on 100 billion dollars over a few short years.

Bruce Schneier, a technology blogger, summed this up as follows:

> *... trying to make digital files uncopyable is like trying to make water not wet.*

People seem willing to pay for subscriptions to Netflix, Steam, and Spotify. These services provide content in a convenient, and relatively inexpensive, way. It takes effort to find and download content illegally, and files that are found are sometimes low quality, incomplete, or carry malware. Gabe Newell of Valve (the producer of Steam) made the following comment:

Piracy is a service problem.

Section 3.6. Inference and Information

Modeling people as rational agents who maximize objectives given constraints seems to give a good description of how large groups behave on the average. Individuals deviate from this all the time, but economists leave these more difficult problems to physiologist.

Constraints are largely defined by real-world facts. For example: If you don't work, then you will need to find someone who is willing to feed you anyway, or you will starve. If you drive fast, then you are more likely to crash your car. If you don't make yourself agreeable, then you will not have many friends. As an empirical science, economics has developed many tools to make inferences from data. Humans must also understand the constraints they face, and for that, they also must make inferences from data.

Objectives, on the other hand, can be anything at all. There is nothing illogical or irrational about loving pizza, wanting to paint yourself blue, fervently believing that your God, or following a code of ethics that require you to do things that might otherwise appear to be against your own interests.

As a behavioral science, economics has also explored how individuals actually process and respond information. These responses are sometimes wrong in the sense that they cannot be well-explained as an optimization of any consistent objective. Nevertheless, these suboptimal decision responses are systematic and predictable.

Behavioral economists call these deviations **Cognitive Biases**. We suppose that while the math might be wrong in any given instance, these behavioral patterns must have conveyed a survival advantage that resulted in their being encoded into our cognitive pathways through natural selection. This turns out to have great significance in the development of platforms and business models in the technology sector.

This section explores how the cognitive processes humans have evolved to estimate constraints, and maximize objectives, hold up in the modern information age.

Subsection 3.6.1. Evolution and Statistical Processes

Most of us expect that if a coin is flipped, the odds that it will come up heads is 50%. Why is this?

The most likely reason is that we have seen coins flipped before. We have empirically observed that about half the time, a flipped coin lands on head-up. If we are more sophisticated, we might notice that the underlying **Statistical,** or **Data Generation Process,** can be described as a symmetric binomial distribution.

What if the coin was not quite balanced? We might start with a guess that coin flips would follow a symmetric binomial process as a **Prior Expectation**, but if we observed that 60 out of 100 flips landed on heads, we would probably update our prior and describe the process was an asymmetric binomial distribution.

Why do humans have priors, and why do we update these priors? Because if we did not, we would all have been eaten by saber-tooth tigers and selected out of the gene-pool. Our brains are a product of evolution, and so the way they make inferences are optimized for surviving in the environments in which they evolved.

Homo sapiens sapiens, the subspecies of Homo sapiens from which all modern humans descend, evolved about 100,000 years ago in Africa. All humans lived in small groups (50 to 100) of hunter-gatherers until agriculture developed in the Fertile Crescent around 10,000 years ago. Even then, almost all humans lived in relatively small groups until at least 5000 years ago. The great majority only had contact with family, and a small group of people who lived nearby, until very recently. Only in the last one or two hundred years have communication and transportation technologies developed to an extent that broader contacts were widely available.[6]

Throughout our evolution, our knowledge of the world has always come from the small group that made up of our extended family or tribe. Given that interactions were repeated and long-term, disincentives to mislead or lie were strong. Tribe members whose information or viewpoints were false or unfounded would quickly become known (or more probably would already have been eaten by saber-tooth tigers).

Since there were only 70 or so people that any given individual had access to, asking five or ten tribe members for an opinion would yield an unbiased, representative sample of the wisdom at the tribe's disposal. Objective, real-world, observations (is it safe to swim in a river, or does it have crocodiles?) would be similarly unbiased.

[6] World population is estimated to have been between 1,000,000 and 10,000,00 in 10,000 BC, and perhaps 400,000,000 by 1,000 AD. In 2023 world population is estimated at close to 8,000,000,000. Low population density contributed to our historic isolation from larger groups of people.

The implication is that we are evolutionarily conditioned to believe what people tell us, especially if they appear to come from our tribe. We also are likely to think that our first-person observations represent an unbiased sample of the underlying data.

Subsection 3.6.2. Conditional Probability and Correlation

The belief that an unbiased coin will turn up heads 50% of the time is statement about the probability of an outcome independent of any other influences. That is, it states the **Unconditional Probability** of an outcome. For example, we might estimate the odds that a random person we meet on the street speaks Portuguese to be 3% (the worldwide fraction of Portuguese speakers). If we knew that the random person was from Brazil, however, our estimate, conditioned on this fact, might go up to 95%. This is known as a **Conditional Probability**.

For example:

- There were 847 deaths per 100,000 men in 2019 compared to 603 deaths per 100,000 women. The probability of death is different when we condition on gender. We might conclude that being male leads to higher mortality, and lament the way that society victimizes men.
- If we look at actuarial tables, and find, for example, that there are 40 deaths per 100,000 at age 20, compared to 1467 at age 70. We might conclude that being old leads to higher mortality, and lament the way that society victimizes the elderly.
- We could continue on looking at other demographic factors to refine our conditional expectations regarding mortality. For example, the poorest 1% of people have a life expectancy about 10 years lower than the richest 1%. We might also note that smoking cigarettes reduces life expectancy by about 6 years, and the poor smoke at about three times the rate of the rich. Educational levels are positively correlated with life expectancy, but also with income …

What is killing people? Is it age, gender, poverty, lifestyle choices, educational opportunities, or something else?

In fact, all of these factors are predictive. Moreover, they are likely to interact in complicated ways, and affect conditional probabilities to different degrees. Fortunately, we have more sophisticated statical techniques, such as **Regression Analysis**, to explore the correlation of arrays of variables to outcomes of interest, and their relative contributions to the predictive model, to untangle the more complicated interactions between them.

If instead we look only at the unconditional probabilities, we risk arriving at conclusions and policy solutions that may not be justified. For example, if you believe in equality of outcomes, you advocate for a massive increase in spending on men's healthcare in hopes of raising their life expectancy to be equal to that of women. A more detailed analysis of conditional probably, however, might show that there were certain male behaviors or choices that explain much of the difference in mortality rates, independent of gender. Such analysis would point to underlying causes that could be addressed by policy, and suggest how resources could be most effectively deployed.

At the individual level, understanding the constraints that one faces is absolutely essential to optimizing whatever objective a person might have. For example, I might want to be a rich and successful rock star. Data shows that success is correlated to being young, good-looking, outgoing, creative, and musically gifted. I might wish that I was all those things, but wishing is a notoriously bad business plan.

Even if I had all of these characteristics in spades, the data show that about .1% of professional musicians get about 90% of the wealth and attention. The unconditional probability of achieving success is slight. I might tell myself that someone has to be lucky, and why shouldn't it be me? If there is a worse business plan than wishing, planning to be lucky is it.

Given this data, I might regretfully choose an alternative career. For me to make the right decision, however, my data must be correct and representative of the real-world facts. Over or under optimistic estimates lead to different types of suboptimal decision-making. Whatever my preferences, I am better off if I make decisions based on correct data.

The world is complex. Arguments and decisions made on the basis of estimates of unconditional probabilities are very likely to be wrong. It may not be feasible to access the required data or subject it to rigorous statistical analysis in many, or even most, cases. Nevertheless, thinking about how complicating factors might contradict, or substantially change, superficial conclusions puts you at the top of the evolutionary heap.

Subsection 3.6.3. Media, Social Media, and Truth

Throughout the thousands of years that our species has been in existence, assuming that people are telling the truth, and our own observations represent unbiased samples of the true state of nature, have served us well. In the last one or two hundred years, mass media has made this less true. New information technologies that have become ubiquitous in the last two decades make these assumptions increasingly dangerous. Unfortunately, biological evolution is too slow of a process to keep up with technology. Extracting signal from noise will have to rely on higher reasoning, and conscious choice, rather than instinct, going forward.

The printing press, faster transportation technologies, radio, and television, combined with increased urbanization, made mass communication of messages cheap and easy. This led to widespread dissemination of commercial advertising, political and religious ideas, and even propaganda. Each time we see such a message, we add it to our set observations about the state of the world. Of course, these observations are not independent or unbiased. In fact, they are duplicated observations of a single, not necessarily correct or honest, viewpoint. Unless we are careful, we tend to conflate hearing a message many times with hearing it from many sources.

Social media intensifies this effect for two reasons.

- The messages in our feeds, and the results of our searches, often do come from different sources. They look much more like independent observations than advertising through mass media. What we forget is that platforms *choose* which messages to send us, and which to suppress. If 2% of the world thinks that tigers are just misunderstood kittens, but it is the opinion expressed by 80% of content in our feed, it is difficult to fight the programming that has generated winners in the genetic sweepstakes for thousands of years. In this case, however, you may end up being eaten by a tiger if you follow your instincts.
- Mass media is just that: mass. It has to communicate a unitary message. For example, electronic spectrum is scarce, and using it is expensive. Broadcast media has no choice but to send the same message to all its listeners. Since broadcasters had to choose single message, they also tended to choose ones with wider, rather than narrower, appeal. Although print media can produce a variety fliers or posters with different messages, it would have been very difficult 100 years ago to selectively deliver them to the correct audiences. If instead the fliers were distributed randomly, or even universally, the message would be lost or ineffective. Social media, on the other hand, presents a different set of content to each human, and uses AI to decide exactly what that content should be.

Subsection 3.6.4. Behavioral Economics and Cognitive Biases

Successful social media, content providers, search engines, and other platforms, understand very well how nature has trained their users to interpret and respond to information. Those that do not are out-competed by those that do. At the very least, platforms use machine learning is to maximize your engagement if only because they have a responsibility to their shareholders to maximize their profits. Whatever it is that you need to hear or see to keep you on the platform, is what you will get, even in the absence of any hidden big-tech agenda.

Technology companies also understand how certain sorts of stimulation can elicit responses that causes users to do things that are not rational in the conventional sense. Economists, I am ashamed to say, have been instrumental in showing technology companies how to increase profits by taking advantage of certain behavioral quirks that seem deeply embedded in our behavioral programing. A leading example is:

Confirmation Bias: A cognitive bias that causes people to give higher weight to, and even to actively seek out, information that confirms their prior beliefs, and to ignore or discount conflicting evidence or views.

Confirmation bias can be seen as the opposite of what **Critical Thinking** purports to be. It is not clear why confirmation bias confers higher genetic fitness, but it is such a pervasive human characteristic, that it must do so in some way. Confirmation bias does not seem to discriminate on the basis of ideology or educational achievement. All of us are subject to it to a greater or lesser extent.

The result is that if your media feed, search results, or content suggestions, play into your preconceptions, you tend to stay engaged and remain on the platform. Media companies are more than happy to tell you that you are smart, pretty, misunderstood, and high-value, and that all the people you don't like are stupid, evil, and will get their just deserts.

Platforms are also very good at figuring out what it is you want to hear. Confirming your biases, makes you even more eager to hear more, and less willing to consider alternative viewpoints. This kind of induced information addiction is a significant element in all successful social media platforms.

Other behavioral quirks exploited by social media, and electronic platforms more generally, include:

Loss Aversion: A cognitive bias that causes people to view losses as subjectively more harmful than the benefit of objectively equivalent gains.

People seem to be strongly motivated to hold on to what they have. Going backwards a step is evidently much more painful, than the pleasure we might get from going forward a step. This leads people to be overly cautious, and even conservative. People seem to prefer accepting the same old mediocre outcomes to taking a chance on a much higher expected outcome if there is any downside risk.

The most common example comes from financial economics. Loss aversion causes people to hold on to assets that have lost value since purchase even though selling them and reinvesting in better ones would increase expected return. Realizing such losses is more painful than the potential benefit of recovering from an investment mistake. A similar phenomenon is observed in people who inherit or are given things like fine art, or a fancy car. If they had received the cash equivalent, they would never have purchased such things, but they nevertheless seem unwilling to sell items they inherit in order to buy things they would otherwise prefer.

A closely related bias is:

Sunk Cost Fallacy: A cognitive bias that causes people to see already committed, and unrecoverable, investments of resources as a reason not abandon a project for one with a more promising return.

Both of these biases result in people making objectively less than optimal choices because of how they *arrive* at a decision point. Think of how such path-dependence affects the behavior of people who have invested time, money, prestige, or effort, in learning a game, becoming an influencer, backing a social movement, building a project, or developing a following.

Framing: Surrounding a choice with context with the intention of leading people to a desired outcome.

Framing is not so much a cognitive bias as it is an attempt to present information and choices in ways that make the "right" decision more likely. This might be done through the exploitation cognitive biases like the ones above, selective presentation of surrounding information, social pressure, or other methods.

The conclusion is that not only do new technologies take advantage of our trusting nature, and belief that our personal observations reflect the true state of nature, but also facilitate the systematic exploitation of our cognitive flaws. Evolution has not provided us with natural protections from being deceived or exploited by new technologies. Critical thinking in the sense of using higher brain function to overcome our natural tendencies is the only defense that seems to be available.[7]

Subsection 3.6.5. Bayes Rule and Posterior Probabilities

We have already discussed the idea of conditional probability. For example, we might think that everyone in Texas rides horses. Suppose we saw someone riding a horse? Would we then conclude that he must be from Texas? The answer is no. Other people ride horses as well. The fact that a person is riding a horse, however, certainly may inform our estimate. If Texans are more likely than others to ride horses, a person on a horse is more likely to be a Texan. It is not enough on its own, however, to draw a conclusion.

Our current assessment of the conditional probability that Texans ride horses is called a **Prior**. Our estimate of the conditional probably that a person on a horse is a Texan based on this prior is call a **Posterior**. **Bayes Rule** is a mathematical identity that relates prior to posterior beliefs. We will see that it imposes a consistency on beliefs that incorporates the way that the underlying data was sampled. This can allow us both to update incorrect priors, and radically change our understanding of reported outcomes.

Mathematically, we can express our prior belief that all Texans ride horses as follows:

$$\text{P(Rides horses} \mid \text{from Texas)} = \text{P(R} \mid \text{T)} = 1.$$

Bayes Rule relates this prior to the posterior conditional probability that a person on a horse is from Texas as follows:

$$P(T|R) = \frac{P(R \mid T) \times P(T)}{P(R)}$$

where:

P(T | R) : the posterior conditional probability that a person is a Texan, given that he is a Rider.

P(R | T) : the prior conditional probability that a person from Texas is a Rider.

P(T): the prior unconditional probability of that a person is from Texas.

P(R): the prior unconditional probability that a person is a Rider.

[7] It is worth emphasizing that evolution also affects corporations. Technology companies may or may not have evil intentions. They may or may not use their power in ways that seem gratuitously harmful. However, they must use their technologies to maximize profits, or they create an opportunity for a competitor. Economic evolution is much faster than biological evolution, but is just as unforgiving.

It is easy to find census data that tells us that about 10% of the US population is from Texas, and about 2% of the overall population rides horses. Given our prior, we have all the data we need. We can estimate the posterior as:

$$P(T|R) = \frac{1 \times .1}{.02} = 5$$

Wait. What? How can a posterior probability, or any probability, be greater than 1? It can't be. Since our two unconditional population probabilities are factual data, it can only be that our P(R | T) = 1 prior that says all Texans ride horses is a false stereotype. Thus, we have learned that our prior about Texans cannot be correct and should be updated.

Suppose we fly down to Dallas, and we find that while about one in twenty Texans do ride horses, about 95% drive pickup trucks. We revise our prior to reflect this: P(R | T) = .05. This gives us the correct posterior:

$$P(T|R) = \frac{.05 \times .01}{.02} = .25$$

In words, the vast majority of Texans don't ride horses, but they are more likely to than the rest of the US population. For example, suppose there were 300 million Americans. Then 30 million would be Texans. A total of 6 million Americans ride horses (2%), and 1.5 million of those would be Texans (that is, 5% of all Texans). Thus, Bayes Rule tells us that while Texans only make up 10% of the population, they make up 25% of the US horse riding population. The conditional probability that person is a from Texas, given that he is riding a horse, is therefore 25%.

An implication of this exercise is that manipulating priors is a very inviting human attack surface. Suppose I want you to believe something like <u>the sky will fall unless we do X</u>, or that <u>all people of type Y are dangerous criminals</u>. If you start out with these biased priors, or I can persuade you to adopt them through social pressure, social media, education, religion, or good old-fashioned propaganda, I can control your actions to the extent they are informed by your beliefs. This is epically effective if:

- You live in an echo chamber, perhaps because you are social isolated, a member of a closed community, get your information from curated sources, or filter your observations of the real-world because of confirmation bias.
- There are very few observations available that allow you to update your priors. For example, the sky can only fall once. Until then, it may only be in the process of falling, and so it is difficult to update a prior about whether the sky might or might fall if we don't do X. Similarly, if there are very few people of type Y in your community, or if all your observations are selected, and therefore subject to sample bias, you may never observe a person of type Y not being a criminal.

Another implication is that Bayes Rule helps us check the internal consistency of our beliefs. Inconsistent beliefs are immediately evident since they will violate the equality. In the example above, we saw that it was mathematically impossible to reconcile the stereotype that all Texans ride horses with the facts that Texans are 10% of the population, and 2% of Americans are riders. Since Bayes Rule is a mathematical identity, any three of its constituent values implies the fourth value. This gives us an hypothesis to test empirically, or intuitively, to support or reject our estimates. Even knowing only two values implies something about the ratios that the other two components must have.

In many situations, we have good, well-verified, priors. In such cases, Bayes Rule can help us in a different way. Suppose, for example we wanted to know if biometric iris scans are a good way of authorizing access to a computer system, or a physical facility.

There are two ways that a scan can fail. First, it may erroneously decide that a scan is a match with scan on-file for an authorized person. This is called a **False Positive**. Second, it may erroneously decide that a scan is a not match with any scan on-file for an authorized person. This is called a **False Negative**. Testing done under controlled conditions establish the likelihood of these errors is of following magnitude:

False Positives: One in a million ➞ P(False Positive) = 10^{-6}

False Negatives: One is a hundred thousand ➞ P(False Negative) = 10^{-5}

Now, suppose that someone passes an iris scan. Should we think that there is only a one in a million chance that he is an imposter? Bayes Rule says: not so fast. We also need the following information to make this judgment:

P(Authorized) = The probability that an Authorized person is being scanned.

P(Unauthorized) = 1 − P(Authorized)) = The probability that an Unauthorized person is being scanned.

That is, we need to know the nature of the population that we subject to our scans. We can use the false positive and negative data to find the following conditional probabilities:

$$\text{P(Pass} \mid \text{Authorized)} = 1 - 10^{-5} = 0.999999$$

$$\text{P(Pass} \mid \text{Unauthorized)} = 10^{-6} = 0.000001$$

We now must calculate the unconditional probability that a random person drawn from our sample will Pass the iris scan. If P(Authorized) is the unconditional probability that random draw is in fact an Authorized person, then we get the following answer:

$$\mathrm{P}(\text{Pass}) = \mathrm{P}(\text{Pass}|\text{Authorized}) \times \mathrm{P}(\text{Authorized}) + \mathrm{P}(\text{Pass}|\text{Unauthorized}) \times \mathrm{P}(\text{Unauthorized}).$$

Give all this, can then calculate the conditional probability that a person who Passes the iris scan is actually Authorized as follows:

$$\mathrm{P}(\text{Authorized}|\text{Pass}) = \frac{\mathrm{P}(\text{Pass}|\text{Authorized}) \times \mathrm{P}(\text{Authorized})}{\mathrm{P}(\text{Pass}|\text{Authorized}) \times \mathrm{P}(\text{Authorized}) + \mathrm{P}(\text{Pass}|\text{Unauthorized}) \times \mathrm{P}(\text{Unauthorized})}.$$

To get a sense of why this useful, suppose that 50% the people who walked up to a door and submitted to an iris scan were, in fact, Authorized. Then:

$$P(\text{Authorized}|\text{Pass}) = \frac{.999999 \times .5}{0.999999 \times .5 + .000001 \times .5} = \frac{0.4999995}{.5} = 0.999999$$

In this case, it is almost a certainly that the person who Passes the iris scan is Authorized.

Suppose instead that the scanner was hooked up to an API what could be accessed though the Internet. Millions of bots might try to get lucky by presenting random iris images for scanning. Suppose that for every Authorized user who got scanned, there were 10 million Unauthorized bots that also were "scanned." Then:

$$P(\text{Authorized}|\text{Pass}) = \frac{0.999999 \times 10^{-7}}{0.999999 \times 10^{-7} + .000001 \times .9999999} = \frac{0.0000001}{0.0000011} = 0.0909091$$

In words, for every Authorized user who is scanned (and almost always passes) there are 10 million imposters, of whom one in a million, or about 10, Pass. Thus, for every 11 positive scans, only one, or about 9%, is an authorized user.

The point of this exercise is to show the critical role that the nature of the underlying sample and our priors play in understanding data. If we don't know how a study was conducted, and especially how the sample was selected, we really can't conclude much of anything. We tend to fill in these priors through guess work and intuition. These guesses are often influenced by confirmation bias or framing in ways we may or may not be aware of.

A more positive point is that if we do know the true underlying distributions, we can use Bayes Rule to make good judgments. For example, the analysis above suggests an iris scan might be a good security measure if used in person, in the lobby of a facility, but not good at all as purely electronic authentication test.

The same analysis suggests that surveillance using iris scans to look for criminals may not be a very good idea. If you have to scan ten million people when you are looking for a single individual, you will end up with a great many false positives despite the extreme accuracy of your biometric test.

Subsection 3.6.6. Experts

We often seek the opinion of an **Expert** to help us understand the world. This can be very effective, and as technology becomes more complex, the average person will have little choice. For example, can you confirm that AES256 encryption is effectively unbreakable given current hardware and cryptographic attack algorithms? The math is beyond me, at least.

Why do we trust experts? After all, as non-experts, we can't judge whether they are telling us the truth or not. If we could, we would not need an expert. Thus, we use secondary methods to judge. For example:

- We seek recommendations from people we trust attesting that an expert is credible and competent.
- We look at credentials, experience, or education, in the relevant field, and endorsements by other individuals or institutions that we judge to be credible.
- We look at their record and history. Do they have a good reputation that took effort to earn? Have they ever been caught lying or making a mistake before? The more costly it was to establish their expert standing, the less likely it seems that they would throw it away by lying to us. Of course, we have to make sure that they have not already lost credibility, or can lie to us without suffering repercussions.

This breaks down if experts have strong enough positive incentives to misinform, or negative incentives to tell the truth. When rewards for supporting the conventional wisdom are rich, or the punishment for dissent is severe, experts should not be trusted. This also breaks down when confirmation bias influences our choice of which expert to listen to. Captive experts who we reward for telling us what we want to hear are not acting as experts.

Subsection 3.6.7. Artificial Intelligence

The fundamental problem with AI is that they are reflections of their creators. AI is anything but a neutral source of information. AI systems can be programmed to weigh certain types of evidence more heavily or weakly, or to put responses through a filter reflecting specific political or social view points. Users can never know the biases built into the algorithm, nor how comprehensive or credible the data made available to the system might be. It is simply beyond us to conduct any sort of audit.

It is very important to be aware that almost all the content and data you see in any electronic form has be selected, framed, presented, or even created, by systems driven by AI. What you don't see has also been selected, and is largely out of your control.

It is a mistake to interpret the information platforms feed you in the same way our ancestors did their first-person observations of reality. Bayes Rule tells us that those who continue to use the heuristic decision processes that allowed our species to survive and thrive in a per-technological world, will simply be following the queues that their platforms of choice provide.

But then again, Bill Gates, Elon Musk, Jeff Bezos, Mark Zuckerberg, and the others who build and program these AI systems, are good guys. What could possibly go wrong?

Section 3.7. Economics

Subsection 3.7.1. Present Value

Money today is worth more than money tomorrow. This bias toward the present arises for two reasons. First, we observe that people are impatient on average. All else equal, people would rather consume a dollar now rather than next year. However, people also exhibit individual **Internal Rates of Time Preference**, which is the rate at which they would be willing to exchange a dollar now for more than a dollar next year. Second, there are many ways to invest money so that it grows year to year. The opportunity cost of giving up a dollar today is this expected return on investment.

Income that arrives in future time periods is **Discounted** relative to current income. The **Present Value** of income that arrives several periods in the future is found by compounding the periodic discount rate. We can calculate the present value of a stream of income (or costs) over many periods by summing up the present value of the income in each period.

Formally, the present value of a revenue stream $\left(R_1, R_2, \ldots, R_T\right)$ is the following:

$$PV = \sum\nolimits_{t=0}^{T} \frac{R_t}{\left(1+r_t\right)^t}$$

where:

$t \in \{1, \ldots, T\} \equiv \mathcal{T}$: periods or years in which revenue arrives with T being the final year

R_t : revenue received (which could be negative) in year t,

r_t : discount rate in year t,

Let's use this idea to evaluate how long copyright terms might generate incentives to create new works.

<u>**Consuming Revenue Flow**</u>: Obviously, for this to be true at all, the work must continue to create revenues for the lifetime of the copyright, or at least long into the future.

<u>**Substantial Future Revenues**</u>: These future revenues must also be substantial. At a 7% discount rate, \$1 in revenue 25 years from now is worth the same as about. \$.17 today, and \$1 in revenue 50 years from now is worth less than \$.03 today. In other words, discounting drastically reduces the value of these future revenues, and so severely limits their power to incentivize creativity. It is hard to imagine how revenues, 70 or 100 years after a work is created would have any incentive effect at all.

<u>**Predictably High Future Revenues**</u>: The creator of the work must be relatively certain about these future revenues. If there is only a 1% chance that a work will produce \$1 in revenue 50 years from now, then the expected value is only 1% of the discounted value of less than three

cents, That is, the expected value of this revenue is \$.00026. Uncertainly about future revenue streams further limits their power to incentivize creativity.

Agency and Incentives: The creator must care about these revenues. Recall that the creator is dead for most of the extent of the copyright term. For these revenues to matter, the creator of the work must care about the welfare of decedents who have not even been born yet. If a corporation is the creator, the decision-maker must think that he will be rewarded today for an uncertain stream of revenues in the distant future long after he has left the company.

In short, it is very hard to make the case that long copyright terms are a significant incentive for the production of new creative works. They are, however, a windfall for corporations lucky enough to own the rights to the vanishingly small number of works that happen to produce significant revenue decades after they were created.

Subsection 3.7.2. Public and Private Goods

Rival goods are often called **Private Goods**. Examples include housing, clothing, food, cars, and electricity. The common property is that if I eat a hamburger, you cannot eat it as well. A hamburger is a zero-sum game. More formally:

$$\sum_{i \in \mathcal{I}} H_i = \bar{\dot{H}}$$

where:

$i \in \{1,\ldots,I\} \equiv \mathcal{I}$:	agents in the economy
H_i :	the number of hamburgers eaten by agent *i*
$\bar{\dot{H}}$:	the total number of hamburgers produced

Nonrival Goods are often called **Public Goods**. Examples include national defense, entertainment content, radio broadcasts, and knowledge. The common property is that the fact that I know something, like $E = MC^2$, in no way prevents or impedes you from knowing it as well. Thus, the simplest version of the consumption constraint can be written:

$$K_i = \bar{\dot{K}}$$

where:

K_i :	the amount of knowledge consumed by agent *i*
$\bar{\dot{K}}$:	the total amount of knowledge that exists.

Provided that a non-rival commodity is a good, all agents will choose to consume all that if produced.

If public goods are produced in sufficiently high quantities, however, agents may become satiated. such commodities need not become bads, but they are no longer goods. It is neither desirable, nor possible to know everything that was ever discovered. It is unlikely that any agent would want to listen to every radio broadcast, hear every song, or watch every video ever created. Almost all public goods are **avoidable** in the sense that agents can consume as much as they wish, and then ignore the rest. For avoidable public goods like these, the consumption constraint is really:

$$K_i \leq \overline{\dot{K}}.$$

Many public good are also **excludable** in the sense that the permission of the producer is required in order to consume them. For example, If television shows are encrypted by the broadcaster, they can only be decoded by agents who have paid for subscriptions. Even though all agents in the coverage area can receive the transmission, they can be excluded from using the content the broadcast contains.

On the other hand, some nonrival commodities are public bads, smoke pouring out of a factory, for example. In many cases, agents are unable to avoid consuming any public bads that are produced. For example, global level of atmospheric CO_2 is about 400 ppm. This can be seen as a nonrival public bad. No agent on the planet can avoid the full effects of this 400 ppm level.[8]

Subsection 3.7.3. The Cathedral and the Bazaar

In Eric Raymond's paper entitled “The Cathedral and the Bazaar”, he focuses on two fundamentally different approaches to writing software. The **Cathedral Approach** is top down and hierarchical. The Pope tells the Archbishops what he wants, and the order goes down the chain of command to be implemented by the Parish Priests. Everyone involved works for the organization and either follows instructions, or finds another employer.

The alternative is what Raymond calls the **Bazaar Approach**. In a bazaar, all the goods are visible and open for inspection. There is no central authority in charge. If anyone thinks that he has a good idea, he is free to set up a shop. Shop owners talk to one another, observe each other’s strategies, share ideas, and even partner together if they find it worthwhile. Customers and vendors negotiate, and if a customer does not like what a merchant offers, he can always go to a different vendor (or even set up a competing shop).

[8] Excludable public bads do not have much empirical relevance. If a public bad is avoidable, then it does not matter whether it is excludable. Agents simply ignore it. On the other hand, a nonavoidable and excludable public bad might arise if the producer of the bad somehow had the ability to prevent it from harming specific agents. For example, suppose a mad scientist invented an airborne virus that gives the infected an overwhelming desire to know every detail about the lives of the Kardashians. Once the virus is released, everyone in world will eventually be infected. However, if the scientists also developed a vaccine, then he could exclude anyone he wishes from suffering this dire fate. People would have to pay the scientist to exclude them from the public bad. Let us hope such cases continue to be empirically irrelevant.

Microsoft Windows and the Linux OS are leading examples large-scale software projects involving thousands of programmers coordinated using the Cathedral and Bazaar model, respectively.

These are two very different ways marshaling and organizing resources for production. Each has their own strengths and weaknesses.

Companies and private businesses are mostly organized on the Cathedral model. Capital comes from investors who are usually given part ownership, and control of the company. Management is chosen for its ability and vision, and is supposed to work in the interests of the shareholders. Complicated systems wages, benefits, promotions, bonus, and monitoring, provide incentives for employees at various levels to work to further the companies interests.

The upside is that a well-managed company has clear lines of responsibility, can coordinate the efforts of teams and individuals towards a well-defined goal, and can maintain a continuity of purpose. The downsides are that hierarchies are rife with principle-agent problems, incomplete information, and moral hazard. It is difficult at every level to know what employees and managers are really doing, and to correctly incentivize them. The larger the company, and the more complex the product, the more difficult this becomes.

Charitable organizations, civic movements, and certain artistic and technolgical projects run on the Bazaar model. Often, the leader, or core team has a clear vision of what needs to be done, and seeks like-minded people to donate their time and resources to accomplish it. Contributors come and go, and contribute with varying intensities. Since all efforts are voluntary, the contributor decides exactly how, and what, to contribute.

The upsides are that important public goods that would not otherwise be produced are come into existence. This might be anything from building houses for the poor, or running a youth Klezmer band, to creating an open source operating system. The downsides are that people have to be convinced to work, and work towards a common goal, without direct compensation. If people lose interest, the project can lag or die.

Subsection 3.7.4. Voluntary Contributions and Freeriding

The Bazar model of production outlined above depends on voluntary contributions. The standard assumption in economics is that people are self-interested. Nevertheless, we see lots of people voluntarily contributing effort, money, and other resources to projects produce public goods, or even simply private benefit for others. Can this be explained if agents are just utility maximizers?

Consider an economy with one public good, and one private good. agents have identical utility functions, and face the following constraints:

$$U_i(X_i, Y), \quad X_i = \omega - c_i, \quad Y = \sum_{j \in \mathcal{I}} c_j$$

where:

$i \in \{1,\ldots,I\} \equiv \mathcal{I}$:	agents in the economy
X_i :	private good consumed by agent *i*
Y :	public good consumed
c_i :	private good contributed to producing the public good by agent *i*
ω :	private good endowment of each agent

To see what this implies, suppose that there were 100 people who would value a certain type of software. They plan to start a project and make this FOSS. Each agent $i \in \{1,\ldots,100\} \equiv \mathcal{I}$ has the following, identical, utility function, and so solves this problem:

$$\max\ U_i(X_i,\ Y) = X_i + Y^{1/2} \text{ subject to } X_i = \omega - c_i \text{ and } Y = c_i + c_{-i}$$

Note that the total amount of public good equals the sum of any given agent i 's contribution, and the contributions of the other 99 agents:

$$c_{-i} = \left(\sum_{j \in \{1,\ldots,100\}} c_j \right) - c_i$$

Each agent takes the contributions of the other agents as fixed when deciding on his own contribution. Substituting in the constraints, each agent's problem becomes:

$$\underset{c}{\max}{}_{i}\ \omega - c_i + (c_i + c_{-1})^{1/2}$$

Taking the derivative with respect to c_i and setting this equal to zero:

$$-1 + \frac{1}{2}(c_i + c_{-i})^{-1/2} = 0 \Rightarrow 1 = \frac{1}{2}(c_i + c_{-i})^{-1/2} \Rightarrow$$

$$2 = (c_i + c_{-i})^{-1/2} \Rightarrow \frac{1}{2} = (c_i + c_{-i})^{1/2} \Rightarrow \frac{1}{4} = c_i + c_{-i} \Rightarrow$$

$$c_i = \frac{1}{4} - c_{-i}$$

In words, agent *i* should contribute just enough to bring the total contribution up to 1/4. It is unclear how the cost will be actually be divided. If one agent contributes the entire amount, then all the other agents contribute nothing. sharing this contribution equally would be another stable solution.

We see that even selfish agents may contribute positive amounts to public goods productions. Unfortunately, these contributions are far below the socially efficient level, and so there is a great deal of freeriding.

Suppose instead that agents agreed to vote on a tax to be paid equally by all 100 agents. If they did so, each agent's optimization problem would become the following:

$$\max_{t}\ \omega + (100t)^{1/2}$$

which implies the following solution:

$$-1+\frac{100}{2}(100t)^{-1/2}=0 \Rightarrow 1=\frac{100}{2}(100t)^{-1/2} \Rightarrow$$

$$\frac{1}{50}=(100t)^{-1/2} \Rightarrow 50=(100t)^{1/2} \Rightarrow 50^2=100t \Rightarrow$$

$$t=25$$

Suppose that $\omega = 50$. Then if agents rely on voluntary contributions, their utility levels will be between $U_i = 50.25 = 50 - .25 + (.25)^{1/2}$, and $U_i = 50.5 = 50 - 0 + (.25)^{1/2}$, depending on how much of the total contribution is borne by any given agent. On the other hand, the tax system approach gives each agent a utility level of $U_i = 75 = 50 - 25 + (100 \times 25)^{1/2}$.

Given this, how do FOSS projects survive? The answer seems to be **Warm Glow**. That is, agents get utility from the act of contributing to, as well as from consuming the public good. This might be due to feeling like a good person, caring about the welfare of others, to gain professional visibility, enjoying the activity involved in contributing, or other reasons. It may even be a **Cold Prickle** from the fear of being seen not to contribute. This alters the utility function as follows:

$$U(X,c,Y) = X + V(c) + Y^{1/2}$$

where $V(c)$ is the utility that an agent gets from the act of contributing. This serves to increase each agents' voluntary contribution level, and off-sets, to some degree, under-provision of the public good.

Warm glow is an important motivator. It is major reason that people write code for software, tithe at church, contribute to charities, build houses for Habitat for Humanity, serve food to the homeless, and vote in elections.

Chapter 4. Computers and Hardware

Computers of various kinds are at the foundation of ICT today. Modern computers can be defined as electronic devices for storing and processing binary data. Data is **Binary** if it consists of a series **Bits** that take one of two values: on or off, or 1 or 0. This is distinguished from **Analog** data which is represented as a continuously variable signal or level.

Binary data is typically grouped into "words" or "octets" consisting of eight bits, collectively called a **Byte**. A byte, in turn, can take one of 256 distinct values (2^8) from 00000000 to 11111111. For example:

$$\begin{aligned}0000\ 0000 &= 0\\ 0000\ 0001 &= 1\\ 0000\ 0010 &= 2\\ 1111\ 1111 &= 255\end{aligned}$$

ASCII (American Standard Code for Information Interchange) is a system that encodes[9] letters and numbers to one of the 128 values possible with a seven bit binary word.[10] This allows computers to use binary data to exchange textual information in a mutually comprehensible way. **Unicode** is an extension of ASCII that uses as many as four bytes (allowing 2^{32} = 4.3 billion possible values) to encode characters and glyphs (including emojis) with the objective making all living languages in the world printable and translatable into binary code.

The concept of binary information had its practical start with **Claude Shannon**'s 1947 paper: "**A Mathematical Theory of Communication**" Before then, there was no real concept of what a unit of information was. Was the number of words in a book a measure of its information content, or the number of letters? How would this compare to image or a song? Is a square inch of an image equal to 1000 words or one minute of sound? Clearly these things are incomparable.

Shannon's contribution was to realize that every kind of analog information could be represented digitally. Words could be encoded as ASCII files, sound as MP3s, images as JPEGs or PDFs, and so on. While there were many possible formats and codecs to convert analog information into digital

[9] See Chapter 8.

[10] The eighth bit is called a **Parity Bit**, and is used for error detection. This works as follows: the first seven bits are added together and the result is either an odd or even number. If the parity check is even, then the last bit is set to one or zero to make the sum of all eight bits an even number. For example, the capital letter "Q" has a seven bit ASCII representation of 1010001. This adds up to three. Thus, to make this an eight bit binary number (a byte) with even parity, we would add a one to the end, 10100011, which makes the sum equal four, an even number. This is the simplest form of error checking and tells you if exactly one, three, five, or seven bits have flipped. For example, if the first bit flipped from a one to a zero, 00100011, the byte would fail the parity check since the sum of the eight bits is now odd. The receiver would then ask you to retransmit your message. See the end of the chapter for more detail on binary and Boolean math.

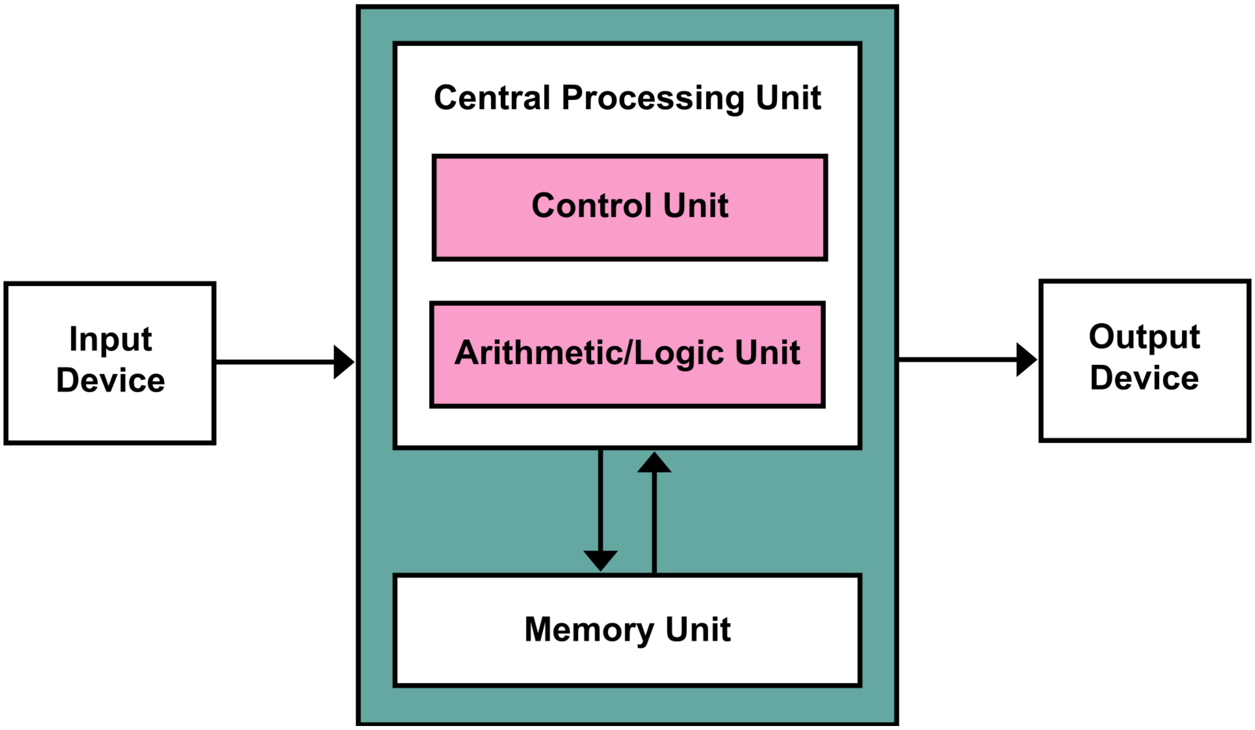
files, the result is information that can be stored, used, and interpreted by computers. The impact of this cannot be overstated. It is the foundation of the information age. Chapter 8 explores this in more detail.

Section 4.1. Computers

At a physical level, the classical description of a computer was given by John von Neumann in 1945. He proposed an architecture that was analogous to the human brain. Specifically, he suggested computers are composed of the following basic components:

- Arithmetic Logic Unit (ALU)
- Control Unit (CU)
- Memory (MU)
- Input (I)
- Output (O)

The first three of these might be classified as associative neurons, and the last two as sensory and motor neurons, respectively.



This basic architecture has been combined in a number of different ways. Most important are the following:

Mainframe Computers: Large and powerful computers used to handle large datasets and bulk processing, especially processing that requires high input/output reliability. These are typically used by large organizations and enterprises.

Supercomputers: Large and powerful computers used to handle problems requiring intensive calculations. These are typically used in research and cryptographic applications.

Personal Computers: Relatively cheap computers designed to be used by one person at a time instead of supporting multiple users in a time-sharing environment.

Cloud Computing: A pool of easily usable and accessible virtualized resources (such as hardware, development platforms, and/or services) that can be dynamically reconfigured to adjust to variable loads for optimum resource utilization.

Grid Computing: A system of distributed and parallel computing in which a virtual computer is built from a cluster of networked computers working cooperatively to execute large tasks. Computers can enter and leave the networks, and often provide only a fraction of their computational and other resources to the grid collective. Grid computing is typically used in large scale, but low priority applications.

In the next few sections we give a brief discussion of these basic components, the technology behind them, and how they affect the structure and use of computers.

Section 4.2. Central Processing Unit

The **CPU (Central Processing Unit)** of a computer can be thought of its brain. It combines the functions of von Neumann's **Arithmetic Logic Unit (ALU)** and **Control Unit (CU)** on one chip, although as separate component structures. Modern CPUs also include a small amount of memory. The CU is responsible for **Fetching, Decoding, and Executing** instructions. The ALU, on the other hand, is a specialized set of circuitry that executes certain kinds of arithmetic calculations when required.

The number of instructions that a CPU can execute per second is governed by an oscillating quartz crystal that sends out a fixed number of pulses per second. These pulses, called **Clock Cycles**, are distributed to various components in the computer to synchronize their actions.

Modern CPUs run at a clock speed of 1 to 5 GHz, meaning that there are one to five billion clock cycles per second. This speed is limited by the time it takes data channels to transit from 0 to 1, that is, the time it takes binary signal to enter the channel and then exit at its destination, leaving the channel ready for a new signal. The rate at which the CPU is able to disparate heat is also a limiting factor.

As of 2020, most CPUs intended for personal computers use either a 32 or 64-bit architecture This indicates the number of bits that the CPU can process in one clock cycle. A 32-bit[11] chip, for example, has data path (or **Front Side Bus)** that allows 32 bits (or 4 bytes) to be read and processed at once.

In addition to affecting a computer's speed, this architecture determines the number of individual memory locations that the CPU understands and can address. Thus, a 32-bit chip can only use 2^{32} gigabytes of RAM (in practice, it is actually a bit less than 4.3 GB for technical reasons) while a 64-bit chip can address about $2x10^{19}$ locations.

Several CPUs (or **Cores**) can be built on a single chip. This increases processing power, but less than in proportion to the number of cores. The reasons relate to the clock speed of the chip and the quality of the parallel processing strategy.

Instructions are fetched and then parceled out to the cores. One instruction is executed each clock cycle, but the jobs sent to the individual cores will typically require a different number of steps to complete. Thus, when one core finishes its part of the job, it must wait until the other cores finish as well so that the results can be combined and returned to the Control Unit. These cores are **Hung Up,** and so are idle in the meantime. The result is that each core is actively operating a smaller percentage of the time than it would in a single core CPU system.

Section 4.3. Memory

There are a number of different kinds of memory, but in all cases, memory is where programs and data are stored when they are in active use.

RAM (Random Access Memory): This is the primary memory space of a computer. The "random" part means that the CU can access any memory location it wants in any order. Data stored on magnetic or paper tape, in contrast, must be accessed sequentially. RAM is usually packaged on circuit boards called **DIMM**s **(Dual In-line Memory Modules)**, RAM can be accessed quickly relative to hard drives, but is still fairly slow in comparison to the clock cycle of modern CPUs. RAM is also a **Volatile** form of memory meaning that it requires power to maintain the data it stores. If RAM loses power, all the stored data is lost.

Cache Memory: This is another form of RAM. The difference is that cache memory is placed on the CPU chip itself instead of being installed as a separate module. The L1 cache is physically closer to the CU, uses larger transistors, wider etched connections, is more expensive, and is therefore added in smaller amounts than L2 cache. They both are part of the active memory system, and contain elements of programs and data that may also reside in RAM. The difference is that they can be accessed much faster.

[11] Shout out to Hing So.

The CU uses algorithms to anticipate what data it is likely to need in the near future and "prefetches" it from the hard drive or RAM and writes it to the L1 or L2 cache. This makes it quickly available and thereby reduces the likelihood that the CPU will be hung up doing nothing while it waits for the next instruction it needs to arrive from the HDD. The gain to system performance can be very large if this predictive use of cache has a high "hit" rate.

ROM (Read Only Memory): This is a type of non-volatile memory that is seldom used today. It was intended to hold permanently coded programs that could not be changed. Most modern systems use **EEPROM (Electrically Erasable Programmable Read-Only Memory**), a type of volatile memory that needs a small charge provided by a battery to maintain its data. EEPROM can be rewritten a limited number of times, and is used to hold the system's **BIOS (Basic Input/Output System).**

Section 4.4. Storage

Storage devices maintain programs and other data that are not currently being used by the CPU in non-volatile memory. They are best classified as input and output devices in the von Neumann schema since they are where the CC goes to get input for RAM, and eventually to write results as output.

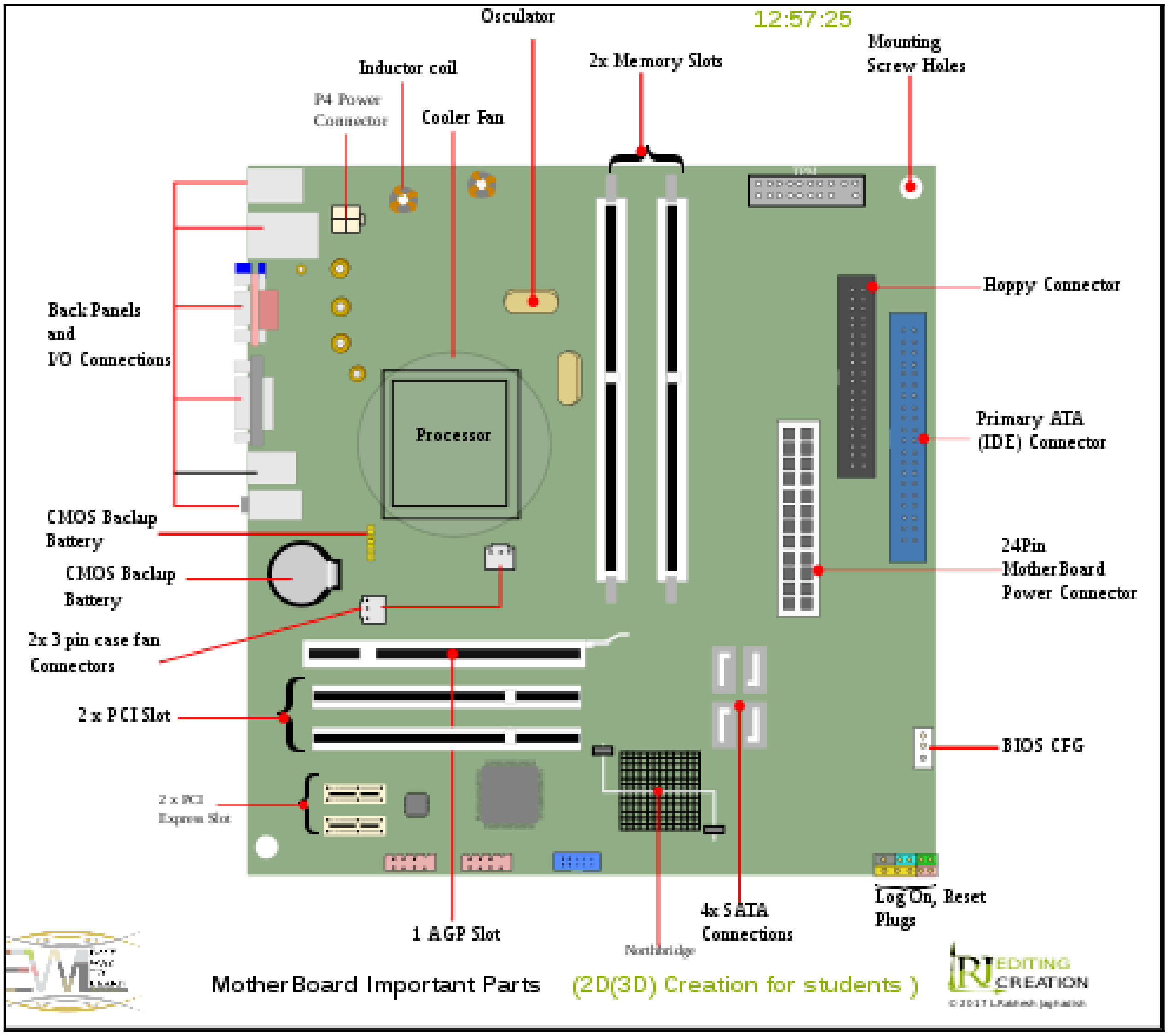
The earliest types of computer storage devices were punched card, punched paper tape, magnetic drums, selectron tubes, and magnetic cores. The **HDD (Hard Disk Drive)** now in common use has its origins in the mid-1950s. Floppy disks came along in the 1970s and CD-ROMs in the 1980s. The **SSD (Solid State Drive)** had it origins in the 1980s and early 1990s, but only recently became cheap enough to see general use.

Storage has become extremely inexpensive recently. In 2023 HDDs cost roughly 1−2¢ per GB and SSDs about 7−12¢ per GB. This has made it practical to keep almost any number of MP3s or pictures that an individual is likely to own indefinitely. Video files are large, but even at 8 GB or more, an HD movie can be stored on HDD for a less than a quarter.

Many companies such as Google, Amazon, Mega.nz, and Dropbox, offer various amounts of cloud storage at low, or no, cost. Sometimes this cloud storage is tied to specific uses such as keeping music or video purchased from the company, or keeping emails or other data associated with its applications. In other cases, consumers are free to use this space as they wish. Cloud storage comes with many different types of SLAs that relate to latency in access time, up-time, write times, location, and duplication of storage, and even whether the data is online at all. Costs range from 2−5¢ per GB per year for cold storage to $1.20 for high availability SSD.

This has altered the way users think about data. Rather than being something tied to a location or a specific machine, data is now a thing that users can expect to have instant access to from any of their electronic devices. In the past, if you wanted to read a book, listen to a song, or watch a movie, you had to own a physical copy. This made it useful to collect large libraries of content and data on physical media. Now, you can access most popular books, music, and video, on the web. You can subscribe to it, rent it, buy it, or steal it. There is less reason every year to actually own content or maintain your personal data on your own storage devices.

The CPU and various types of memory are located on a computer's motherboard. The motherboard, in turn, connects these components to inputs and outputs through a set of buses, discussed below.



Section 4.5. Input

Input devices allow computer users to provide instructions and programs to their machines. Patch cables were used in the earliest days of electronic computing which gave way to punch cards and paper tape. More recent types of storage devices such as magnetic tape, drums, and drives are used both to input programs to RAM, and to record the results as output.

Humans also need to provide input to computers in order to guide and use their programs. The most important classes of input devices for human interaction with computers are keyboards, which allow a user to enter text, and pointing devices, which allow a user to interact with a screen image created by the computer.

Computer keyboards have their antecedents in the teletype machines of the early 20th century. Computers in the 1940s and 1950s often used keyboards to punch tape or cards to be read by the computer. By the 1960s and 1970s **Printing Terminals** and **Thin Client Terminals** were in use for time-sharing mainframes.

Printing terminals were keyboards set up on stands that printed whatever a user typed, character by character, on tractor feed paper and then sent this as electronic input to the mainframe when the user hit the return key. It then printed the computer's response. In other words, the paper took the place of a screen.

A thin client was just a keyboard attached to a small monochrome television display. It works just like a printing terminal, passing text between the user and mainframe, but used a screen place of paper. By the late 1970s and 1980s the more familiar arrangement of a keyboard as a separate device attached to a computer to type instructions became standard.

The first mouse was developed at SRI (the Stanford Research Institute) and was patented in 1970. It was not very useful until **GUIs (Graphical User Interfaces**) became the standard way to interact with computers in the 1980s. The mouse is a type of pointing device in the same class as trackballs, touchpads, joysticks, trackpoints, and touch screens.

Microphones and cameras have also become more important recently. Voice recognition systems are good enough now that it is practical to dictate text to a computer instead of typing it. Unfortunately, we do not yet have a widely accepted way of controlling a computer using voice commands, although mobile device manufactures continue to work on the problem. Otherwise, the two main uses for microphones and cameras are communicating with other users (Zoom, for example) and creating content that can be stored, shared, edited, and mashed-up with various applications.

Scanners convert text and images printed on physical media (paper, for example) into digital form. The resulting files can be used just as any other image, or they can be passed to an **OCR (Optical Character Recognition)** program to convert any text they contain into an editable file. Scanning makes it possible to convert paper records and pictures that are subject to damage, expensive to store, and difficult to access into electronic files that can be indexed, searched, and stored, without degradation indefinitely.

Section 4.6. Output

Output devices allow computer operators to access the results of the instructions and programs they provide to their machines. The same type of storage devices used to give machines inputs, from punched cards to HDDs and SSDs, can also record their output. This output may be fed directly into other applications, but if users wish to interact with it, it needs to be given a more accessible form.

The most important classes of output devices are monitors and printers. Teletype and printing terminals combined both functions, but gave output line by line and very slowly. Early computer monitors in the 1960s and 1970s used **CRTs (Cathode Ray Tubes)** similar to television sets. These devices were monochrome, fragile, heavy, large, power hungry, and had low resolution. A number of other technologies found limited application including plasma displays, **LED (Light Emitting Diodes),** and vacuum fluorescent displays, but the CRT was overwhelmingly dominant.

LCD (Liquid Crystal Displays) use compounds that can be molecularly oriented using heat or electricity. The degree of molecular alignment affects the amount, and polarization, of the light that can pass thorough a layer of LC. By using an array of transistors to control the structure of LCs deposited at each individual **Pixel (Picture Element)** of a back-lit display screen, controlled amounts of differently colored light can be allowed to pass through each.

HD screen resolution is 1920 × 1080 pixels, arrayed roughly 250 microns apart in more than two million separate locations. Each must be independently, and rapidly, controlled to make an image. Fortunately, **TFT (Thin Film Transistor)** technology now makes this relatively cheap and easy to do. New video standards such as 4K/UHD offer $3840\,x\,2160$ pixel screens, and we are just beginning to explore how we can adapt these technologies for various forms of immersive **Virtual Reality** interfaces.

Hard copy output technology has also advanced. Early printers were like typewriters with a fixed character set. Only the characters built into the device (on keys, a daisy wheel, or type ball, for example) could be printed.

Dot-matrix Printers got rid of this limitation by using a printer head with a vertical line of ten or more pins that would move horizontally across a sheet of paper to print a line of text. Dot-matrix was an electromechanical technology in which the these pins would physically strike an ink ribbon placed in front of the paper. This meant that any character that could be approximated by a matrix of dots could be printed. This was an inexpensive way to print, but was also slow and low quality.

Ink-jet Printing become commercially viable in the 1970s. This used electromagnetically aimed streams of microscopic ink droplets to print characters. Ink-jets were also inexpensive and slow, but gave output that was a bit higher in quality than dot-matrix printers. By using multiple streams of ink, color printing was also possible with ink-jets.

The most widely used printing technology today is **Laser Printing**. By the mid 1980s, the cost of laser printers was still several times that of ink-jet or dot-matrix printers, but had come down enough to be within reach of individual users. The high quality and speed of output helped them gain general acceptance. Laser printers use a technology similar to Xerox copy machines.

In both cases, intense light in the shape of the image to be printed is directed at a printer drum. The light causes the drum to become electrostatically charged. The drum is then exposed to very fine toner particles made from plastic and carbon black which have been given the opposite charge. The particles are attracted to the charged parts of the drum which is then rolled over a sheet of paper. The particles are deposited and then fused to the paper using heat.

The difference between a Xerox machine and a laser printer is that the former uses light reflected from the physical object being copied to directly charge the printer drum, while the latter "writes" text and images as a set of small dots on the drum using a carefully controlled laser. By using multiple drums and different colored toner, color laser printing is also possible.

Although printers themselves are fairly inexpensive, the ink and toner they use are not. This is an example of a "tied product" strategy in which services or products needed to use a piece of equipment are provided exclusively by the manufacturer.

The latest output technology is **3-D Printing**. This uses a kind of ink jet guided in three dimensions instead of two to lay down successive layers of fine-grained polymer or other material which are then fused together by heat.

Sound is another form of output that computers can generate. In early computers, this was in the form of a "bell" or beep which indicated different types of errors or system alerts. Modern computers use sound cards to produce mono, stereo, and even surround sound output which can be fed to speakers, a headphone jack, and other pieces of audio equipment.

Recall that computer files are all digital. Sound, however, is analog (that is, it has a continuum of possible levels instead of a finite and discrete number). A digital audio file is converted to an analog audio signal using a **D/A Chip (Digital to Audio Conversion Chip).** Conversely, when sound is recorded on a digital device or sent over a digital network, it first must be converted into digital data. This is done with an A/D chip. For example, twisted copper pair telephone lines actually send an analog sound signal from user to user. Now, phone calls mostly go as packet data over digital connections and this is only possible because of D/A and A/D chips.

Section 4.7. Ports and Buses

The essential components of a computer come down to the CPU, the system memory, and the various input and output devices that are attached. These components are linked together by what are called **Buses** (from the Latin *omnibus*, meaning "for all") which are communications systems that transfer data between components inside and outside a computer, and between computers.

The fastest of these are the **BSB (Back Side Bus)** that connects the CPU to L1 Cache and the **FSB (Front Side Bus)** which connects the CPU to L2 Cache, which in turn connects to the RAM and slower buses that connect other system components. External buses take many forms and have varying speeds. The term “bus” encompasses both the physical part of the connection and the protocol it uses.

In contrast, a **Port** is a logical connection point. Thus, your computer might have USB port 1, USB port 2, serial port 3, and so on. Making a connection often requires that a special kind of software called a **Driver** be installed so that the computer understands the control systems and capabilities of the specific device connected to a port. **PnP (Plug and Play)** is a standard that facilitates the automatic installation of these drivers.

Video cards are very fast and data intensive, They use a fast bus such as **PCI (Peripheral Component Interconnect)**, PCIexpress, or **AGP (Accelerated Graphics Port).** Monitors connect to video cards using buses such as **VGA (Video Graphics Array)**, which is designed to send analog data to a CRT, **DVI (Digital Visual Interface)** and **HDMI (High Definition Multimedia Interface)**, which carries digital sound and other data in addition to digital video information. All of these video buses are relatively fast. Slower buses such as **SCSI (Small Computer System Interface)** and **SATA (Serial Advanced Technology Attachment)** connect storage and some other internal components.

External peripherals such as printers, external disks, CD, DVD, and flash drives, are connected with even slower buses such a parallel, serial, and **USB (Universal Serial Bus).** Pointing devices, keyboards, have low data requirements and can be connected with USB and even wirelessly using bluetooth. Routers and **LAN (Local Area Networks)** are connected together with Ethernet, various Wi-Fi protocols, and even cellular networks, all of which are relatively slow compared to internal bus connections.

Below we give a table that lists the approximate speed (bandwidth) of different types of buses. Speed is given in terms of the number of bits that can be transferred over the bus per second. Recall that it takes eight bits to make one byte.

Bus Types and Speed

Bus Type	**Speed**		**Bus Type**	**Speed**
Teletype Machine	50 b/s		Cable Modem	60 Mb/s
Morse Code Telegraph	210 b/s		Wi-Fi IEEE 802.11g	54 Mb/s
POTS Modem 12k	12 Kb/s		LTE Cell Phone	173 Mb/s
POTS Modem 56k	56 Kb/s		Ultra4 SCSI 2	320 Mb/s
2G GSM Cell Phone	14.4 Kb/s		USB 2.0	480 Mb/s
3G UMTS Cell Phone	384 Kb/s		Wi-Fi IEEE 802.11n	600 Mb/s
EDGE Network Cell Phone	474 Kb/s		Gigabit Ethernet (1000BASE-T)	1 Gb/s
Parallel Port	1 Mb/s			
Serial Port	1.5 Mb/s		PCI	1 Gb/s
ADSL	1.5 Mb/s		SATA 3.0	6 Gb/s
DS1/T1	1.5 Mb/s		AGP	2 Gb/s
Ethernet (10BASE-T)	10 Mb/s		FireWire 3200	3.1 Gb/s
DVD Controller (1×)	11 Mb/s		USB 3.0	5 Gb/s
Wi-Fi IEEE 802.11b	11 Mb/s		DVI	5 Gb/s
USB 1.1	12 Mb/s		HDMI 2.0	18 Gb/s
Bluetooth 4.0	24 Mb/s		PCI Express 1.0	64 Gb/s
HD DVD Controller (1×)	36 Mb/s		PCI Express 2.0	128 Gb/s
Blu-ray Controller (1×)	36 Mb/s		PCI Express 3.0	256 Gb/s
Narrow SCSI	40 Mb/s		400 MHz 64 bit FSB	26 Tb/s
DS3/T3	45 Mb/s		1666 MHz 64 bit FSB	107 Tb/s

Black = Wired Communications/Data

Blue = Wireless Communications/Data

Red = External and Internal Buses

b/s = Bits per second

Kb/s = Kilobits per second

Mb/s = Megabits per second

Gb/s = Gigabits per second

Section 4.8. A Final Note on Security

The **Intel Management Engine** is a hardware backdoor that has been built into all of its processors since 2008. AMD has a similar backdoor called **AMD Secure Technology**. These hardware-based backdoors are intended to allow corporations to update and manage the configuration of distributed networks of computers without needing the user's permission, without the user's knowledge, and without needing to go through the computer's OS.

Access to the Management Engine is controlled by encrypted credentials distributed to system administrators approved by Intel. Of course, Intel can also access the Managements Engine. Intel could also lose control of the keys to a hacker, or could be forced to give credentials for certain systems to the government. If your house has a backdoor, you have to depend on the good faith, courage, and competence, of any keyholder.

Intel's backdoor is well-known, but backdoors can also be built in secret. For example, it was discovered in 2010 that many Lenovo laptops were sold to the U.S. military had a chip on the motherboard recorded all the data and sent it to China. Huawei 4G and 5G chipsets in routers, base stations, and antennas, were found to allow Huawei access to global telecom networks where they were installed.

Cellphones almost always have hardware and software backdoors. These can be implemented in the operating system, may be mandated by the government, and cannot be turned off by the user. You should always assume your cellphone is insecure and that everything you do with it is exposed to the OS provider (Google, Apple, etc.), the phone's manufacturer (Samsung, Huawei, etc.), the carrier (AT&T, T-mobile, etc.), and governments (USA, China). Hackers can also use these backdoors, but at least they have to work at it.

Hardware is an attack surface that is very difficult to defend. To be secure, you have to have access to, and fully understand, the chip design. Then, you have to verify that the design was used to make the chips that actually ended up in your device. Once the chip package is closed, there is no way to know what is inside. You must physically track the chip from the time it is produced, to its installation in a motherboard, and then physically track the device with this motherboard from the manufacturer, until it is delivered to you. A substitution could be made at any point.

Section 4.9. Economics

Subsection 4.9.1. Increasing Returns to Scale

Production technology can display **Increasing, Constant,** or **Decreasing Returns to Scale**. (**IRS**, **CRS**, and **DRS**, respectively). This depends on the proportional effect of changes in the level of inputs have on the quantity of outputs. Formally:

Increasing Returns to Scale (IRS): $f(kx) > kf(x)$.

Constant Returns to Scale (CRS): $f(kx) = kf(x)$.

Decreasing Returns to Scale (DRS): $f(kx) < kf(x)$.

where:

$x \in \mathbb{R}^N_+$:	bundle of inputs
$y \in \mathbb{R}^1_+$:	level of output
$k \in \mathbb{R}^1_{++}$:	strictly positive scalar
$f : \mathbb{R}^1_+ \Rightarrow \mathbb{R}^1_+$:	production function describing a process that turns inputs into outputs

Many types of ICT hardware manufacturing display IRS. In some cases this is due to a type of first copy cost, since designing a complicated electronic product is expensive, and must be done regardless of how many units are manufactured.

The complexity manufacturing hardware, chips, and components, also results in companies getting better at it over time. Employees get better at doing their jobs as they get more practice, management and engineering staff learn ways to streamline, or otherwise improve the production process, as they observe it in action. This phenomenon is called **Learning by Doing**.

For example, making chips requires that the dies and templates used to make each layer be precisely aligned, the chip forge be isolated from vibration, and be kept in a clean room free from contaminants. The first batch of chips produced typically have a high failure rate. Over time, problems are tracked down, and the success rate goes up. This means that the same inputs yield a larger amount of useful product over time. The more inputs you use, the faster you learn, and so the process displays IRS.

Subsection 4.9.2. Binary and Boolean Math

Recall that base-ten numbers actually represent place values:

$$2509 = 2\times10^{3} + 5\times10^{2} + 2\times10^{1} + 9\times10^{0} = 2000 + 500 + 00 + 9$$

Base-two numbers are similar. The differences that the places are powers of two instead of ten. Generically, consider an eight bit byte: Byte = (b1, b2, b3, b4, b5, b6, b7, b8). Then,

$$\text{Integer}_2(\text{b1, b2, b3, b4, b5, b6, b7, b8}) =$$
$$\text{b1}\times2^{7} + \text{b2}\times2^{6} + \text{b3}\times2^{5} + \text{b4}\times2^{4} + \text{b5}\times2^{3} + \text{b6}\times2^{2} + \text{b7}\times2^{1} + \text{b8}\times2^{0}$$

For example:

$$\text{Integer}_2(0010\ 0101) =$$
$$0\times128 + 0\times64 + 1\times32 + 0\times16 + 0\times8 + 1\times4 + 0\times2 + 1\times1 = 32 + 4 + 1 = 37$$

and so:

0000 0000 = 0

0000 0001 = 1

0000 0010 = 2

0000 1010 = 10

0110 0100 = 100

1111 1111 = 255

0000 1001 1100 1101 = 2509

Binary addition and subtraction depend on **Boolean Algebra**. If I add 5 and 7 in base ten, I get a number that is larger than any of the allowed place-holder values (0 through 9). Thus, as I add 5 to 7 and I get to 9, have to carry a 1 to the ten's place. In other words, I added 3 to seven, and then cycled back to 0 in the one's place, and added 1 to the ten's place. I then continued to add the final 2 to the 0 in the one's place to get my solution: 12.

Base-two has only two place holder values: 0 and 1. Thus, when I add 1 and 1, I cycle back to 0 in the "one's" place (more accurately, the 2^{0} place), and add 1 to the "two's place". Using this approach, I can add two binary numbers as follows:

$$\begin{array}{rl} & 0000\ 0010 = 2 \\ + & \underline{0000\ 0110 = 6} \\ & 0000\ 1000 = 8 \end{array}$$

Multiplication and division are similar. Base-ten is what we are used to, but it has no special place in mathematics.

It is also useful to have a sense of the magnitude of the powers of two, and we finish the chapter with the folowing table:

Possible Values for Byte-Strings of Various Sizes			
Bytes	**bits**	**Number of Values**	**Examples**
1	8	256	Byte, ASCII, UTF-8
2	16	65.5×10^{9}	UTF-16, Port Numbers
4	32	4.3×10^{9}	Unicode, UTF-32, IPv4, x86 chip architecture.
8	64	1.8×10^{19}	AMD64 chip architecture
16	128	3.4×10^{38}	IPv6, SHA126
32	256	1.2×10^{77}	SHA256, AES256, ECC 256
64	512	1.3×10^{154}	Less common hashing and encryption systems
128	1024	1.8×10^{309}	kilobit, RSA1024

Note that it is conventional to call 1024 bits a megabit (kb), and 1024 bytes a kilobyte (kB). More generally, $2^{10} = 1024$ is described as a thousand, $(2^{10})^2 = 2^{20} = 1{,}048{,}576$ as a million, and $(2^{10})^3 = 2^{30} = 1{,}073{,}741{,}824$ as a billion, when referring to bits or bytes.

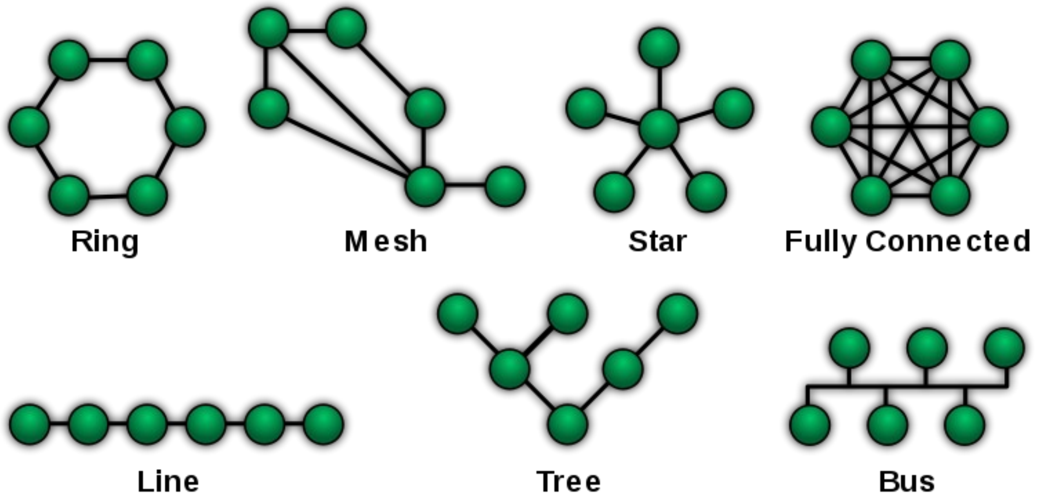
It is also noteworthy that a single kilobyte ($8 \times 1024 = 8192$ bits) can take $2^{8192} \approx 10^{2466}$ values. To put that in perspective it is estimated that there are 1.5×10^{18} gallons of water in the Pacific Ocean, 10^{38} stars, and 10^{82} atoms in observable universe.

Chapter 5. Networks and Infrastructure and Architecture

Section 5.1. Network Basics

Telecommunication and computer Networks consist of **Nodes**, and **Links**. arranged is various topologies. Most communications networks are bidirectional, although some allocate capacity asymmetrically (ADSL allocates more bandwidth for download than upload, for example.)

Graph Theory describes networks in terms of **Vertices** and **Edges**. An edge is defined by a pair of vertices, A and B. If these pairs are unordered {A, B}, then the graphic is undirected, and travel on the edge in both directions is allowed. If these pairs are ordered (A, B), then the graph is directed, and travel is only possible from the first vertex, A, to the second, B. Basic network topologies include the following:



Section 5.2. Distributed Systems

Most databases used by large companies and government agencies are not single centralized systems. Keeping valuable data all in one place is dangerous for a number of reasons. Instead, Visa/Mastercard, banks, insurance companies, Amazon, and so on use **Distributed Systems** to hold copies of their data in several locations. The major advantages are:

Redundancy: Destroying one data center does not destroy the data.

Accessibility: Data centers may be subjected to denial of service attacks. Congestion or damage on the internet backbone can also cause some data centers to be slow to respond due to latency or be entirely cut off (partitioned) from the rest of the system.

Cost: Having data available locally for customers to use can reduce internet traffic and spread processing load more optimally.

Distributed Systems rely on a network between data nodes, and the study of distributed systems is a major subfield in computer science. Building such systems to be efficient and robust turns out to be difficult. This is largely because there are several key features we would like distributed systems to have:

Consistency: Every read receives the most recent write or an error.

> (When a user contacts one of the databases in the network, he either gets the current data, or the system returns an error that lets the user know that the system does not know the latest data.)

Availability: Every non-failing node in the system is available for queries.

> (If a data node is the network has not crashed, it can be contacted by users, and will give a response to any querry. It will not necessarily be correct, however, in the sense that it reflects the most recent update of the data.)

Partition tolerance: The system continues to operate in the presence of network failure(s) that result in any number of messages being delayed or lost.

> (The system will maintain consistency, even in the event of a partition of the network that prevents certain nodes from communicating from one another.)

Unfortunately, there is a basic result in computer science called the **CAP Theorem** (Gilbert and Nancy 2002) which formalizes a conjecture by Brewer (Brewer 2000) and says that a distributed data store can at best guarantee two out of three of these at one time.

It would also be nice if a distributed data systems could satisfy:

Termination: Every component will eventually decide on a value.

> (Every node in the system will eventually decide on how its data should be updated. No node will every be **Halted** and unable to make an update.)

Safety: Different components will never decide on different values.

> (The system will never fail to come to a consensus and end up disagreeing about the correct current state of the data.)

Unfortunately, the **FLP Theorem** (Fischer, Lynch, and Paterson 1985) tells us that both termination and safety cannot be satisfied in an **Asynchronous Distributed System**, within a bounded time, that is robust to the existence of at least one faulty component. A distributed system is asynchronous if its system clocks may be inaccurate or out of sync and messages may be delayed for unknown and arbitrarily long periods of time. In other words, if the system is part of the real-world.

These two **Impossibility Theorems** on networks apply both to cloud-based, redundant data systems run by TDIs, and to blockchains, which are decentralized and require no TDI.

This leaves us having to make compromises among these criteria. Are we willing to sacrifice some availability for a guarantee of consistency? Are we willing to allow long delays while systems try to terminate or recover from a partition, in order to guarantee safety? Another direction is to try to build more robust systems where partitions or unbounded latency are possible, but unlikely. Malicious actors called **Adversaries** are also aware of these fundamental impossibility theorems and the **Attack Surfaces** they open.

Subsection 5.2.1. Network Types

Physical **Nodes** in a communications network can be anything from routers, computers, or mobile device, to printer, smart speakers, or connected TVs. Networks, however, only see a collection **Network Interface Controllers** (**NIC**) that are built into these devices.

Devices may have more than one NIC, and each is assigned a unique identifying number called a **Media Access Control (MAC) Address**.

MAC Addresses are assigned by device manufacturers, and this creates a privacy risk. Cell phones and laptops broadcast their MAC as they search for networks to attach to. Your device continually advertises its presence, and this allows it, and its owner, to be tracked. More recently, some manufacturers have added MAC address randomization to their devices, and even allow users to choose MAC addresses on the fly. Nevertheless, MAC addresses are a largely over-looked security risk.

When a device (really a NIC) joins a network, a router assigns it an internal **Internet Protocol** (**IP**) **Address** using **Dynamic Host Configuration Protocol (DHCP)**, The IP address is an internal endpoint to which network traffic can be addressed, while the MAC address is more like a serial number or name for a NIC.

The Physical **Links** in a communications network are both wired and wireless. Wired connections types include CAT5 and CAT6 Ethernet, cable, fiber, and copper, twisted pair, phone line. Wireless connection types include 802.11 based Wi-Fi, Bluetooth, and Zigbee, PCS, and satellite radio. The physical characteristics, such as the range, and capacity, of these link types vary. From a network standpoint, they are all the same.

There are many types of networks used in communications systems. The most common are the following.

LAN (Local Area Network): An interior network of computers and devices that share data and interface with one another. a router or gateway typically stands between a LAN and the WAN. These gateways manage internal traffic among nodes within the LAN, and serve as a proxy to for traffic between external sources and interior nodes. The final destination within a LAN for external data is invisible to outside world.

WLAN (Wireless Local Area Network): A LAN that uses a wireless system (typically, one of the IEEE 802.11 specifications) to link nodes in the internal network.

WAN (Wide Area Network): The broader Internet that lies outside of any given LAN. Routers/ gateways are assigned **(Internet Protocol) IP Addresses** by an **Internet Service Provider (ISP)** that identifies them as a specific Internet endpoint.

P2P Network (Peer-to-Peer Network): A network in which all nodes have equal status, and there is no cental server or hub. Often these networks are *ad hoc* in the sense that nodes come and go at will, and form connections randomly to a small number of nodes. They are robust in that they have no central point of failure, and no central list of node IP addresses that might allow censorship or generalized attacks on the network. Blockchains often use a P2P approach to disseminate user transactions, and newly mined blocks, throughout the network of nodes and miners.

Server/Client Network: A network in which a single node, called the server, forms connections with all the other nodes, call clients. All traffic between clients, and to any outside network, must go through the server.

Mesh Network: A network in which nodes link opportunistically to whatever nodes they can reach, and then serve relays between nodes that are not able to contact each other directly.

Subsection 5.2.2. Domains and IP Addresses

Domain names are basically human-readable nicknames for IP addresses. The real address needed to find a location on the Internet typically uses the **IPv4** standard, and consists of a 32-bit number, *129.193.255.098*, for example. Each of the four parts in this address is an 8-bit number and so can take values between 0 and 255.

Users access Internet content by typing a **URL (Universal Resource Locator)** such as:

timewaster.com/LOLcat2983/

into a browser. The first two parts of this are a **Domain Name**, while the last is the name of a specific HTML file or other object that exists on the timewaster.com server.

When the request gets to your router, it sends a query to a **Domain Name Service (DNS)** server. These servers keep tables with the IP address to which each domain name is currently pointed. (Actually, this information is divided over a network of several servers, but the effect is the same.[12]) These associations can be changed by the owners of domain names when they move to different servers or different ISPs. The DNS server then sends back the current IP address which your router uses to create a properly formatted packet requests to send over WAN directed at the target server.

ICANN (Internet Corporation for Assigned Names and Numbers) was established in 1998 to allow private individuals to purchase the exclusive right to use domain names. Addresses in some **TLDs (Top level Domains)** such as *.com*, *.org*, and *.net*, can be purchased by anyone, while others such as *.edu*, *.mil*, and *.gov* can only be purchased by qualified groups. Each nation also has its own TLD such as *.us*, *.fr*, and *.uk*.

Private individuals can submit applications to ICANN to run new TLDs. Submitting an application costs $185,000, and if more than application is received, ICANN awards the TLD either by arbitration or auction. You can now purchase a URL in such TLDs as *.attorney*, *.beer*, *.church*, *.dating*, *.email*, *.fail*, *.pink*, *.porn*, *.sexy*, *.shoes*, *.vodka*, and *.wtf*, for example, Domain names for most TLDs can be purchased for $10 to $30 per year, although *.car*, and *.rich* about $2,000 and *.na* (Nigeria) and .th (Thailand) cost around $5,000 per year.

There are at total of 4.3 billion possible IP address available under IPv4. A block of 65,536 addresses between *192.168.0.0* and *192.168.255.255* are private addresses that routers are allowed to assign to nodes within the LANs they create. These addresses are not visible outside the LAN, are reused millions of times inside LANs. No node on the WAN is allowed to use one of these addresses externally. (This leads to the saying that "*there is no place like 192.168.0.1.*")

Four billion addresses may seem like a lot, there are billions of connected IoT devices that require a direct connection to the Internet, and more coming online very day. We are now moving to the **IPv6** standard which allows for 128-bit (16 bytes) addresses giving 3.4×10^{38} possibilities.

You can see how central the DNS is to the Internet. As a result, users must be sure that they trust the DNS they access. The worst case is that a Trojan or virus changes your network setting to send you to a captured DNS. This kind of **DNS Spoofing** allows a hacker to give you an IP address of a fake site when you type in the URL of your bank, for example. A slightly more subtle type of attack is when a legitimate DNS is cracked using an exploit and the part of the look-up table is rewritten. This is called **Cache Poisoning.**

Subsection 5.2.3. The Last Mile

The backend of the Internet is owned by a variety of companies and is always being upgraded. The so-called **Last Mile** that connects this infrastructure to each subscriber's router is a different

[12] DNS servers are owned by private companies that provide their look-up services for profit. Most consumers do not notice this because their ISP also runs DNS servers, and bundles their cost in with their service fees. You can choose your own DNS through your browser configuration if you wish.

story. Until recently, the last mile had to be traversed using legacy twisted copper pairs, and coaxial cable. Upgrading the last mile requires using public rights of way, digging up local streets, and installing new connections to each household. As you might expect, this is an expensive process, and requires permits and permissions from states and local jurisdictions.

In the 1980s the only connectivity option was the telephone modem which allowed connections of up to 56 Kb/s. This was expensive, low quality, and tied up your phone line. Eventually, ISPs developed **Digital Subscriber Lines** (**DSL**) that used the same twisted copper pair **Plain Old Telephone Service** (**POTS**) connection to carry two analog data streams. The first used frequencies in the normal human hearing range (about 20 Hz to 20 kHz) and carries voice. The second uses frequencies outside the range of human hearing (roughly 26 kHz to 1100 kHz) to carry data. This approach allows voice and data to be carried at the same time, Various implementations such as **ADSL (Asynchronous DSL)** allowed data transfer rates between 1.5 and 24 Mb/s.

Almost all households had a POTS connection, and most also had a coaxial cable TV connection. Beginning in the last 1990s cable broadband started to become available, It is possible to make good connections over hundreds of miles with transfer rates of up to 70 Mb/s *per channel* using coaxial cable compared to the one or two miles and 24 Mb/s *in total* that twisted pair lines offer.

In the last fifteen years or so, AT&T, Google, and other providers, have at last started to install new fiber optic connects to households. Fiber has almost unlimited capacity, and bottlenecks only occur at the local neighborhood switches, and entry points to the backbone.

Less important ways of covering the last mile include satellite service and cellular networks. As of 2019, satellite service provides about 10-30 Mb/s and costs about $2-6 per GB of data consumed. Elon Musk's Starlink satellite network may change this.

Cellular LTE 4G[13] service in the US costs around $10-15 per GB, and provides 15-30 Mb/s, depending on the quality of the connection to the local tower. For users in isolated locations, these may be the best or only options. By way of comparison, cable and fiber provide 20-1000 Mb/s at a cost of $5¢-20¢$ per GB.

This gives us four alternatives for users to cover the last mile between their router and the WAN. Satellite and wireless, however, are expensive and slow, and so are only used when other options are not available. For most consumers, the last mile is covered by a duopoly consisting of a legacy telephone company, and the local cable TV provider.

Subsection 5.2.4. The Backbone

[13] Note that "5G" is not a communications protocol standard like LTE, or 802.11. It is more of a marketing term that means something like "you may get higher PCS transfer speeds how and when we deploy more modern technology."

After data leaves your LAN and traverses the last mile, it ends up at a **Neighborhood Switch** controlled by your ISP. This collects all the local traffic and sends it back to one of your ISP's **PoP (Points of Presence).** A PoP is a set of routers and switches, usually located at a local telephone or cable company's facility, with access to a fast link to the Internet backbone.

From there, the data travels between high capacity **Core Routers** using fiber backbone links for the most part. The data may stay on your own ISP's part of the backbone, or may need to use routers and backbone owned by other ISPs to get to its destination. This is accomplished though **NAPs (Network Access Points)** and **IXPs (Internet Exchange Points)** which provide interlinks between various subnets owned by different providers.

Eventually, the core routers deliver the data close enough to its destination (maybe going through another NAP) that it can be handed off to a PoP owned by the ISP that provides access to the end-user. From there it is sent to the destination LAN, and to the targeted node within.

Larger companies and content providers such as Netflix and YouTube serve as their own ISPs and connect directly to backbone. Data centers serving government organizations, large companies, or large user groups may be located in, or adjacent to, facilities owned by providers with direct access to the backbone.

The more general question of who owns and maintains the Internet is an interesting one. ISPs own and maintain the links to the PoPs and to the point that their networks interlink at NAPs and IXPs.

ISPs can be small, such as regional telephone companies, or large, like Comcast. Some ISPs specialize in providing service to a specific company or group such as a government organization, a municipality, or campus. In any event, the cost of building and maintaining these switches, servers, and communications lines to the backbone, are paid by some ISP.

The **Internet Backbone** itself consists of high capacity core routers, NAPs, IXPs, and the (mostly optical fiber) connections that link them all together. This infrastructure is owned and maintained by a variety of government agencies, academic organization, telecommunications companies, and other commercial entities.

The Internet backbone is a large, overlapping, redundant, network of high speed routers and data connections, whose ownership is spread over many different companies and organizations. This is by design since it gives the Internet resilience and reliably, while preventing centralized control by any single authority.

ISPs must use one another's networks to serve their customers. Traffic between users who have different ISPs must at least transits both networks, and mostly likely, other networks in between. There are two main approaches to facilitating network sharing:

- **Peering Agreements**: Large ISPs and backbone providers allow one another free transit on their networks, This is because they expect to take approximately as much traffic from, as they put into, the other networks. There were only seventeen companies who had networks that are comprehensive enough to make such peering agreement with one another as of 2020. These include major US and foreign carriers such as AT&T, Deutsche Telekom, NTT Ltd., Orange Sprint, Tata Communications, Telxius, and Verizon. These companies are called **Tier 1 Network Providers.**

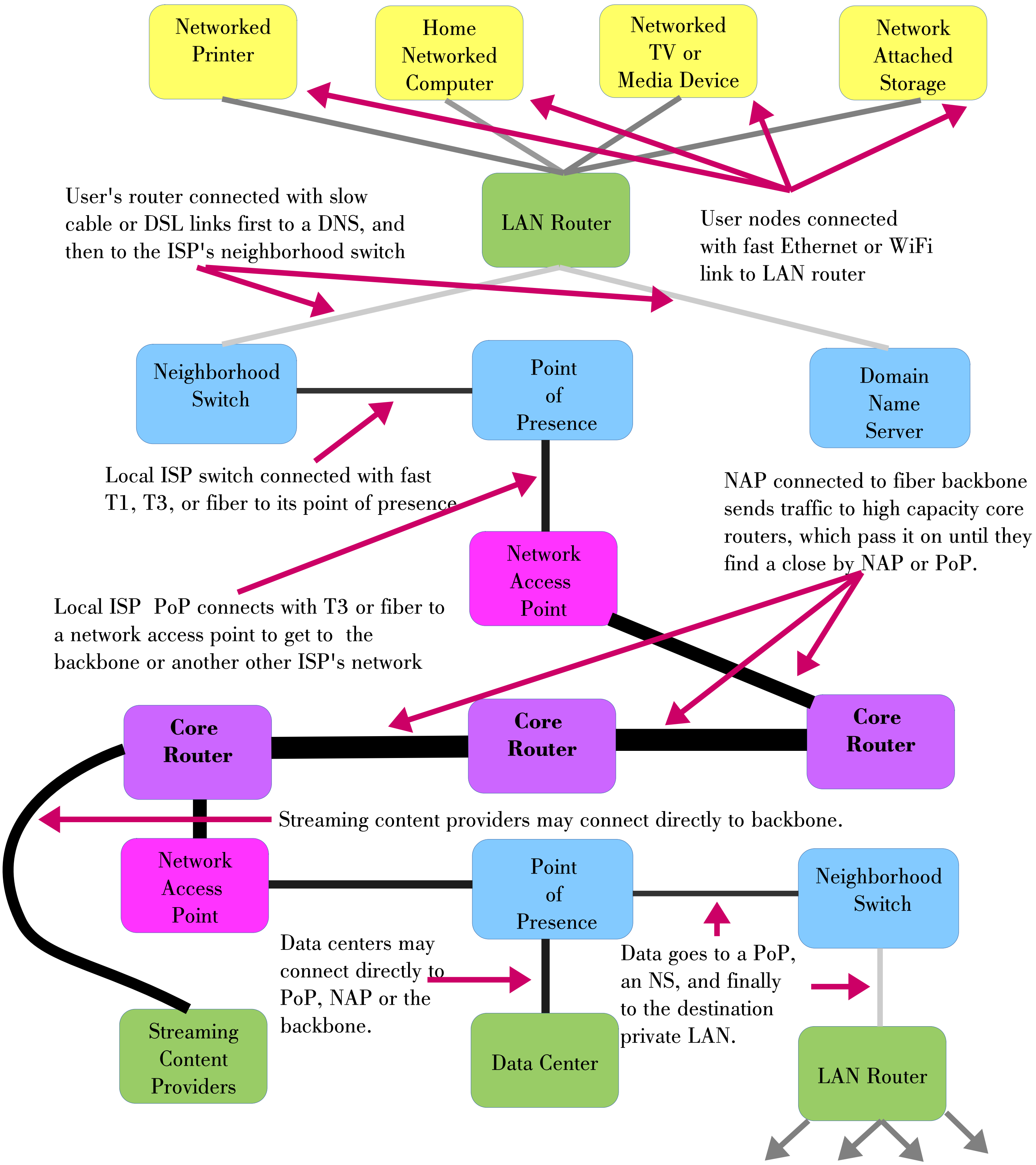
- **Transit Agreements:** For smaller ISPs, traffic is likely to be asymmetric. These ISPs sign transit agreement with one another and tier 1 provider to measure how much data goes in and out of NAPs and IXPs, and stipulate monetary payments to cover any imbalance.

Unfortunately, the decentralization and freedom of the Internet are under attack from many directions at present. Some countries put up national firewalls (such as the **Great Firewall of China**) that prevent access the any URL or website the government finds offensive. This can be done either by controlling the choke-points where internet traffic enters or leaves a country, or by setting up monitoring and filtering infrastructure at ISPs or in separate government sites.

Governments can also require ISPs, content providers, gateways, search engines, and/or social media companies to filter, censor, or report content and usage. More recently, private companies have taken it upon themselves to filter the internet and decide who should have access to the communications platforms and services they provide.[14]

[14] The Freedom House, the OpenNet Initiative, Reporters without Borders, and other NGOs concerned with internet freedom continuously monitor and evaluate how different countries rank in this area. Perhaps not surprisingly, couturiers like China, Cuba, Iran, Saudi Arabia, and Syria have been declared "Enemies of the Internet" for a very long time. More recently, Pakistan, Russia, and India have joined this list along with both the United Kingdom and the United States. Australia, France, South Korea, and Norway, among others, are currently under surveillance for this designation.

Basic Architecture of the Internet



User's router connected with slow cable or DSL links first to a DNS, and then to the ISP's neighborhood switch

User nodes connected with fast Ethernet or WiFi link to LAN router

Local ISP switch connected with fast T1, T3, or fiber to its point of presence

NAP connected to fiber backbone sends traffic to high capacity core routers, which pass it on until they find a close by NAP or PoP.

Local ISP PoP connects with T3 or fiber to a network access point to get to the backbone or another other ISP's network

Streaming content providers may connect directly to backbone.

Data centers may connect directly to PoP, NAP or the backbone.

Data goes to a PoP, an NS, and finally to the destination private LAN.

Section 5.3. The Internet of Things and Embedded Systems

Devices from rice cookers with fuzzy logic, washing machines, and dishwashers, to traffic signals, industrial robots, and vehicles, are “Smart” as because they are built around an **Embedded System**.

Embedded System: A simple, low cost, low power, computer built into a mechanical or electronic device to control its function.

These systems usually run simplified versions of Linux, C++, or scripting languages like Python or PHP. Some embedded system have no user interface at all, and simply do a task without any user input. Others have a few buttons or dials, a touch screen, or have a virtual interlace that may be very elaborate, but must be accessed via a web connection.

Examples of devices with embedded systems include digital watches, rice cookers with fuzzy logic, washing machines, and dishwashers, traffic signals, industrial robots, and vehicles. The Raspberry Pi and the Arduino microcontroller, in its many variants, are examples of standalone systems that are similar to those that are embedded in devices.

Smart devices can be programmed do to things contrary to a user's interests, or to disobey a user's wishes. For example, anti-alcohol interlocks on car ignitions disobey drivers' wishes, but probably for good reasons, and hardware implemented copy prevention and content filters have both good and bad aspects. When smart devices operate in isolation, and do not communicate with the outside world, these dangers are generally small. You can always get rid of an offending device.

Internet of Things (IoT): A network of physical devices that can access the Internet without human intervention.

Most smart devices are also connected. Connected devices are able to autonomously share the data they gather with one another, and communicate with their manufacture, and the broader Internet. The motivation is to coordinate their actions, and better anticipate the needs of the owner. Users are generally unaware of the type and volume of information collected, and, have very little control short of turning a device off.

Connected devices also present a vulnerable attack surface that is behind the router, and so within the trusted LAN. User’s seldom change the default passwords on such devices, nor update firmware to fixed exploits. Many such devices are captured by hackers and added to botnets that send spam, and participate in DDoS attacks. Some devices are configured so that the manufacture can push updates to a system within a LAN without the knowledge or permission of the owner. Such updates can change the way a device functions completely, including how it collects and shares data.

In Chapter 2, we discussed some the concern raised by connected devices. One way to frame the problem is that the devices you own are not your agent. In some cases, they are like children who lack a filter to prevent them to repeating things they shouldn't. They are neither for you nor agist you, but they can't keep a secret. In other cases, devices are agents working for a mostly benign master who only wants to make some money by anticipating your needs.

How great a threat to privacy, security, and personal freedom the Internet of Things turns out to be is still unknown. From a technological standpoint, however, almost anything we do are say will soon be seen and heard and so may be recorded and transmitted by some device. Such devices may be turned against us by marketers, employers, coworkers, hackers, hostile governments, terrorists, or even law enforcement agencies. On the other hand, they may make sure the toast and coffee are already made by the time we get out of the shower. So, there's that …

Section 5.4. Economics

Subsection 5.4.1. Hold-Up

Hold-up is a situation where one agent has control over an indispensable part of the joint process, especially an economic process. An agent in this position can demand all, or almost all, of the net profits or benefits from the joint project. If the other agents refuse, they get zero profit. Thus, if the hold-up agent leaves them any profit at all, they are strictly better off, and should therefore agree.

For example, suppose a firm wanted to construct a road or power transmission cable. It would have to acquire all the land, or at least the rights of way, along the planned route. Suppose that the firm had acquired rights from every landowner but one. The remaining landowner could demand a price so high that the firm would be almost indifferent between agreeing and making almost no profit, and refusing and making zero.

Of course, all the landowners are initially in the same position, and each may hold out hoping that he will be this lucky last landowner. Knowing this will happen, the firm would probably decide not to start the project at all. The landowners would miss out on selling their land at a more reasonable, but still lucrative price, and the firm would not make any profit at all. Everyone is worse off.

ISPs are like these landowners. Content companies make considerable investments in programing, infrastructure, and customer acquisition. The ISP, however, has monopoly control over the last mile. If allowed, it can demand extra fees from content providers who have made these sunk investments. Content companies might agree in the short run, but would not find it profitable to continue to produce programming. Any profits from new investments would be similarly appropriated by the ISP.

When potential partners in value chains like these cannot sign binding agreements to prevent hold-up, the result is that markets are abandoned, new products are not developed, and innovation and economic growth is slowed. This is one of the main reasons that governments regulate communication and transportation providers, and other industries in a position to engage in hold-up.

Subsection 5.4.2. Lock-in

Lock-in is a situation in which agents have to make decisions in the present that will be difficult or expensive to reverse in the future. If agents are myopic, or uniformed, they may choose an alternative that is most attractive in the short-run, and neglect this put them in a vulnerable position in the longer run.

For example, if you marry someone, it is costly to undo your action. If you have kids, you are locked-in even more strongly. Investing in **Specific Human Capital (**skills or knowledge that are only of value to your current employer) produces employment lock-in. Choosing a social media platform locks you in because you develop a social network is difficult to move to a different platform.

It is important to distinguish switching costs from sunk costs. Sunk costs are unrecoverable expenditures or investments. Lock-in, on the other hand, exists when a flow of benefits from a decision (choosing a platform, for example) grow over time, and can only be recreated on a different platform at some cost, over time. The switching cost is the net loss, discounted over time from changing platforms or making a different decision.

The ICT sector is rife with lock-in. For example:

- **Operating systems**: Once you choose an operating system, you develop expertise in its use, buy software that is compatible, link to compatible systems, and so on. The more time passes, the more difficult it is to switch to an alternative OS.
- **Hardware**: Computers, peripherals, embedded systems, and devices become part of larger information systems and so must work together smoothly. Sticking to the same hardware also means less need to learn how to configure, troubleshoot, and integrate new types of machines. This is one reason people choose to use only Apple, or only Microsoft produces.
- **Ecosystems**: Ecosystems such as those provided by Google, Apple, Microsoft, and others offer users smooth integration and interoperability of data and applications. It is difficult and expensive to switch from one to the other, or to mix elements of these ecosystems together.
- **Social Media and SaaS**: Facebook, Twitter, Sales Force, and other SaaS companies keep data in proprietary formats. If a company wanted to change vendors, extracting its data, and building a new software system around it is an expensive and difficult task. In addition, employees get used to the work flow and interfaces of these proprietary systems. This also makes it costly to switch systems. Such platforms also connect you with others, and leaving the platforms generally breaks those connections. The longer you use a platform, more costly it becomes to leave it.

- **Loyalty Programs**: One way to create artificial lock-in is a loyalty program. Examples include frequent flier miles, and stamps on your Subway or Starbucks rewards card. Eating at Subway everyday results in free lunch when the card is filled. If you stop eating at Subway, your points never hit the threshold, and so are wasted. In turn, this creates a mild lock-in that discourages you from starting over at a different lunch place and accumulating rewards there.

From an economic standpoint, the company and customer are playing a game (although the customer may not be fully aware of this fact). The company wants to sign up as many users as it can. In the early stages of this relationship, the company wants the customer to be a happy as possible.

At some point, the company will decide that it should turn from building a customer-base, to exploiting it. The company raises prices, imposes terms of service that work to its advantage, leverage user data in new ways, etc. Provided what the company leaves on the table for its customers exceeds the flow of benefits from starting over at an alternative platform. costumers stay, and the company profits. If a company can somehow raise prices for existing customers, while offering cheaper service to new ones, it can start the exploitation phase even sooner.

A sophisticated consumer would anticipate this, and evaluate a platform over the foreseeable long term rather than comparing only current costs and benefits. If consumers did this, companies would be forced to offer guarantees of future behavior or else have consumers choose competitors who did.

One way to think about this is as a **Rational Addiction** problem. Suppose you are offered an addictive drug at a party. Sounds fun, right? The dealer may offer you a low price. However, you notice that he charges higher prices to his already addicted customers, Becoming addicted puts you at the mercy of the dealer. If you can't be sure he will choose not to take advantage of his power, or confident that market forces will prevent him from doing so, you would be well-advised to find an alternative.

By the way, drugs are bad, and you shouldn't do them. This logic applies to any type of consumption where your utility per-unit increases as you consume more. For example, tobacco and alcohol, running and fitness, gaming and sports, therapy and religion.

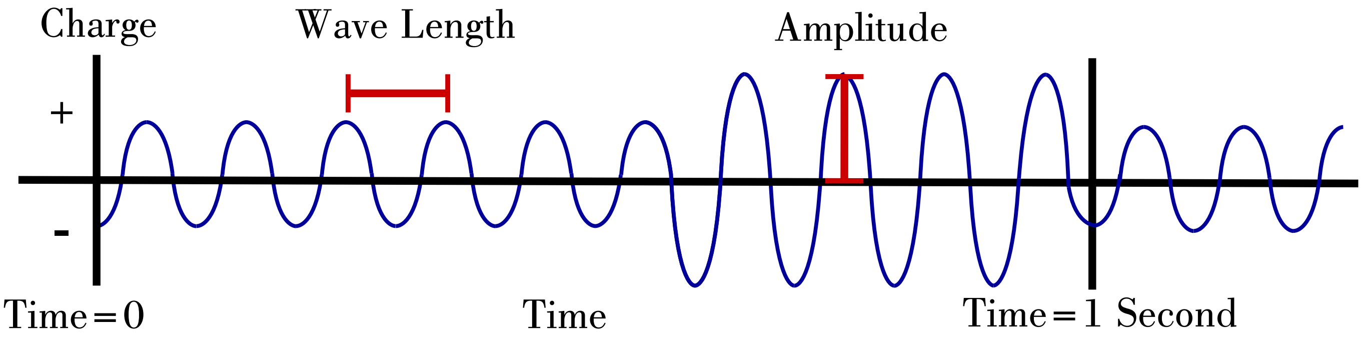
Chapter 6. Wireless, Wired and Spectrum Basics

Section 6.1. Signals and Frequency

This section gives a very superficial overview of **Electromagnetism** as it relates to communications. This is important because the physics involved impose constraints that must be respected by policymakers, businesses, and other spectrum users, and thus, impact of the shape of the economic questions.

The **Electromagnetic Spectrum** includes radio, microwaves, infrared, visible and ultraviolet light, X-rays, and gamma rays. In all cases, electromagnetic radiation is transmitted via photons moving out from the transmission source at the speed of light (just under 300,000 kps).

The difference between the various parts of the electromagnetic spectrum is the frequency of the wave. “What wave?”, I hear you ask. Well, a **Photon** is both a particle and a wave as it turns out. If you want to know more, read Physics for Dummies. So, moving right along … The photons that transmit electromagnetic waves oscillate at certain frequencies. They go from positive to negative and back again in a regular sinusoidal pattern. A single **Wave** is measured from one maximum to the next maximum. The figure below illustrates this:



If a wave oscillates 10 times a second, as in the example above, then we say that signal has a **Frequency** of 10 **Hertz** (abbreviated 10 Hz). Since the wave propagates at about 300,000 kps. A photon in the wave shown above travels 30,000 km by the time it makes a complete cycle from maximum to maximum. In other words, a 10 Hz signal has a **Wavelength** of 30,000 km. You can see that this implies that wavelength and frequency have an inverse relationship.

Signals also can be sent with various levels of power. If you put more energy into creating a signal, both the positive and negative extremes increase in magnitude. The difference between the positive maximum and zero (that is, half the total oscillation of the wave) is called the **Amplitude** of the signal.

Intervals of the spectrum are allocated to various uses. For example, AT&T might have the rights to an interval of 10 MHz, called a **Channel**, from 800 MHz, to 810 MHz in a certain region of the US to use for cellphone communications, or an AM radio station might have a 10 kHz interval between 1030 kHz and 1040 kHz to broadcast its programing. The difference between the upper and lower frequencies of a Chanel is called the **Bandwidth.**

You have probably heard the word bandwidth used to describe the amount of data that can be transmitted per second. This use has come into being because it is closely tied to bandwidth in the radio frequency sense.

A theorem due to Claude Shannon tells us that the amount of information that can sent is proportional to the bandwidth that is allocated to the task. What is surprising about this is that it does not matter where in the spectrum the bandwidth is. Having 10 MHz band in the low frequency part of the spectrum is just as good from an informational standpoint as 10 MHz in the super high frequency area. Bandwidth is bandwidth.

Frequencies in the 300 kHz – 300 MHz range have certain highly desirable technical properties. This **Beachfront Spectrum** is used for AM, FM, and CB radio, VHF television channels, pagers, cordless telephones, alarm systems, door openers, among other things. This is only 10% as much spectrum as the super high frequency bands in the 300 MHz – 3 GHz range (sometimes called the **Affordable Housing** of the spectrum), used for PCS, UHF television channels, Wi-Fi, and GPS satellites. There is only 1% as much beachfront spectrum as there is in the 3 GHz- 30 GHz range, used for things like Wi-Fi, point-to-point microwave.

The point is that only a small fraction of the available spectrum is suitable for the most valuable types of wireless communications. Given Shannon's Theorem, allocations fo the limited information carrying capacity of beachfront spectrum must be carefully considered, and on an ongoing basis.

Section 6.2. Wireless Communications

A wireless signal is produced by passing a current that alternates between positive and negative charges at the desired frequency through an antenna. This produces an electromagnetic wave that eventually encounters another antenna tuned to the same frequency that absorbs the energy that is contained in the wave. The receiver then amplifies and processes the signal, and delivers it to the final user.

Broadcasting a signal is similar to speaking to a crowd. If there are no obstacles between the speaker and the listeners, it is easier to hear what is being said. However, if there is brick wall between the two, it can be very difficult. Two things are happening here.

- First, some of the sound is reflecting off the wall and bouncing back towards the speaker.
- Second, some of the sound energy is being absorbed by the wall (and is converted into a tiny amount of heat). If the speaker has a high-pitched voice, more is reflected and absorbed. However, words spoken in a lower register tend to leak through (think about what a loud party sounds like from next door.) Buildings, trees, hills, people, rain, and even empty air, also absorb and reflect some of the sound energy. This is part of the reason it is difficult to hear someone who is far away.

The technical term for this phenomenon is **Attenuation.** Electromagnetic waves behave similarly to sound in many respects. The shorter the wave, the more severe the attenuation. Extremely low frequencies (long waves) can penetrate ground and seawater to some depth before they are absorbed. Somewhat higher frequencies can bounce off of ground obstacles and the ionosphere and thereby follow the curvature of the earth (the AM radio band, for example.)

As frequency goes up, wave-length gets shorter, and attenuation becomes more significant. Shorter waves give up their energy at a higher rate as they travel through the air. Overcoming this requires broadcasting signals with more power to go the same distance as is frequency goes up.

Even if none of the energy put into a signal was absorbed as it radiated out from the transmitter, the energy put in must be distributed over an increasingly large area. Recall:

$$\text{Surface area of sphere with radius r} = 4\pi r^2.$$

Thus, if the energy per square inch measured at one foot from the transmitter is X, then this same energy has to cover four times this surface area at a distance of two feet and 10,000 times the surface area at 100 feet. In other words, the signal strength at 100 feet from the transmitted is just .01% of what is at one foot from the transmitter. This is called the **Inverse Square Rule** and tells us that strength of a transmission decreases with the square of the distance from the transmitter.

Putting this together, we get the following takeaways:

- Low frequency transmissions go a long way using comparatively little power due to low attenuation
- Information throughput is proportional to bandwidth
- There is not much low-frequency bandwidth available in total, and so it is usually chopped up into smaller chunks which therefore cannot carry much information
- The farther you get from a transmitter, the less of its signal you receive

Section 6.3. Bandwidth as a Public or Private Good

Recall that private goods are rival in consumption. The sum of production of a private good over agents can be no larger than the amount produced. Public goods, on the other hand, are nonrival in consumption. Each agent can consume up to the total amount of the good produced. Formally:

$$\sum_i X_i \leq \bar{\dot{X}} \text{ and } \forall\, i,\ Y_i \leq \bar{\dot{Y}}$$

respectively, where:

$i \in \{1,\ldots,I\} \equiv \mathcal{I}$:	agents in the economy
X_i :	private good consumed by agent i
Y_i :	public good consumed by agent i
$\bar{\dot{X}}$:	total amount of private good produced
$\bar{\dot{Y}}$:	total amount of public good produced

Commercial **Broadcasting** uses a **One-to-all**, **Open-channel,** model, in which a single transmitter uses the channel continuously, and all receivers in range can hear whatever is transmitted. The **Federal Communications Commission (FCC)** grants or sells licenses for certain parts the spectrum to private entities for radio, television and other uses. A part of the spectrum is also reserved for unlicensed uses and amateur broadcasts such as citizen band radio.

Channels are private goods from the perspective of broadcasters. If one transmitter is using a channel, then no other transmitter can. On the other hand, the information transmitted would seem to be a public good. My tuning into a channel does not preclude anyone else for turning in as well. Two refinements should be made to this later conclusion, however.

First, although broadcasts are nonrivalrous to receivers, they are not always public *goods.* In fact, they may be public *bads.* It might be that I hate the politics of Sean Hannity, or NPR, or that I think polka music is destroying the moral fiber of America's youth. In this case, broadcasting such content is a bad for me, and a public bad for all who share my views.

Public or private commodities can also be a good if consumed in one quantity, but a bad if consumed in another. For example, watching an episode of Dr. Who might be enjoyable, but what if I tried to binge-watch the entire show? Somewhere between episode 1 and episode 839, I would almost certainly find that I was sick of the Doctor, the Daleks, and all the rest of it.

Second, it may be better to think of radio broadcasts as a **Local Public Good**, or **Semirival Good** to receivers. The simplest case is the following:

$$U(X,D) \text{ or } U(X,N)$$

and

$$\frac{\partial U(X,D)}{\partial D} < 0 \text{ or } \frac{\partial U(X,N)}{\partial N} < 0$$

where:

X : quantity of the semirival good

D : distance of the consumer from the source of the good

N : number of agents the good is shared with

Recall that for the case of network externalities, the utility for a given level of provision of a good, X, *increases* with the number of users N.

If a good is subject to **Congestion** or **Crowding**, on the other hand, the opposite is the case. The negative derivative conditions says that the utility an agent gets from a good decreases with either distance, or crowding.

Cellphone and pagers on **Personal Communications Services (PCS)** networks, and routers, laptops, and other devices on IEEE 802.11b based **Wireless Fidelity (Wi-Fi)** networks use a **Multicast**, or **Many-to-Many**, approach. All stations on the multicast network both transmit and receive data. Data is typically intended for a single receiver, and it may be encrypted to protect its secrecy. All stations on the network receive all encrypted data packets that are transmitted, even if they can't understand them.

Two stations in a multicast network cannot transmit on the same channel at the *same time*. Even though transmission time-slots are purely rival, if the demand for transmission time-slots is low enough, one station can use the channel without interfering with the ability of other stations to do the same when they wish to.

In the extreme, multicast spectrum can be seen as a pure public good until capacity is reached, at which point time-slots become a pure private good. More realistically, as demand for time-slots is goes up, stations may have to wait for a slot to become available. Thus, the more stations, the less useful or valuable the channel becomes. In other words, multicast networks are a semirival good to its users subject to crowding: $U(X,N)$.

Radio broadcasts, on the other hand, become harder to receive, and are lower in quality, the further the receiver is from the transmitter. Thus, they are not really pure public goods, but rather semirival goods subject to negative distance externalities: $U(X,D)$.

Section 6.4. Wired Communications

The inverse square law and attenuation are serious limitations in wireless communications. For this reason, a great deal of electromagnetically encoded information is transmitted by wire. Wires are also limited since they both radiate and absorb electromagnetic energy. This is why POTS (which are just twisted pairs of unshielded copper wire) connections cannot transmit meaningful amounts of data for more than a mile or two. Recall that DSL uses frequencies above the range of human hearing to transmit data. This higher frequency information is easily lost in the noise generated as the wire runs through electromagnetic fields of various kinds on its way to the local switch.

The solution is to use what is called a **Wave Guide**. By way of example, ships used to have **Speaking Tubes** that ran between the bridge and the engine room. Since the engine room was separated from the bridge by decks, bulkheads, and distance, it would have been hopeless for the captain to simply yell his instructions.

By running a tube between the two locations, however, the captain could speak into one end, and be heard at the other. The tube reduced attenuation by creating a column of air to carry the sound, allowing it to bypass decks and bulkheads. It also channeled the sound energy from one place directly to another. Thus, a speaking tube is a wave guide that both reduces attenuation and defeats the inverse square rule. Note that if the speaking tube had holes, in it would both absorb any sounds in the rooms it passed through, and leak some of the sound energy carrying the captain's orders.

Radio signals up to the lower end of the microwave bands (below 100G Hz) can be transmitted through a conductor such as a copper wire. Wire can be shielded from outside interference and leakage by surrounding it with a mesh of conducing material that is then grounded. (This is a kind of Faraday cage.) Cable systems use **Coaxial Cable** shielded in this way to transmit programing and data.

Light has too high a frequency to be transmitted through metal, but can be sent trough **Optical Fiber**. Optical fiber is made from an extremely transparent type of glass, and is specially coated to keep in the signals it carries in, and any extraneous light out.

By transmitting information using any part of the electromagnetic spectrum over the appropriate type of waveguide, three things can be accomplished:

- Transmissions are contained within the wave-guild, and so do not use any broadcast spectrum.
- Signals travel much farther, with less energy input, since energy flows down the guide instead of being wasted heating up the air.
- Waveguides can be strung over mountains, through walls, and around corners and so high frequency spectrum becomes much more versatile.

Optical fiber is the basis of the Internet backbone. A single channel on fiber can transmit as much as 26 Tb/s (that is, 26,000 Gb/s). The reason is that the part of the spectrum we call visible light ranges from 428.5 THz –750 THz (one THz is equal to 1000 GHz). The electromagnetic part of the spectrum used for radio is far below this, in the 3 kHz to 300 GHz range.[15] Visible light therefore has the potential to carry about 1000 times more information than is possible on all of radio spectrum.

[15] Microwaves and infrared fall between radio and visible light, and X-rays and gamma-rays are at frequencies above visible light.

Not only does fiber guide the light signal, and prevent interference and degradation, it can also carry many frequencies (or spectrum ranges) of light at the same time. In other words, many different channels, using different colors of light, can be used on a single piece of fiber at the same time without interfering with one another.

The limits of how many separate channels can be supported at once has not yet been determined, but experiments have used hundreds of channels at once and produced transfer rates as high as 100 Pb/s (100 Petabits/s is equal to 100,000 Tb/s).

Section 6.5. Spectrum Use and Allocation

In 1833 William Forster Lloyd wrote an essay observing that when a pasture was held in common, individuals tended to overuse it to the point where it supported fewer livestock than if it had been managed by a private owner. In general, any resource for which there are not well-defined property rights tends to be inefficiently used.

Tragedy of the Commons: Goods that are collectively owned, or to which no ownership is attached, are termed common property. Such incomplete property rights imply that anyone in the collective is free to use the resource as no cost. If a private or congestible good is held in common, using the good generally causes negative externalities for other members of the collective. For example, every fish I catch from a community pond is one less fish you can catch, and every car that gets on the highway lengthens your commute. Rational, self-interested agents do not take these externalities into account, and use the resource more intensively than is socially optimal as a result. Overuse of common property, and its negative social consequences, are called the Tragedy of the Commons. In contrast, when pure public goods are held in common, the marginal cost of using them is zero, and so it is optimal for agents to use them as if they were free.

Up until 1982, the FCC used competitive hearings to allocate spectrum. Companies desiring spectrum allocations would have to convince the FCC that they would put it to the best use. This gave an advantage to politically connected companies, and also allowed politicians to impose conditions on the grants. Companies and the government influenced each other, and divided up the profit, or rents, that free grants of valuable spectrum could generate.

From 1982, until 1994, the FCC used lotteries in which winners were allowed to resell there spectrum allocations. This had the advantage making it difficult for the FCC and other government agencies to capture spectrum rent through influence peddling, and instead awarded it to random individuals who might have no intention or ability to use spectrum. Of course, spectrum generally ended up being resold to the same companies who would have been granted it anyway under previous rules.

Since 1994, the FCC has used auctions to allocate bandwidth. This had two advantages. First, the company who ends up winning an auction has to be willing to pay more than anyone else, which is usually because it can make the best, or at least most profitable, use of the spectrum. Second, all the revenue from auctions went into the general fund, instead of being shared out among insiders, or randomly awarded to lottery winners.

Optimal auction design is a complicated issue that is well-studied in economics. The best way to divide up an interval of spectrum into channels, or to divide up the country into regions where the spectrum can be used, is not obvious. This is further complicated by the fact that the objects being auctioned have correlated, rather than independent, value. A license to use a channel is one region, is less than 1/6 as valuable as the right to use it in all six regions.

Settling these, and other details, such as the type of auction, terms for bidding, and the prevention of collusion, provided many economists with very lucrative consulting incomes for years. Even more economists were employed by telecommunications companies trying to game the mechanisms that these economists created. Of course, lawyers made even more from all sides.

When governments are in a position to do anything that confers value on private parties, outcomes like this are almost inevitable. Unfortunately, it is not obvious that there are better ways to solve such problems. Don't the players, hate the game.

Subsection 6.5.1. Protocols

Protocols such as 802.11x, LTE, FM, and HDTV are important regardless of whether bandwidth is licensed or unlicensed. To see this, think of bandwidth as a one lane bridge over a river. Traffic can flow over it at a certain rate, but only in one direction at a time. All the cars going in one direction must exit the bridge before any traffic from the other direction starts. The issue is how to allocate bridge crossing time-slots most efficiently.

One way would be to leave things completely unregulated. Anyone would be allowed to cross the bridge anytime they wanted. The problem is that this would lead to collisions. If two cars tried to cross in different directions at the same time, they would block each other, and neither would get across. Neither user benefits from the bridge as a result.

People might respond by getting bigger cars, trucks, or even tanks, so that they could force their way through. The more powerful vehicle would win. If only one agent had access to tanks, it might even be that the other agents would simply give up and not try to use the bridge at all.

To the outside observer, it might seem that the agent with the tanks is the only one who has any interest in using the bridge and conclude that there really is no resource allocation problem at all.[16] On the other hand, if any agent could buy a tank, we would end up in an arms race and a negative

[16] We will see below that LTE-u may affect Wi-Fi users in a similar way. LTE-u routers may end up using all the time-slots on a channel if it never detects any Wi-Fi routers attempting to transmit.

sum game. The most powerful vehicle in any collision would win, but the costs of getting to the other side of the river would rise for all. This was the main motivation for the Radio Act of 1927.

A better approach would be to agree to rules of the road. This might arise through social convention or government regulation. For example, a rule might be that whoever arrives at the bridge first gets to cross. A polite motorist would stop and look down the bridge before entering. If he saw another car, he would wait until it had completed the crossing before starting his. This would stop collisions, and if traffic were relatively light, would be a fairly efficient way to allocate time-slots.

Listen Before Talking (LBT): A part of Wi-Fi protocols that use **Carrier Sense** for clear channel assessment, before a station begins transmitting.

In the 802.11x (Wi-Fi) protocols, LBT is the electromagnetic equivalent of looking down the bridge. If traffic was heavier, the LBT rule might not be enough to prevent collisions. If there were two cars on either side of the river waiting for a third to complete its journey before starting their own, both would try to enter the bridge at the same moment. Either they would crash or notice each other entering the bridge and decide to back off, so the other could cross. If both drivers politely backed off and waited for the other, we would be at an impasse.

Random Back Off (RBO): A part of Wi-Fi protocols that require stations who detect that a channel is in use wait a randomly determined number of microseconds before assessing the channel again.

A better strategy would be to back off, and then look to see if the other driver actually enters the bridge. If it happened that the other car was still waiting, then the first driver could safely cross. Unfortunately, if they both look down the bridge immediately after they back off, they still see one another ready to enter the bridge and back off again. As long as they check the bridge at the same interval, the problem remains.

A solution is to have each driver look at the last digit displayed on his odometer and wait that number of seconds before checking again. This reduces the odds that they check the bridge at the same time to one in ten. If they happen to look at the same time anyway, they could each count the number of bugs on the windshield, and then have another look.

What if traffic was heavy, and lines of cars waiting to cross formed on each side of the bridge?[17] The first come, first served, mechanism given above, even with LBT and RBO, would no longer make optimal use of the bridge. What if instead of sending a single car in each direction at a time, we sent cars in groups of five? It takes five cars in a row only a little bit longer to cross and clear off a bridge than one car. We would only have to do LBT and RBO once for the entire group instead of once for each car. Thus, sending cars in groups would almost quintuple the number of crossings per day.

[17] When routers in a multicast network have a queue of packets waiting to be sent, they are said to have a full buffer. The 802.11 protocols have a very hard time sharing bandwidth efficiently in this situation.

In the case of multicast networks how long a station should be allowed to broadcast before going silent and giving other stations a chance to claim the channel is a key element of fair and efficient bandwidth sharing.

Choosing how many cars should be grouped together to cross at one time is more difficult if cars arrive at each end randomly. There may be longer queues on one side than the other, or there may be no queue at all on one of the sides. Sending five cars one direction and then waiting long enough for five non-existent cars to cross in the other direction would be a waste of capacity. It might be better to send twenty in one direction, and then two in the other, given the relative demands for crossings. How could we do this efficiently?

One possibility is to privatize the bridge. The hope is that if the bridge becomes private property, the owner internalizes the crowding externally, and uses the resource efficiently. In the case of the bridge, the owner might decide to build a tower that allows him to observe how many cars are waiting on each side. He could station an agent at each bridgehead and signal them by semaphore how many cars to release for crossing at any given time. Drivers might pay a toll to use such a bridge.

This plan would completely prevent collisions and would reduce the waste of bridge capacity. How much improvement in efficiency would result depends on how good the owner is at allocating crossing slots and how much time his agents spend receiving and interpreting the semaphore signals.

Whatever the case, the bridge owner is a monopoly, and we know that monopoly pricing leads to inefficiency. Whether the social loss from monopoly pricing is less than social gain from better use of bridge capacity depends on the details of the situation.

All PCS protocols have **Control Channels** that are separate from the **Data Channels**, and must balance gain from efficient use of available bandwidth with the overhead costs of control and coordination. PCS networks also require that someone bear the cost of building towers and other infrastructure.

Perhaps we could accomplish the same thing without privatizing the bridge. Doing so would require somehow knowing the length of the queue on each side of the bridge, and perhaps on the roads leading up to the bridge as well. An engineer would then have to solve an optimization problem to determine the best rules about how bridge traffic should be regulated given the information available. Allowing emergency vehicles to have priority seems like a good idea, and it would also be nice if lower value bulk users would find an alternative to the bridge, or at least not use it at times of peak demand. This means we would have to know if a truck is carrying cargo that could be moved at less social cost by barge instead of on the free, but congestible bridge.

Wi-Fi 802.11 routers have neither the information, nor the computational capacity, to solve this problem. We would need routers to be more knowledgeable about their local environments and to know how use this information intelligently. **Cognitive Radio**, is an emerging technology that might allow decentralized routers to share bandwidth even better than PCS protocols using a control channel.

Subsection 6.5.2. Standards and Regulation

Suppose we decide to privatize all railway, as opposed to automobile, bridges. Even better, suppose we privatized the right to build railway bridges on different segments of the river. One advantage of this is that the private sector would have an incentive to find the best spot to build and would pay for construction and maintenance. The public gets the bridge infrastructure for free, although it must pay a monopoly price for its use.

Should we leave the private bridge builders unregulated? Suppose that each builder had to decide on his own how widely to space the rails he lays down. If many different rail gauges ended up being used, manufacturers would need to produce small batches of locomotives for each wheelbase. This is more expensive than building all rail-stock to a single standard. In addition, each locomotive would be restricted to a small number of bridges and so would provide less value.

Open Standard: A set of technolgical specifications that are publicly available and free to use, often created by government or professional groups.

Proprietary or Closed Standard: A set of technolgical specifications that may or may not be publicly available, and generally involve licensing and fees to use. In some cases, groups of companies will form patent pools that allow use of propriety technologies within the membership as a compromise to gain wider use and network externalities.

The invisible hand is not enough to solve this coordination problem. In fact, competing, but incompatible, standards may be a consequence companies trying to maximize profits. VHS/Betamax, HD/Blu-ray, GSM/CDMA are examples of coexisting standards that were promoted by companies with vested interests. Society benefits when a standard rail gauge is used, and government regulation is probably the best way to get it. This is true even though the railroad bridges are entirely privatized.

Regulation plays a similarly important role regardless of whether bandwidth is used for one-to-all broadcasts or many-to-many multicasts. Suppose that the government allowed local TV and radio stations to choose any broadcasting format instead of HDTV, AM, or FM. Consumers would have to buy separate receivers for each format and this would be wasteful. Suppose that Wi-Fi routers could decide for themselves to use LBT and RBO or how much wattage to use for broadcast. Users who choose the rudest and most powerful routers would get sole use the spectrum. Imposing 802.11 protocols and wattage limitations allows the 2.4 and 5 GHz spectrum to be shared far more fairly and efficiently.

To sum up, protocols, regulations, and standards are necessary for efficient use of spectrum. This is true regardless of whether spectrum is licensed, unlicensed, or reserved for governmental use. Neither the innovative potential of free experimentation, nor the incentives for profitable use offered by the free market, are compelling arguments for a hands-off approach. Bandwidth is scare, and demand is increasing. Choosing good protocols and standards is at least as important as deciding how bandwidth should be allocated to various uses.

Section 6.6. Licensed or Unlicensed?

Deciding how much new bandwidth to allocate to licensed and unlicensed uses is one of the most important choices the FCC has before it. The implications for the economy will be enormous whatever the decision. The bandwidth currently allocated to both PCS and Wi-Fi is being used at levels close to its capacity in many cases, and new uses such is autonomous vehicles and IoT will need even more bandwidth to function. This section will discuss the relative merits of licensed and unlicensed solutions to meet these demands.

Subsection 6.6.1. Why Licensed?

In general, privatizing bandwidth through licensing does not lead to competitive outcomes, economically efficient use of bandwidth, or achieve social objectives in the public interest. It also tends to create monopoly and oligopoly markets for vital communication services. Given this, how can licensing ever be good public policy?

There are two main arguments in favor of licensing. First, some important uses of bandwidth require the build-out of expensive infrastructure. For example, someone must build towers, connect them to backhaul, and so on, if we are to have a nationwide cellular network. The government could do this, but then taxpayers would have to foot the bill (whether they are cell phone users or not). This might be inequitable, and other demands on the public purse may be considered to be more important. Even though private carriers charge users higher than competitive prices for cellular service, consumers may still be better off than waiting for the government to build a local tower.

TV and radio licenses can be justified on the same basis. Producing and broadcasting content is expensive, and the public benefits from these broadcasts. Broadcasters are allowed to use public spectrum and recover these costs through advertising. Cable and landline telephone companies follow a similar model. In this case, permission to use public rights of way is granted instead of the use of public radio spectrum. Companies agree to build and maintain costly infrastructure in exchange for being allowed exclusive, or near exclusive, rights to provide certain types of communications services to customers.

The second reason is that the private owners of bandwidth are able to control how bandwidth is used and shared. PCS carriers use prices to ration usage over subscribers, and employ protocols such as LTE that use centralized coordination to allocate time-slots to users in each cell. This centralized approach allows private PCS providers to use limited bandwidth more efficiently in many cases, and their ownership of gives them an incentive to do so.

In contrast, Wi-Fi and other uses of unlicensed spectrum applications rely on regulations and decentralized protocols such as the 802.11 family to share bandwidth fairly. In many cases, this is less efficient and reduces the net data capacity of a channel. The FCC has had to choose between a monopolistic, second best, licensed model, and a wasteful, regulated, unlicensed approach. Licensing was often the lesser of the two evils. Below, we will argue that new technologies and use cases offer the FCC a better set of choices.

Subsection 6.6.2. Why Unlicensed?

In 1985, the FCC decided to open the **ISM**[18] (**Industrial, Scientific, and Medical Bands** for unlicensed use. The intention was to encourage experimentation and innovation. As long as a few rules such as limits on transmission wattage were followed, users were free to do pretty much anything they wanted. As it turns out, this policy experiment was a phenomenal success.

Simple uses such as cordless telephones, remote controls, and baby monitors, gave way to more sophisticated ones such as Wi-Fi, Bluetooth, ZigBee, and RFID tags.[19] These technologies are completely embedded in our lives now. It is surprising that so much has been accomplished given the relatively small amount of bandwidth set aside for unlicensed uses. In total, only about 100 MHz in the sub-3 GHz bands and another 400 MHz in the 5 GHz band are allocated. More recently, an additional 7 GHz in the 60 GHz bands have been allocated, and this may prove to be very well suited to facilitating oncoming technologies.

An unfortunate legacy of this history, however, is that the FCC is reluctant to imposes more stringent conditions on unlicensed bandwidth use. The value created by the 500 MHz below 6 GHz allocated at present is huge. These bands are crowded, and demand is rapidly increasing. The FCC's light hand in the 1980s gave us all this, but it may be time to reap the fruits of that experiment.

Rules, regulations and protocols should probably be revised to protect Wi-Fi and other current users of this bandwidth, Why continue to search when you have already arrived at your destination? It would be a mistake to conflate "unlicensed" with "experimental" or "unregulated".

[18] The ISM bands are set of frequencies set asides for microwave heating, metallurgical induction process, medical and scientific imagining, and other non-communications uses.

[19] Radio Frequency Identification tags are passive devices that use energy transmitted by a reader to relay stored information on the unlicensed 2.4 GHz band. It is estimated that more than that RFID tags contribute hundreds of billions of dollars the US economy through savings in inventory control, retail loss prevention, reductions in medical errors, and similar applications.

Although the need for experimentation is no longer as strong argument in favor of unlicensed bandwidth, there are still many things to recommend it. Users do not have to pay monopoly, or indeed, any price for transmitting in unlicensed spectrum.

The decentralized nature of Wi-Fi allows for greater freedom and privacy. The fact the Wi-Fi routers are required to use low transmission power levels allows that the same unlicensed bandwidth can be used by simultaneously without interference by devices that are as close together as 50 or 100 meters. In contrast, licensed PCS cells operate at higher power and this limits reuse of spectrum.

Subsection 6.6.3. Choosing Licensed or Unlicensed

Broadcasting, such as commercial radio and television, uses licensed bandwidth, and should probably continue to do so. The costs of producing and transmitting content generates a free-rider problem, and the best solution available seems to be licensing channels to whomever agrees to bear this expense.

The FCC has set aside a significant amount of spectrum for radar, radio astronomy, communications between aircraft and ships, public safety, military, and other governmental uses. In a sense, the government has retained the license to this bandwidth, however, it permits qualified agents to use it in specified and regulated ways. In other words, use of this restricted spectrum is shared using decentralized protocols similar to what we see for unlicensed bandwidth.

While licensing and restricting bandwidth for such use cases is probably the best approach, the question of how much to allocate is less clear. Cable and broadband has replaced broadcast television as the preferred way to get video content, and existing channel allocations do not have enough bandwidth to support high definition broadcasts.

The FCC has wisely decided to move these channels to higher frequencies and repurpose the beachfront spectrum that is freed up. There are many similar allocations that made sense when demand for spectrum was low that should now be reassessed.

The central problem facing the FCC, however, is finding more bandwidth for multicast networks, and deciding whether it should be licensed or unlicensed. There are two factors that create a strong presumption in favor of unlicensed approaches.

First, bandwidth belongs to the people and the FCC is charged with allocating and regulating it in their interests. Ignoring this default, and selling some of this scarce public resource to private entities, is sometimes beneficial to the public.

For this to be so, however, the private entity must bring some comparative advantage or specialized ability to the table that makes it possible to generate more public benefit than well-regulated, unlicensed uses.

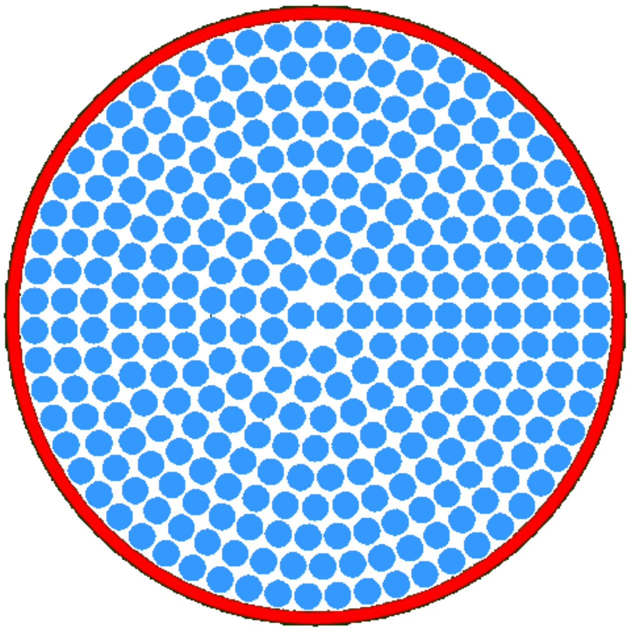
Private benefits to the license owners, such as revenue or profits, contribute to the public's welfare only indirectly in as much as they imply that the use may also have generated consumer surplus or beneficial externalities.

For example, if the government gave, or sold, a national forest to a lumber company, the company would certainly make profits by harvesting the trees. This is not in itself a good argument for selling national forest land. Falling lumber prices that might reduce the cost of housing, allowing public access to otherwise inaccessible lands, and reducing the instances of wildfires, on the other hands, are all public benefits that do argue in favor of this policy.

We could instead allow individuals to harvest trees, but this is likely to be inefficient for a number of reasons. Coordinating such activity to get safe and sustainable lumbering would be difficult, and the lumber company has the advantages of lower production costs that go with larger scale. For example, it can buy heavy automated equipment to harvest and mill the lumber instead of relying on axes and chainsaws. Thus, selling the rights to some part of the national forest might very well be good policy. Taking such lands away from the public, however, requires this kind of clear justification.

Second, allocating bandwidth to low-power, unlicensed, instead of higher-power, licensed, use, is inherently more efficient. As an example, suppose that 10 MHz of bandwidth could be allocated to PCS or Wi-Fi. Suppose a PCS cell has a radius of 1 km while Wi-Fi routers have a radius of 50 meters. It is possible to pack 308 Wi-Fi routers with non-overlapping signals inside such a PCS cell.

In other words, suppose that a 10 MHz channel is capable of transmitting 100 Mb/s if allocated to licensed PCS usage. Then one cell tower could transmit 100 Mb/s divided over any clients within range. On the other hand, allocating the same 10 MHz to unlicensed Wi-Fi use would allow 30,800 Mb/s to be transmitted to clients within the same 1 km diameter region. If the PCS cell had a radius for 2 km then 1245 Wi-Fi routers could fit, and if it was 10 km, then the number would be greater than 31,000.



308 Wi-Fi routers fit inside a 1 km diameter PCS cell

Subsection 6.6.4. Allocating Spectrum in Practice

Broadcast spectrum is a scarce resource that is held as common property by the residents of a country. There are any number of competing uses: remote control devices, commercial radio, police, fire, and public safety communications, TV, radio astronomy, **RADAR (RAdio Detection And Ranging), s**atellite uplink and downlink, CB and ham radio, air traffic control, military and government communications, Wi-Fi, GPS, and cell phones, to name a few.

In the US, the FCC is in charge of allocating, licensing, and policing spectrum. Ideally, this should be done with a view to maximizing public welfare taking into account the technical constraints and benefits that various slices of the spectrum carry with them.

Like any government agency, there is a degree of influence peddling, and regulatory capture. Big companies who use spectrum have a loud voice, and there is always a temptation both to censor "bad", and foster "good", content and practices. This is complicated by legacy allocations of spectrum that made sense at the time, but are now undesirable given technological advances.

It is difficult to convince entrenched interests to give up their coveted slice of spectrum. Things do change, however, though slowly. A good example is the 2010–2015 FCC initiative to move some broadcast television from lower frequencies that are well suited to cellphone communications, to higher frequencies that are more suitable for HDTV, which is more information intensive than analog TV broadcasts.

Section 6.7. Economics

Subsection 6.7.1. Semirival Goods.

The simplest cases of semirival goods are result from negative crowning or location effects:

$$U(X,D) \quad \text{or} \quad U(X,N)$$

where:

X : quantity of the semirival good

D : distance of the consumer from the source of the good

N : number of agents the good is shared with

In this simple case, it is often assumed that:

$$\frac{\partial U(X,D)}{\partial D}<0 \quad \text{or} \quad \frac{\partial U(X,N)}{\partial N}<0$$

The negative derivatives model the idea that value decreases with distance and crowding. For example, the value of a museum (X) to a consumer decreases the further it is from his house (D) or as the number of people who are in the museum at the same time (N) increases.

A more general case would recognize that not all agents crowd each other in the same way. For example, attractive, interesting people make a party more enjoyable, while unattractive louts make it less so. Intelligent classmates can make a course more enjoyable, but if grading is on a curve, smart classmates may be a net negative. You probably care about the characteristics of the people in your social networks, or with whom you share a social media platform. An open source project needs coders, database experts, network specialists, administrators, fund-raisers, and more. The number of each of these types of agents affects the time needed to complete a project, as well as its overall quality.

Thus, we should think of agents as bring attribute that make them fall into different "crowding type" categories. We can describe a group or coalition of agents who jointly consume or project a semirival good with a crowding profile:

$$(N_1,\ldots,N_t,\ldots N_T)$$

where:

$t \in \{1,\ldots,T\}$: crowding type (e.g,, smart, tall, skilled in C++, male, speaks my language, …)

N_t : number of agents in a group of crowding type t.

Note:

- Crowding can affect both utility, and production costs
- Crowding effects may be valued differently by different agents
- Crowding effect can be positive or negative, and can switch for one to the other
- Crowding effects can be complementary or substitutable

Thus, utility functions, and production (cost) functions, might take the following form:

$$U_i(X_i, N_1, \ldots, N_t, \ldots N_T) \quad \text{or} \quad TC = F(X, N_1, \ldots, N_t, \ldots N_T)$$

There is no *a priori* restriction or expectation on the sign of the derivative of these functions with respect to the number of any crowding type.

Subsection 6.7.1. Standards

Technolgical Standards play a positive, and essential role in innovation and growth. Laying down standards allows, or forces, companies to build products that are interoperable. Wide-spread use by many companies generates beneficial network externalities. It also allows experimentation, and new applications that build on an existing ecosystem. **Open Standards** allow large and small companies to compete on a level playing field.

Some standards arise from the market domination of a single firm. For example, Microsoft Windows is the de facto standard for desktop computing. This puts Microsoft in an enviable position that allows it control a large ecosystem, extract profits from its technology, and manipulate the shape of technolgical and market advances to suit its interests.

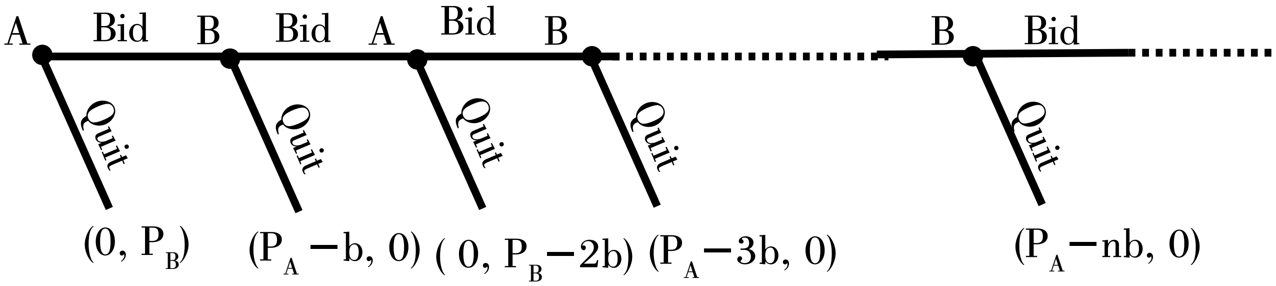
Firms have come to realize that standards setting has large economic implications. If a firm can get their proprietary technology embedded in an official, or de facto, technolgical standard, they can charge significant licensing fees to the entire industry. For example, Qualcomm holds several patents that are essential to 4G, 5G, and has convinced the 3rd Generation Partnership Project (**3GPP**) to include them in its standards.

Because of this value, we have seen a number of **Proprietary Standards** wars. For example, Betamax was a propitiatory videotape standard owned by Sony. In many ways it was superior to VHS, which was a competing open standard. Sony would have made a great deal of money if Betamax had managed to gain critical mass. However, the higher cost, and resistance of the rest of the industry, eventually lead to VHS winning the war, and the ultimate disappearance of Betamax. Perhaps surprisingly, Sony tried the same thing with its Blu-ray video disk standard, which eventually lost to the current DVD standard.

Standards are set by international and national government organizations, legislative and regulatory bodies, industry groups, coalitions of firms, individual firms, and markets. The results are hugely important, both economically, and socially. Despite this, the public knows very little about how, why, and in whose interests, they are established.

Subsection 6.7.2. First Price Ascending Bid Auction

In a **First Price Ascending Bid Auction,** agents bid in turn, the highest bidder wins, and pays whatever he offered. We can write this kind of auction down as a variation on an unbounded centipede game,



where:

P_A, P_B : reservation price or valuation of bidders A and B.

b : minimum bid increment

Agent A goes first and either quits, or bid b. If he quits, agent B gets the item for free and receives a payoff o f P_B.

If instead agent A bids, Agent B is at his first decision node and either bids $2b$ or quits. If he quits, agent A wins the item and pays b, receiving a payoff of $P_B - b$.

If agent B bids, then agent A is at his second decision node, and so on.

The dominant strategy of both agents is to keep bidding until they reach their reservation price, and then quit.

These are a type of **Open Outcry Auction** in which agents hear the bids of their competitors and must respond immediately. Such auctions are also commonly used to sell works of art, used cars, cattle, foreclosed houses etc. They are not well suited to online or virtual situations, but one might wonder why they are so common in real-space. The FCC, for example, uses various combinations open outcry first price acceding auctions, and first price sealed bid auctions to allocate spectrum.

First price ascending bid auctions are efficient in the sense that the agent who values the good the most will always end up with the item. The agent with the second-highest valuation (or reservation price) will never choose to outbid the agents with the highest valuation. Governments, for example, often want the most productive use made of resources they make available for various purposes. The highest bidder for drilling rights on a parcel of land is likely to be the one who can put it to the most economically profitable use.

On the other hand, first price ascending bid auctions are not revenue maximizing. The bidder with the highest valuation only needs to bid just a little more than the agent with the second-highest value (or reservation price). Thus, the maximum price (the reservation prices of the highest bidder) will not be obtained by the seller.

First price ascending bid auctions are also subject to manipulation by **Shills**. A shill bidder is an agent working in the interest of seller who bids only to raise the price. This is like a game of chicken. When only one other bidder remains, the shill tries to guess if the remaining bidder is at his reservation price. If he thinks not, the shill bids, believing that a rational opponent will continue to bid as long as the bid is below his reservation price.

If the shill guesses right and drops out before the other bidder reaches his reservation price, then the final selling price ends up somewhere between the second highest and highest reservation price. If the shill guesses wrong, then he wins the item. However, since he is working for the seller, the sale is not really executed. Instead, the seller simply retains the item, and tries to sell it later.

This outcome has three possible downsides for the seller. First, it delays the sale of the item. Second, it may be that the seller has to pay a commission to the auctioneer. Thus, he might pay a 10% fee, and still be stuck with the item. Third, he may have poisoned the well. If the seller puts the item up for auction again, the original highest value bidder may be suspicious and choose not to join the auction. This knocks the agent who was willing bid the highest out of the auction, which will lower selling price to the third-highest reservation price instead of the second.

Subsection 6.7.3. Regulatory Capture, Agency, and Rent Seeking

Economic Rent is the difference between what a factor is paid and what its value is in its next most valuable use. Rent is almost always created by some external force that distorts markets and keeps them to clearing at equilibrium princes.

Regulatory bodies such as the FCC, FTC, SEC, and many others, have a duty to protect the interests of the public. They are often created to solve an existing market failures such as the tragedy of the commons problem we see with the allocation of bandwidth. The FCC has the power to assign valuable property rights outside of a market context. In other words, the FCC has the authority to assign economic rents to favored actors.

Formally, the FCC is an **Agent** of the public, which is called its **Principle**. **Principle-Agent**, or simply **Agency Problems**, are a well-studied area in many subfields of economics. Agents have their own motivations, and they do not always align with those of their principles. This is why we have inspector generals, audits, and elections.

Unfortunately, these are difficult tools to use well. Instead, we often see **Rent Seeking** behavior where some party creates incentives for agents to work in the parties interests instead of their principle's. In some cases, this takes the form of bribes. While this corrupts the agents, it is just a transfer, and so in not in itself economically wasteful. In other cases, rent seeking take the form of lobbying, influence peddling, political campaigns, and indirect transfers such as job offers. These are economically inefficient, and lead to **Rent Dissipation.**

In the context of regulatory bodies, this phenomenon is call **Regularly Capture**. Agencies are influenced to work in the interests of the regulated instead of in the public interest. This has been going on for as long as people have had governments. Humans are stuck with second-best choices: Allow markets to fail, or have them regulated by less than perfect agents of the government. Sadly, there is seldom a perfect solution.

Section 6.8. Appendix – Spectrum and its Uses

In this appendix, we start by describing the different parts of the electromagnetic spectrum. The following table outlines the conventional names for the various parts as well as the frequency and wave length intervals they cover.

Frequency and Wave Length		
Name	**Frequency**	**Wave Length**
Extremely Low Frequencies (TLF, ELF, SLF, ULF)	0 kHz – 3kHz	∞ – 100 km
Radio (VLF, LF, MF, HF, VHF)	3 kHz – 300 MHz	100 km – 1 m
Microwave (UHF, SHF, EHF, THF)	300 MHz – 300 GHz	1 m –1 mm
Infrared Light	300 GHz –428.5 THz	1 mm – 700 nm
Visible Light	428.5 THz –750 THz	700 nm – 400 nm
Ultraviolet Light	750 THz – 30 EHz	400 nm – 10 pm
Gamma Rays	30 EHz – ∞	10 pm – 0

LF = Low Frequency
HF = High Frequency
T = Tremendously
E = Extremely
S = Super
U = Ultra
V = Very

km = Kilometer (10^3)
m = Meter
mm = Millimeter (10^{-6})
nm = Nanometer (10^{-9})
pm = Picometer (10^{-12})

kHz = Kilohertz (10^3)
MHz = Megahertz (10^6)
GHz = Gigahertz (10^9)
THz = Terahertz (10^{12})
EHz = Exahertz (10^{15})

Below is an outline of the most important uses of various parts of the radio and microwave spectrum.

Extremely Low Frequencies (0 – 3 kHz): This includes TLF, ELF, SLF, ULF. Frequencies this low can penetrate ground and seawater, but the 3kHz bandwidth available is not much and so can carry very little information. To the extent ELF bands are used at all, it is for military, communications with submarines and underground facilities.

Very Low and Low Frequencies (3 kHz – 325 kHz): These frequencies carry a long way and are subject to limited attenuation. However, there is too little bandwidth in this part of the spectrum to carry very much information. They are mainly used for radio location, air, and nautical beacons, and underground cable location,

Medium Frequency (325 kHz – 3 MHz): The lower end is used for air traffic control and emergency channels. The middle part (530 kHz – 1705 kHz) is used for AM radio. This band is a nice compromise between low information capacity and low attenuation. At night, with the help of reflection off of the ionosphere, a 50,000 watt AM station can carry low fidelity music for thousands of miles. The upper end is devoted to police, fire and public safety, amateur radio, and marine communications.

High Frequency (3MHz – 30 MHz): This band is devoted mostly to maritime and military uses, although parts are reserved for industry, citizen's band (CB) radio, and radio astronomy.

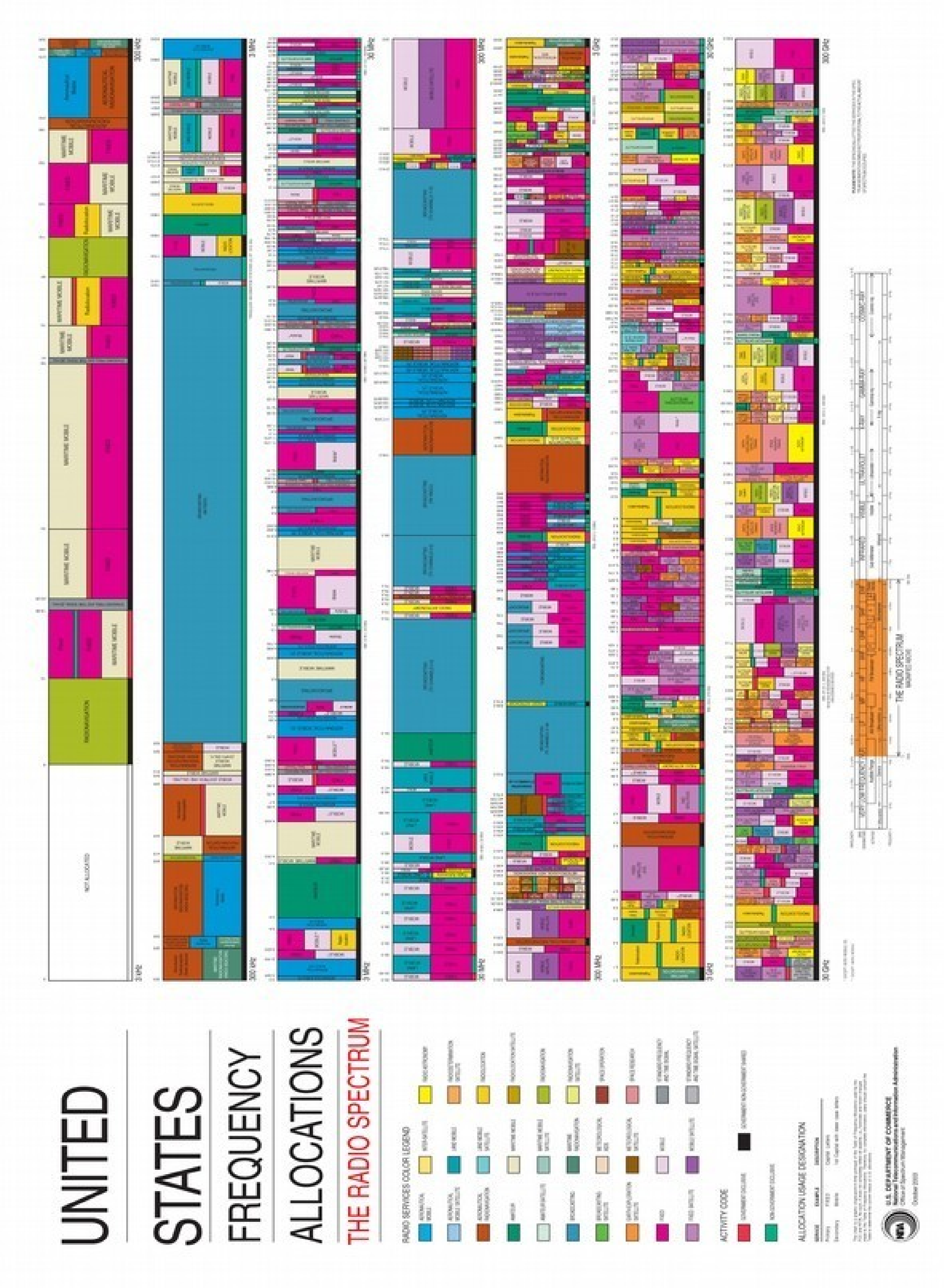
Very High Frequency (30 MHz – 300 MHz): This band goes to wide variety of uses that require a reasonably large information capacity, but can tolerate moderately high attenuation and restriction to line sight broadcasting. Broadcasts in this bandwidth work best over shorter distances where there are few obstructions. Uses include: cordless telephones, alarm systems, door openers, remote switches, VHF television (starting with channel 2 at 60 – 66 MHz), radio-controlled models, FM radio broadcasting (108 – 137 MHz), and space-to-earth communications for satellites, meteorology, research, amateur, military, public and public safety, commercial transposition, business, public, and mobile pagers. Note that the line of sight restriction is the main reason that television and radio are usually broadcast from towers placed on hills or mountains.

Ultra High Frequency (300 MHz – 3 GHz): This band goes mostly to PCS cellular service (starting at 800 MHz), biomedical telemetry, UHF television, satellite, unlicensed short-range, Wi-Fi, military, air traffic control radar, radio astronomy. GPS satellite, point-to-point microwave, and the government deep space network,

Super High Frequency (3 GHz –30 GHz): This band goes mostly to radio altimeters, radio astronomy, government radio-location, satellite, TV broadcast auxiliary service, cable television relay service, Wi-Fi, and point-to-point microwave.

Extremely High Frequency (30 GHz–300 GHz): This band goes mostly to government radio-location, satellite, TV broadcast auxiliary service, cable television relay service, Wi-Fi, unlicensed, amateur, point-to-point microwave, military, and satellite broadcasts.

The reality is a bit more complicated. Below is table that shows exactly how bandwidth is allocated in the US. Feel free to skip this figure. It is included mainly to show what a mess things are.


UNITED
STATES
FREQUENCY
ALLOCATIONS
THE RADIO SPECTRUM
RADIO SERVICES COLOR LEGEND
ACTIVITY CODE
ALLOCATION USAGE DESIGNATION
U.S. DEPARTMENT OF COMMERCE
THE RADIO SPECTRUM

Chapter 7. Systems Software

Section 7.1. BIOS and Bootstrapping

When a computer is first turned on, all volatile RAM is empty. The system is hardwired to automatically read the **BIOS (Basic Input Output System)** held in non-volatile or battery powered memory and begin executing the instructions it contains.[20]

This usually begins with a **POST (Power On Self Test)** which identifies, checks, and initializes basic system devices and hardware such as the CPU, RAM, video display card, keyboard, hard disk drive, and so on. If everything checks out, the boot process continues with the BIOS providing basic drivers and interfaces that allow the central control unit on the CPU to find and access the storage device that contains the operating system. Normally, the process completes with the CC loading its OS into RAM and then proceeding as a full-blown system.

The BIOS is considered **Firmware,** meaning that it has a degree of permanence, but can be altered. This is done using an interface that can be called up at certain points in the boot process, or by replacing, or **Flashing**, the BIOS currently in memory with a new version. The BIOS contains a number of setting that an advanced user might wish to change. For example:

- Choosing the order in which the system looks for a boot device. This allows you to ask the system to boot from a USB or CD drive instead of the default HDD, which in turn allows you to run alternative OSs or replace the current one.
- Over-clocking the CPU and configuring the way the system uses and accesses other hardware components.
- Setting system passwords.
- Setting fan speeds, the system clock, and disabling certain pieces of hardware.
- Updating or replacing the BIOS.

For embedded systems in IoT devices especially, the BIOS is can be a particularly vulnerable attack surface. Consumers seldom, if ever, access these devices directly and so never change the default administrative passwords they were shipped with.

[20] Most personal computers are now shipped with a BIOS based on **UEFI (Unified Extensible Firmware Interface)**. UEFI implements a “secure boot” feature that make it more difficult for rootkits to load during boot-up, but also more difficult to install or run OSs besides Microsoft Windows. Microsoft, and some device manufactures, also push firmware updates to the UEFI which are almost impossible to prevent. This gives Microsoft access to your computer at the boot level, which means that it effectively has total control over your device.

Section 7.2. Operating Systems

A computer's operating system contains the basic set of instructions that allow it to use and interact with its various physical components, and permit application programs to share these resources efficiently. An operating system has five basic components:

Kernel: The kernel is the core of an operating system. It connects programs to hardware, and decides if a program is allowed to use a given resource, and if so, how the resource is to be shared.

Device Drivers: A driver is software that provides an interface between a specific piece of hardware, and the kernel. Drives must be customized to give access to all the features and capabilities of a given type of printer, monitor, HDD, keyboard, or other device, and also to comminate with each version of each operating system. Thus, a device maker must provide separate drivers, for Windows 10, Windows 11, Android 13, MacOS 12, and 13, various flavors of Linux, and so on.

System Utilities: These are background programs that run services that multiple application programs are likely to need. Examples include printer spools, encryption, data compression, antivirus, and disk management tools. Including utilities in a standardized form with the OS reduces the risk of conflicts that might arise if each application included its own redundant subsystem to accomplish similar tasks.

User Interface: UIs allow humans to interact with an operating system, and the hardware it controls. The UI contributes to, but is distinct from the **User Experience (UX)**, which depends on the usability, work flow logic, performance, and other more technical elements of an operating system, or piece of software. Most modern OSs use a **GUI (Graphical User Interface)** that allows the operator to use keyboards and mice to give commands and interact with application programs.

Application Program Interfaces: APIs are software-to-software interfaces that allow applications to communicate with one another in agreed-upon ways. Operating systems have APIs that allow applications to access elements of the user interface, receive inputs from keyboards and mice, use clipboards, and various system services.

APIs are also found in almost every program and application, Internet platform, and network system. For example, Amazon provides APIs that allows customers to search for merchandise, and to pay for items in a shopping cart. Cryptocurrency wallets have APIs that facilitate creating and signing transactions, and then sending them to the network for execution. **SQL (Structured Query Language)** servers provide APIs that allow internal applications, or external users, to querry a database, and pars the answers that it returns.

Note that APIs create a unique kind of fragility in integrated systems. **Master Data Management (MDM)** and **Enterprise Systems Management (ESM)** solutions usually incorporate SaaS components, and other cloud services. They are seldom self-contained systems built, and run, on local servers. The companies behind these cloud components may decide to stop supporting certain services, or to change elements of the API, for any number of reasons. Each time an essential service is integrated into a complicated system through an API, an additional point of failure is created at the same time.

The two most important aspects of an OS are the kernel and the user interface. The user interface affects how easy the OS makes it for user to do the things he wishes with his computer, so it is obvious why this is important. Why the kernel matters may be less obvious. The kernel controls what the application programs can and cannot do (and by extension, what the user can and cannot do). Here are some reasons this makes a difference:

Speed: A well-designed kernel connects hardware to program requests efficiently with minimum overhead. It decides how to allocate resources when multitasking to meet the user's priorities.

Security: A well-designed kernel will not allow unwanted programs to install themselves. It will monitor programs to make sure they do not gain access to parts of memory or other hardware that they have no permissions for.

Control: A well-designed kernel allows the user to have full control over his system. Users should be able to terminate any process, to see all the files on all of his storage devices, and be able to gain administrator privileges to modify or delete them if he chooses.

Disclosure: A well-designed kernel will make the functions and actions of its programs and systems transparent. It should never take any action without the user's knowledge and permission,

When the world was new, there were many operating systems. These were generally very simple and written with one type of minicomputer in mind. There were not many computers in existence and so interoperability and standardization were not major considerations. Most these OSs are no longer with us. There are four major OS families, of which only two remain significant.

Subsection 7.2.1. DOS: 1970-1999

DEC (Digital Equipment Corporation), released BATCH-11/DOS-11 in 1970 as an OS for its very successful PDP-11 minicomputer. CP/M was a similar command line driven OS written with microcomputers in mind in 1974. CP/M was very popular at the time and ran on the Altair 8800, the Apple II, the Commodore 128, the Osborne 1, and the TRS-80, for example.

Famously, IBM tried to enter into negotiations to license CP/M for use on its new PC in 1980, however, IBM could not get the makers of CP/M to sign non-disclosure agreements. Instead, it contracted with a small company that made an implementation of the basic programing language.

This company quickly purchased an OS called **QDOS (Quick and Dirty Operating System)** from Seattle Computer Products for $30,000. They modified it to meet IBM's requirements and licensed it to them for $40 per copy installed on each new PC. The product was released as both MS-DOS and PC-DOS, and strong similarities to CP/M were noted. Bill Gates and Microsoft continued to develop DOS until the mid 1990s when GUI based OSs finally came into general use.

Microsoft's first GUI OS was Windows 1.0 released in 1990. It was really just a shell written on top of DOS, rather than a standalone OS. It had many quirks and problems and was not very successful. Windows 3 and 3.1 were also DOS shells, solved most of the problems in Windows 1, and was extremely successful. Both of these OSs were widely used from 1991 to 1995 or so.

Microsoft and IBM teamed up in the late 1980s to develop OS/2 which was a DOS based OS and later included a GUI shell. The project did not really pan-out, but it did absorb a lot of attention and resources, especially from IBM.

Apple DOS was another command line OS similar to CP/M and PC-DOS, but written independently at Apple for its Apple II computer. It was used from 1978 to 1984 and had many quirks and idiosyncrasies that made it difficult to use.

In 1984, Apple released “System 6”, an OS with a GUI designed for its Macintosh computer. This went through several revisions and was rebranded “macOS” in 1986. It was very popular and successful. This line of OSs has to be counted being independent of the DOS family, but it came to an end in 1999, as we shall see.

Subsection 7.2.2. Windows: 1995-

Starting with Windows 95 (1995), Microsoft's OSs can be counted as a new family, essentially independent of DOS. It was true that Windows 95 used PC-DOS as a bootstrap (that is, the BIOS loaded DOS instructions and drivers into RAM to allow the system to access the hard drive and load the Windows kernel. Once the system was up and running, the GUI and kernel interacted directly rather than through DOS.

Windows 98 (1998), Windows Millennium Edition (2000) were improvements in some ways, but also brought with them an array new issues.

Windows XP (2001) and was the first Microsoft OS built without DOS in any form. Instead, Microsoft used **NT (New Technology)** which Microsoft developed to be a highly portable, cross-platform, foundation for operating systems. XP was faster, more stable, and more secure, than ME, and also had a well received new look for its GUI.

Windows Vista (2007) was XP's intended successor, but was widely disliked by consumers, Windows 7 (2009) and fixed many of the problems in Vista. Windows 8 (2012) moved to a tiled style of GUI that was designed with touch screens in mind. It received mixed reviews.

Widows 10 (2015) was the first version to be cloud centric. It had tight integration with Microsoft 365, and the rest of Microsoft's cloud based ecosystem. Windows 10 has been criticized for the amount of telemetry and user data that it collects. Windows 11 (2021) broadened and tightened these integrations with Microsoft's ecosystem.

Subsection 7.2.3. UNIX: 1969-

UNIX is an operating system written in the "C" programing language at AT&T Bell Labs in the early 1970s mostly for internal use. The fact that it was written in C instead of assembly language made it easy to port to different chip sets and computer systems. UNIX has a modular design, with different functions being implemented by logically separate parts of the code. This has made it much simpler to debug, upgrade, add, and subtract features.

UNIX now follows the **POSIX** (**Portable Operating System Interface)** standard written in 1986. POSIX is a set of standards and protocols for APIs, utilities, shells, services, bus communications, and other low level aspects of the operating system. The use of POSIX makes it relatively easy to port programs and applications between UNIX-like operating systems. These factors have created a broad community that can usefully interact together, and makes collaborative open source projects such as Linux and Apache much more feasible and valuable.

At first, AT&T shared UNIX with other research groups and universities. Operating systems were not considered money making products at this time, and their study and development by the community at large was encouraged. The University of California at Berkeley, for example, installed a version of UNIX on one of its minicomputers in 1974. Faculty and students immediately began to modify, rewrite, and add to the code. This eventually became **BSD** (**Berkeley Software Distribution**) which was a free version of a UNIX-like operating system.

By the late 1970s, AT&T realized the value of UNIX and began to license it. Both Sun Microsystem's Solaris operating system and NeXT Inc.'s NeXTSTEP operating system are derived from UNIX through these licenses. NeXT Inc. was a company that produced high-end workstations founded by Steve Jobs in 1986 after he left Apple. The Sun SPARCstation that used the **SunOS** (1989) also competed in this high-end market.

Apple acquired NeXT Inc. in 1996, and Steve Jobs returned to the company he founded. Apple had been working a major revision of its macOS for many years, but with limited success. Jobs made the decision to scrap these efforts and start fresh by building a new OS based on NeXTSTEP and therefore on UNIX. **Apple OS X** (2001) and all subsequent Apple OS releases have been mostly closed source, proprietary, but based on the UNIX kernel,

AT&T's commercialization of UNIX caused a great deal of concern in the academic community which thought that code should be free and open for study and modification. This lead Richard Stallman to start the **GNU (Gnu is Not Unix)** project in 1983 to create a UNIX-like OS that was completely free of any propriety, patented, or copyrighted code. The project was successful in many respects, especially in producing a GUI. Making a working kernel turned out to be more challenging.

Fortunately, Finnish graduate student named Linus Torvalds produced and released a free and open source kernel he had written called **Linux** (1991). This was derived from an OS called **Minux** which was a micro-kernel version of UNIX. When the Linux kernel was bundled with the GNU tools, utilities, libraries, and GUI, the total package makes a complete operating system most often called Linux, although purists argue that it is really GNU/Linux.

In the meantime, ownership of AT&T's UNIX business passed to Novell in 1990 and finally to **SCO (Santa Cruz Operation)** in 1995. These companies claimed that they held the intellectual property rights to parts of the code included in Linux, and both sued, and threatened to sue, users and makers of Linux products. The specifics about what exactly was being infringed upon were a bit vague, but many companies paid license fees to SCO and Novel to avoid being taken to court.

At the time, this was called a **FUD (Fear, Uncertainty, and Doubt)** campaign. Making ownership unclear and creating a potential for liability was enough to convince many companies to pay to avoid trouble, and many simply avoided Linux products just in case.

By creating FUD, vendors of official UNIX-like systems would win by default. The outcome of the suit itself was unimportant, in fact, dragging it out was in the interest of SCO. Fortunately, the strategy was unsuccessful. SCO sued IBM in 2003, but by 2007, the major claim was thrown out. The suits continued until 2011, but few in the industry continued to be worried.

Linux has had a huge impact and is distributed in many ways. There are several different versions, aimed at different types of users. Notable **Distributions** or **Distros** include Red Hat, Fedora, SUSE, Debian, Ubuntu, Mint, Mandriva, Slackware, and Gentoo.

Android OS (2008) is an open source fork of Linux specialized to touch screens and mobile devices developed by Google. It can be used and modified freely, but the Android brand and much of the software it includes, such as Google Play and Google Mobile services, are proprietary and closed. **Amazon Fire OS** (2011) is a fork of Android specialized to touch screen digital content consumption devices.

Chrome OS (2009) is another Google project, but one aimed at thin client laptops that use web-based applications, instead of locally installed software. This was released as a proprietary OS, but at the same time, much of the code was released as **Chromium OS** in the form of an open source project.

Thus, Google hoped to benefit from the input of the open source community, while at the same time, keeping a closed, proprietary, version for its own use. The objective of Chrome seems to be to move users into a browser-centric mode of using laptops. Users already spent a great deal of time on social media sites and watching content, but Google wanted to complete this move by offering office applications, cloud storage, and an integrated suite of calendars, email, social media, and other web-apps that would lock users into the Googleverse.

Subsection 7.2.4. Markets and Market Share

To summarize, there are really only two significant families of operating systems today. The first is the UNIX family which gave birth to the Apple and Solaris OSs. Linux is another child of UNIX, which in addition to spiting into many sibling distributions, has spawned a large set of grandchild forks including Android, Chrome, Chromium, and Fire OS. The culture of this family tends to open source and free, although some derivatives are not. The second is the Windows family which is closed source, propriety, and produced by Microsoft.

We conclude with a table that gives a sense of the market share of these different OSs on various platform types as of 2019. Note that the numbers do not sum to 100 because there are other minor OSs in some markets (Blackberry's mobile OS and the Wii OS for gaming consoles, for example).

Operating System Market Share in 2019

	Windows	Apple	Android/Chrome	Linux
Desktops and Laptops	80	9.7	.3	2.1
Tablets	.1	71	29	.1
Mobile	.2	22	76	0
Gaming Consoles	96	3.3	0	0.8
Servers	36	20	0	40
Supercomputers	0	0	0	100

Operating systems are often used as an instrument for price discrimination. Most common is versioning where bundles of features that are likely to appeal to the average user are put together in a “home” version of the OS. Features that are likely to be needed by enterprises, or for use in high volume servers, are put together in more expensive “pro” or “server” versions. This is a form of identification by market segmentation, that is, third degree price discrimination.

Quantity discounts are also offered, which is a form of second degree price discrimination. Finally, Microsoft in particular has an army of salesmen whose only job is to negotiate individual prices and bundles with companies in different sectors, and in different countries. This is a form of first degree price discrimination.

Section 7.3. Server Stacks

LAMP is an acronym that stands for Linux, Apache HTTP Server, MySQL, and PHP. Collectively, these four pieces of software are called a **LAMP Stack** and provide all the functions needed to run a web server. In a LAMP stack, Linux provides the basic operating system, Apache HTTP Server is responsible for taking and responding to requests for webpages and other types of content from users on the network. MySQL holds and retrieves information from a database in response to queries delivered by PHP scripts, possibly triggered by different types of user input.

A key feature of this particular stack is that all four components are free and open source. Some of these components can be switched out for other FOSS alternatives. For example, Python or Perl can be substituted for PHP. It is also possible to substitute proprietary software to get an equivalent stack. The most common is called a **WIMP Stack** and uses Windows, Microsoft Internet Information Service, MSSQL and PHP. A compromise between these two is called a **WAMP Stack** (Windows, Apache, MS/MySQL and PHP). **MAMP** and **XAMP** stacks use the macOS and OS X, respectively, along with some combination of server, database and scripting applications.

We have already discussed OSs in detail in the previous section, and so we will focus on the other three elements of the stack below.

Subsection 7.3.1. HTTP Server Software

HTTP Server Software is a class of programs that take and respond to **HTTP (HyperText Transfer Protocol)** requests from clients. An instance of server software running on a computer is refereed to as a “server”.

Several servers may run on single computer, a single computer may be dedicated to run a single instance of server software as its exclusive task, or a single instance of a server may be spread over several real or virtual machines in a cloud environment. The latter is especially common for high traffic websites or for database servers dealing with large demands or complicated data structures. Web servers primarily host websites, gaming software, large enterprise level applications, and email and file sharing systems.

The most popular server software in 2019 was Apache followed by nginx, both of which are open source and free, followed by Microsoft's IIS, which is closed source and propriety, Apache is used on about 40% of servers, nginx on about 30%, and IIS on about 13%.

Subsection 7.3.2. Scripting Languages

Scripting Languages are a family of high-level programing languages that allow programs to interact with the operating system, other programs, and even peripheral devices by issuing a set of commands that the computer executes in sequence.

The most important scripting languages now in use are general purpose, and are most often employed as “glue languages” that link together programs and system components. Leading examples include PHP, Python, Perl, Ruby, VBScript, and JavaScript. They are especially useful in the following applications:

- Quickly generating and testing potential solutions to programing problems which will be folded into to larger complied programs if they are successful.
- Writing small, simple programs to do repetitive tasks.
- Providing a link between a web-server and a database application or sever. When used in this capacity, they are sometimes called “glue languages”.

Subsection 7.3.3. Databases

At the most basic level, a **Database** is both a collection of data, and the way it is structured and organized. There are several main approaches.

Spreadsheets: This is a simple kind a data object in which entries are placed in a table of **cells** with a set number of rows and columns, and are not really databases. Spreadsheets are tool for visualizing and doing simple manipulations on small sets of data. Data elements are not strongly typed (does a column contain floating point numbers, phone numbers, text restricted to a certain length, etc.). nor can relationships across data elements be defined. Spreadsheets are intended for a single user and purpose. A class grade book is a good example.

Relational Databases: an approach in which data is spread across more than one table. These tables are “related” via a **Key-Value Pair** architecture. For example, each user who submits a paper to a journal gets a unique username, which becomes the key. The table that stores his address information, and the table that stores the set of submissions the journal has received, use this key to relate specific rows of the address or submission data to each user. It might also use the paper number as another key to relate submissions to a table of referees providing reports, editors handing papers, or published issues of the journal.

Almost all enterprise databases are relational and use the **SQL (Structured Query Language)** standard for data management. SQL handles, data definition, data manipulation, and data queries. Examples of SQL implementations include MySQL, PostgreSQL, MSSQL, Oracle, SAP, and IBM DB2.

Databases fit into the server stack by answering queries, and sending results to HTTP servers that provide elements for dynamically built webpages. For example, a SQL server might give a list of flights, times, and prices, that are used to build a response to customer querry on a travel platform.

The relational database model was first proposed by Edward Codd at IBM in 1970. IBM developed an elementary relational database called System R, but it was still the test phase as of 1979. In large part, this was due to limitations in storage and computational technologies at the time.

Larry Ellison, the founder of Oracle Corporation, actually beat IBM to the market and released the first version of his database software in 1978. This built on Ellison's and his partners' understanding of Codd's academic papers describing the theory of relational databases. IBM's first commercial database product was released in 1981 and was quickly renamed DB2. Both Oracle and DB2 were written as mainframe programs.

XML (Extensible Markup Language): An approach based on a generalization of HTML which allows for very detailed and flexible metadata tags. These can be exploited to create a large file that contains tagged entries which can be searched, edited, and easily moved across platforms. The downside is that these files can be very large, and the query and administrative applications available are not as advanced as they are for relational databases. An example of an entry in an XML data file is below:

```
<CD>

<TITLE>Sitting on the Dock of the Bay</TITLE>

<ARTIST>Otis Redding</ARTIST>

<COUNTRY>USA</COUNTRY>

<COMPANY>Atlantic</COMPANY>

<PRICE>7.90</PRICE>

<YEAR>1987</YEAR>

</CD>
```

More recently, big data applications have started working with very large, and unstructured data, sometimes called **BLOBs** (**Binary Large Objects**). These are **NoSQL** databases, and for the most part, are not amenable to XML tagging.

Big Data: Google, Amazon, government agencies, and other large organizations have enormous amounts of data stored. Often this is held in incompatible systems that are poorly linked, or in data structures that do not interoperate well. Extracting useful information from big data is difficult and resource intensive. One technique is called **Data Mining.** This might be described as using artificial intelligence, machine learning, human intuition, and brute force, to discover patterns hidden in very large datasets.

Section 7.4. Economics

Subsection 7.4.1. Price Discrimination.

What if the monopolist could sell to different agents at different prices? Suppose, for example, I owned the only two cars in town, and they are worth $\$500$ on the outside market. I want to sell them to two people, Alice who is willing to pay up to $\$2000$, and Bob who would pay at most $\$1500$. What do I need in order to be able to charge these consumers different prices?

Monopoly Power: If I try to charge above MC and there are other competitive firms that can enter the market (or who already exist), they can undercut my price and still make profits. Thus, I must have market power if I hope to charge anything other than MC.

Identification: If I cannot tell who the high demand agent is, both will claim to be the low demand agent. Thus, I can either sell one car at $\$2000$ or two cars at $\$1500$. If I can tell which agent is which, on the other hand, I can just set the price at their individual reservation levels, and tell them to take it or leave it.

No Resale: Even if I can identify an agent's willingness to pay, if the low demand agent can walk in and get the low price and then turn around and sell to the high demand agent, then I will not be able to take advantage of the high demand agent.

If all these conditions are satisfied, exactly how a firm maximizes profit usually depends on the strategy of identification. There are three broad categories of price discrimination. Bear in mind that most of what we see in the real-world falls somewhere in between these categories.

First Degree Price Discrimination: A marketing strategy in which every consumer is charged exactly his reservation price.

The good news is that if a firm can perfectly identify agents' willingnesses to pay, and therefore engages in perfect price discrimination by charging each agent exactly his reservation price, there is no market inefficiency. The firm has an incentive to sell goods as long as there is any agent with a marginal benefit above the firm's marginal cost As a result, the firm produces the competitive free market quantity. The bad news is that the firm takes *all* the surplus, and the consumer surplus is zero. Consumers as a group are worse off than if they faced a single monopoly price.

For example, Amazon, among others, employs a team of economists to analyze the browsing, purchasing, and other behaviors of its users. It experiments with different prices for goods in order to determine what someone with a given data profile would be willing to pay. Putting together thousands of experiments, with data from millions of users, and billions of interactions, Amazon is working hard to figure out your personal reservation price for various goods, as well as how presenting and steering your search results might affect your eventual purchases. Users going to Amazon from different zip codes or countries, with different collections of cookies, get different offers, prices, and experiences. Even worse, once you log in, the data set that allows Amazon to do these things is enriched by the entire record of everything you have ever done on the site. On behalf of economists, at least, I apologize for this.

Second Degree Price Discrimination: A marketing strategy in which different units of the good are priced differently.

Such pricing schemes may take the form of quantity discounts or what is called **Versioning** of products. Both of these are examples of a more general strategy called **Bundling.**

Bundling: A marketing strategy in which specific amounts of one or more goods are sold in a package at a fixed price instead of individually at per unit prices.

Versioning: A marketing strategy in which a firm offers a set of related products with different bundles of features.

The point of versioning is to extract the highest possible price from high demand consumers while still being able to sell goods to lower demand consumers at a lower price. For example, a firm might offer a cell phone without a camera for a low price, but also a higher priced version with a camera for the premium market. Low demand and high demand agents identify themselves by the buying choices they make. Resale is irrelevant since no high demand consumer wants the camera-less phone.

Quantity Discounts: A marketing strategy in which a firm charges high per unit prices to consumers who wish to buy small amounts of a good, but lower per unit prices to consumers who are willing to buy large amounts.

For example, you can buy a can of Coke from a vending machine for $\$2$. You can also buy a case of 12 cans of Coke at the grocery store for $\$6$. Clearly, the per-unit price is much lower in the store, but you have to buy bundles of 12 cans at once to take advantage of this. People who use vending machines are thirsty, and lacked the foresight or energy to bring a Coke with them. People who buy cases typically have cars, can shop around for the best price, and have many alternative drinks right in front of them to choose from when they buy.

Thus, we have people with both high and low willingness to pay for a can of Coke. If Coke tried to charge $\$2$ a can in the grocery store, many people would buy Pepsi, the store brand of soda, or some other drink. Coke would lose these sales. There is no need to offer discounts at a vending machine if the day is hot, and there is no other vending machine in sight. Those who came unprepared, have a high willingness to pay. Again, people identify themselves by their buying choices and resale is generally impractical, although some leakage may occur.

Tying: Product tying is the practice of making the purchase of one product a mandatory condition of purchasing a second product.

The classic example of tying is razor blades. Gillette sold shavers that held the blades at a loss, but charged a high price for the proprietary blades that it required. In so doing, it identified high demanders who shaved a lot though their voluntary purchase of blades, and at the same time charged them more for the whole product. It was able to do so without losing the market for lower demand users who would not have purchase a shaver if Gillette had tried to extract monopoly profits with this part of the product.

Other examples include IBM in the 1960s that charged a relatively low price for its mainframes, but required its customers to purchase all the punch-cards the computer needed from IBM at a monopoly price. Thus, IBM in net charged for computing services in proportion to the size of the demand of each individual customer. This was better than setting a single monopoly price for mainframes which would have excluded many lower demand customers. Game consoles today are priced in a similar way, with companies making profits on subscriptions and game purchases rather than hardware.

Third Degree Price Discrimination: A marketing strategy in which a firm charges different prices for the same good in different markets.

Firms can benefit if they can identify markets, or markets segment with different elasticities of demand. Each market is offered a different price, and does not have access to prices offered in the other markets. Preventing resale is sometimes easy, but is often tricky.

For example, the price for textbooks is different in different countries. The wiliness to pay for a textbook in a poor country like India is much lower than it is in the US. The demand curves are quite different. Thus, publishers often put out an Indian edition that sells for a fraction of price of the US version. This allows them to sell to, and profit from, both markets.

What prevents resale? The cost of shipping a book from India to the US is a deterrent as well as the fact that the overseas editions are often printed in paperback and on cheaper paper. Publishers have also made the claim that such books are sold under the condition that it only be used in the country they were intended for, and export out of the country is illegal.

Fortunately, this claim turns out to be a violation of the first sale doctrine. As a result, a secondary market has arisen in which entrepreneurs in India and other countries buy local editions in bulk, and ship then by container back to the US, Europe, and other more expensive markets.

Chapter 8. Encoding, Encrypting, Hashing, and Security Protocols

Encoding, encrypting and hashing are different ways of transforming data and are performed with different objectives in mind.

Encoding: Data is encoded so improve its usability in some way.

Encrypting: Data is encrypted in order confine meaningful access to authorized users. An encrypted file is refereed to as **Ciphertext**, while the unencrypted file is refereed to as **Plaintext.**

Hashing: Data is run through a hashing function to generate a kind of digital fingerprint that is essentially unique to the data file. The point of hashing is not to hide data, but to allow verification that the data has not been tampered with in any fashion. The hash of a file cannot be "unhashed" back into the original file.

Section 8.1. Encoding

Encoding can be reversed (at least approximately) using publicly available technology. For example:

ASCII: An encoding standard that maps 127 letters, numbers, symbols to different values of one, eight bit, byte.

Unicode: An encoding standard that maps a more general set of symbols to different values of up four bytes.

MP3: An encoding standard that maps samples of analog audio streams made thousands of times a second to a stream of digital measurements. Encoding sound is a **Lossy** process in that the analog signal contain an infinity of data that is reduced to a finite sample. Reversing the encoding returns an approximation of the original signal.

Digital files are all long binary strings of zeros and ones. Files typically have a type, such as PDF, DOC, GIF, etc. This type defines the encoding schema that is needed to interpret all these zeros and ones. For example, a TXT file is read as a sequence of ASCII characters, while a DOC file may contain headers and metadata followed by characters that are **Parsed** as Unicode, ASCII, UTF-8, etc.

Serializing: the process of encoding a data structure of object using some well-divined schema so that it can be stored or transmitted in a form that can be later be reconstructed (deserialized) by another computer or system.

Section 8.2. Hashing

Hashing is a method used to verify the integrity of a message or file. The **Hash Function** itself is public knowledge and does not need any kind of key to work. When a message or file (called the **Pre-image**) is run through a hash function, it produces a fixed-length output string (called the **Hash-Digest**).

SHA256 Hash Examples

Hash(Nothing is easy) =
80e9f91a7086d500f6761caccb9ab8c4c07b04d3269d94c814e72293b8a558bb

Hash(Nothing is easy!) =
071dd76622bb5adb8172fc49ad4076a250abc3362804ff037d67ee1afb35b485

Notice that the two pre-images differ only by the addition of an exclamation point, but the hash-digests are completely different.

Hash(Never appeal to a man’s better nature. He may not have one. Invoking his self-interest gives you more leverage.) =
188ce5f0bcb2dee714cc31085e359527b724ac41f1ed511330193346ff7e9dff

Notice here that the last pre-image is a phrase with 111 characters (and so would be represented by 111 bytes if encoded in ASCII). The hash-digest is represented by 64 hexadecimal numbers, just as the first two shorter pre-images were. This illustrates that the hash-digest is always the same length regardless of the size of the pre-image.

The Secure Hash Algorithm-256 (SHA-256) is one widely used approach are said to have following properties:

- Running any file, regardless of length, through SHA-256 returns a 256 bit binary string.
- Very similar files produce quite different hashes in an unpredictable way. In fact, SHA-256 is designed so that hashes of different files are, in effect, randomly and uniformly distributed over the set of all possible 256 binary strings of which there are 2^{256} , or about, 10^{77}.
- The hash of any file is unique. The same input always produces the same output. For this reason, the hash is sometimes referred to as a file’s **Fingerprint**.
- Although hashing a file always gives the same result, two files may have the same hash. This is called a **Collision**. In practice, however, this extremely unlikely to occur.
- Hash functions are not invertible. It is impossible to recover a file from its hash.

Two of these points need slight qualification.

First, collisions must occur in theory. A typical MP3 file is tens of millions of bits long. Obviously, it is impossible to do a one-to-one mapping between the set of every possible 10 million bit sting and every possible 256 bit string. However, If I took a hash of any file and then started looking for another file with the same hash, I could test $10^{77}/2$ files and still have only a 50% chance of finding one. Thus, it is extremely unlikely that you will ever actually see a collision.

Second, hash functions are not random. In fact, one of their most important characteristics is that they are deterministic. However, if you took an arbitrary file, modified it one bit at a time, and then looked at the distribution of the resulting hashes, they would be approximately uniformly distributed between (0, … , 0) and (1, … , 1) where these are each 256 bit long binary strings. In other words, they deterministically map files of arbitrary length into an unpredictable, almost uniform, distribution of every possible 32 byte string.

One final note. **Hexadecimal** (**Hex**) is a standard way to encode binary strings to make them more readable to humans. There are 16 hexadecimal numbers, (0-9, and a-f), each representing a unique four-bit binary value. Thus, two hex numbers are required to represent one byte. Using the example above we can translate hex to the original binary:

188ce5f0bcb2dee714cc31085e359527b724ac41f1ed511330193346ff7e9dff ≡

0001 1000 1000 1100 1110 0101 1111 0000 1011 1100 1011 0010 1101 1110 1110 0111
0001 0100 1100 1100 0011 0001 0000 1000 0101 1110 0011 0101 1001 0101 0010 0111
1011 0111 0010 0100 1010 1100 0100 0001 1111 0001 1110 1101 0101 0001 0001 0011
0011 0000 0001 1001 0011 0011 0100 0110 1111 1111 0111 1110 1001 1101 1111 1111

where:

0001 ≡ 1, … , 1100 ≡ c, …, 1111 ≡ f, etc.

Section 8.3. Encryption

Encryption is meant to keep private files private. If data stored on a hard drive, or other fixed media, is encrypted it is said to be **Encrypted at Rest**. If you encrypt packets or messages before they leave your system, they are said to be **Encrypted in Transit**. An encrypted file can be made be publicly available, but without a key of some sort, the encryption cannot be reversed, and so the file is unreadable.

At the most basic level, encryption transforms a message or data file using an algorithm known to both the sender and receiver. The classic **Transposition Cypher** is the simplest example. The idea is that each letter is mapped to a different letter in the alphabet. For example, the alphabet could be transposed one position so that *a* becomes *b*, *b* becomes *c*, and so on. Thus, the plaintext "Local Zoo" becomes the ciphertext "Mpdbm App".

Subsection 8.3.1. Symmetric Encryption

Modern cyphers are more sophisticated, but at root, use a similar approach. Plaintext is subjected to a series of substitutions and permutations determined by a **Key** which is a string of bits of specified length. For example, **AES 256 (Advanced Encryption Standard 256)** uses a 256-bit key, implying that there are $1.2 x 10^{77}$ different ways that the substitutions and permutations might be executed on plaintext to produce ciphertext.

AES 256 is seen as a completely secure cypher system given current computer technology. The only way to decrypt a file is to know the right key. Breaking encryption therefore requires that the key be discovered by brute-force guessing.

Suppose that a hacker had enough computational power to make 10^{12} (one quadrillion) guesses per second. Such a hacker could make about 3×10^{19} guesses per year. To have a 10% chance of guessing which of the $1.2 x 10^{77}$ possible keys was correct, the hacker would have to test $1.2 x 10^{76}$ keys. This would take approximately 4×10^{56} years. The universe is only about $13.7 x 10^{9}$ years old, and so it is unlikely that a hacker would care very much about the about finding the key, even if he eventually succeeds.

There is no need to hide the encryption algorithm itself. In fact, it is essential for it to open to inspection and audit by experts. Otherwise, there is no basis for users to be confident if has no backdoors or potential exploits. Of course, the value of any given encryption key must be known only to authorized parties.

AES 256 has two main advantages. First, it can be used to encrypt files of any length, and even streaming data. Second it is designed to be computationally efficient. That is, very few computational cycles are required to encrypt or decrypt a given amount of plaintext if the key is known. Without the key, however, brute-force attempts to invert the cypher are computationally impractical.

Although AES is a very secure system, it depends on the sender and receiver sharing knowledge of a common key (sometimes called a **Shared Secret**). Since the key is used both to encrypt plaintext, and decrypt ciphertext, AES is and example of what is called **Symmetric Cryptography**.

Subsection 8.3.2. Asymmetric Encryption

How can users agree on a key while keeping it secret from everyone else? Obviously, sending the key in an unencrypted form exposes it to interception. On the other hand, the sender cannot send the key to the receiver in an encrypted form unless the receiver has a key to decrypt encrypted key. We have a chicken and egg problem.

There are several ways to solve this. For example, agents might meet in person and agree upon a key. An even more sophisticated approach would be for users to meet in person and agree upon a book and an algorithm such as: "Use the first 256 characters on the N^th line of page X where n is today's date, and X is a page number to be sent in an open email". Unless the book the two agents have agreed upon is known to others, the page number is useless information. This system allows the sender and receiver to generate a new key each time they communicate.

Both of these solutions depend upon users having at least one completely secure, unencrypted exchange of information. This is impractical if users do not know in advance that they may wish to communicate securely. For example, I may wish to send a secure email to someone I have never met, or give a credit card number to a merchant I have never used before.

Where it is impractical for users to meet securely and agree upon a shared secret, **Public-Private Key Encryption (PPK)**, which is a form of **Asymmetric Cryptography,** can be used instead. The drawback is that decrypting text without a shared key takes much more computational effort. In addition, the amount of plaintext that can be encrypted is less than the size of private key. As a result, PPK encryption is often used only to begin a secure communication session by sending a symmetric encryption key to be used by both sides for the remainder of session.

The real magic of PPK encryption is that the public and private keys have special mathematically relationship. Not only is the private key the one and only way to decrypt a message encrypted with the complementary public key, but the *public key is the one and only way to decrypt message encrypted by the complementary private key.* This symmetry will turn out to be essential to blockchain, **SSL/TLS Certificates** (**Secure Socket Layer/Transport Layer Security Certificates**) and many other applications.

At the highest level, public key encryption works like this:

1. The receiver generates two large numbers using a PPK generation algorithm. One is called a **Public Key**, and is made freely available. The other is call a **Private Key** and is kept secret by the receiver.
2. A sender uses the receiver's public key to encrypt his message.
3. The receiver uses his private key to decrypt the message.

The two most common PPK encryption schema are the following:

Rivest-Shamir-Adleman (RSA): A PPK encryption schema first developed in 1977 which is much less computationally efficient, and requires a significantly longer key sizes, to give the same level of security as symmetric key encryption. For example a 1024-bit RSA key gives the same security guarantee as an 80-bit AES symmetric key, while a 2048-bit RSA key is as good as a 112-bit symmetric key.

Elliptic Curve Cryptography (ECC): A newer approach to asymmetric encryption that is more computationally efficient and also provides more security for any given key length. A 224-bit ECC key, for example, is roughly as good as a 112-bit symmetric key.

Subsection 8.3.3. Client-Side Encryption

Cloud backup services like OneDrive and Dropbox, and enterprise applications built on Azure or AWS, hold sensitive client data, and use encryption to keep it secure. A common approach is to use a shared symmetric key to encrypt data that is transmitted between a client's computer and the cloud service. This encrypts data in transit, but not at rest. If either the client or cloud service is hacked, the data is revealed.

A better approach is to use a symmetric private key to encrypt data end-to-end. That is, encrypt the data on both the client's computer and the cloud provider's server, and only transmit data in encrypted form. This encrypts data in transit and at rest on both the local, and cloud, disks.

The problem with this approach is that if the cloud service knows the shared encryption key, it can read your data. Even if you trust the cloud provider, a hacker may find a way to steal the encryption keys, or a government agency may compel the provider to turn over your unencrypted data.

Thus, an even better approach is **Client-side Encryption** which keeps the key in hands of client instead of making it a shared secret with the cloud provider. This key is used to encrypt data on the client's local disk, and only encrypted files are sent to the cloud server. Since the cloud provider does not know the private key, it is unable to read the client's files.

If the cloud provider is hacked, only encrypted files without keys are compromised. Even if the government ordered the cloud provider to turn over client data, it could only give up encrypted files. Since the cloud provider does not have the key, it is technologically impossible for it to turn over cleartext client data.

The way this works in practice is that you create an account on some provider using a long, and secure, password. This password is used locally as a seed to generate a symmetrize key. (Don't worry, the cloud provider only sees a hash of your password, and so cannot reconstruct your key.) Everything you send to the provider, including a directory listing, is encrypted locally before it is uploaded. You can request specific encrypted files using the tags in the encrypted directory listing you sent. These files, are then sent to you in encrypted form. Your local machine can reconstruct your key as needed from your password, and so can decrypt any file you download.

Subsection 8.3.4. Cool Tricks with Hash Functions

Hashing can be used in a number of interesting ways:

Verifying Documents: Suppose you sign a partnership agreement, and later on have a dispute. You and your partner both present copies of the contract to a judge, but they say different things. How can the judge determine which is genuine? Suppose the judge had access to a hash of the contract as it was on the day it was signed. He could then compare it to hashes of the contracts presented to him by you and your partner. The genuine contract will produce the correct hash, and any altered contract will produce something else. You may ask how the judge might have access to a hash of the contract he knows was taken on the day it was signed. Read on about digital signatures and blockchain below.

Verifying Logins: When a user tries to log into a system, the system compares the password the user provides to the one stored in its files. The problem with this is that the password file might be stolen by hackers. Thus, it is good security practice to store only hashes of user credentials. Users still type in passwords, but the system then hashes them and compares them to the hashed passwords stored in its files. If the password is correct, the hashed password will match a stored hash. If the hashed password file is stolen, it is of no use. The password cannot be determined from the hash, and the system requires a correct plaintext password to grant access, not the hash itself.

Securely Resetting Passwords: Sometimes when you forget a password, you can request that it be sent from the website to your verified email address. This is not the most secure practice since email often transits the Internet in plaintext and can be stored on any intermediate server that passes the message along. However, if a website follows the practice of only storing hashes of passwords, it literally cannot send a user his password since it does not know it itself. This is why you often get a message asking you to click on a one-time link, that quickly expires, and then choose a new password. This password is immediately hashed and stored. For security, this system relies on three things: that the email address has previously been validated, the use of two part authentication in many cases, and the fact that a hacker would have to intercept and use the link immediately while having access to your email account.

Privacy Preserving Data Matching: Suppose we all joined a social network and wanted to know if any of the people in our address books had also joined. None of us, however, wanted to reveal our own email address or the contents of our address books to the network. We could instead submit a hashed version of our address books. The network would be able to see that one person's email address was in another person's address book by finding identical hashes. However, the network would not be able to read the email addresses themselves.

Efficient Database Search: Suppose we had a database that contained names, email addresses, email message texts, documents, audio recording, case histories, and other large and dissimilar items. Suppose we wanted to find the record that contained a specific document or email message. Running a standard search looking for a match in the database would be computational intensive, since it would require comparing data elements that are many kilobytes or megabytes long against the document we wanted to find. We could instead create a database that contained hashed versions of the all the data elements and search for a match with a hash of the document we were looking for. Using a **Hash Table** requires comparing data elements that are only 256 bits long, and special search algorithms have been developed that are very efficient.

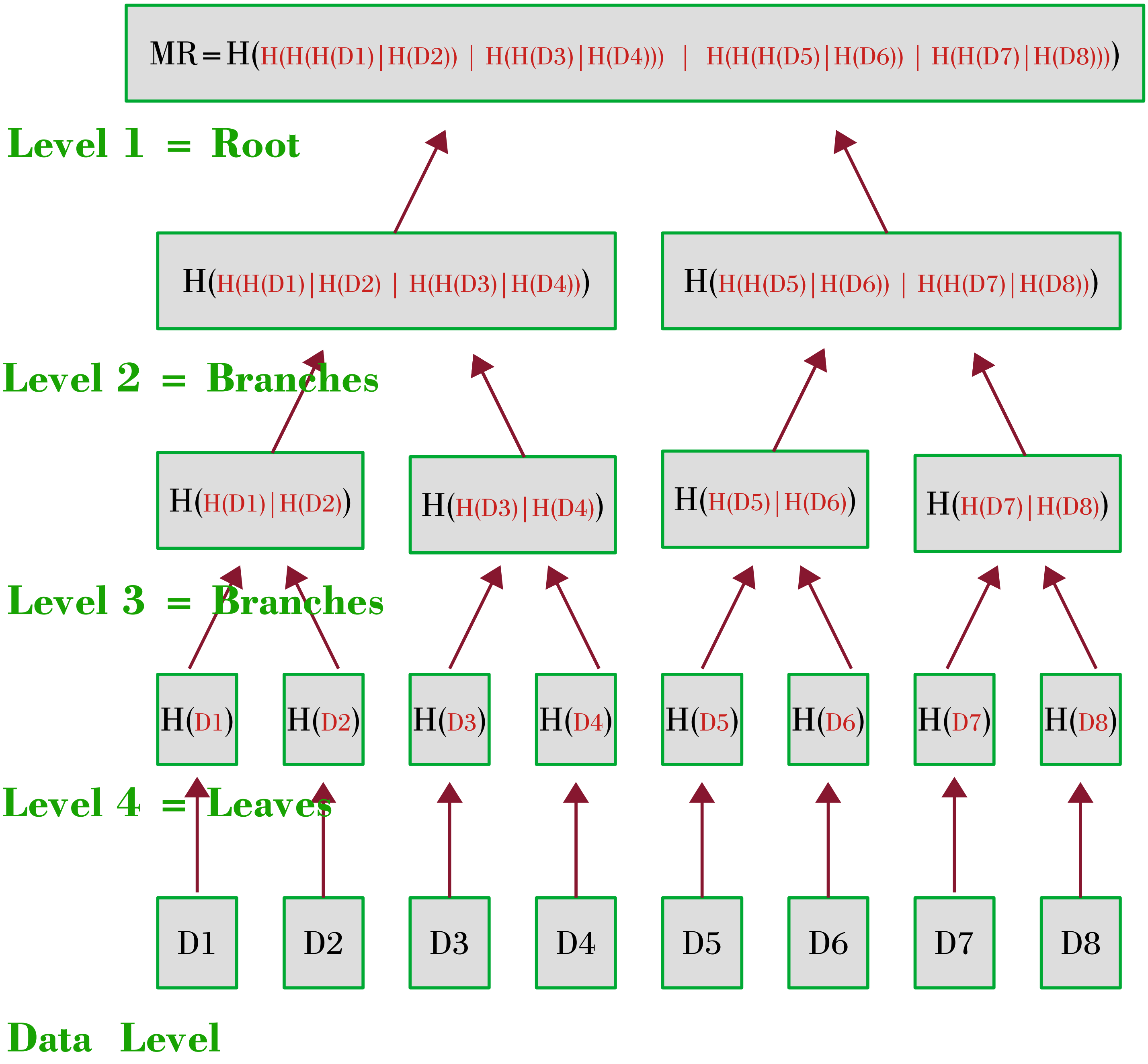
Verifying Data Structures: Merkle Trees are used in blockchain and other data system applications to prove that specific data elements are included in the data structure. Verification requires knowledge of the **Merkle Root** of Merkle tree as well as series of hash-digests collectively called a **Merkle Proof**. Databases often contain gigabytes of private information spread over millions or billions of separate elements. The value of Merkle proofs is that they allow the data maintainer to prove that any specific element that happens to be of interest is in fact part of the data structure with a very small, privacy preserving message. For example, proving that a data set with a billion elements contained a given element would require a proof with only one kilobyte of data. The subsection below explains how this works in detail. It is very cool, but not really necessary to understand unless you like this kind of thing.

Subsection 8.3.5. Merkle Trees and Proofs

Consider the Merkle tree below. It is built from a data structure that happens to contain eight data elements, but could contain any number. The basic data elements could be of any type or size: blockchain transactions, medical records, even movies, music, and types of other content.

The Merkle tree is constructed in layers:

- First, a hash-digest of each data element is created (think of these as leaves on a tree).
- Second, these hash-digests are paired up, and then each pair is hashed again. In the example, we have eight original hash-digests (leaves). This allows us to form four **Concatenated** pairs (concatenated a technical word that simply means written sequentially as a group). This creates the next level of the Merkle tree, (think of this as branch).
- Third, we do the same thing with the four hash-digests we just made. That is we, concatenate them into two pairs, and make a new level with two hash-digests.
- Finally, we concatenate the remaining two, and then hash them to get a single top level hash called the **Merkle Root**.



An Example of Merkle Tree

Note the following:

- The example is based on eight data elements and the resulting Merkle tree has 15 hashes ($(2\times8)-1$) on four levels, one less than the lowest power of two that gives a number ate least as large as the number of data elements $(8 = 2^3)$.

- In general, suppose that a data structure had N elements and that $2^{(p+1)} < N \leq 2^{p}$. Then the Merkle tree would need $L = p + 1$ levels, and $H = 2p - 1$ hash-digests ($H - N$ of these would be leaves with “null” hashes or fillers, but ignore this detail for now).
- Each of the hashes in the Merkle tree requires 32 bytes (if SHA-256 is used) and so a Merkle tree for a data structure with a million elements would have a bit over two million hashes, 21 levels, and require only 64 MB of information in total. The structure itself might contain gigabytes of data, so a Merkle tree is a very data efficient way of getting a detailed fingerprint of the data structure as a whole.

Suppose you sent a transaction to a blockchain, and you wanted proof that it was included in a committed block and so the transaction was finalized. Alternatively, you might want proof that you registered to vote, or that a filing for a permit or legal action had be recorded and accepted by the relevant agency.

You could ask who ever maintains the data structure if your data is there, but this would not allow you to prove anything if the data was erased or modified later. You could ask the data maintainer to send you copy of the entire data structure cryptographically signed (as outlined below). This would allow you to prove that your data was part of as data structure, but it might require transmitting gigabytes of that is irrelevant data to you. The maintainer might refuse to do this for reasons of cost, or because some of the data was private or privileged.

The main value of a Merkle tree is that it allows parsimonious (that is, data efficient) proofs that a specific piece of a data is in part of a larger data structure without the need to reveal details of any other data the structure contains.

Consider our example again. Suppose that you wanted to know that D5, your Bitcoin transaction, for example, was in a given block. A Merkle Proof of inclusion would require the Bitcoin node (the data maintainer) to send you exactly four hash-digests, a total of 128 Bytes (that is, an eighth of a kilobyte). Below we show how his works:

The Merkle proof consists of the following four hashes, and some indicator of whether the three branch hashes are concatenated to the left or right:

$$\boxed{MR = H(H(H(H(D1) \mid H(D2)) \mid H(H(D3)+H(D4))) \mid H(H(H(D5) \mid H(D6)) \mid H(H(D7) \mid H(D8))))}$$

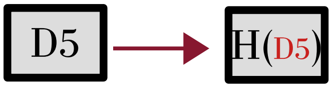
$$\boxed{H(H(H(D1) \mid H(D2)) \mid H(H(D3) \mid H(D4)))} \quad \text{(L)}$$

$$\boxed{H(H(D7) \mid H(D8))} \quad \text{(R)}$$

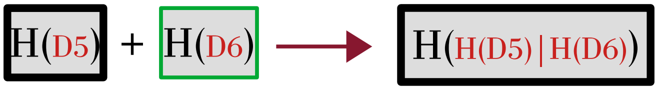
$$\boxed{H(D6)} \quad \text{(R)}$$

Note that sending these hashes does not let the receiver know anything about the data they were created from. The receiver requesting proof would do the following (the items in black frames are known to receiver in advance or are created by the receiver, while the green frames are items that were sent as part of the Merkle proof).

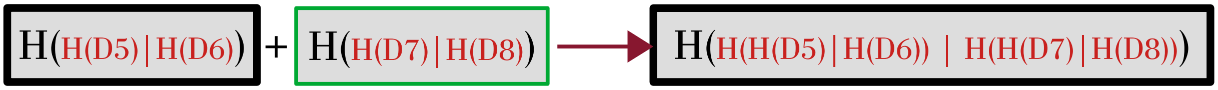
Step 1: Take the hash of your data (which the receiver knows and is trying to verify).



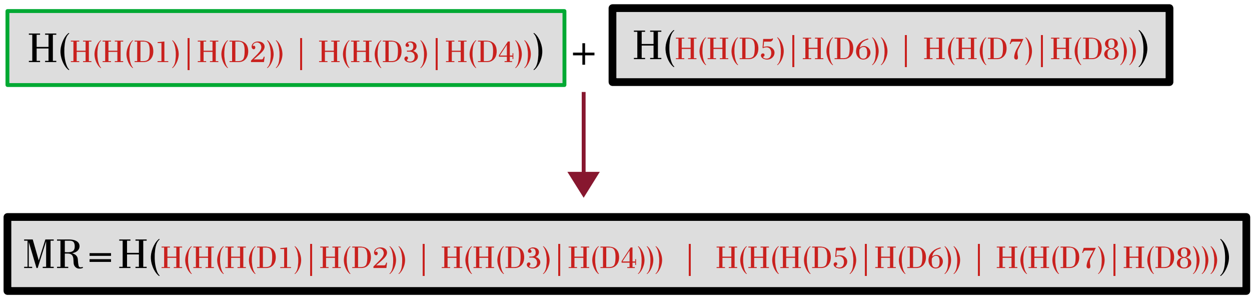
Step 2: Take the hash just made, concatenate to the right of the level four Merkle hash from the proof, and take their hash.



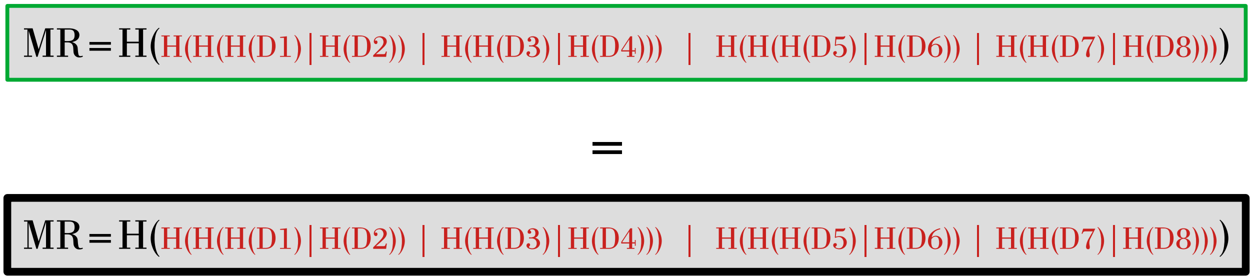
Step 3: Rinse and Repeat—Take the hash you just made, concatenate to the right of the level three Merkle hash from the proof, and take their hash.



Step 4: And again to get your own version of the Merkle root—Take the hash you just made, concatenate to the left of the level two Merkle hash from the proof, and take their hash.


H(H(H(D1)|H(D2)) | H(H(D3)|H(D4)))
+
H(H(H(D5)|H(D6)) | H(H(D7)|H(D8)))
MR=H(H(H(H(D1)|H(D2)) | H(H(D3)|H(D4))) | H(H(H(D5)|H(D6)) | H(H(D7)|H(D8))))

Step 5: Compare the Merkle root you just created with the one sent by the data maintainer.


MR=H(H(H(H(D1)|H(D2)) | H(H(D3)|H(D4))) | H(H(H(D5)|H(D6)) | H(H(D7)|H(D8))))
=
MR=H(H(H(H(D1)|H(D2)) | H(H(D3)|H(D4))) | H(H(H(D5)|H(D6)) | H(H(D7)|H(D8))))

What you need to know: You need to know the full plaintext of the data for which you want the proof of inclusion. You also need to know the Merkle root of the data structure. This last part is key. Merkle roots are published, widely distributed, and signed for this reason.

What this proves: What is being proved here that a data structure with a given Merkle root includes a specific piece of data. If your data is latter omitted or changed, the new modified data structure will have a different Merkle root. This only helps if the fact you can prove that the maintainer has changed the data, invalidates the data, and any harmful effects that it might have had if accepted. For example, blockchains are supposed have immutable data, and so such a change would violate protocol, and result in sanctions for dishonest nodes.

Why it works: Remember that hash functions create uniformly distribution hash-digests. Thus, it is computationally impractical for the data maintainer (or anyone else) to find a 32 byte string by guesswork that would hash correctly when concatenated and hashed with the hash of your data and continue to hash correctly all the way up the tree to the Merkle root. This process of trying to find a pre-image of a hash-digest is exactly the same computational problem as solving Bitcoin's cryptographic puzzle. but set to maximum difficulty. That is, it is effectively impossible to claim that your data is in a structure with a given Merkle root unless it is true (meaning that it would take too long and cost too much to construct a false, but apparently correct, Merkle proof).

What this buys you: Parsimonious proof of inclusion. Proving that your data is in a data structure with one billion elements requires a Merkle proof with 31 hashes, just under one kilobyte of data.

Section 8.4. Applications that Combine Encryption and Hashing

Sone very interesting things become possible when encryption and hashing are used together.

Digital Signatures: Signing paper documents is the traditional way of indicating agreement or acknowledging receipt. Signatures can be forged, so banks and notary publics require that people show identification documents such as passports or driver's licenses. These documents often include photographs, signatures, physical descriptions, and even fingerprints. In effect, these documents are an **Attestation** by a government agency or whoever issued the documents that it believes that the person named on the ID document is the same one in the photograph, uses a signature that looks like the one on the document, has a certain fingerprint, and so on.

There are several weaknesses to this approach. First, we have to trust in the truthfulness and due diligence of the document issuer if we are to believe its attestation. This is why banks ask for official government IDs instead of your work ID or AAA card. Second, the document could be forged. Third, the ID could be stolen. These last two problems might be solved if we could request a copy or image of the document from the issuing agency. This would make it immediately apparent if the document being presented is forged, altered, stolen, or revoked.

In the digital world, public private key (PPK) pairs combined with hashes can be used to sign documents, messages, transactions, and any other type of digital object. Signing and verifying signatures works as follows:

1. The signer produces a hash of the document.
2. The signer encrypts the hash with his *private key*.
3. The signer attaches this encrypted hash to the unencrypted (cleartext) document
4. To verify the signature, the reader decrypts the hash using the signer's *public* key. The decrypted hash could only have been encrypted in this exact way by the holder of the complementary private key (that is, the signer). Thus, the reader knows that this is the correct hash of the document as it was signed by the private key-holder.
5. Finally, the verifier produces his own hash of the unencrypted document. If this hash matches the decrypted hash in the signature, then he knows the document is exactly what was signed and has not been changed in any way.

For this to work, the verifier must have access to the public key that can decrypt the encrypted hash, and believe that this public key belongs to a specific person or entity. The verifier also has to believe that the owner of the public key had control of the corresponding private key when the document was signed.

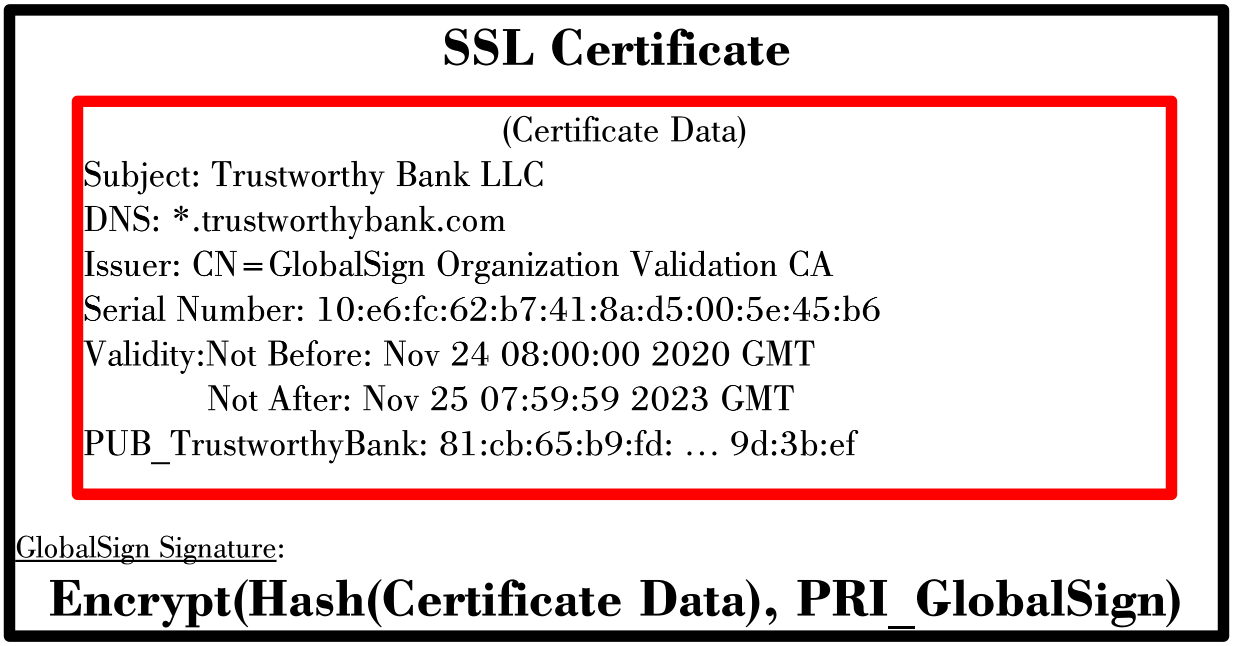
If a hacker managed to get his hands on the private key, he could use it to sign documents just as easily as the true owner. Thus, if we think the key has not been stolen or compromised, and we are confident that we know the real-world person or entity who owns and controls the key, then we can be equally confident that the person or entity signed the document verified by matching hashes.

TLS/SSL Certificates: There are small files that can be attached to an email, requested from a web or email server, or from a server set up specifically to hold SSL certificates. (Actually, TLS been used since 1999, but they are still commonly referred to as SSL certificates). SSL certificates are issued by a **CAs (Certificate Authorities),** such as VeriSign, Comodo, Globalsign, Google, or Microsoft, and consist of six basic elements[21]:

- The **domain or host name** that the certificate was issued for.
- The **real world identity** of the certificate holder.
- The **name of the CA** issuing the certificate.
- A **serial number**, and an issue and **expiration date** for the certificate.
- The **public key of certificate holde**r.
- A **digital signature** from the certificate-issuing authority. This includes a hashed version of the certificate data encrypted using the **certificate issuer's (the CA) private key**.

[21] As of 2019, there are more than 200 root certificate authorities. It is no longer clear that all are trustworthy or that users can consistently identify correct certificate credentials.

An SSL certificate looks something like:



SSL Certificate

(Certificate Data)

Subject: Trustworthy Bank LLC
DNS: *.trustworthybank.com
Issuer: CN=GlobalSign Organization Validation CA
Serial Number: 10:e6:fc:62:b7:41:8a:d5:00:5e:45:b6
Validity:Not Before: Nov 24 08:00:00 2020 GMT
Not After: Nov 25 07:59:59 2023 GMT
PUB_TrustworthyBank: 81:cb:65:b9:fd: … 9d:3b:ef

GlobalSign Signature:

Encrypt(Hash(Certificate Data), PRI_GlobalSign)

How can we be confident that we know the true, real-world, identity of the public key used to decrypt the hash in a digital signature? Just like physical signatures, we need some kind of digital ID document to give us confidence that the owner of the public key is really who he claims, and that the corresponding private key is under his exclusive control. SSL certificates are used as follows to help us accomplish this:

1. A user contacts the certificate holder's server/website a*nd obtains the certificate*.
2. The user finds the CA's public key. Alternatively, the SSL certificate contains information about the CA which can be used to find the CA's certificate online (which contains its public key).
3. The user applies the CA's public key to decrypt the signature (which is the hash of the certificate data encrypted with the CA's private key).

$$\text{Decrypt(GlobalSign Signature, PUB\_GlobalSign)} = \overline{\text{Hash(Certificate Data)}}$$

4. The user hashes the certificate data (no key needed).

$$\text{Hash(Certificate Data)}$$

5. The user compares the hashed certificate data he generates, with the decrypted hash made by the CA:

$$\text{Hash(Certificate Data)} = \overline{\text{Hash(Certificate Data)}}?$$

6. If they match then the user knows the following:

a. The holder of the private key, PRI_GlobalSign, which corresponds to public key, PUB_GlobalSign, attests that PUB_TrustworthyBank is the *owner of the domain *.trustworthybank.com.*

b. If the user believes that PUB_GlobalSign is the public key of Global sign, then the user also believes that the real-world entity called GlobalSign attests to (a).

c. If the user trusts GlobalSign as a real-world entity, then the user trusts that (a) is true.

d. Finally, given all the above, the user believes he can securely use PUB_TrustworthyBank to encrypt a symmetric encryption key that can be sent to *.trustworthybank.com, and then used to establish a secure communications channel.

Note two things:

First, all of these steps are done automatically by your browser. Browsers often come preconfigured with a list of trusted CA's along with their public keys. These can be changed by the users through the setup menus, but the browser takes this list as user input of which CAs to trust. If everything checks out as described above, a secure connection is established. A cute little lock will then appear on the user's address bar telling him that an **HTTPS** protocol session has begun.

Second, this procedure **does not** establish the real-world identity of the public key-holder, or the website. It only establishes that the CA attests that an entity self-identifying as Trustworthy Bank LLC owns the URL, *.trustworthybank.com, and has offered a public key. PUB_TrustworthyBank, to establish HTTPS sessions with the web server that supports *.trustworthybank.com.

CAs do not have the time or reassures to verify that Trustworthy Bank LLC, is in fact the Delaware corporation that runs certain brick and mortar bank branches, and has control of *.trustworthybank.com at any given moment. Any random bad actor could register the URL BofAmerica.-com, or BofA.io, and get a correct SSL certificate issued to him. The CA does not opine as to whether this registering entity is the same Bank of America the user happens to have in mind. This is why businesses typically register or buy as many URLs that might generate confusion, across all the most popular TDLs (.com, .net, .org, etc.)

What TLS/SSL certificates actually prove that a CA attests that and a URL is officially registered the and claimed identity, who has in turn, offered a public key. Even if this good enough (and it usually is in practice), this process just backs the identity verification problem up one level. We still have to decide if we trust the CA, and we have found the correct public key for this trusted CA.

The **PKI (Public Key Infrastructure)** is a system of hardware, software, policies, protocols, and participants that issue, revoke, and identify the owners, of public keys.

The PKI based on what is called the **Web of Trust**. The CA who signed a certificate will also have one or more SSL certificates issued by other CAs attesting to public key of the signing CA. The issuers of those certificates, in turn, will also have SSL certificates issued by other CAs, and so on. Provided we can find at least one CA in this chain that we believe is trustworthy, then the rest of the structure is verified. There is no central authority in the web of trust by design. If you cannot find CA in the attestation chain for whom you believe you have a correct public key and whose honesty you trust, SSL certificates are of no help.

Section 8.5. Economics

There are a number of interesting economic principles that encryption and hashing technologies depend on.

Asymmetric Costs: Encryption works at an economic level because it is computationally easy (and therefore economically cheap) to encrypt and decrypt data if you have the proper key, but computationally impractical (and therefore prohibitively expensive) to decrypt ciphertext without the key. Similarly, it is easy to verify that a given file hashes to a specific digest, but essentially impossible to find a pre-image that hashes to a given hash-digest.

Precommitment: Cryptographically signing a document creates a nonrefutable record that parties agreed at a certain time, to a contract specifying obligations and compensations, that may depend on facts that are not yet known. Notary publics offer the same type of service, but notarized documents only have the attestation that a human notary was satisfied that the identification documents offered belonged to the parties signing the contract. Neither a copy, nor a hash of the contract is kept by the notary public, and so it could be altered by either party. Notarized or digitally signed contracts may or may not be enforced, or even be enforceable, by courts. but digital signatures remove any doubt about what is being adjudicated.

Incomplete Information: When agents do not know everything about an economic environment, we say that information is **Incomplete**. When agents know the same things about an economic environment, whether or not they know everything, we say that information is **Symmetric**. When agents know different things about an economic environment, we say that information is **Asymmetric.** Asymmetric information implies that at least some agents do not know everything, which implies that information must also be incomplete. When an agent knows something that no other agent knows, we say that he has **Private Information**, which implies that information is also incomplete and asymmetric. Agents with private information often wish that there was a way to credibly convey it to another party sicne counterparites may be unwilling to engage in an otherwise profitable transactions without it.

- Merkle proofs are a way to prove the existence and content of data structures without needing to share other confidential information. Merkle proofs are durable in the sense that either party can keep copies of a proof, and then later prove what the data state was when it was created. If digitally signed, they are a durable proof of what the signatory attests to regarding the data state.
- The PKI and Web or Trust are ways of proving identities, and so resolve what would otherwise be private information that could not be credibly conveyed.

- Client-side encryption allows users to send private information to untrusted entities for storage and backup without being concerned that it might be made public by a faithless provider.
- SSL certificates and encrypted tunneling remove the requirement that we trust communication networks we don’t control. Thus, cryptography vastly expands the market for network services, which otherwise would only carry data for which confidentiality was unimportant.

Section 8.6. Appendix – Protocols and Applications to Communications

SSL (Secure Sockets Layer) and **TLS (Transport Layer Security):** Internet protocols used to establish secure sessions between clients and servers. Once the user receives the server's certificate, he uses the public key it contains to encrypt and send a 128-bit or 256-bit key which will then be a shared secret and can then be used for symmetric encryption for the rest of the session.

SSH (Secure Shell): A method of creating a secure encrypted tunnel between a client and a server. SSH uses **Tunneling** or **Port-Forwarding** to connect a client to a server through an insecure public network in such a way that routers directing the packets on the public network are unaware that they are in fact encrypted private network traffic. The difference between SSL/TLS and SSH is that the latter is not an Internet protocol. SSH traffic goes between the client and server on an unencrypted connection. However, the traffic itself is encrypted by the sender, and decrypted by the receiver upon arrival. An SSH tunnel is designed to support many kinds of traffic, including file transfers, shell commands, and HTTP data.

VPN (Virtual Private Network): This is an extension of a LAN over public infrastructure. Most commonly, an employee will use the Internet to connect directly his employer's server, and thus, to his employer's private network. Although it is not required, in most cases, the connection is made over a secure tunnel using SSL, TLS, or some other protocol. This allows the remote user to operate as a trusted, but remote, node within a private network.

Note that commercial VPN services actually combine aspects of a VPN as described above, and a proxy server, described below:

- When you log into a commercial VPN, you technically become part of their LAN. However, the only resource you are given access to is the gateway, though which you contact the WAN. You typically would not interact with other VPN users, or access hardware or data that might be somewhere in the VPN's network,
- All traffic is encrypted between your home router and the VPN's router, and usually between the VPN's router and the Internet URL you are trying to contact. The VPN's gateway, however, sees unencrypted versions of both your requests, and the responses.
- VPNs typically offer users a choice of servers, physically located in several jurisdictions. Not only do VPNs mask the IP address of the user requesting content from an external URL, but they can make it appear that the requests are originating from different states or outcries.

Proxy Server: When a router or server that connects to the WAN, it does so through an IP address provided by the ISP from which it gets Internet access. The ISP, therefore, knows which URLs are contacted, and what data is transmitted. The ISP can also block access to certain URLs if it wishes.

A user, however, may choose to alter his Internet configuration to route all requests through a proxy server. When a user requests a URL, the request is transmitted directly to the IP address of the proxy server, which then contacts the desired URL, and then sends the requested content back to the user. Thus, all requests from the user go to the proxy server, and so the user's ISP cannot tell who the use's router is really contacting. If the connection to the proxy server is encrypted, then the ISP will not know the content of the traffic either.

If the proxy server happens to be in a different country than the user, then all servers will treat requests as if they are coming from the location of proxy server and not the user. Sites that are blocked by the user's government or ISP can be accessed through the proxy server. As a bonus, this also allows a user to defeat content providers' habit of releasing new products at different times, for different prices, and under different terms, in different markets. A user in France would be able to get access to shows that have been released in the US, but embargoed in France, by using a US based proxy server.

Thus, the proxy servers allow a user can gain privacy and much greater access to content. Of course, the proxy server provider can read any unencrypted content sent or received by the user just like an ISP, so some caution is required. In addition, the better proxy server companies charge for access, and the ones that are free, are generally slow. Finally, governments know about proxy servers and often block access to their IP addresses when they are discovered.

Tor (The Onion Router): This is an elaboration on the idea of a proxy server. Tor is free and open source software that allows users to access the Internet through a series of proxies using encrypted links. Tor works as follows:

1. When a user wishes to send a message, or contact a certain URL, his Tor client first contacts a directory server with a list of Tor nodes, and their public keys.
2. The user's client selects a path using several Tor nodes to the target URL.
3. Suppose there are three Tor nodes on the path, an Entry node, Guard node (in reality, there are several), and an eXit node, and a Destination node.
   a. The user generates a symmetric key, SYN_D and uses it to encrypt his message. The user also has obtained the public key of the destination node, PUB_D, and uses it to encrypt SYN_D. Finally, the user adds a plaintext header identifying the final destination (although using special Onion Address system rather than the DNS/IP infrastructure). This is the innermost layer of the onion, LAY_D.
   b. The user generates a symmetric key, SYN_X and uses it to encrypt LAY_D. The user also has obtained the public key the eXit node, PUB_X, and uses it to encrypt SYN_X. Finally, the user adds a plaintext header identifying the exit node. This is the next layer of the onion LAY_X.
   c. The use does the same to generate LAY_G, and LAY_E for the guard and entry nodes.

d. Having constructed the onion message, the user sends LAY_E which consists of SYN_E encrypted with PUB_E, and also LAY_D plus a plaintext pointer to the guard node, all encrypted with SYN_E.

e. The entry node decrypted SYN_E using PRI_E, known only to him. The plaintext routing instructions tell him the next link in the chain. Hi forwards LAY_D to the guard node, who follows the same procedure.

4. Any response is sent back down the same chain to the user using a similar procedure. Tor chains are altered every ten minutes or so for added security.

Note that as each node decrypts its layer of the onion, it only sees where the message should be sent, and an encrypted symmetric key and message. He cannot decrypt either, and the node also does not know where it is the chain, nor how long the chain is.

The Tor network is built on free and open-source software, developed by the Naval Research Laboratory, and released in 2002. Its original intention was to serve as a conduit for political dissidents, journalists, and others, to communicate to the outside world without detection. Tor still serves this purpose, and is also used to preserve user privacy and anonymity in broader circumstances.

Tor is used for criminal purposes as well. Most famously, the Silk Road was a Tor-based website that specialized in selling illegal drugs. If you make the right connections on Tor, you can find guns, counterfeit money, stolen goods, botnets, and almost anything else that you can think of. This is more of a change of venue for a small slice of criminal activity than something really new. Illegal goods and services of all kinds are available through many other digital platforms, and have be available in the physical world as far back as civilization goes. What you could get depended on knowing where to look, and having the right (wrong?) people vouch for you. So it is today.

Chapter 9. Authentication

Section 9.1. Basic Elements

We begin by defining some of the basic elements.

Authentication: Verifying that a user is who he says he is, and is authorized to do what he is requesting. Authentication is based on one of three classes of information:

- **Something You Are**: Biometrics such fingerprints, facial geometry, iris morphology, or even DNA.
- **Something You Know**: A password, or the answer to a security question, a private encryption key.
- **Something You Have**: A smart cards, cell phone, physical key, or access to an email account.

Two, or Multipart-Authentication: An approach that requires that a user pass more than one authentication test, usually chosen from different information classes.

Permissioning: A more granular use of authentication in which verified users in a system are given different levels of access and authority over the data and resources in a system.

Permissioning tables sometimes define a set of privilege or authority levels, and associate each user with one of these levels. Permissions can also be assigned on a user by user basis, and may depend on context. For example:

- A customer might be allowed to access only his own account records, not account records in general
- A doctor might be allowed to access a specific patient’s files, but only if he is the patient's doctor of record at the time
- A technical support person might only have access only records relating to the customer he is currently on the phone with.
- On the other hand, All board members might be given access to all of a company's financial records based on their status as members.

Section 9.2. Something You Are

Users sometimes provide scans of their fingerprints, retinal patterns, or faces, to the computer as they set up their accounts. Each time they log in after this, they allow the computer to scan their thumb, eye, or face, and compare it the stored scan. Biometrics such as these can be used in place of passwords, or as an element of two-part authentication.

Biometric authentication might seem to be convenient and secure. After all, you do not have to remember your fingerprint, and no one other human has a fingerprint exactly like yours.

Of course, there is an enormous potential for misuse of facial recognition, and similar technologies by oppressive governments, business who may wish to identify and exploit customers, and others who have reasons to erode personal privacy. Leaving aside these larger issues, there are also concerns with biometrics as a safe and secure form of authentication:

- If a password is stolen or compromised, you can simply reset it. If your fingerprints are stolen somehow, you are out of luck. It is very difficult change your fingerprints, retinal pattern, or face.
- Many current biometric scanners can be spoofed by pictures of faces, retinas, or fingerprints. Latex, and even Play-Doh, impressions of fingerprints are not hard to make, and are even more successful as spoofing scanners.
- Even if scanning technology improves, at root, the scanner is just a connected digital device. It only really “sees” the digitized file that it sends to the system. It cannot see the scan itself or verify that the input came from the hardware. If the scanner. or the connection between it and the network, is not secure, then a hacker could simply feed a stolen or created file to the system or intercept the file of an actual scan, and replace it with another.
- There are reports that a few people have had fingerprints surgically altered, and of course, plastic surgery on faces is old news. While it might be difficult to imitate another person with these techniques, it is not at all difficult to become someone else. Terrorists on watch lists and people with outstanding warrants, and criminals, can hide from authorities by altering their biometrics. A person with a new fingerprint cannot be connected to any of his history via biometrics in this event.
- All this aside, biometric systems are only reliable if the records they depend upon are secure. A hacker could replace the file containing the scan on your fingerprint with this own. This takes identity theft to a whole new level. Alternatively, a hacker could take your fingerprint file, and substitute it for his. or that of a dangerous criminal. Just as with any authentication system, the files, or access levels, connected to your scanned biometric credentials can also be altered.

It is worth mentioning **FIDO (Fast ID Online) Alliance,** a non-profit organization that is developing a set of secure authentication standards to address some of these problems. FIDO compliant authentication schemes keep biometric scans, templates, (as well as public keys and other types of identifiers) encrypted on user devices, and so never let them get uploaded or transferred to a database out of the user's control. All authentication happens on the client side instead of in the cloud. While FIDO can't help you change your fingerprint or retinal pattern if it somehow gets captured or associated with a different identity, it can make such capture and misuse less likely.

Section 9.3. Something You Know

Passwords, security questions, and private encryption keys are examples, of things that you know that are used for authentication. Passwords and answers to security questions are usually stored in hashed form on authentication servers. The plaintext supplied by users as they log in is hashed and then compared to the corresponding value in the hash table.

PPK private encryption keys are used a little differently, and mostly for different purposes. Like passwords, they are information known only to the user. Unlike passwords, they are long binary strings, often rendered in hex, that are extremely difficult for a human to remember, much less transcribe accurately. Instead, the are kept on some (hopefully) secure storage media, and used in **Trusted Execution Environments**, or **Confidential Computing Secure Enclaves**, to sign transactions, create encrypted requests gain access to resources, or directly authenticate.

Passwords, and security questions are frequently compromised. Users write them on Post-it notes. tell friends, send them unprotected email, and so on, and systems containing records of passwords or questions are sometimes hacked, or are compromised by faithless insiders. Even if the password is kept secure, a hacker may be able to discover it directly by guessing.

The first set of problems can be largely addressed simply by not being stupid (er, I mean, incautious). Hide your passwords, do not tell them to anyone, and design systems that store authentication data in an encrypted or "hashed" states.

The last problem (hacking) can be addressed by being a little clever. Far too many users have terrible passwords, which they seldom change, and use repeatedly across many platforms and applications.

Subsection 9.3.1. What Makes a Password Terrible?

People are terrible at choosing and managing passwords for devices, accounts, documents, and encryption. This may be because people are unaware of the nature of the security threat, and the technologies that hackers have available. Fortunately, there are ways to choose secure passwords that a user can remember. Let's begin by outlining what you should not do when choosing a password.

Use a Common Password: For example, *123456*, *password*, *qwerty*, *letmein*, *abc123*, *111111*, *master*, and *trustno1* are all among the top 25 most common passwords, which collectively make up something on the scale of 2.5% of all passwords used. Lists of the 1000 or 10,000 most common passwords are well-known to hackers.

Reuse a Password: Surveys by Google and others estimate that 65% of people, and 76% of millennials, use the same password for many, or even all, of their accounts. When forced to update passwords, about half of users change a single digit or character. This means that if *any one* of the sites you visit is hacked or is itself untrustworthy, then *all* of your online financial, social media, communications, and probably much more, becomes open and insecure. In other words, your identity is only as safe as the weakest link.

Use a Dictionary Word or Common Term: A password like *cubs rule, Titanic, Rocket ship, Rachel, Semper fi, antidisestablishmentarianism,* or *July 17, 1991*, might appear to be pretty good. How could a hacker know your favorite team, movie, first girlfriend, anniversary, or that you like USMC space vehicles? These might seem to be obscure and unguessable, however, there are just not that many dictionary words, proper names, common phrases, or dates. There are on the scale of one million English words, of which only 10,000 to 50,000 are in common usage. There are about a third of a million dates per century. Thus, while a hacker might not be able to associate any one of these million or two possible passwords to you personally, if he makes a few hundred thousand guesses, he will break into your system.

Use Simple Substitutions or Variations on the Above: You might try to make your password more obscure by using: *Cu3s RU1e*, or *Titanic123* instead of *cubs rule* or *Titanic.* This does create more password possibilities, but not as many as you would think. This is because while *C* or *c*, or even *K* or *k,* might be used to spell *Cubs*, few people would use *q, Z* or * in place of the *c.* It is much harder to remember that you substituted *Zubs* or **ubs* for *cubs,* than *Kubs* or *Cubs.* Similarly, *123, ABC, 000* and other easy to remember strings are often added to dictionary words to get passwords like *Enterprise000.* It is far less common to add strings like *7^o* or *p$4k2.*

To crack a password, several shortcuts are tried before resorting to pure brute-force. First, hackers go through a list of known common passwords. Next, they try a list of dictionary words, phrases, dates, and numbers. If this does not work, they try common substitutions and variations on these lists. At his point, maybe a billion or so guesses have been attempted. Even seemingly obscure words and complicated substitutions will come up on such a list.

Choosing a one in a billion password should provide enough scrutiny, right? It must take a very long time to make a billion guesses! It turns out that it takes much less than you might imagine. Applications are available which can easily feed a target site 1000 guesses per second. It would take 11.5 days to go through a one billion word list. If the hacker can get the password file offline or gain possession of the device that holds the file (a local attack), then a high-end desktop can run an application that can make 100 billion guesses per second. If the hacker has access to a grid network, or NSA type computers, 100 trillion guesses per second or more are possible.

You might wonder whether this is something to be concerned about in the real-world. After all, many systems and services lock a user out if three bad password guesses are made in a row, or enforce a delay of several seconds between guesses.

Unfortunately, many systems have APIs, or include applications, that do not use such lockouts. Hackers can knock on these side-doors remotely through insecure networks as often as they wish. In addition, a hacker can attack thousands or millions of accounts at once. Billions of guesses are still made, but are spread over many different accounts.

Statistically, making a billion guesses for one account yields the same result on average as one guess made to one billion different accounts. Thus, lockouts have little effect in aggregate. Most corporate and public systems encrypt the millions of passwords they hold. However, if system security is weak, or if a Trojan or social engineering attack is successful, a hacker can steal, and then work with, the file offline and use the much faster approaches outlined above.

Subsection 9.3.2. How to Choose a Good Password

So what is a user to do? The best advice is this:

FORGET ABOUT OBSCURITY AND FOCUS ON COMPLEXITY

Consider this password: **=pOIK93j@*. Each element can be a lower or upper case letter (52 possibilities) a number (10 possibilities) or a symbol (20 or so possibilities depending on which symbols are allowed). Thus, each element can be one of 82 different characters. If the password is 10 characters long, then there are $82^{10}=1.4\,x\,10^{19}$ different passwords possible. That is, 14 billion, billion. Even with 100 billion guesses per second, it would take six months to try all the possibilities. The obvious problem with this sort of password is that while it takes a lot of effort for a hacker to guess, it also takes a lot of effort for a user to remember.

What is needed are passwords that are complex in a computational sense, but are easy for a user to remember. Consider this password: *happyfoodzebrapark*. This is a bit silly, but maybe not so hard to remember. As a human, you only have to remember four words. A hacker, however, has to figure out 18 characters. There are about 20,000 four and five-letter words in English. Of course, there is no need to stick to only four and five-letter words, and you can add capitals and numbers if you wish. For the purposes of illustration, however, here are the consequences:

There are:

- 10 numbers.
- 26 lower case letters.
- 62 upper and lower case letters and numbers.

- 82 upper and lower case letters, numbers, and symbols.
- 20,000 four and five-letter words in the English language.

Suppose we consider passwords with different numbers of elements drawn for each of these sets. The table below shows the number of passwords that are possible.

Password Complexity						
Types of elements	**Number of elements**					
	4	**6**	**8**	**10**	**12**	**20**
Numbers only	10^{4}	10^{6}	10^{8}	10^{10}	10^{12}	10^{20}
Lower case only	$4.6 x 10^{5}$	$3.0 x 10^{8}$	$2.0 x 10^{11}$	$1.4 x 10^{14}$	$9.5 x 10^{16}$	$2.0 x 10^{28}$
Letters and numbers	$1.4 x 10^{8}$	$5.6 x 10^{10}$	$2.2 x 10^{14}$	$8.4 x 10^{17}$	$3.2 x 10^{21}$	$7.0 x 10^{35}$
Letters, number, and symbols	$4.5 x 10^{8}$	$3 x 10^{11}$	$2.0 x 10^{15}$	$1.4 x 10^{19}$	$9.2 x 10^{22}$	$1.9 x 10^{38}$
Four and five-letter words	1.6×10^{17}	6.4×10^{25}	2.6×10^{34}	$1.0 x 10^{43}$	$4.1 x 10^{51}$	$1.0 x 10^{86}$

In the case of four, lower case, four or five-letter words we can get 160 quadrillion different passwords, If we removed the word restriction and allowed a fully random selection of 20 upper and lower case letters, numbers, and symbols, on the other hand, the number of passwords possible increases by twenty orders of magnitude. However, you can probably remember froghaveblob zooms, while 0Q9folda*7;3aBds1l#9 is all but impossible to recall. How much is gained if you try anyway?

Consider a hacker using various technical approaches at brute-force guessing. These approach allow different numbers of guesses per second. The list below shows how many guesses per year each of these approaches makes possible:

$3 x 10^{10}$ total guesses at <u>one thousand</u> guesses per second in a year. (remote attack)

$3 x 10^{16}$ total guesses at <u>one billion</u> guesses per second in a year. (strong remote attack)

$3 x 10^{18}$ total guesses at <u>one hundred billion</u> guesses per second in a year. (strong local attack)

$3 x 10^{21}$ total guesses at <u>one hundred trillion</u> guesses per second in a year. (NSA)

This means that even if you have bad network security practices but used a password with four, four or five-letter words, a typical hacker would need more than two million years to have a 50% chance of guessing your password. If you added one word, but had your device stolen by a very determined hacker, it would take him 500 years. If you added a sixth word, it would take the NSA itself 3000 years.

Here is a summary of best practices:

- Password length and obscurity are only paths to password complexity and are not important in themselves.
- Choose passwords from candidate sets of at least 10^{16}, and ideally, 10^{25} or more. (Four to six random words for example.)
- Never reuse passwords
- Never choose passwords that are common or have anything to do with you (especially with any personal details that might be found through web searches or guessed).
- Use a password manager app that employs client-side encryption (LastPass, for example) and secure it with a strong password you can remember.

Section 9.4. Something You Have

In the pre-digital age, we mainly relied on physical documents presented by humans to establish identity. A human might present a driver's license, passport, or ID card issued by company. Another human would inspect them to determine whether the credentials were genuine, and if they belonged to the human presenting them.

Physical keys were another way of determining authority, but without identifcaion. Anyone one in possession of a key to a lock had the ability unlock it, and gain access to whatever it protected. Possession of the key conferred this ability, but did not require any identity information about the user.

Cryptographic keys are classified as "something you know", but they function in the same way. Whoever knows the key is able to sign transactions, or access encrypted data, without having to establish an identity.

Plugging in a USB key, swiping a card, or holding an RFID chip next to **Near-Field Communication** (**NFC**) reader are ways of proving to a digital system a user in possession of an item that authorizes access. Such items can be lost or stolen, and users will only become aware of this the next time they try to gain access. digital systems are not equipped to check if the biological human using the item is the one to whom it was issued.

Most authentication systems in this class use either **HMAC-Based One-Time Password** (**HOTP**)[22] or **Time-Based One-Time Password** (**TOTP**) protocols. Both approaches start with a seed that is hashed, and then mapped to a six or eight digit number. HOTP has a counter that is incremented each time an OTP is generated. The current value of the counter is added to the seed so that the OTP generated will have a new value each time. TOTP is time based. Every 30 or 60 seconds, a new timestamp is added to the seed, which is then hashed to get a new value.

[22] **HMAC** stands for **Hash-Based Message Authentication Code**

The most common implementation of HOTP and TOTP is to have the authentication server generate an OTP, and send it to a user via SMS, email, or even a phone call. In this case, only the server needs to know the seed. It is also possible for the seed to be a shared secret. This allows OTPs to be generated independently by user.

The most widely-used approaches to "something you have" authentication are the following:

Security Key: A USB key usually following the FIDO2 standard that facilities HOTP and other two-factor authentication protocols. Certain websites can be configured to accept 2FA from data on the key plugged into a device, and some security keys can communicate with the devices via **Near-Field Communication** (**NFC**) and so do not need to be physically plugged in. Computers and other hardware can be set up so that they only allow access unless a security key is physically plugged in.

Tokens and Smart Cards: Devices that generate OTPs on the client side, usually using TOTP. Tokens and Smart Cards require power to work, and must also have internal clock which is synchronized fairly closely with the authentication server. Generally, they are provisioned with a seed, and then delivered to a user. Thus, the seed is a shared secret between the server and the token or card, but not with the user. TOTP credentials expire when the timestamp is incremented, which is typically one minute, or less.

Cell Phones: One of the most common ways for a user to get an OTP is to request one from a server, and have it sent to his mobile phone in the form of an SMS (text) message. Many servers also give users the option of receiving an automated phone call to get an OTP. Typically, HOTP is used, and so OTPs become invalid only when a new OPT is requested. Severs impose expiration times of 10 to 15 minutes by policy for security reasons.

Email: OTPs can also be sent to an email address, using essentially the procedures as cell phones. In this case, the "something you have" is access to a "verified" email account. Of course, you may be able to access your email from a computer, a mobile device, through public terminal by logging into webmail, and so on. The same is true to some degree SMS and phone calls. Often you can log in to a provider's site, and read stored SMS texts, or access voice mail through other devices where OTPs may have been left.

Email is not a secure way to receiving OTPs. Emails can be captured though man-in-the-middle attacks, and read by the SMTP server that hosts your address (Google, or our employer, for example). Email passwords may be compromised, and if the hacker does not do something aggressive like sending out emails on his own, the user may never know that someone is passively reading his email and intercepting his OTP messages.

Mobile phone have a different array of problems. Like email, they can be subject to man-in-the-middle attacks, and your provider is able to ready any texts you receive containing OTPs. If you leave your phone unlocked, and unattended, anyone could pick it up, start your banking app, and move money out of your account with the aid of the OTPs that arrive by text. Less than five minutes would be needed, so the user may not even be aware that anyone but him ever used the device.

If a phone is lost, but locked, bad actors usually can't pull off this trick. Many people are so attached to their phones, that they seldom put them down. A lost phone would be noticed very quickly.

The new problem that arises is that while a bad actor may not be able to access your accounts if he finds, or steals your phone, you are also locked out of your accounts.

The best case scenario for a lost phone is to buy a new one, and them convince your provider to switch over your number. This may require you to authenticate using an OTP, which you can no longer receive since your device is lost. The next step is to go to a physical store, and try to persuade the staff that you are who you say you are, and have the authority to switch over a phone number. If this does not work, you will have to get a new number, and then contact every bank, social media platform, and anyone else who uses OTPs to make them recognize your new phone number.

An even more nightmarish situation is for your service to be canceled. If your phone is on someone else account, they own the number, not you. If your carrier is mad at you, or you cannot pay your bill, your phone number is also locked up. If you are considered politically undesirable, or socially unacceptable, a provider may cancel your service because "you do not reflect their core company values". Governments may also require that providers cancel your service for these or other reasons. Should a tax evader be allowed to spend money on PCS service when he owns the government money? Should a criminal have access to a network that could in theory allow him to contact his accomplices, or threaten his victims?

The point of this is that we have come to rely on mobile phones to do things that are essential to functioning as a financial, or social, actor in a technological society. As a technology, however, it is insecure, and our access to it is fragile.

There are a number of software-based OTP services that break this dependence on a specific mobile device. Google Authenticator, Duo Mobile, Authy, Yandex, and FreeOTP are all companies that use mobile apps, computer programs, websites, or even hardware, to solve the problem. The basic approach, however, is to obtain the seed from each company you need to authenticate to, and then generate OTPs as needed locally. The seeds can all be backed up, and then transferred to new devices as needed. This makes OPT generation portable, and under the control of the user.

While many companies and platforms support the use of authenticator apps, many do not. It is not yet possible to do entirely without mobile phones for two-factor authentication, but it is possible, with some effort, to take control over significant part of your digital identify.

Section 9.5. Economics

Subsection 9.5.1. Second Price Sealed Bid Auction

Google Ads and Ebay both use a **Second Price Sealed Bid Auction**, where the highest bidder wins, but pays only the second-highest bid plus one bid increment.

This implies that it is a dominant strategy for agents to bid their true value. To see this, suppose that you bid lower than your reservation price. If you still won the auction, your payment would be one bid increment larger than the second-highest bid. Thus, you would pay the same as if you had bid your true reservation price.

Suppose instead you lost, and the winning bid was below your true reservation price. Then, you get zero payoff, while if you had bid your true value you might have won the auction purchased the good for less than your reservation price. This shows that you can never gain, and may lose, if you bid below your reservation price. The logic showing that bidding above your reservation prices is dominated by sincere bidding is similar.

Formally the game is defined as follows for the two agent case:

$i \in \{1, 2\}$: players or agents

$S_i \equiv \mathbb{N}$: strategies

$F_i(s_A, s_B) = \{0 \text{ if } s_i \leq s_j, \text{ and } P_i - s_j - b \text{ if } s_i > s_j\}$: payoff functions

In both first and second price auctions, a **Reserve Price** is sometimes used. Typically, the winning bid must at least equal the reserve price or the auction is called off and the item left unsold. The reserve price is usually kept secret from the bidders. There is a theorem in auction theory that says that there exists a reserve price that can be added to any given auction that yields strictly more revenue to the seller than the auction without the reserve price. Google makes more than $100 billion each year from such auctions, and the addition of a reserve price increases this by tens of billions.

It is still a dominant strategy to bid your true value even if there is a reserve price assuming a one-shot game. However, if a game is repeated (as Google Ads auctions are), the seller learns to how to set reserve prices at levels that maximize profits to the determent of the buyer. In this case, truthful bidding is not a dominant strategy.

Things become more complex in repeated second price auction games because the information each side has about the distribution of reservation prices evolves.

Chapter 10. Banking and Credit

Section 10.1. Charge Card, Credit Cards, and Debit Cards

Before 1950, credit was extended to customers by individual merchants. Most often, bills were settled at the end of the month. This meant that people received bills from, and had to write checks to, each merchant they dealt with.

Diners Club was founded in 1950 with the idea that it would be more efficient for merchants to extend credit through a common company. In this way, merchants would not have to track or bill individuals, and would be paid directly and promptly by Diner's Club. The company, in turn would centralize all the charges made by an individual, and send him one bill. Diner's Club charged merchants a fee of 7 percent for this service. Customers were charged $\$5$ per year and were required to pay off their charges each month. **American Express** operates in a similar way and was founded in 1958.

Bank of America issued the first true credit card in 1958. This eventually become the **Visa Card**, **Mastercard** was founded in 1966 as a competitor. Visa and MasterCard are different from American Express and Dinners Club in two important ways. First, the cards are issued through a network of member banks, rather a single company. Second, they are **Credit Cards** as opposed to **Charge Cards**. You do not have to pay off your Visa or MasterCard balance at the end of the month, and you can automatically borrow money from the issuing bank up to your credit limit.

Charges are processed by the Visa and MasterCard network, which also makes payments to merchants (usually through intermediaries that offer **Merchant Accounts** to businesses). The issuing banks are responsible for billing, collecting payments, and extending credit to customers. Merchants generally pay 2 to 3 percent plus a $.25 fix fee per transaction for these services, and customers pay interest, late fees, over-the-credit-limit fees, and sometimes annual membership fees. The advantages of credit cards are that:

- Customers are only liable for at most $\$50$ of any fraudulent charge. Debit cards receive no such protection.
- Customers can carry less cash, which reduces risk of theft or loss, and is more convenient.
- Customers can finance larger purchases immediately, and pay them off over several months (or years). Consumers would have had to save up, or apply for a formal loan, in the past.
- Customers get one bill each month instead of one from each merchant they have a charge account with.

- Customers often get cash back, frequent flier miles, or other rewards for using cards. In addition, by using a credit card for a purchase, customers receive extended warranties on goods, and liability insurance on rental cars.
- Customers who are defrauded by a merchant, or sold defective goods, can ask Visa and Mastercard to investigate and issue a chargeback if they agree. If a customer pays in cash or with a debit card, he has no way of making such a recovery.
- Merchants are assured of payment provided there is no fraud on their part, and they followed the rules when authorizing the charge. Merchants do not have to think about the credit worthiness of his customers, or if their check is likely to bounce, something they are ill-equipped to do.
- Merchants reduce their transactions costs by accepting credit cards since all the charges are sent electronically to one place for payment. They do not have to keep their own accounts or issue their own bills.
- Merchants do not need to have as much cash on hand, which protects them from theft by employees and outside criminals.
- Merchants who do not take credit cards today risks losing significant business. Customers expect to be able to pay with credit cards and may choose to avoid merchants who do not accept them. In addition, credit cards allow customers to make large purchases even when they do not have money in the bank to pay.

The Visa/Mastercard networks routinely handle tens of thousands of **Transactions Per Second (TPS)**, and can handle more than 100,000 TPS if needed.

Machine learning and artificial intelligence are now used to recognize potential fraud. As a result, some card issues have been able to drive fraudulent charges well below 1% of the total transactions. This is part of the reason that issuers offer cash back to users. Merchants are charged 2-3% per transaction which leaves close to 1% in pure profit after fraud is paid for. Card issuers return this to users to induce them to choose their cards over those of their competitors.

Debit Cards: These allow consumers to deduct charges directly from their bank account. The Visa or MasterCard networks are frequently used to process these charges. This allows customers to use them anywhere that accepts these credit cards and use most ATMs regardless of which bank they belong to.

Preloaded Credit Cards: These cards also go through the standard payment networks. On the positive side, preloaded cards offer the same consumer protections as ordinary credit cards. They are used by many people as a kind of substitute for a bank account. It is even possible to have a paycheck automatically credited to your balance every month. If the card is lost or stolen, it can be replaced, and so it is more secure than cash. On the negative side, preloaded care often include significant fees for using ATMs to get cash, for making purchases, and for reloading the cards.

Gift Cards: These are a variation on preloaded cards and are sold in fixed denominations. Some gift cards are tried to a specific merchant (such as an Amazon or iTunes gift card), and others can be used anywhere that the corresponding credit card is accepted. Gift cards often charge a fee for each use, and have monthly maintenance charges that are assessed even if the card is not used at all. In addition, gift cards must be zeroed out and cannot be reloaded. This means that you must know the balance on a card and ask that no more than this be deducted at the point of purchase. As a result, many people end up with gift cards having a small balance left over that is difficult to use.

To combat this, people sometimes buy more than they would have otherwise to make sure that everything is spent. Otherwise, the balance is eventually eroded by the monthly maintenance charge. Estimates are that about 20% of gift card balances are never redeemed. Gift cards cannot be replaced if lost or stolen.

PayPal is primarily a provider of merchant account services similar to Authorize.net and Square. The company has also tried to develop an internal payment ecosystem that draws from users' bank accounts directly, or from money paid to users' PayPal accounts by other users (for example, from eBay sales, donations made to a cause or charity, and other merchant activities). Since PayPal does not have to pay the credit card networks any fees for internal transfers, they get to keep any such fees they might charge. Google Wallet and Apple Pay do not provide financial services as such, but allow users to store credit card and other information electronically, and facilitate its use to make purchases.

There are obviously strongly network externalities at work here. There are two major, and two lesser, players in the US credit card market. Competition is not strong. Merchants essentially must take Visa and Mastercard if they want to do business, and will often take AmEx and Discover despite their higher merchant fees.

Privacy in financial services is a major concern. Many people use their credit cards for almost everything they purchase. Credit card bills show much liquor you buy, where your kids go to school, when you went on vacation, where you went, and what you did, if you are on prescription medication, subscribe to any naughty websites, buy guns, or make donations to a specific church, Greenpeace, or the Republican Party, etc. Obviously, this information could be used by marketers to create a social profile. It could also be used for criminal prosecution, civil litigation, divorce, child custody, stalking, and so on. You have very few secrets from Mr. Mastercard, and Ms. Visa knows all.

Most of the rest of consumers' financial lives runs through banks. Checks are becoming less important over time as **(ACH) Automated Clearing House** payments through bill pay systems become more widely used, and people opt for direct deposit of their earnings. The Federal Reserve System runs the largest ACH and there is only one private ACH in the US. This is quite a concentration of data in only two places. Adding the two major credit card brands, four organizations collectively know just about everything financial worth knowing about almost everyone in the country.

Wire Transfers are an older technogoy for sending and receiving money that is similar in many ways to ACH. Domestic wires within the US largely go through the Federal Reserve Bank's Fed-Wire system. The Europian Union, and other nations and coalitions of nations have similar systems. Internation wires ussually us the **Society for Worldwide Interbank Financial Telecommunication** (**Swift**) network.

Wires are typically used for larger transfer of funds between individuals, organizations, enterprises, while ACH payments tend to be used to pay bills and for smaller transfers. Wires are more secure in many ways, and allow direct transfers from any account to any other account. ACH payments go from accounts to ententes who have set up the infrastructure to receive payments rather than specific accounts.

Although wires are more secure, they are slow, inconvenient, and expensive. You can only send a wire during banking hours, and not on bank holidays. Wires initiated after two or three in the afternoon typically go out the next day, and it can take as much as an hour to set one up. Wires cost between $3 and $30 to send, and the same to receive. A typical wire will take between one and three business days to land, and sometimes considerably longer. Crypto-enthusiasts often claim that "Bitcoin fixes this". This is mostly true, almost true, or could be completely true, depending on the circumstances.

Federal Reserve Notes, also known as **Cash**, are backed with the "Full Faith and Credit of The United States Government" and are "**Legal Tender** for all debts, public and private". Dollar bills are an example of what is known as a **Bearer Instrument**. If you have a dollar bill in your physical possession, you can give it to someone else without asking anyone permission, and without creating a record of the transfer. The receiver then has full ownership of the instrument, and identical rights. This is unlike checks, or credit cards, which transfer balances, or create debt, on a central ledger.

Cash allows you to make anonymous transfers of value in exchange for goods and services. If you buy your liquor with cash, no one will know how much drink (at least there will be no record of it). In this sense, cash allows you to maintain at least some of your financial privacy.

Of course, you have to obtain cash somehow. For example, if you get cash from a bank or ATM, you leave an electronic record. In this sense, cash is similar to cryptocurrency. It allows you to create a degree of privacy and anonymity in your financial life, however, the on-ramps, and off-ramps, create significant data leakages.

Cash is medium of choice for illegal transactions, money laundering, and evation of taxes (far more popular than cryptocurrency). Governments would prefer that you create records of your financial transactions, in part, to prevent this. This is why China, and other nations, arr exploring the idea of **Central Bank Digital Currencies** (**CBDC**), India demonetized its 500 Rupee note (worth about $7), and the US government requires that a source of funds be determined before large amounts of cash can be legally accepted for certain types of transactions.

Section 10.2. Identity and the Unbanked

In most countries, banks are not allowed to open accounts without doing **KYC/AML (Know Your Customer/Anti-Money Laundering)** due diligence. The exact regulations vary between jurisdictions, but they all involve some form of checking and verifying the identity of potential customers, making sure that they are not criminal or terrorists, and that the account is not used in illegal ways to evade taxes, commit fraud, or to facilitate crimes or terrorism. This includes preventing transactions with sanctioned countries, companies, or individuals.

It is estimated that about 35% of the world's population did not have bank accounts of any kind as of 2017. Citizens of low income countries, the poor in general, and women in particular, are disproportionately affected.[23] Refugees and the displaced can lose contact with their banks or have accounts locked or confiscated. Unfortunately, AML/KYC checks automatically prevent banks from giving accounts to the billion or more people who have no official IDs.

Being **Unbanked** leads to **Financial Exclusion**, and makes it difficult for people to save money, get credit, seek jobs, start their own business, invest in education or health, manage risk, receive remittances or money transfers, keep their wealth safe, and in general, forces them into the shadow economy. Not surprisingly, the World Bank takes the position that:[24] "Financial inclusion is a key enabler to reducing poverty and boosting prosperity."

This unfortunate situation is not the fault of banks. Setting up a bank is costly. There are any number of regulatory and compliance measures that must be satisfied, and this requires lawyers, accountants, payment of fees, taxes, and sometimes bribes. Physical infrastructure is also required including buildings, employees, and IT systems.

On-boarding a new account holder at the very least requires KYC and AML, and this alone costs somewhere in the range of $40 per customer. Of course, these costs must be passed on to the customer. Even in the US, accounts with low balances may pay monthly fees of $10 or more, plus fees for each transaction. Holding $1000 in an account for a year could easily cost $200 or more in fees. The fixed cost of setting up accounts may simply not justify the benefits to either the bank, or the customer, when balances are too small.

This leaves the poorest and most vulnerable with a limited set of expensive and unattractive options. For example, the use of mobile phones to make money transfers is a growing business (especially in East Africa). However, costs are 1-2% on a $100 transaction and can range up to 10% on smaller transactions. Such services are not designed to make transfers over borders, or in different currencies, and require the user to place his faith in the phone company as a TDI.

[23] Gender gaps for bank accounts in developing countries average 9% and can range up to 30%. In low income economies more generally, less than half of the population are typically banked (see Demirguc-Kunt, *et al*. 2017).

[24] The World Bank also sees financial inclusions as an enabler of seven of its seventeen sustainable development goals (see http://www.worldbank.org/en/topic/financialinclusion/overview).

Receiving remittances from overseas can also be quite expensive for the unbanked. Transaction fees range from 4% to 8% on average in 2018, and significant additional charges are added for foreign exchange services, using credit cards and debit cards to fund transfers, and dispensing cash to the receiver. As a result, the unbanked often choose to transact only in cash. This puts them at risk of being robbed, which can be especially punitive if they are attempting to build savings, or accumulate enough cash to establish a legal identity.

Section 10.3. Economics

Subsection 10.3.1. Constraining Choices

To an economist, gift cards are a strange phenomenon. By converting a $50 bill into a gift card you make it more difficult to use, likely that part of it will not be used at all, and also constrain the recipient's choice of how to spend your gift. Even if the card is not tied to a particular store or merchant, it cannot be used to pay for things where credit cards are not accepted (rent, for example).

More formally, it is a theorem that the best choice from a set of options is always at least as good as the best choice from a subset of those options. Gift cards can only be used to buy a subset of the goods that an equivalent amount of cash would. You do the math.

The only advantage gift cards have over credit cards is that they can be purchased and used anonymously (but then, so can a $50 bill!). The only advantage gift cards have over cash is that they can be used to make purchases online. Thus, the only time it makes sense to give a gift card instead of cash is if you want the receiver to be able to buy things online anonymously. Other than this, cash or credit cards are better in every way.

Chapter 11. Blockchain Basics

In Section 5.26 we introduced the idea of distributed systems. This chapter deals with blockchain which uses **Distributed Ledger Technolog**y (**DLT**), a type of distributed system. One of the main attractions of blockchain is that, depending on implementation, is that it can function without a TDI. Blockchain is big topic, so this chapter will focus on the basics.

Section 11.1. What is Blockchain?

Blockchains are electronic ledgers that group transactions together into **Blocks** and append them sequentially to an existing chain. From a functional standpoint, blockchains are direct descendants of the paper **Ledgers** using **Double-entry Bookkeeping** invented in the fifteenth century by the Milanese Franciscan monk, Luca Pacioli. Ledgers have a particularly simple data structure, recording only account numbers/owners and current balances. They lack the meta and semantic web data of XML, the keys and relational table structures of SQL-type databases, and even the columns and rows of spreadsheets.

Conventional databases are typically kept on a central server under the control of a single party. This party has the power to determine who has access to various parts of the database, as well as the ability to alter or delete records. Such data is only as trustworthy as the party in control. In addition, since the data is kept in one place (or at least by a single entity), such databases have a central point of failure. As a result, state actors, courts, criminals, and hackers may be able to censor, alter, or even prevent access to such data.

In contrast, blockchain ledgers are updated and stored by widely distributed sets of agents sometimes called **Nodes**, **Validators**, or **Miners**. Transactions are sent by users to these nodes who communicate them to one another through peer-to-peer networks. Nodes examine sets of candidate transitions and decide if they are valid. The network of nodes as a whole comes to a consensus agreement, somehow, and then uses the set of valid transactions to update the **Current Ledger State**. In computer science, blockchains are described as **State-Transition Machines** that use distributed ledger technology.

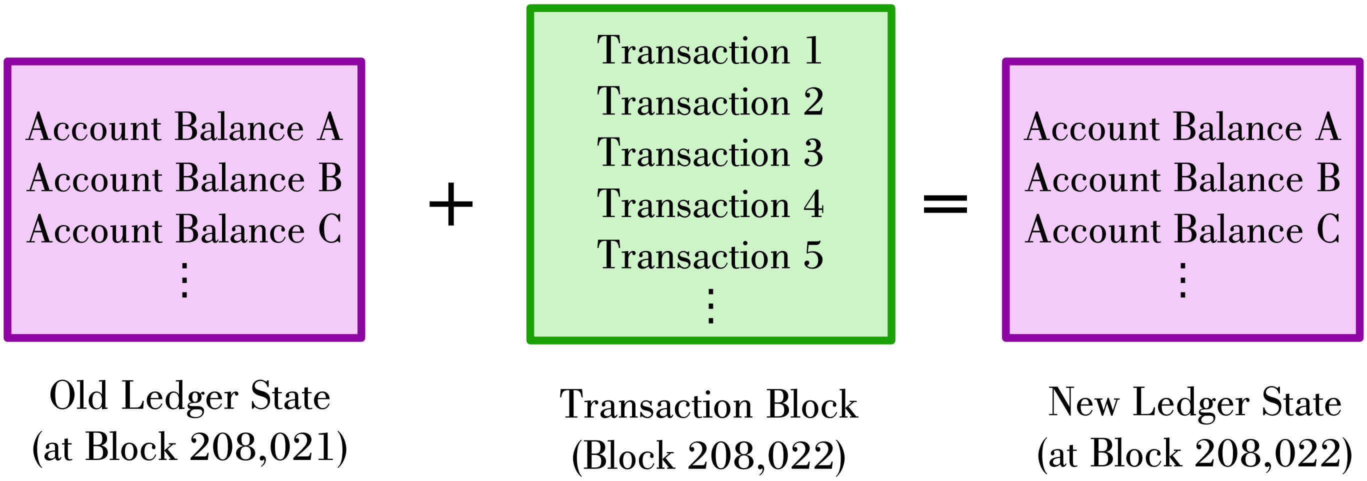
Although blockchains dramatically reduce the richness and utility of data, they offer several advantages. In particular, well-constructed blockchains allow users to interact and exchange value with each other without the need for trust. and without requiring permission from any central authority. Blockchains can create a transaction record that is immutable in the sense that it would be computationally impractical to change historical data, as well as allowing users to make transactions while maintaining a kind of **Pseudo-Anonymity**.

All the advantages that blockchains offer depend on honest transaction verification and block writing by the network of nodes. Blockchains use a number of different validation protocols to ensure this. Protocols lay out the set of rules that users and miners are supposed to follow, and often include incentive structures to make sure that they do.

Section 11.2. How Blockchain Works

At the highest level, blockchain is just a ledger that groups transactions together into a sequence of blocks, and then uses them to update the ledger state.

This begins with users who submit transactions to nodes who then distribute them to rest of the validating network. In general, one node is chosen as the **block-leader**. who proposes the next block. This is responsible for choosing a group of valid transactions from the set of unprocessed transaction (sometimes called the **Mempool**), bundling them into a block, and distributing it to the network of nodes. If the nodes agree, they append the block at then end of the chain, and use the transactions it contains to update the ledger.

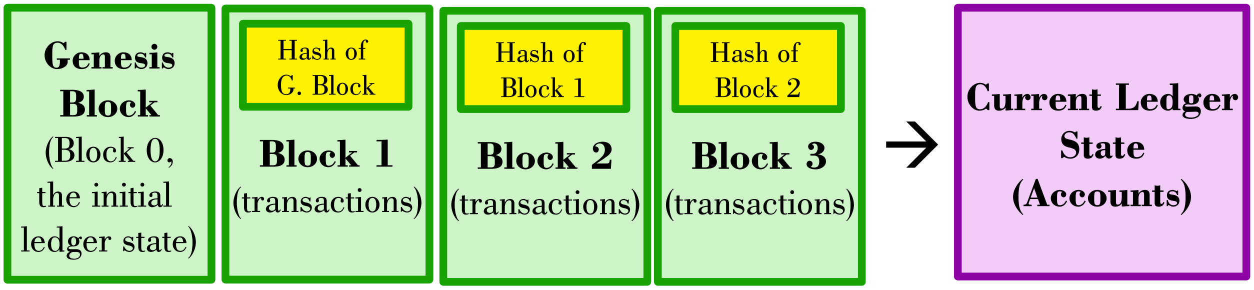


Blockchains are designed to be **Append-Only** through a cryptographic process of **Recursive Hashing**. The basic idea is that the hash of the previous block[25] in included in the next block. This creates a recursive hash tree that makes blockchains tamper-evident.

Suppose, for example, there is a blockchain with a **Block Height** of 5000, and I wanted to change a transaction in block 4000 for some reason. This would mean that the hash of block 4000 would change, and so the hash included in block 4001 would not match its original. I would therefore have to put the new block 4000 hash into block 4001. This would change the hash of block 4001, and so I would have to hash it again, put this into block 4002, and then continue this process all the way up to the current block, 5000.

[25] Some chains use the Merkle root of the last block, or the hash of the last block header.

In other words, I cannot insert new blocks, or alter existing blocks, without recreating the hash tree all the way up to the end of that chain. A new block is said to be **Committed** to the chain when a hash of the previous block is added, which then fixes its order in the chain.[26]



Bitcoin accounts numbered, and do not name an account's owner directly. This allows users to make transfers of value without it creating a direct record of the identity of the sender or receiver. However, it is often possible to deanonymize a user based on analysis of the details of his transaction traffic, the source of funds, or off-chain information. For this reason, Bitcoin accounts are only considered to be **Pseudo-Anonymous**.

For example, if you transfer funds from your bank to a crypto-exchange like Coinbase, both Coinbase and your bank know your identity since they are required to do KYC/AML on all their customers. If you transfer coins with your identity, and Coinbase may be required to report this to IRS.

From there, any transfer out of, or into, that Ethereum address is publicly observable, and can also be associated with you. There are ways to obscure your trail, and machine learning and data analytics are tools that partially lift this veil. This is also one of the motivations for the many privacy coin projects that have been created using **ZKPs** (**Zero Knowledge Proofs**) or other approaches.

In general, accounts are controlled through PPK pairs generated by the user. The public part of the key is the account number, and is kept in the ledger along with a balance. (Sometime, the account number is derived by the public key, but is not the actual key itself.) Users create a transaction request. and then sign it with their private key. This allows nodes, validators, and anyone else who has access to the ledger, to verify the signature using the data in the publicly recorded transaction that is committed to a block in the chain.

[26] Ethereum, EOS, Tron, and many other blockchains, use the traditional ledger approach of keeping a list with running account balances that transactions add to and subtract from. Bitcoin, Litecoin, Zcash, and several others, use what is called the **UTXO** (**Unspent Transaction Output**) approach for their ledgers. Basically, each Bitcoin transaction "consumes" by sending some of its balance to one or more other accounts, and then sending the any remaining balance to a user account to a new account owned by the sending users. More accurately coins are added to existing UTXOs, and the "change" leftover goes to a new UTXO controlled by the original user. This seems complicated, but there are technical advantages and disadvantages to both the account and UTXO approaches.

What makes a transaction valid varies somewhat over different types of blockchains, but generally, the following are required:

- The transaction is correctly formatted and signed.
- The sending account has enough of a balance to cover the amount of the transaction. In other words, you can't spend more than you have.
- You are not attempting to spend your balances more than once.

This last bullet point is called **Double Spending** and sounds like it should be covered by the bullet point above. Double spending has special significance in blockchain since the ledger is held in many places by many nodes. If all of these ledgers are not fully synchronized, nodes may approve transactions on different versions of the ledger that spend the same coins twice.

In most blockchain protocols, **Forks** are possible in which different blocks are attached to the existing chain by different sets nodes. We will see below that consensus protocols attempt to prevent or quickly resolve forks in order to prevent double spending.

Blockchains are designed with public verification of transactions in mind. Fundamentally, a blockchain is a publicly viewable set of transactions that are given an order of execution that collectively imply the current ledger state. The ledger is meant to be kept in many different places by anonymous nodes which makes them difficult to censor or shut down.

Blockchain removes the need for a TDI to maintain the data. There are also **Private** or **Permissioned** versions of blockchains which are protected by firewalls and are maintained by a small number of nodes who know and trust each other, but we will leave this discussion for later.

Section 11.3. Protocols

Blockchains use **Consensus Protocols** that seek to assure that all nodes in the network come to a shared view of the blocks and the current ledger state. There are four traditional approaches with many variations. This section will give a very brief overview.

Subsection 11.3.1. Proof of Work

Proof of Work (PoW) blockchains, such as **Bitcoin** and, up until September of 2022[27] **Ethereum**, require that the **Miners** who help maintain the ledger compete with one another to solve a difficult cryptographic puzzle for the right to propose the next block in the chain and receive mining rewards and transactions fees as a result.

[27] Ethereum did a hard protocol fork on September 15, 2022, in which its original Proof of Work consensus was replaced by its version of Proof of Stake.

The solution to the puzzle is sometimes called a **Nonce** (**Number used only Once**) and is specific to each block of transactions. The only way solve the puzzle is through brute force guessing and checking.[28] Thus, finding the solution is positive proof that the winner expended a great deal of computational work in the process. On the other hand, it is trivially easy for other miners to verify that a nonce is the correct one for any proposed block once they see it.

The winning block-leader communicates his block to other miners using a gossip network. A gossip network only requires that each of the thousands of miners/nodes knows the IP address of a few other nodes, who in turn know the address of a few others, and so on. No central registry of nodes or addresses exists, and nodes can anonymously join the network by announcing their presence and connecting to a small number of nodes anywhere in the network.

Newly mined blocks are transmitted from node to node until everyone in the gossip network is aware of them. Similarly, users can send transactions requests to any node they happen to know, and count on them being gossiped to the rest of the network.

Once a miner receives a newly mined block, it is supposed to verify that the transactions it contains are valid under the blockchain's protocol, the nonce has been correctly calculated by the block-leader, and then commit the block to the version of the chain that it keeps. Each miner then stops working on the block it was trying to mine since it can no longer be appended to previous block once it commits the block it just received. All miners then choose a new set of transactions, put them into a new candidate block, and start to work on mining it instead. Unfortunately, all the work unsuccessful miners put into the previous candidate block are wasted, and the miners receive no reward for their efforts.

Subsection 11.3.2. Proof of Authority

Proof of Authority (PoA) blockchains all have their roots in Castro and Liskov's 1999 work on **Practical Byzantine Fault Tolerance (pBFT)**. (We will say more about the key idea of BFT in general below.) Nodes are run by a small set of non-anonymous, registered, real-world agents. Nodes take turns proposing new blocks of transactions which are committed to the chain if they receive the approval of the required majority.

To the extent that there are incentives not to improperly alter the ledger, they come from a common desire on the part of the nodes to have an honest record, and a fear that they will suffer repu-

[28] For the nerds out there, the puzzle works like this: Take the hash of the block of transactions that you are attempting to mine. The hash-digest is random, so the odds that the leading digit happens to be zero is 50%. The odds that the first two are zero is 25%, and so on. The puzzle asks the miner to add a small amount of random data (a nonce) to the block such that when the whole thing is hashed, it has 50, 70, or some other number of leading zeros followed by the rest of the 256 bits of the hash. The odds that any random pre-image will happen to have 70 leading zeros are one out of $2^{70} \approx 10^{21}$. Thus, miners add random data until they find a nonce that gives them the required number of leading zeros. The difficulty of the puzzle is adjusted periodically so that on average, a block is mined every ten minutes. This means that the probability of success depends directly, and exclusively, on the hashrate available to the miner compared to the network at large.

tational damage if they were to behave dishonestly. This is most often seen in **Permissioned Blockchains** such as IBM's **Hyperledger Fabric** and **Ripple** that run behind firewalls and sometimes encrypt all the data in the chain.

Subsection 11.3.3. Proof of Stake

Proof of Stake (PoS) blockchains require nodes that participate in validating the chain to stake coins, but putting them is s kind of escrow account. Both the probability that a node will be chosen to be the block-leader, and the share of the votes that accept or reject proposed blocks are equal to share of the staked coins the has escrowed. The leader circulates his block via a gossip network, and if 2/3rds of the stake weighted vote approves, then the block is committed to the chain.

For example, Suppose that a blockchain platform issues 100 million coins, and 90 million are purchased by ordinary people to use the pay for the services provided by the platform. The other 10 million are purchased and staked by node. If a node happened to stake 500,000 coins it would have a 5% chance of being selected as the block-leader, and have a 5% say in the vote over whether a given block is correct and should be committed to the chain.

There are endless variations on this basic protocol including those employed by **Ethereum, Tendermint**, **Honey Badger**, **Cosmos**, **Algorand**, and **NEO**, for example. These variations are generally aimed at addressing different attack surfaces, or improving efficiency of transactions throughput. Typically, if 2/3rds of the stake-weighted vote agrees that a proposed block is correct, it is committed to the chain and the ledger updated to reflect the new transactions. PoS protocols follow the same sort of recursive hashing strategy as PoW and PoA so that if any block is changed, all the blocks that follow must also be rewritten.

Subsection 11.3.4. Directed Acyclic Graphs

Directed Acyclic Graphs (DAGs) are a non-blockchain approach to DLT and come in permissionless and permissioned varieties. **Iota** and **Hashgraph** are examples of each, respectively.

The basic idea is to create a topological ordering of transactions where new transactions (called vertices in the language of graph theory) are linked to existing transactions by hashing them together. The starting vertex (the existing transaction) is said to be linked to the ending vertex (the new transaction) by an "edge" which places the starting transaction before the ending transaction in the topological order.

Users are only supposed to hash a new transaction to an existing transaction if they believe that existing transaction, and every transaction that it is backwards linked to it, are valid. Eventually, assuming that users choose which valid transactions to hash to in a sufficiently random way, a directed acyclic graph is created so that one can start at any end point of the graph and be able to find a path that links to any other transaction that is sufficiently old.

Under certain assumptions, this creates an unambiguous ordering of these historical transactions. The key is that if we know this correct ordering, we can execute the transactions sequentially to find the current state of all accounts in the ledger. Double spends will not be included since one of the transactions will be deemed to have come earlier in the order which makes the transaction that attempts to spend the tokens in the account a second time *per se* invalid.

Subsection 11.3.5. Proof of Honesty

Proof of Honest (PoH) is a non-consensus approach to building blockchains. Candidate transactions are crowd-sourced rather than being chosen by a block-leader. Validation of transactions and updates to the ledger are done independently by each node in the active network. There is no need to vote to accept or reject any other node's conclusions or chain view. Since the protocol is deterministic, all honest nodes must come to identical conclusions.

Nodes play a kind of unanimity game in which they audit any nodes that come to a conclusion different from their own out of the network. Nodes on the honest fork are able to prove this to users, while nodes on dishonest forks cannot. As a result, if there is even a single honest node in the network, it will have a provabley correct chain-view. Users are therefore able to ignore any false chain-view presented by dishonest nodes.

The unanimity game (called the **Catastrophic Dissent Mechanism**) implements truth-telling in coalition-proof equilibrium (see Chapter 12 for a discussion of game theory as it relates to blockchain). PoH is 99% Byzantine fault tolerant as opposed to 50% for PoW, and 33% for PoA, and PoS. PoH is used by the Geeq project, for which your humble author serves as Chief Economist. Since this creates an obvious conflict of interest, this book will contain no further discussion of Proof of Honesty.

Section 11.4. Pros and Cons of Different Approaches to DLT

Blockchain and other forms of DLT have various advantages and disadvantages that make them suitable for different sorts of applications. In this section, we outline and evaluate these differences.

Subsection 11.4.1. Immutability

One of the chief claims of blockchain is that it creates an **Immutable** record of transactions and other data. Different types of blockchains and distributed data systems approach this is various ways. None of them, however, create records that are truly immutable. It is more accurate to say that the records they contain are difficult to mutate once written, or that the records they create are tamper-evident.

A good way to understand the immutability claim for PoW blockchains is to think of each block of transactions as a page in a paper ledger book. This ledger book has three special characteristics. First, each page is made up of thousand-dollar bills. Second, all transactions are recorded in indelible ink, so pages cannot be altered or reused once transactions are written on them. Finally, a tiny copy of the previous page is written at the top of the next page in the ledger (actually, this is the "hash" discussed above).

Suppose that I wanted to change a transaction recorded 50 pages back from the most recent one. First, I would have to create a new blank page of thousand-dollar bills to write my new transition on. (This is equivalent to spending the large number of compute cycles required to recalculate the correct nonce for the new block.) Since this would change the hash of the page, I would also have to create 49 additional new blank pages so that the recursive hash tree was consistent.

This implies that the more deeply a transaction gets buried under new pages in the ledger, the more expensive it is to alter it without detection. Thus, the ledger is not immutable, it is simply very expensive to mutate. Eventually these costs become prohibitive. Unfortunately, this security guarantee comes at a high cost. Each page costs hundreds of thousands of dollars to create, mostly in the form of wasted electricity. An finding a nonce for a bitcoin block (equivalent to creating a new "page") block currently costs about $250,000. Bitcoin nodes collectively use more electricity than smallish countries like Ireland or Argentina each year.

PoA blockchains rely on vetting nodes for honesty before they are admitted as validators. The hope is that nodes will stay honest to preserve their valuable reputations. For example, twenty banks might agree to write a shared ledger of the transfers they make between one another. It might seem unlikely that any bank would ever behave dishonestly, but dishonesty is sometimes in the eye of the beholder.

For example, several of the banks might decide that they want to reverse transactions to one of their partners because they view it as having participated in a fraud. Courts or legislation might require that banks reverse certain transactions that involve illicit goods, that might be construed as money laundering or tax evasion, or that went to undesirable or sanctioned individuals, groups, or nations.

In PoA blockchains, changing the ledger only requires getting the agreement of the majority of the nodes. There is no other cost or impediment. Thus, PoA ledgers are not immutable, and transactions can never really be considered to be finalized. The integrity of such ledgers requires trust in the honesty of the majority of nodes. As a result, PoA can never be a real foundation for the kinds of trustless interaction between agents that blockchain is supposed to provide.

PoS blockchain ledgers can also be mutated, rewritten, or forked if enough of the stakeholders agree. Like PoA, the ledger says whatever a qualified stake-weighted majority says it does. Often, a minority of stakeholders can halt block writing as well, although they may not be able to rewrite transaction history.

The many variations of PoS follow different strategies to make such collusion difficult or expensive. Ultimately, if more than one third of the stake-weighted voters manage to collude, such ledgers offer no security guarantee to users at all. How likely this is to happen is a matter of debate, but there are many attack surfaces in PoS protocols, and no real proof that the incentives for good behavior are sufficient to prevent coalitions of self-interested nodes from manipulating the ledger.

The immutability guarantee of DAGs is founded on transactions being independently verified by many agents who create the graph of interlinked hashes. The idea is that altering the graph would require the complicity of an impractically large number of agents. Unfortunately, there are serious problems with this dependency.

First, if more than a third of the hashes/edges are contributed by dishonest users, they can prevent or alter consensus conclusions about the validity and order of transactions. This makes DAGs vulnerable to **Sybil** attacks in which one agent pretends to be many Some platforms attempt to deter this by adding a limited a PoW element to the protocol, but this adds significant costs, wastes electricity, and in any event, mitigates, rather than solves, the problem.

Second, permissioned DAGs with a fixed set of presumably honest nodes run into the same problems as PoA. That is, the DAG they produce can be rolled back or rewritten if the majority of nodes choose to, or are forced to by legislative, legal, or criminal actors.

Third, permissionless DAG implementations, such as Iota, end up having a privileged set of trusted or highly reputable nodes who ultimately decide on the validity of graph edges, and therefore, the state of the ledger. This places a small set of actors in a position to alter or choose the correct state of the ledger, and so does not provide a strong immutability guarantee.

Distributed private databases, of course, are never immutable by construction. Whatever organization controls the servers can rewrite or erase any data it pleases. To the extent that it is difficult to do so, it is because users would lose trust in a bank or brokerage house if it became known that it altered its data arbitrarily. This is one of the reasons that paper statements and records are traditionally provided to clients. To the extent these are difficult to forge, they provide record of what the database said at a specific point in time (sometimes referred to as a **Checkpoint** in blockchain) and so can be used to prove that alterations took place that were inconsistent or unjustified.

Subsection 11.4.2. Distributed

A second claim of blockchain is that it is distributed and decentralized. If a blockchain, ledger, or any type of data, is stored redundantly on a widely distributed set of servers, it becomes more difficult to censor or destroy. This is especially true if it is kept by anonymous nodes in different jurisdictions. **DDoS (Distributed Denial of Service)** attacks against thousands of servers at once are expensive, and may be impractical.

Bitcoin and Ethereum both had on the order of 10,000 nodes on their networks as of January 2023. Other blockchains have much smaller networks. Nodes check to make sure that transactions in committed blocks are valid, keep copies of the blockchain, participate in the gossip-network that communicates new user transaction requests and mined blocks, and generally keep watch and share information about their blockchain.

These are all purely altruistic acts. Nodes do not mine blocks, build the chain, nor receive transactions fees and mining rewards. These go exclusively to whichever miner happens to be the first to find the nonce for the next block which it then communicates to network using nodes. Miners in PoW chains may also choose to run a node, but this not strictly necessary to mine blocks.

Recall that blocks are mined by solving a “guess and check” hashing problem. It turns out that the nature of this puzzle requires using specialized mining rigs called using **ASIC**s (**Application Specific Integrated Circuits**) that are orders of magnitude more efficient than general purpose computers. Modern Bitcoin mining rigs can produce on the order of 100 petahashes (equivalent to making 10^{13} guesses at the puzzle) per kilowatt-hour of electricity consumed.

Since the probability of a given miner finding the next block is equal its own hashrate as ratio to the overall hashrate of the network (which was about 3×10^{20} or 300 exahashes per second in January 2023), profitable mining can only be done in optimized server farms located where electricity is as inexpensive as possible.

This has lead to an extreme concentration of miners into pools. In 2021, was estimated that approximately 65% of the Bitcoin’s hashing power was located in China. China outlawed mining in late 2021, and as of January 2023 about 45% are located in the US or Canada, and 30% in Russia or Kazakhstan.

Miners typically join **Mining Pools** that work together to produce blocks, and share risks and rewards. The top four mining pools mined 80% of all Bitcoin blocks in early 2023. Thus, while nodes may be widely distributed, hashing power is not. A relatively small number of mining pools could certainly mount a 51% attack on Bitcoin, and gain complete control of the ledger. They might even be forced to by their governments.

PoA is relatively centralized by construction. Typically, permissioned networks consist of 10 to 25 nodes, and only those nodes have copies of the chain. PoS approaches give voting power to agents in proportion to how many tokens they have staked, rented, or had delegated to them. Even if tokens are widely distributed, only a relatively small number of agents will typically be keeping copies of the blockchain, and actively participating in block proposing and transaction validation.

As we say above, existing DAGs either have a small set of permissioned validators maintaining the graph, or have privileged agents with disproportionate power over transaction finality. Having a truly open DAG with a broad set of anonymous agents contributing transactions and verifications is problematic for two reasons:

- First, all such agents would have to maintain copies of all the data in the graph in order to verify transaction hash trees. This is resource intensive and not even feasible for IoT devices with limited bandwidth, processing power, and storage capacity.[29]
- Second, open DAGs are extremely vulnerable to Sybil attacks since power can be concentrated in this way by a malicious agent.

Finally, private decentralized databases provide redundancy, but all the servers are coordinated by design. A single court order could result is all of these copies being taken down or destroyed, Thus, decentralization of data storage does not contribute to the uncensorablity data over all.

Subsection 11.4.3. Trustlessness

PoA, permissioned and permissionless DAGs, and private databases all require the users to trust the honesty of the validators. PoW and PoS, on the other hand, attempt to set up protocols and incentives so that users have a degree of confidence that self-interest, or the difficulty of getting away with dishonesty, will protect the integrity of the ledger. If at least 51% to 67% of nodes are moved by these incentives, then users can count on a ledger's correctness.[30]

Subsection 11.4.4. Scalability

Bitcoin can process about seven transactions per second (TPS), and Ethereum, approximately fifteen. **Lightning Networks** are supposed to handle larger numbers of transactions through **Side Channels**, but costs, security, *de facto* centralization, and the lack of expensive-to-establish escrow connections between counterparties, make it unlikely that lightning networks will allow PoW

[29] See Attis Elsts' 2018 Medium piece "Lessons learned from evaluating IOTA on Internet of Things devices"

[30] There are many attacks that do not require this degree of dishonesty (see Eyal and Sirer 2014, and Houy 2014, for example). In any event majority attacks are, in fact, a significant problem. Bonneau (2018) estimates that it would cost $1 million per hour to rent enough capacity on EC2 to mount a 51% attack on Ethereum, and $1.5 billion to purchase enough capacity to mount such an attack on Bitcoin. Given that Bitcoin's token cap was approximately $500 billion as of August 2023, $1.5 billion seems like a relative bargain to gain full control. Other authors have placed the cost of a rental attack on smaller PoW blockchains such as EthereumClassic, Monero, and Dash, at less than $10,000 per hour (see https://www.crypto51.app/). A number of such attacks have actually occurred on such chains.

blockchains to scale significantly.[31] Some newer PoS and PoA chains can manage several hundred to a few thousand TPS, and in principle, DAGs can scale infinitely.

Ultimately, the binding constraints on scale are the bandwidth and the storage needed to process large numbers of transactions. Public blockchains (including DAGs) that use anonymous nodes are bound by the resources available to the nodes in their network. This could be overcome through a truly federated structure of independent, interoperable blockchains, but as long as there is a single chain, or a master chain, to which all others must ultimately report, it will not be possible to overcome these limitations.

Permissioned approaches, including private databases, have a much better ability to scale. This is because nodes and servers can be set up on high capacity cloud platforms that can provide as much bandwidth, computation, and storage as needed. The Visa network has a capacity of many tens of thousands of TPS, for example. Of course, this comes at the cost of centralization and the need for users to trust in the honesty of the nodes.

Subsection 11.4.5. Cost

PoW protocols are fundamentally expensive. They use significant computational resources by design as the basis of their security models. Transaction costs to users on these networks vary. In 2023, but the average cost of a transaction on the Bitcoin chains were on the scale of $1, but have been as high as $30, and in 2021 reached $69. Ethereum coin transaction fees are similar, but transferring ERC20 tokens requires invoking a smart contracts. Fees for token transfer can easily be three to five times as much as for coins. The fundamental costs of PoA and private databases, on the other hand, can be very low. This is because these networks are typically small, and costs per transactions amount to resource cost of processing and storage on ten or twenty servers.

What users get charged, however, is another matter. Visa charges 2% to 3% plus 25¢ per transaction. Part of this is because of the monopoly power that they have, but another reason is that they are providing valuable and costly services in addition to simply maintaining the ledger. This is primarily in the form of insuring users and merchants against fraud. On blockchain, if someone has your private key and uses it to take coins from your account, you are out of luck. Visa, on the other hand, makes an effort to minimize fraud, but then eats the cost of any that manages to get through.

PoS solutions charge whatever fees they wish and sometimes have block writing rewards similar to PoW. Costs are reduced when the approach limits block writing, validation, and chain storage to a small set of wealthy stakeholders. If only 100 nodes communicate and store the chain, the costs

[31] See, for example, Jamie Redman's 2018 Bitcoin.com piece "Looking Beyond the Lightning Network Hype: Every Day Users Experience Issues", or Jonald Fyookball's 2017 Medium piece "Mathematical Proof That the Lightning Network Cannot Be a Decentralized Bitcoin Scaling Solution".

are much lower than if 10,000 do so (as with Ethereum and Bitcoin). On the other hand, some approaches such as **Algorand** require that all users be ready to participate in the validation process and this might mean that hundreds of thousands of nodes must communicate and keep copies of the chain and bear the cost.

DAGs are similar in this respect. Permissioned DAGS with small pre-determined validators sets can process transactions cheaply. Open approaches require huge amounts of communication and storage by many nodes/users and sometimes include additional PoW costs to reduce Sybiling.

Subsection 11.4.6. Evaluation

There is no perfect solution. On one extreme, there are TDIs and private databases. These are relatively inexpensive, can scale, but require a high degree of trust that the TDI will make the data continuously available in unadulterated form. In the middle, there are permissioned PoA blockchains and DAGs. These are also relatively inexpensive and can scale to varying degrees. Users are not left at the mercy of single TDI but must trust that a sufficiently large majority of a small, nonanonymous group will behave honestly.

On the other extreme are permissionless, public PoS and PoW solutions. PoS is more costly and less scalable than PoA or private databases, and PoW even worse in those dimensions. On the other hand, such protocols do not directly require trust in the honesty of the validators, instead depending on the incentives the protocols provide for honest behavior. When validators are anonymous, numerous, and widely distributed, state actors, courts, and criminals should find it difficult to force nodes to behave in ways outside of protocol, or to censor the blockchains and ledgers that they keep.

Section 11.5. Smart Contracts

Nick Szabo, proposed the idea of a **Smart Contract** as “a set of promises, specified in digital form, including protocols within which the parties perform on these promises”. If code could be executed in a decentralized environment, with provable, and irreversible outcomes, then trustless interactions could be extended far beyond simple exchanges of value. Ethereum’s implementation of smart contracts in a Turing complete language made his vision real. It has lead to an entire ecosystem of ERC20 tokens, ERC-721 NFTs, decentralized exchanges, and experiments in decentralized finance.

The idea that the intentions of two parties could be encoded, and then executed remorselessly, without favor or deviation, is powerful. Extending blockchain’s functionality would set us free from reliance on centralized authorities in even more dimensions.

Examples of smart contracts on Ethernet:

- Some smart contracts are simple, and do such things as allow users to escrow tokens, create multi-signature accounts, or make bets that are settled by oracles.
- **Fungible Tokens:** A more complicated application is **ERC20** compliant token contracts which allow the creation of non-Ethernet cryptocurrencies (sometimes called **Alt-Coins**) that are traded and maintained on the Ethereum blockchain. There are many thousands of ERC20 tokens, each with its own smart contract, with a combined value of tens of billions of dollars. At times, the market cap (total market value) of these tokens has exceeded that of Ethereum itself. The ERC20 standard was instrumental in fueling the ICO explosion of 2017 and 2018 and continues to be a key piece of infrastructure that facilitates experimentation and growth in the blockchain sector. Collectively, the clear majority of smart contract calls on Ethereum are to various ERC20 contracts.
- **Decentralized Exchanges** (**DEXs**): Contracts such as Uniswap and IDEX facilitate the trading of ETH and ERC20 tokens without a **Centralized Exchange** (**CEX**) such as Coinbase or Binance. CEXs are required to do KYC/AML on their clients and generally hold tokens in **Custodial** accounts similar to what brokerage houses do with securities owned by their clients. Custodial accounts are not under the direct control of the owner. The CEX must authenticate and agree to any transaction request. Billions of dollars worth of cryptocurrency have been lost by CEXs that failed to properly secure these custodial accounts. In contrast, users can trade directly from Ethereum accounts under their complete control, without permission, and pseudonymously, on a DEX. Not surprisingly, DEX contracts see significant usage on the Ethereum chain.
- **Non-Fungible Tokens** (**NFTs**) NFTs are tokens that are not divisible, and so must be owned and transferred as whole. On the Ethereum chain, Fungible Tokens are instantiated by and ERC-20 smart contracts, while NFTs use the ERC-721 standard. An NFT record typically contains a hash of some digital object, an image, for example, and is usually digitally signed, or otherwise authenticated, by the creator. The smart contract connects these records to Ethereum addresses in the ledger, and allows the user/owner to transfer them to another address, if he wishes. What "ownership" means depends on the use case. They do not give you any kind of physical control over the digital object that is hashed (anyone can copy the image if they get access to the file), nor do they convey a copyright, under current US law. On the other hand, if a government made it a law that whoever "owned" an NFT that tokenized a deed or title, legally owned the house or car it describes, then you could transact ownership of physical objects on a blockchain.
- **Decentralized Applications (DApps)** is a catch-all term for smart contract that can do almost anything. Examples include games like **CryptoKitties** and gambling applications, logistics and chain of custody, **Decentralized Finance** (**DiFi**) such as **Maker DAO** and **YAM**, **Distributed Business Processes,** such as real estate transactions, **Machine to Machine Markets**, and so on. While there is great potential here, we are still at the opening stages. Transaction costs must decrease, and transactions volume capacity and security must increase before the true potential of DApps will be fully realized.

Smart contracts on Ethereum work as follows:

- A user writes a computer program in a **Turing Complete,** high level language, like **Vyper** or **Solidity**, that has a set of functions, variables, and internal logic. In a sense, this is similar to an API in that the functions are like hooks that other users can call to alter variables and invoke code.
- The user compiles this smart contract program into low level bytecode, and then executes a special transaction that **Deploys** it on the Ethereum blockchain. The bytecode, and the initial state of any variables, are literally written into a block, so it becomes both "immutable", and publicly visible.
- Smart contracts have their own accounts and addresses. Contracts are not controlled by private keys, and anyone can call on a contract's functions once it is deployed.
- When a user wishes to invoke a smart contract, he creates a special transaction that calls on the contract's functions, may include parameters and other inputs, and may authorize coin transfers out of his own account governed by the rules of the smart contract.
- The contract bytecode is then loaded and executed by miners in what is called the **Ethereum Virtual Machine (EVM)**, sometimes refered to as the **World Computer** by fanboys. The outputs of the contract are written into a block which may then modify the state of a contract's variables or the balances in ordinary user accounts.

While miners are the ones who decide what transactions and contract results to record in blocks, there entire network of 10,000 Ethereum nodes is supposed to independently verify that the smart contract was correctly executed. This is possible because the compiled code is publicly visible and fixed in the block where it was first deployed. Thus, thousands of computers run the same code, record, and then store, the same results in the distributed copies of the Ethereum chain they maintain.

As you can imagine, this uses a lot of resources. An average Ethereum block is around 25 kB in size, so storage space for bytecode is scarce. Since thousands of nodes independently run each smart contract when it is called, and the results are communicated to thousands of peers and users, executing smart contracts uses a great deal of computational power and bandwidth. Running smart contracts is *extremely costly* compared to running equivalent code on private web server. To align incentives, smart contract transactions have high transactions fees that are proportional to amount of storage and computational efforts they require.[32]

Unfortunately, with great power comes great responsibility, and smart contracts developers have a decided mixed record. In fact, smart contracts have turned out be the one of the most significant points of failure in blockchain. Here are a few of the more spectacular examples, and the resulting loses:

- DAO; June 17, 2016: 3.6 million ETH worth $79.6 million, (now about $6 billion)

[32] We will not go into the details of how these prices are determined, what "gas" is, how it fuels and incentives users to use the EVM wisely, the positives and negatives of having a Turing Complete smart contracting language, and raft of other very interesting but complicated topics in the interests of not exhausting the reader.

- Parity; July 19, 2017: 150,000 ETH, worth, $30 million. (now $259 million).
- Estimated losses in 2021: $680 million. (Exploring Security Practices of Smart Contract Developers, T Sharma, Z Zhou, A Miller, Y Wang, arXiv preprint arXiv:2204.11193, 2022 – arxiv.org).
- Poly Network Binance Bridge; August 10, 2021: $600 million, eventually returned by the hacker.
- Wormhole Solona Ethereum Bridge; February 3, 2022: $320 million.
- Ronin Axie Bridge; March 20, 2022: $610 million.
- Beanstalk Protocol; April 17, 2022: $180 million.
- Nomad Bridge, August 3, 2022, $200 million.

Why is something so promising in theory, so disappointing in practice?

- **Human Error**: Contract creators do not always think of every possible edge case. Sometimes developers simply make mistakes. There is a great incentive for hackers to discover any exploits once a contract starts to carry real value.
- **Fraud**: In some cases, there is a suspicion that these vulnerabilities were intentional, and insiders are responsible for the thefts.
- **Lack of Understanding**: Smart contracts perform exaclty as they are designed to do. Very few users read the code, or can verify what this design actually is. Signing a real-world contract when you don't fully understand what you are agreeing to puts you at risk. Engaging with a smart contract you can't comprehend is no different.

These factors inevitably create a trust layer in what should be a trustless technology. Users must trust that a contract was written by very competent, and honest people, and the description of the contract's function is complete, and accurately describes the code. Few, is any, users have the ability to independently verify smart contracts, and the results are plainly evident in their history of failure.

The way smart contracts have been implemented on many platforms creates an even deeper, but less obvious, trust layer. A contract's state (for example, who owns how many tokens, or which NFTs) is a result of the sequence of transactions that have targeted it since it was deployed. That is, to verify a contract's state, you would have to replay all historical transactions in the correct order. Full nodes, in fact, do exactly this, and keep an off-chain record of contracts' states at any given block height (using a Patricia Merkle trie structure, in the case of Ethereum.)

Users, however, do not have access to the necessary data, or the computational resources, to calculate a contract's state. Instead, users simply query secondary sources like **Etherscan** to learn what assets an account controls.

Not only is independent verification unpractical, users have no realistic way of getting cryptographic proof of what Etherscan tells them. In other words, users simply trust that Etherscan is telling them the truth. This is not much better that trusting Bank of America to tell you your balance, which would not be so bad were it not for the fact the whole point of blockchain is that trust should not be required.

Going one step deeper, it is theoretically possible to generate proofs on Ethereum, and most other blockchain platforms, that would remove this last trust layer. The difficulty of producing the required Merkle proofs depends on the data structure of blocks and ledgers. In the case of Ethereum, for example, it is fairly straightforward to prove the state of a user account (that is, the balance of ETH that it holds).

Recall, however, that token and NFT holding are recorded in an off-chain, Patricia Merkle trie, and are not included, or even referenced, in the user account that controls them. This means that an independent Merkle proof for each type of token and each NFT would from the state root of the corresponding smart contract, to the key-value pair that describes an account's holding, stored in the Patricia Merkle Trie. For example, if an account holds 20 types of tokens, and 50 NFTs, 70 additional Merkle proofs are required.

Worse, the state root of each smart contract account must also be proved. In total, Merkle Proofs of 71 account states, plus 70 more Merkle proofs of each key-value pair in the contract state would be required. Each of these proofs might have a kilobyte of data, and so the complete proof could require 100 kB, or more, for a single user account. It is understandable that Etherscan does not provide such proofs as a matter of course to users, and that users do not ask for them.

Here is the central point: this complexity is entirely due to the use of smart contracts as a way to generate tokens and NFTs. Ownership of assets, is simply not part of a user account's data by design. There is no other way to prove ownership than to dive into each smart contract's state. Thus, not only do users not verify a contract's state, they don't even ask Etherscan to provide proof of its conclusions. Instead, they rely on trust.

If blockchain is to be a useful counterweight to the increasingly centralized financial and content platforms we are all forced to use, it must remove all trust layers. Otherwise, we are just replacing the devil we know for one we can only hope will be better. At the same time, blockchain must go much further than simple transfers of value, useful though they are. This means we must find an alternative to smart contracts, without sacrificing the functionality they provide.

Section 11.6. Use Cases for Blockchain

Cryptocurrencies are the most widely known use of public blockchains, but it is probably also the least interesting. for all their flaws, smart contracts open almost limitless possibilities.

A very simple smart contract could be designed that would allow users escrow tokens that would be released when certain conditions were satisfied. For example, Alice and Bob might bet on who will win the world series. Both would create a transaction that would transfer tokens to an account controlled by a smart contract. The contact would be written to transfer all the tokens to Alice's account if the American League won, and to Bob's account if the National League won. Alce and Bob agree that a statement of the outcome signed by the New Your Times' public key will serve as proof of which team won the series.

The New York Times is said to acting as an **Oracle** in this case since it is a mutually trusted agent who delivers off-chain facts to on-chain applications. To activate the contact, the winning user would create a transaction containing the NYT's attestation and the contract would then automatically make the required transfers.

Leading use cases for smart contract based DApps include:

- Distributed business processes such as logistics chains, real estate transactions, provenance, and medical records. Blockchains allow actors who work for different organizations to contribute to a process without the need for central agent coordinate or control required data. Blockchain creates a provable source of data and process consistency without the need for trust. Each agent in a logistics chain, for example, can put signed attestations of receipt and transfer of cargo the prove that they accomplished their part of the task, and so any failure is responsibility of some other party.
- Accountability and audit trails for things such as financial transactions, maintenance records, royalties, content sales, legal and tax compliance, health and safety, and following best practices or required procedures. Blockchains can create immutable, contemporaneous, records that make it possible to know and prove a wide variety of things.
- Public records such as land and auto titles, legal records, smart city data, and public services deployment. Blockchain allows governments at all levels to increase transparency, improve fair and equitable allocation of resources, and encourage civic engagement.

Examples of specific use cases include:

IoT and Connected Infrastructure Devices: Suppose that transformers, substations, water mains, gas pipelines, communications grids, roads, bridges, rail lines, dams, and so on, were instrumented to create useful data showing their performance, state of health, potential for failure, or actual failure. Suppose that real time records of maintenance performed on these systems could also be created. Putting this data in a public blockchain allows the agencies or companies responsible to prove that they have been good stewards. Power sometimes fails, but if the failure is due to a detectable problem or a lack of maintenance, then the power company might be liable. Similarly, train and auto crashes, failed or slow communications networks, fires that might have been started by transformers or gas leaks, etc. can crate liability which can be warded off with credible records.zzz

Machine-to-Machine Markets: Blockchain opens up the possibility for devices to work as agents in their owners' interests. When you are traveling in a foreign country, for example, your cell phone could use an app to wirelessly reach out to and negotiate with other cell phones for bandwidth on the local network. Payments would be securely made in a cryptocurrency without the need for either device owner to get directly involved. Buyers would get connectivity despite being out of network, and sellers would make a profit. Other examples include allowing computers to sell unused compute cycles or excess storage to other computers, and matching cars to parking places or to slots on a conjested highway.

Peer-to-Peer Content Platforms: Individual content producers such as musicians, writers, photographers, visual artists, and video creators, are forced to use platforms such as YouTube, SoundCloud, Shutterstock, and Etsy to distribute their work. These platforms impose their own rates of payment, promote content as they choose, and can also remove or censor content they don't like. Blockchain **Micropayments**[33] allow creators and consumers to connect directly without a platform imposing rules or taking a cut. This makes it possible for talented creators to make a living, and gives consumers access to a much wider array of offerings.

Public Records: Land and car titles, certain tax and legal records, and licenses and permits are examples of records that the public has a right to see. Although some of this information might come from devices, most is generated by humans. Blockchain has two things to offer here. First, putting full copies of public records in an immutable and replicated blockchains gives the public easy and equal access to information they have a right to inspect.

As it stands, getting such access often requires going to a specific office and requesting a specific record. This makes the information effectively invisible to the majority of the public. Second, it allows public records to be updated and amended in an externally verifiable way. As it stands, information brokers buy and aggregate such records and then sell access to facilitate background and credit checks.

The problem is that brokers have little incentive to spend the effort to keep their data up-to-date and there is nothing a citizen can do to force the broker to remove inaccuracies. A citizen is unlikely to even know what this aggregated record contains. As a result, a person might be denied jobs, loans, or benefits because of an incorrect record of a DUI conviction, a lien, an unpaid student loan or other debt, or a court ruling. This might be due to new information the broker does not have, an identity theft, or even an error made by the agency that generated the data. When such records are keep current, visible, and accessible, citizens are able to find inaccurate data, get it corrected, and have it propagated,

On the other hand, when there is a central party that can, or should, be trusted with data, there is no need for blockchain. Your employer keeps your personnel records in a private database, and manufactures keep records of inventory, sales, parts, accounts payable, etc. in the same way.

[33] Micropayments refer to transferring small amounts of value (a few cents, or less). This is not possible using credit cards or even Ethereum due to the high fixed cost of transactions. Blockchains with transactions fees of a few hundredths of cent will make it possible pay for a single song, a few minutes of streaming content, pay as you go gaming, and small services such as search. This has the potential to supplant the subscription and ad-based business models that currently prevail, and open up entirely new sorts of markets.

This is just as it should be. If several parties trust each other, such as a consortium of banks or hospitals, permissioned, distributed, synchronized data systems do an excellent job. However, it is hard to see a reason to use a private permissioned blockchain which introduces the overhead of a consensus layer and requires reducing the complexity of the data, so it will fit into ledger when there is mutual trust between all parties.

Chapter 12. Blockchain, Game Theory, and Incentives

Section 12.1. Byzantine Fault Tolerance

Blockchain is an outgrowth of a subfield of computer science called distributed systems. The main objective of this field is to design systems with many different, possibly geographically distant, components, that nevertheless reliably synchronize their views and agree on the current correct state of a database.

It is well-understood that both physical machines (called nodes) and the networks they use to communicate can, and will, fail in various ways. Nodes may crash or simply be turned off. Messages sent by nodes may be delayed, or even lost, by the network. More severe network failures can even result in a **Partition** where nodes may only be able to communicate with a subset of the entire network.

Robust distributed systems should be **Fault Tolerant**. That is, systems should continue to operate at some level, or recover quickly and elegantly, despite **Fail-Stopped** nodes, network latency, and temporary partitions.

Faults are framed as random and unavoidable real-world events. In particular, faults are generally not considered to be intentional actions taken by self-interested, possibly malicious, actors. For distributed databases where nodes are deployed by a single company, this is a perfectly reasonable focus. Why would a company work against its own interests?

Blockchain ledgers are maintained by distributed, often anonymous, nodes run by actors who do not share common interests. For example, each individual actor might benefit if he could get away with a ledger update that transferred coins into his own account improperly.

The main challenge in blockchain is to find a way to get the network of nodes to come to a common and correct consensus about whether sets of transactions are valid and how they should update the ledger state despite these conflicting interests. If many of the agents running nodes are dishonest, they may be able to corrupt or capture the ledger. The nodes controlled by dishonest agents are faulty in the sense that they are not behaving according to rules of the protocol, but the fault is a choice, not a random act of nature.

Lamport, Shostak, and Pease (1982) describe what they called **The Byzantine Generals Problem** as follows: Suppose that ten generals of the Byzantine Empire are surrounding a city, some of whom may have been bribed not to attack. Suppose the participation of at least seven generals is required in order to conquer the city. Otherwise, the attack fails. Such a battle plan is said to be 30% **Byzantine Fault Tolerant** (**BFT**).

Subsection 12.1.1. BFT in Proof of Work

One of the key difficulties in blockchain, and distributed ledgers in general, is that there can be more than one correct way to order and execute transactions. For example, if you have $1000 in your checking account and your write two checks of $750, honoring the first, and rejecting the second, or the honoring the second, and rejecting the first, are equally valid and correct. The order in which the are evaluated does not matter, but it does matter that all versions of the ledger consider them in the same order.

This means there could be two equally valid, but mutually inconsistent, views of transaction ordering and the ledger. In other words, a blockchain could **Fork** and append correct blocks to a common root with different sets of transactions that were all valid. The question then becomes, how does the protocol decide which fork is **Canonical** in the sense that it takes precedence over any other internally correct ledger and chain view.

Ethereum (until recently), Bitcoin, and other PoW protocols follow what is called the **Longest Chain Rule**. This stipulates that the chain instance with the largest number of correctly committed blocks is considered to be canonical.

Forks occur in naturally Bitcoin due latency in the network. Two blocks at some height N+1 might be successfully mined at almost the same time by different mining pools. Nodes and miners in the network might receive either one of them first, and appending to block N, the curret last block committed to the chain. This is entirely within protocol, but results in two forks (call them fork A and B, and the blocks $N+1^A$ and $N+1^B$).

Fortunately, forking usually gets resolved quickly. Suppose that block $N+2^A$ (which can only be appended to block $N+1^A$) is mined and distributed before block $N+2^B$. Then nodes who currently are maintaining fork B should discard block $N+1^B$, attach block $N+1^A$ (which arrived late) to block N instead, and then attach block $N+2^A$ to block $N+1^A$. In other words, all nodes will resynchronize on fork A, and fork B will be **Orphaned**.

It is possible that a second pair of blocks, or even a third or forth pair, might be mined close to simultaneously, but the probability of this diminishes with the number of blocks. Eventually, one or the other fork will have more valid blocks available, and nodes should discard the shorter chain as it becomes increasingly improbable that it will ever catch up.

The key motivating factor here is that only the longest fork is canonical by protocol. If a miner builds off of a shorter fork, any mining rewards he might have earned are lost when the fork is orphaned. Similarly, any transactions that are in a block committed only to the shorter chain become invalid when fork is orphaned. Forks almost never continue for more than three or four blocks. For this reason, a transaction that is buried at least six blocks deep is considered to be almost certainly **Finalized** and immutable.

This brings us back around to the BFT of PoW protocols. Since the probability of mining a block is a function of hashing power, whoever has a majority of the hashing power will eventually create the longest chain with a probability that approaches certainty. Thus, if the majority of the hashing power is honest, the longest chain will also be honest. As a result, PoW protocols are said to be 49% BFT since they can tolerate anything short of a majority of Byzantine nodes.

Subsection 12.1.2. BFT in Proof of Stake

In PoA and PoS protocols, a two-thirds majority of the stake-weighted voters or nodes is usually required to commit a block. To see why this is so consider the following:

- First, suppose a 50% majority is required to approve a block, and 10% of the nodes are dishonest.
    - A dishonest block-leader could create two correct, but inconsistent, blocks, and then send one block to half of the honest nodes, the other block to the other half of honest nodes, and both blocks to all the dishonest nodes.
    - Each block would 45% of the vote from the honest nodes, since they follow protocol and approve valid blocks.
    - Each block would also get a full 10% of the vote from dishonest nodes. Signing two blocks at the same block height is called **Equivocating**, but there is nothing to prevent dishonest nodes from voting twice.
    - The result would two different blocks with 55% of the nodes apparently voting yes. Thus, a fork is created with two otherwise correct chains that have incompatible transactions.
    - Note that it would actually only take a single dishonest node to accomplish this only a 50% majority is required. Thus, PoS with a 50% threshold are only 1% (or one node) BFT.
- Second, suppose a 60% majority is required to approve a block, and 20% of the nodes are dishonest.
    - A dishonest block-leader creates two blocks follows the same strategy:
    - Each block gets 40% of the vote from the honest nodes, and 20% of the vote from dishonest Equivocating nodes. Again we have a fork.
    - Note that we must have at least 20% of nodes equivocate for this to work, so PoS with a 60% threshold is 20% BFT.
- Third, suppose an 80% majority is required to approve a block, and 60% of of the nodes are dishonest.
    - A dishonest block-leader creates two blocks follows the same strategy.

  - Each block gets 20% of the vote from the honest nodes, and 60% of the vote from dishonest Equivocating nodes.
  - Clearly, we must have at least 60% of nodes equivocate for this to work, and so it would seem that PoS with a 80% threshold is 60% BFT. This is sort of true, but …

- Fourth, suppose that suppose an 80% majority is required to approve a block, but 21% of nodes are Byzantine.
  - Byzantine nodes could simply not vote. In this case, no block, honest or not, would ever receive 80% of the vote. The chain would therefore be permanently stalled.
  - If M% is the required majority, then $100 - M$ is the fraction required to stall.
  - On the other hand, $2M - 100$ is the faction needed to create a fork through equivocation.
  - M = 67% is the magic compromise between these two types of failures. In this case, it takes 33% to stall the chain, and 33% to successfully equivocate. Increasing or decreasing M makes on of these two attacks easier, and so 33% BFT is the best that is possible for any PoS or PoA protocol.[34]

Section 12.2. Games and Algorithmic Game Theory

Blockchain protocols have their roots in **Algorithmic Game Theory** which adapts traditional **Noncooperative Game Theory** for use in computational environments. The costs of calculating best responses, equilibrium outcomes, operations of the validation protocol, and the complexity of the algorithm itself, are central concerns. Agents using such protocols without a complete understanding of how they work may have difficulty identifying fully optimal actions.

As a consequence, agents are often modeled as following *ad hoc* behavior patterns. If fully rational play exceeds the cognitive ability of agents, they might instead be modeled as being either malicious (Byzantine) players or honest players who simply follow the rules. It is also seen as reasonable to balance computational costs against security and nonmanipulability.

Protocols that give desirable outcomes with high probability are considered to be good enough for the real-world applications. If the odds that one, or a few, bad actors can change the outcome of a protocol are sufficiently small, then the protocol is considered to be robust to malicious behavior.

Algorithmic approaches tend to pay less attention to certain other elements of games and mechanisms. The next few sections explore the implications.

[34] Note we are rounding here. What you really get with M= 2/3 is a BFT of 1/3, so that $1/3 + \varepsilon$ node would be able to both stall or fork the chain.

Subsection 12.2.1. Multiple Equilibrium

If you read white papers produced by blockchain projects describing their platforms, or even the technical computer science papers showing the properties of their protocols, you will often come across a statement that reduces to the following: "There exists a Nash equilibrium of the implicit blockchain network game in which enough nodes follow protocol to allow the blockchain to function correctly". Another way of saying this is that good, correct, or honest behavior, is **Incentive Compatible**.

Of course, a protocol would not be of much value if honesty was not Nash equilibrium. However, what are if honestly turns out to be only one of several Nash equilibria? Should we consider each of these equilibrium outcomes equally likely, or should we give them different weights? The nature of Nash equilibrium is that if we happen to be at one, it has a kind of stability. Thus, there might be a certain path dependence at play in equilibrium selection.

The existence of multiple equilibria in blockchain protocols is not widely appreciated, but it is a critical problem. To see this, consider the following stylized blockchain game:

> Suppose there are 100 nodes. One of them gets chosen at random to propose a block and at least 50 other nodes must vote yes for the block to be approved. Nodes get paid 1 coin if they vote yes on a block that is approved, or if they vote no on a block that is not approved. They are penalized 1 coin otherwise (that is, if they are on the losing side of a vote).
>
> The block contains transactions that are chosen by the block-leader, and can collectivley move at total of 1000 coins to any accounts on the chain, including to accounts held by nodes. The only valid transaction, however, moves 1000 coins from Alice to Bob, who do not happen to run nodes. All other transaction that might be created by the block-leader are dishonest and incorrect under protocol.

Nash equilibrium 1: All nodes use the following strategy: Propose an honest block if they are block-leader. Vote yes on honest blocks, and no on dishonest blocks.

- If a node is the block-leader, he knows that all other nodes will vote yes if he proposes an honest block, and no if his block is dishonest. His best response is propose (and in so doing, votes yes) an honest block (the leader makes 1 coin instead of losing 1 coin).
- If a node sees an honest block, he knows that all other nodes will vote yes. His best response is therefore also to vote yes (the node makes 1 coin instead of losing 1 coin).

Nash equilibrium 2: All nodes use the following strategy: Propose a block that dishonestly transfers 10 coins to the account of each node if they are block-leader. Vote yes if only if this exact dishonest block is proposed, and no all other blocks, honest or dishonest.

- If a node is the block-leader, he knows that if he proposes this specific dishonest block, it will be approved, and he will make 11 coins. If he proposes any other block, it will be voted down and he will lose 1 coin. His best response is propose if to propose the dishonest block.
- If a node sees this dishonest block, he knows that all other nodes will vote yes. His best response is therefore also to vote yes (the node makes 11 coins instead of losing 1 coin).

⋮

Nash equilibrium ∞: All nodes use the following strategy: Propose some specific *"Block Z"* if they are block-leader. Vote yes on *Block Z*, and no all other blocks.

- Same made above applies: If a node is the block-leader, he knows that if he proposes *Block Z* it will be approved, and he will at least 1 coin. If he proposes any other block, it will be voted down and he will lose 1 coin.
- If a node sees *Block Z* , he knows that all other nodes will vote yes. His best response is therefore also to vote yes (the node makes at least 1 coin instead of losing 1 coin).

In other words, not only are proposing and voting yes for honest blocks a Nash equilibria, proposing and any block is a Nash equilibria. There are a litteral infinity of Nash equilibria supporting any outcome that is feasible. Put another way, honesty makes up zero percent ($\frac{1}{\infty}$) of all the possible equilibria. There is no particular reason to think the desired equilibrium is in any way special. All Nash equilibrium are created equal.

The solution is to design a protocol that has the property that the "good" outcome is the one and only equilibrium. **Economic Mechanism Design** is a subfield of non-cooperative game theory that considers such problems.

A mechanism (similar to protocol) is said to **Implement** an outcome Z in some equilibrium type X if Z is the unique outcome that can be supported as an equilibrium of type X of the mechanism. For example, implementing honest blockchain validation in Nash equilibrium would mean that the only behavior that is proof against unilateral deviation is universal honesty by all nodes. In other words, if all other nodes are honest, then you should be also honest, and any other outcome (half honest and half dishonest, for example) is unstable since at least one node would find that it is a best response to change his behavior.

Subsection 12.2.2. Nash, Dominate Strategy and Coalitions

The uniqueness of equilibrium, or at least a proof that all the possible equilibrium outcomes are acceptable in some sense, is fundamental. Protocols and mechanmisms in general are too manipulable to be useful in the real-world otherwise. Systems with many bad, but stable, outcomes are inherently fragile and risky. Unfortunately, even if all equilbria of an mechanmism are good, the equilibrium concept itself may not be robust enough for the application

In a Nash equilibrium, all agents choose best responses. That is, no single player would wish to unilaterally deviate and choose a different strategy provided that all other players continue to play their existing strategy. This seems to require that each player guess what the others will choose before he can figure out his best response. This might lead to mistakes or instability. Indeed, this is exactly the type of perturbation that could cause a game to transit from one Nash equilibrium to another.

If a player had a strategy that was a best response no matter what the other players did, his decision would be much easier. We call this a **Dominant Strategy** since playing it dominates every other strategy, in every circumstance. If all players in a game have, and decide to play, dominant strategies, the outcome is a called a **Dominant Strategy Equilibrium (DSE).**

The famous **Prisoner's Dilemma** is a classic example of a game with a DSE. Recall that the game involves two thieves who are caught by the police. They are taken to separate interrogation rooms and offered a deal. If they confess, and their partner does not, they will be allowed to turn state's witness and go free, while their noncooperative partner will go to jail for ten years. If both confess, they will both be convicted but given leniency in the form of a five-year sentence. If neither confesses, then they will only get one year in jail each since then they can only be convicted of lesser offenses.

The key observation is that each of them should confess regardless of what the other one does. Even if their partner stays silent, confessing allows the other thief to go free, while remaining silent himself results is sentence of one year.

Designing a game, protocol, or mechanism, such that the good outcome is the only DSE is challenging, and not always possible. Nevertheless, it is still not enough. While individual agents are always better off following dominant strategy, **Coalitions** of agents might do better collectively if they coordinate their efforts.

In the case of the prisoners' dilemma, suppose that the thieves were members of the Mafia and expected to be arrested many times over the course of their lives. If they could jointly agree in advance never to confess, then each time they were arrested they would only lose a year. Failing to make such an agreement means they would lose five years each time. Thus, while confessing is a DSE, it is not **Coalition-Proof.**

Bitcoin and PoW blockchain are supported by anonymous miners and nodes. There is no way to know if nodes or miners are Sybils run by the same real-world agent, or are truly independent. Miners explicitly form pools and coordinate their efforts to maximize payoffs. Similar coalitions are formed by stakeholders in PoS blockchains, and delegated PoS protocol variants explicitly facilitate the formation of coalitions. In short, it seems hard to image that any protocol where good behavior by nodes, stakeholders, and/or miners, was not a **Coalition Proof Equilibrium** could provide a robust security guarantee.

Subsection 12.2.3. Sequential Games

All the concerns about games and blockchains outlined above consider only the static case of a one shot-game. In reality, blockchain protocols are better modeled as **Sequential Games** that take place in stages over time. Bitcoin, for example is a game that is both dynamic and probabilistic. Miners have a chance of being allowed to propose the next block and earning a block reward that is proportionate to the hashing power they bring to the game.

Mining requires buying and installing ASICs, and so deciding how much to invest and when requires a miner to anticipate the future decisions of other miners whose investments compete. Miners must also guess at what the value of future mining rewards will be, and how users and governments might respond to actions by collations of miners, or to real-world events. In other words, blockchain is a repeated coalitional game with asymmetric and incomplete information, as well as an anonymous and endogenously formed set of players who may have different strategic objectives and payoff functions.

Needless to say, modeling all this formally, much less devising a protocol that implements the right outcome in the right kind of equilibrium, is no easy task. Blockchain continues to be a very dynamic and experimental field on both the academic and commercial sides. We will not be able to solve these problems here. Instead, we offer a simple example to give a real-world sense of how important information and expectations are in such settings.

Recall that under the Bitcoin protocol, the first miner to find the nonce for the next block is allowed to create a certain number of new Bitcoins (6.25 BTC currently) as a reward. The miner credits these to his account and other miners commit the block containing this action to the existing chain.

Suppose that all the miners were to get together and decide that they would start to give themselves rewards of 50 BTC per block going forward. Moreover, they would not commit any blocks that did not have a 50 BTC reward, and would simply ignore them instead. You can already, see based on the previous discussion, that this would be a Nash equilibrium for the simple, one-shot version on the game.

In the real-world sequential game, there are other things we must consider. Increasing the block reward is not allowed by protocol. Doing so is dishonest under the rules of Bitcoin. Moreover, both nodes and users can easily see that these are blocks are incorrect, and by protocol, should ignore them.

If nodes and users followed this rule, then miners would not benefit from their actions. Even if the only visible version of Bitcoin is a long chain with incorrect blocks, users may not accept any transactions or accounts recorded there as valid. In other words, the real-world value of coin balances recorded on this incorrect chain might be zero.

On the other hand, users may see this action as out of protocol, but essentially harmless. No coins are being stolen, the nonce is being correctly mined, and there is no alternative Bitcoin ledger on which to spend their coins. Really, all miners have done is formed a union to raise their wages from 6.25 BTC to 50 BTC per block. If users have to choose between walking away from their bitcoins or living with a small change in protocol imposed unilaterally by the miners, who is to say they will not choose the latter?

Here is the point: In a sequential game, both outcomes are plausible equilibria. If miners believe than users will accept their "dishonest" actions, and users believe that other users will continue to accept BTC payments from a "dishonest" ledger, then forming the mining union is an equilibrium. On the other hand, if miners believe that at least 51% of the miners would reject and ignore any "dishonest" blocks they produced, or that users would not see them as legitimate, then following existing protocol is an equilibrium. Any incentives that miners have to behave honestly depend critically on how users react to the existence of forks and errors, or more accurately, how potentially dishonest miners estimate that they will.

The fact that users can identify dishonest miners and forks does not empower them to do anything within protocol about it. Users are left with a set of bad choices. Theory does not help us to reject very many belief structures as wrong or irrational, and there is not much empirical evidence to help us estimate which belief structures are likely to emerge either.[35]

Subsection 12.2.4. Metagames

The discussion above considers how rational, self-interested agents should respond to the incentive structures provided by blockchain protocols and mechanisms. This all relates to incentives within the blockchain itself. Agents running modes and mining rigs, however, live in the real-world, and so, we need to consider the **Metagame** in which a blockchain is embedded.

One of the promises of blockchain is that it can be used to do things like hold public records such as land and car titles, and tokenize financial markets so that stocks and bonds can be represented as tradable items on blockchains, and be securely transacted without a brokerage house or other intermediary. It might also be used to control and monitor IoT devices, and various sorts of infrastructure such as electric grids.

What this suggests is that in the future, the value carried on blockchains could be in the trillions of dollars, and would represent real-world assets, not just cryptocurrencies native to the chains. This creates a whole new set of incentives for an adversary or bad actor to try to compromise a blockchain.

It may very well be that a tight protocol makes it irrational for nodes to try to steal coins. The costs of a 51% attack on Bitcoin might far exceed any residual value that controling a discredited

[35] Similar scenarios can be constructed for PoS and DAG protocols. In fact, they are easier in many ways, but, the same fundamental logic and conclusions hold.

ledger might have to an adversary. What if such an attack could transfer off-chain assets such as stocks or land titles? This raises the incentives for an adversary to an entirely new level, and the security guarantees needed to trust blockchains used for these purposes must be sufficient to discourage such an attack.

It might be that an adversary has objectives that are entirely unrelated to any direct gain that might be made by stealing from a blockchain. A hostile state actor, a competing platform, a commercial competitor, an anarchist, etc., might simply want to create chaos.

What would it be worth to Russia or China to crash to NYSE, or to North Korea to gain control of the US electrical grid? Maybe a consortium of banks considers it their duty to point out how insecure decentralized financial platforms are compared to the traditional banking sector. The NSA might be directed to destroy a platform that facilitated money laundering or financial transfers to terrorist groups by the executive branch. Even the highest end of the cost estimate for destroying a chain is far less than the cost of acts of war, or even the cost of maintaining an infrastructure to mount more conventional cyberattacks.

Section 12.3. Blockchain Governance and Code as Law

In this section, we consider two question. First, what should we make of the claim that in blockchain, "**Code is Law**", free from human bias or intervention? Second, should we aspire to this goal, or is there a role for **Governance** in blockchain?

Subsection 12.3.1. Is Code Law?

It is often claimed that what makes blockchain trustless is that code is law. The rules of the protocol are locked into code run by nodes and miners. Smart contracts automatically execute, and do exactly what they are written to do, predictably, and without fear or favor. In other words, blockchain is supposed to run according to unalterable deterministic rules agreed to in advance by the participants.

What is Bitcoin really?[36] Is it Satoshi Nakamoto's 2008 paper? Is it the program (called a client) that a Bitcoin mining node runs? The truth is, no one really knows and, at best, it is a moving target.

Nakamoto's paper describes a consensus mechanism, but does not specify how it should be implemented. It is a vision, or an intention, to be instantiated in code by developers, and then run by the validation network following the protocol he describes.

[36] We use Bitcoin here as an example to be concrete, but the discussion applies to all protocols.

Bitcoin is not "the code" run by nodes for at least two reasons. First, there are more than ten Bitcoin clients currently in use, written in different programming languages, for different operating systems, and optimized for different chip architectures. Second, all of these clients are patched and updated continuously. None of these versions defines what Bitcoin is, or what it should do. If they disagree or process transactions differently, none of them can claim that they are definitive and correct in any abstract sense.

Each of these client versions is really an interpretation of Satoshi's intention, or more accurately, what a group of nodes agree is the best evolution of his intention. In computer science, code is understood to be an **Imperative** instantiation of a way of doing something. That is, the code is a way of doing something, but does not in any way define what should be done. Client code does not, and is not intended to, define what Bitcoin is. More to the point, Bitcoin code is not law. How could it be? It changes almost daily.

Even if we agreed that one version of **Bitcoin Core** (the most popular client) defines exactly what Bitcoin is, we would still be building on sand. This is because Bitcoin depends on external code libraries to do things like encryption, signature checking, hashing, and communications. These are not written or maintained by some group of Bitcoin developers (and in any event, the existence of a privileged, centralized group with such responsibility, and authority, runs against the very idea of Bitcoin). Instead, these repositories are maintained, patched, and upgraded, by independent developers who have nothing at all to do Bitcoin.

This creates two obvious problems for projects like Bitcoin that aspire to be trustless and immutable. First, as these libraries change over time, they do different things. Even if some version of Bitcoin's code-base were to be frozen, its dependencies would not be.

Second, these dependencies create a significant attack vector. One of these repositories could accidentally or intentionally deploy an update with a bug that could damage client functions, or make them vulnerable to adversaries. In effect, the Bitcoin community must trust in the integrity and the competence of Bitcoin Core developers and dependency maintainers, in addition to miners who choose which client to run, and how.

Technical papers are abstractions that express intentions. but are neither specifications, nor definitions. of truth or function. Imperative code does what it does, but there is no standard that allows us to know if what it does is correct. We can't trust what we can't verify, and we can't verify what has never been defined.

If our goal is to have a fixed set of rules we all agree to live by, come what may, then we must have a formal and precise definition of what a protocol is supposed to do that can be checked against what a client actually does. Computer scientists refer to this **Declarative** programing. In practice, some protocols get closer, and some fall very far short, of meeting this standard. However, this is the objective that must be met for truly immutable and trustless blockchain.

Subsection 12.3.2. Governance on Blockchain

Many projects take the opposite view and argue that blockchain protocols should be designed to be mutable. At a higher level, if a protocol allows for changes in response to real-world data provided by oracles, by trusted authorities, through voting or some other form of governance, then changing the rules is really following the rules.

The case for governance is a sensible one. Blockchain protocols may need to be updated as technology or use cases change, smart contracts may contain bugs or allow unintended exploits, and there may be good arguments to rollback transactions that use stolen keys, pay ransoms, launder money, or fund terrorist activity. Creating a way to fix these problems seems like a very good thing. Moreover, allowing the people who use a blockchain to have voice in how it works sounds democratic, inclusive, and fair.

Unfortunately, there is a fundamental problem with any system of governance: **If changes can be made for good reasons, they can also be made for bad ones.** Governance requires that we trust the good intentions and wisdom of other humans. How is that working out for you? The basic arguments against governance are the following:

- **Theoretical Impossibly**: A great deal of economic research in voting theory and public choice proves that it is essentially impossible to create non-manipulable mechanisms that are in any way fair, equitable, or efficient. The most famous are the Arrow Impossibility Theorem (Arrow 1950) and the Gibbard-Satterthwaite Theorem (Gibbard 1973 and Satterthwaite 1975). These results applies to voting by majority, unanimity, rankings, iterative veto, and any other social choice function that maps the preferences of voters into outcomes. It should be immediately obvious that majoritarian governance is at best 50% BFT.
- **Empirical Experience**: It only takes looking at how governance works in practice to confirm this. Are you happy with the national leadership of the USA, UK, EU, Russia, China, or Canada? Is your local school board, or favorite charitable organization, well-run? Is the decision-making process at your university, your company, even within your family, satisfactory? How do you feel about the judicial system, or regulatory bodies? Some are better, and some are worse. What they have in common is that they confer power on people with varying degrees of qualification, ethics, objectivity, and public mindedness, to control the actions of others. Trusting in such a process in the context of blockchain or real life is a scary proposition.

If a governance system is good in the sense that it successfully aligns the interests of the governors with the governed, it may seem like having a degree of trust is a reasonable price to pay in order to let blockchains and applications evolve and improve.

Unfortunately, such a system is impossible. This is for the very simple reason that the governed don't have identical interests. For example, suppose that ten users take a loan from another user and that this is controlled by a smart contract. Then ten people might wish to have the contract canceled, or the loans paid off for ten cents on the dollar, while only one would wish the contract to remain unchanged. People who might benefit from the availability of loans in the future would find that lenders were no longer willing. There are winners and losers and no objective way of deciding if the change is good or bad overall.

At the protocol level, some developers may wish to build DApps with high transactions volumes for low value data (IoT telemetry, or microtransactions), while others may prioritize high security (tokenizing land titles, or stock certificates). Both parties are sincere and well-motivated, but their interests do not align. This may lead the protocol to evolve in ways that make it unusable for some developers who have already invested significant time and effort in building DApps.

Uncertainty about how a platform might change, which functions will be prioritized, and which might no longer be supported, can lead rational developers to choose to build on platforms without governance, where they can at least be confident in the permanence of the protocol as a foundation.

Of course, governance may simply be bad. Agendas can be controlled, Who is likely to spend the time to make protocol upgrade proposals that benefit the entire system? Who is going to take the trouble to read and understand them? We have a classic freeriding problem where most stakeholders may choose not to vote, or vote in a misinformed or uniformed way. If a 2/3 majority is required, then nothing may ever change. If a plurality of voters is enough, then a minority may hijack the platform.

All of this is to say that we live in a highly imperfect world. A compromise of some sort must be found to allow societies to agree on rules to prevent anarchy, and yet still respond to new challenges and changes in social priorities as they arise. This is the main reason that nations have constitutions that lay down basic rights and responsibility that cannot be altered without significant, and costly effort. Less important rules are more mutable. Drawing the right line is tricky.

In the blockchain universe, the fixity of the code is supposed to provide similar guarantees. If protocols, transactions, or ledgers can be changed once they are agreed to, immutably and trustlessness are lost. If smart contracts can be modified at all, they are not contracts, merely expressions of intentions that agents had when they made the agreement. On the other hand, the extreme view that “code is law” prevents protocols from fixing serious problems, and responding to new, and widely held, user demands.

Section 12.4. Participation and Incentives

The previous Sections have considered agents as participants in games. This requires thinking about the actions and reactions of many agents, both in coordination, and as individuals. Strategy sets can be complex, information uncertain, externalities between agents involved, and so on.

In this section we consider the much problem of creating direct, individual, incentives to participate in network activities. This is analogous to market equilibrium analysis where agents take prices as fixed, and respond optimally.

Subsection 12.4.1. Equilibrium Hashing Power in Bitcoin and PoW Blockchains

Bitcoin miners receive 6.25 BTC for each block they successfully mine.[37] Since the puzzle difficulty is calibrated to so that a block is mined every ten minutes on average. About 50,000 blocks committed, and 300,000 BTC mined, per year. At January 2024 prices of around $44,000, this is worth about $13 billion.

The costs of mining Bitcoin are mainly the capital needed buy and deploy specialized ASICs, and in the price of electricity to run them. How much hashing power will the Bitcoin mining network have in equilibrium? The answer is what ever amount of hashing power costs $13 billion. Most of this $13 billion will be spent on electricity, but about a third will go to buying, and supporting, mining devices.

To see this, suppose that the network only spent $9 billion. If I joined the network and spent a billion dollars on mining, I would have 10% of the hashing power, and so would receive 10% of the mining rewards ($1.3 billion). I would therefore make a net profit of $300 million. Similarly, if more than $13 billion worth of electricity was consumed, mining would be unprofitable and miners would shut down until the network was back in equilibrium.

Implications:

- The cost of a 51% attack on PoW blockchains with mining rewards varies as with the price of the coin it supports. This implies that declines in coin price leads to a proportional drop in security. Of course, the market-cap on the blockchain would decline in approximately the same proportion, so this might seem okay. If the blockchain supports smart contracts or tokenized securities (using NFTs, for example) that have value that does not vary with the coin's price, however, this is problematic. In this case, as the coin price drops, the benefit/cost ratio of attacking the chain becomes larger.
- Bitcoin's protocol calls for a **Halvening** of the block reward every 210,000 blocks (about every four years). Block rewards are scheduled to drop to 3.125 BTC on April 26. 2024. The incentive to provide hashing power, and therefore security to blockchain, abruptly drops by half every four years. Eventually, transaction fees will be the only reward Bitcoin miners receive. If these turn out to be relatively modest, then we can expect the hashing power devoted to the security of Bitcoin to become similarly modest in the future.
- Bitcoin, Ethereum, and most crytocurrency prices are highly volatile. Mining pools must nevertheless make the risky decision to purchase ASICs, which are durable, and have very few uses besides mining cryptocurrencies. This implies that the mining is a risky investment with an uncertain rate of return. If the people who build mining pools are risk averse, we will have less than the equilibrium level of hashing power outlined above, which makes the chain that much less secure.

[37] Miners also receive transactions fees, but these are small compared to the mining reward.

What we see here is that you get the security level you pay for in PoW blockchains. Whether it is enough is a good question. Bitcoin's market-cap in Janurary 2024 was about $860 billion, but even the most pessimistic calculation of the cost a brute force 51% attack is at most $11 billion.

Subsection 12.4.2. Equilibrium Staking in PoS Blockchains

The consumption of electricity is foundation of the PoW security model. Whether or not this is wasteful is complicated question. Nevertheless, many people would prefer an alternative with less environmental impact.

Proof of Stake blockchains come is an enormous number of variations. The simplest is to allow users to lock up coins in a kind of escrow and receive voting power and rewards proportional to the amount of their of locked stake. Typically, only a fraction of the total coinbase gets staked, with the rest circulating as a currency, or used to buy services on the blockchain.

Rewards may come from a newly created coins or tokens (similar to mining), from a stock held in reserve by the platform, or from transactions fees, among other things. Stakeholders also provide services such as maintaining the ledger, transmitting copies to other stakeholders or users, and computing smart contracts. Stakeholders that behave badly, for example, by proposing or voting for blocks with invalid transactions, are sometimes penalized by having their escrowed stake confiscated.

How much will be staked in equilibrium? As in the case of hashing power, staking will take place until the marginal benefit equals the marginal cost. The benefit of staking is the expected value of the reward. The costs are mainly the opportunity costs of posting the stake, and the real costs of providing any required services to the platform.

As an example, suppose that a blockchain paid a total of $100 million worth of coins to the stakeholders per year, but that it cost $25 million for stakeholders collectively to provide validation and record keeping services. This implies that the stakeholders divide an annual profit of $75 million.

Staking requires that coins be purchased and locked-up, which prevents stakeholders from investing the equivalent funds in other ventures. In 2020, the Dow Jones Industrial Average generated a 9.7% return including dividends, the S&P 500 index 18.4%, and the Nasdaq Composite 45%. The corporate prime interest rate was about 4%, and banks are willing to lend to businesses at rates of between 5% and 12%, depending on risk. Let's take the opportunity cost of money at 10% to make things easy.

We conclude that $750 million would be staked since this equates the rate of return for staking, with outside opportunities for investment.

Implications:

- Someone has to pay for the cost of staking. Ultimately, this has to be the users of the platform through transactions fees or something similar. For example, if there are $10 billion worth of transactions per year on the platform, a 1% transaction fee would be sufficient to raise $100 million.
- Paying stakeholders through newly minted or released tokens is equivalent to an inflation tax. If $100 million new tokens are released to pay stakeholders, them value of tokens adjusts so that the real value of the tokenbase is constant (see the discussion of the quantity theory of money, in the next chapter). This is equivalent to collecting transactions costs from users through an erosion of the value of their tokens, instead of directly through fees.
- As with PoW, uncertainty cause the amount of staking to decrease, all else equal.
- If a platform allows delegation of stake, or if coins can be borrowed or rented, it may become possible to gather enough stake to mount a 34% attack for short period of time relatively cheaply. While 34% of staking power can, in some circumstances, be enough to take over a chain, it is always enough to halt block writing. In particular, a 34% attacker can prevent any blocks from being committed to the chain that include transactions that would return the coins he borrowed or rented to their original owners, increase staking by other agents to bring the attacker below 34%. or punish his bad behavior by **Slashing** or confiscating his stake. Once this control is gained, the attacker cannot be dislodged except by an out-of-protocol hard fork.

Again, you get what you pay for. Paying the guards at Fort Knox minimum wage if a bad idea if you value your gold. For PoS security models to be credible, they must have a sustainable to way to incentivize enough stake-holding that an attack would be unprofitable.

Subsection 12.4.3. Participation and Freeriding

Too many blockchain protocols depend on altruism for functions and actions that are essential to making them work. Needless to say, this should cause some concern. Properly incentivizing agents requires two things at a minimum. First, it must be possible to correctly observe or measure the desired actions. You cannot reward (or penalize) what you cannot see. Second, the marginal benefit must at least equal the marginal cost of taking the desired action.

Here are some example where this fails:

- Bitcoin and Ethereum nodes participate in a gossip network, store and communicate blocks and transactions, and check that committed blocks are valid. If this node network did not exist, it would be very difficult to monitor the actions of miners, and get candidate transactions into the system. Nodes, however, receive no rewards for their participation.
- Mining empty blocks is cheaper and quicker than choosing transactions, validating them, putting them into blocks, and only then begin to mine them. The mining rewards are the

same, however, regardless a block contains. Unless the marginal benefit (the expected transaction fee the miner realizes) from including a transaction is sufficient to overcome this, processing transactions becomes an altruistic act by miners.

- DAGs require users to hash their transactions to a random endpoint of the graph that connects to a history of other valid transactions. There is no incentive for users or other network participants to convey large segments of the graph to one another, and no incentive for users to choose a random end point instead of a single well-publicized one. There is very little incentive to check that history of transactions leading to endpoint are all correct, even if the user had full graph data available.

- Some platforms allow users effectively to rent the network as a kind of virtual machine, or facilitate the renting of computational power from other users. The motivation is to have many independent computers run the same code so that processing is done in a neural way similar to smart contracts. If there is agreement on the outcome, this is supposed to give everyone confidence that the results are correct without the need for a TDI or to run the code one's self.

  Nodes are paid when they report their results and achieve consensus. This creates an opportunity for freeriding. Suppose one of the nodes processed the code, and passed the result to other nodes for confirmation. These other nodes cam choose to spend the compute cycles themselves to run the code and verify this conclusion (as they are supposed to), or they simply vote yes, instead. Doing so saves them considerable compute costs, but gives them the same reward.

  In fact the first node could just make up an answer, and expect the other nodes to vote yes without verifying its conclusion. The only way to know that the network is not processing the code, would be for original user to process it himself and notice the nodes got it wrong. But then this would defeat the purpose of using the platform in the first place. The difficultly in actually knowing what nodes do makes it impossible reward correct behavior, or punish bad behavior.

- As we point out above, any platform that has governance will find that most agents freeride. Either they will not vote, or they will not take the time to vote wisely. The benefit to any individual is lost in the sea of other voters. Being a good citizen is costly, and one individual’s small voice is unlikely to greatly affect the outcome. Large stakeholders, and groups that expect to benefit from a proposal, are more likely to vote, and this undermines the democratic intentions of governance. (The same is true in real-life, by the way.)

Section 12.5. Conclusion

One might be tempted to conclude that blockchains are a disaster from game theoretic standpoint. The examples above are only the tip of the iceberg. There are many more problems that are shared by whole classes of protocols, and each platform implementation introduces its own special set of issues. Given this, how can we consider blockchain to be an important new technology?

There are three basic contravening factors:

First, many platforms are not truly decentralized. In some cases, blockchains have privileged nodes and notaries by design. In others, core developers and nodes effectively have control of the both the protocol, and the ledger. In effect, there is an informal kind of governance on many platforms, including Bitcoin and Ethereum. The costs and benefits of forking protocols, reorganizing chains, and reverting transactions, all depend on how platform users and token holders might react to the change. This provides a real, but informal, incentive for those with such power to use it wisely. Of course, this does not provide a trustless guarantee, or cryptographically locked security, It may, however, be good enough, at least for now. Real-world banks, governments, and other institutions, are also trusted. but insecure, organizations. Blockchain, as it stands, is better in some ways, and worse in others, but in any event, offers a different set of possibilities and tradeoffs.

Second, this chapter considers attacks by self-interested, rational actors. While it is worrisome that there are so many attack surfaces, behavioral economics teaches us that humans are more complicated. It may very well be that many agents who run blockchain nodes are “honest”. They may believe that blockchain is cool, and interesting, and are willing to lend their support, and not be at all tempted by the potential benefits of joining a coalition to subvert a ledger. Depending on the kindness of strangers is not as strong a protection as “code is law”, but historically, it has often been the only option, and it has often been sufficient.

Third, most emerging technologies, AI, ML, cloud computing, and big data, for example, tend to support large, centralized, data-driven enterprises. Facebook, Amazon, Google, Microsoft, Apple, and Twitter, employ these technologies on a massive scale, and reap the rewards of the consequent network externalities. Once established, such companies are very powerful, and difficult to displace. This puts them in a position to impose their business models, terms of services, preferences, and values, without regard to the interests of their user-base. Blockchain is the only significant technology which has decentralization baked into its DNA. It may be the only refuge that individuals have from powerful, centralized companies, governments, and institutions. It is worth giving blockchain the time and space to develop.

Section 12.6. Economics

Subsection 12.6.1. Noncooperative Static Games

A One-Shot, Simultaneous Move Game: is a triple $(\mathcal{I}, S, F)$ where:

$i \in \{1,\ldots,I\} \equiv \mathcal{I}$:	**Players or Agents**
$c \subseteq \mathcal{I}$:	**Coalition of agent, a subset of the Grand Coalition**, $\mathcal{I}$.
$s = \{s_1,\ldots,s_I\} \in S_1\times\ldots\times S_I \equiv S$ where $s_i \in S_i$:	**Strategies**
$F \equiv (F_1,\ldots,F_I)$ where F_i: $S \Rightarrow P$:	**Payoff Functions**

We denote by P some **Payoff Space.** This could be any sort of finite or infinite set where the elements represent amounts of money or utility, market shares, discrete objects like houses or art, possible grades of a test, jobs, binaries such as wining or losing a game or a war, etc.

We can now define types of deviation strategies:

Deviation Strategy Profile from $s \in S$ for Coalition $c \subseteq \mathcal{I}$: $\hat{s} \in S$ such that $\forall i \notin c,\ \hat{s}_i = s_i$. and $\forall\ j \in c,\ \hat{s}_j = \hat{s}_j$.

Note that $c \subseteq \mathcal{I}$ could consist of a single agent, or the grand coalition, as well as an arbitrary subset of agents. The special case of deviation strategy for a single agent $i \in \mathcal{I}$ is defined as follows:

$$(\hat{s}_i, s_{-i}) \equiv (s_1,\ldots,s_{i-1},\ \hat{s}_i,\ s_{i+1},\ \ldots, s_I)$$

In words, a deviation from a strategy profile $s \in S$ in which the i^{th} element $s_i \in S_i$ deleted, and replaced with an alternative strategy $\hat{s}_i \in S_i$.

The following are the three most import solution concepts for noncooperative one shot games:

Nash Equilibrium (NE): $s \in S$ such that $\forall\ i \in \mathcal{I}$ and $\forall\ \hat{s}_i \in S_i,\ F_i(s) \geq F_i(\hat{s}_i, s_{-i})$.

Dominant Strategy: $s_i \in S_i$ such that $\forall\ \hat{s}_i \in S_i$ and $\forall s_{-i} \in S_{-i},\ F_i(s_i) \geq F_i(\hat{s}_i, s_{-i})$.

Dominant Strategy Equilibrium (DSE): $s \in S$ such that $\forall\ i \in \mathcal{I}, s_i \in S_i$ is a dominant strategy.

To define coalition-proof equilibrium, we will need to define a refinement of deviation strategies:

Credible Deviation from $s \in S$ for Coalition $c \subseteq \mathcal{I}$: $\hat{s} \in S$, a deviation strategy profile from $s \in S$ for coalition $c \subseteq \mathcal{I}$ is a credible deviation if $\forall\ \widetilde{c} \subseteq c$, $\nexists\ \widetilde{s} \in S$, a deviation strategy profile from $\hat{s} \in S$ for coalition $\widetilde{c}$. such that $\forall\ i \in \widetilde{c},\ F_i(\widetilde{S}) > F_i(\hat{s})$.

In words, a deviation strategy $\hat{s} \in S$ from $s \in S$ for coalition $c \subseteq \mathcal{I}$ if no subcontinental of c has a deviation from $\hat{s} \in S$ that makes all of its member better off then they are at $\hat{s} \in S$. Put another way, the deviating coalition does not rely on self-sacrificing behavior on the part of any subcoalition to support the stability of the deviation. We can now define the following:

Coalition-Proof Equilibrium (CPE): We say $s \in S$ is a coalition-proof equilibrium if $\forall\ c \subseteq \mathcal{I}$, $\nexists\ \hat{s} \in S$, a credible deviation from $s \in S$ for coalition $c \subseteq \mathcal{I}$ ft $\forall\ i \in c,\ F_i(\hat{s}) \geq F_i(s)$ and $\exists\ j \in c$ such that $F_j(\hat{v}) > F_j(v)$.

Nash equilibrium is proof against unilateral deviations, that is, deviations where one agent chooses a new strategy assuming the remaining agents keep their original one.

Coalition-proof equilibrium are proof against credible deviations by any coalition, including single agent coalitions, and the grand coalitions.[38] The restriction to credible deviations reduces the number of possible challenges to coalition-proof equilibrium. The motivation for this restriction is that it would be impossible to hold a deviating coalition together if some of its members were able could improve their payoffs by defecting from the deviating coalition.

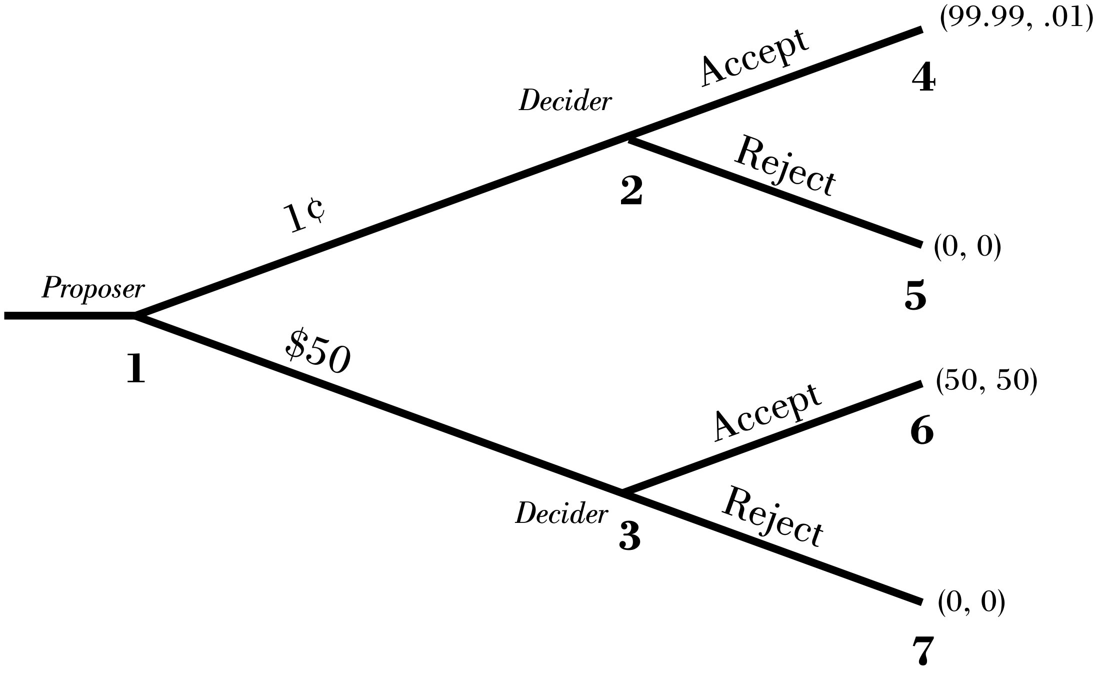
Subsection 12.6.2. Noncooperative Sequential Games

A **Sequential Game** is defined by the following list of items:

$$(\mathcal{I}, \mathcal{A}, \mathcal{D}, \mathcal{T}, Player, Action, Next, S, F)$$

where these represent a set of agents, actions, decision nodes, and terminal nodes, and a list of permissible actions at each node, linkages between nodes, strategies, and payoffs for each agent at each terminal node. Defining a sequential game formally, especially one with incomplete information, beliefs, and histories upon which players can condition strategies, is notationally heavy, and well not do so here. Sequential games can often be represented in **Extensive Form** as a kind of tree made of decision nodes and actions.

[38] The fact that coalition-proof equilibrium is proof against single agent deviations implies that it is a subset, or a refinement, of Nash equilibrium.



The example above is called an **Ultimatum Game**. The object is to divide $100 between two agents. The Proposer moves first (the game starts a decision node 1) and chooses one of two strategies: equal division, or keeping all but a penny for himself. If he chooses to be greedy, he puts the Decider in the subgame that starts at decision node 2. If he chooses to be fair, the Decider ends up in the subgame starting at decision node 3. At both nodes, the Decider can accept the proposed division, or reject it, in which case both agents get zero.

Note that if the Decider finds himself in subgame 3, his best response is to accept since that gives him $50 instead of zero. Accepting is also a best response in subgame 2, since the Decider gets 1¢ instead of zero.

The Proposer can use **Backward Induction** to figure out that accepting is the best response for the Decider in both subgames. His best response at decision node 1 is therefore to offer 1¢, since his payoff is larger in this game.

This logic depends on the Proposer's belief that the Decider will choose a best response in all subgames. What if the Decider announced that if the Proposer gets greedy, he will reject the proposal? He threatens to burn down the world, even he burns with it. On the other hand, the Decider will accept a fair division.

If the Proposer believes that the Decider will follow through on his threat, his best response is to propose fair division. Getting $50 is better than getting zero.

There are two Nash equilibria in this game.

- The first is where the Decider and the Proposer choose best-responses in all subgames. This is called **Subgame Perfect Equilibrium**, and precludes any non-credible threats. The outcome in this case is terminal node 4, and an unequal division of the prize.
- The second is where the Decider makes a non-credible threat to behave against his own interests in subgame 2, and the Decider believes that the Proposer will follow through. In this case, the best response of the Proposer is fair division. Since the game never ends up at node 2, the threat by the Decider to be irrational should the game arrive there is costless. It does not matter what you would have done in a situation you never face, since you never have to pay the cost of following though on your threat. It can, however, affect the choices of others, and the subgame you do end up facing. Thus, fair division paired with the Decider accepting fair division, but promising to reject the unfair division are mutually best-responsive, and therefore a Nash equilibrium.

Subsection 12.6.3. Arrow Impossibility Theorem

Kenneth Arrow investigated how of groups of people might make joint decisions that were in some sense fair. Abstractly, you can think of this problem as searching for desirable ways of mapping the set of preferences of a group of agents into a ranking over social alternatives. One might instead be satisfied with identifying a single “best” choice over a set of social alternatives.

Voting, in its may forms, is one mechanism to establish such a ranking. Markets, in their many forms, are another. Economic mechanisms, non-cooperative bargaining solutions. social choice functions, are other possibilities. There are an infinite number of ways to choosing or ranking social outcome based on agents’ preferences.

The question Arrow asked is: What are the minimum properties that we would want such a SWF to have. He comes up with the following list:

Unanimity: If all voters prefer A to B then the social ranking should place A over of B.

No Dictator: There must not be a dictator, that is, a person whose ranking of A and B is always the social ranking of A and B, regardless of the preferences of others. (That is, for every conceivable constellation of preferences for agents.)

Transitivity: If society prefers A to B and B to C, then society prefers A to C.

Independence of Irrelevant Alternatives (IIA): If society ranks A over B when C is not available, then it should not choose B over A if C happens to become available.

Unrestricted domain: The Social Welfare Function mapping preferences to social rankings must at least be able to consider every logically possible combination of individual rankings over problems with at least three alternatives.

Arrow argues that these are most mininal set of properties that any method of ranking social alternative should satisfy. There are many others than can be imagined, regarding equity or fairness, but it is hard to imagine that anyone would reject of Arrow's criterion as also being necessary requirements. Arrow proved the following Theorem:

Arrow Impossibly Theorem: There does not exist a Social Welfare Function satisfying the requirements listed above.

What this means is that no social mechanism that satisfies the most basic set of consistency and desirable criteria, can solve our problem. This failure is profound. It suggests that it is not a failure of imagination that is responsible for our dissatisfaction with the allocative outcomes our market or government institutions, but that no such institution exists.

In the case of proposed blockchain governance mechanisms, the flaws obvious and fatal to anyone who has studies voting theory or mechanism design. You can't always get what want, but if you try sometimes, you just might find, that you still can't get what you want.

Chapter 13. Cryptocurrency and Monetary Theory

Why are dollars worth anything to anybody? After all, they are just pieces of paper that have been blessed by the *Treasury Gods*. There were about 5.5 trillion dollars worth in circulation as of August 2023 in the form of US currency (M0). A broader measure of money called M1 that includes bank deposits in checking and money market accounts is estimated at about 20 trillion dollars. The electrons that encode these digital dollars must have been blessed by very tiny *Treasury Godlets*.

Money is both a **Store of Value** and a **Medium of Exchange**. We only accept dollars in exchange for goods and services because we trust that others will accept those same dollars from us when we want to buy something. We also trust that money will hold a reasonable part of its value over time so that saving dollars to be used for consumption in the future makes sense. If we did not trust these things, we would insist on getting something else, such as gold or beans, in exchange for goods and services, and would store these physical commodities to be exchanged for future consumption instead of real or virtual dollars.

Put another way:

MONEY IS TRUST

In principle, we could have a system in which favors were traded back and forth between people:

> … the doctor treats my cold, the plumber fixes the doctor's sink, I teach economics
> to the plumber's son, the plumber’s son makes a skinny frappuccino with two
> pumps of mocha syrup for the doctor …

This chain of favors works, and leaves everyone better off, as long as everyone provides favors of equal value to the favors they receive. Since the people we do these favors for will typically not be the same as the ones we ask favors from in this hypothetical economy, we would have to know that an agent who asks a favor has accumulated enough unredeemed favors for others to be of equal worth. Then, of course, we would need a way of deciding how much an economies lecture is worth in comparison to a doctor's visit. You can imagine how difficult valuing and keeping track of favors done and received would be.

Money, on the other hand, provides a convenient system that allows us to determine the relative value of things (through markets) and also to keep track of how much each individual has provided to, and taken from, the economic system. Money also serves the very important function of being a **Unit of Account**.[39]

US Dollars are what is called a **Fiat Currency**. "Fiat" is Latin for "let it be" (and Italian for "small car"). Dollars exist because Federal Reserve Bank of the United States decides that they should. It does not need to back the dollars it issues with gold or any other commodity. Thus, a downside of dollars is that if the Fed wanted to, it could run the printing presses overtime, and spend the new fiat dollars that it produces. This would cause inflation since more dollars would now be chasing the same number of goods. Dollars would lose value compared to real commodities. This in turn would limit the utility of dollars as a store of value.

Widespread counterfeiting of a currency would have the same effect, and would make using the currency as medium of exchange risky as well (since sellers would refuse to take any counterfeit bills they happened to detect). Fortunately, estimates suggest that only about a .01% of US currency is counterfeit. Another downside of currency is that it is easy to steal. On the other hand, the fact that using cash does not require revealing your identity, or create a record, is a distinct upside of fiat currency as a medium of exchange.

Section 13.1. Quantity Theory of Money

Whatever form money takes, there is one fundamental rule that it must follow: the **Quantity Theory of Money (QTM)**. Calling this a "theory" is misleading. The QTM is actually an accounting identity which is true by definition. As a result, the QTM is the engine that drives the value of both fiat and cryptocurrencies at the most fundamental level. Formally, this can be expressed in the following equation:

$$MV = \frac{T}{P}$$

where:

T : Transaction Volume (number of units of commodity transacted annually using the currency)

V : Velocity of Money (number of times a dollar bill or token is transacted per year)

M : Money Supply (for example, the total number of dollar bills or cryptotokens in circulation)

P : Price (the rate of exchange between the currency and commodities or other currencies)

To see how this works, consider the following example:

1. Suppose I launch GoatToken.io. The idea is that Goat traders will use my GoatToken instead of dollars to buy and sell goats. Rapid finality and goats on a chain! I issue 10,000

[39] There is nothing in the abstract that makes physical US dollars, electronic balances in bank accounts, bitcoins, gold, poker chips, cigarettes, or Yap Island Rai stones, the best choice to serve as money. Less abstractly, they each have advantages and disadvantages.

GoatTokens ($M = 10{,}000$). It turns out GoatToken.io is a success and 50,000 goats are bought and sold each year on my platform ($T = 50{,}000$).

2. I look myself up on CoinMarketCap.com and discover that my tokens change hands 10 times per year. That is, 100,000 GoatTokens change hands each buy and sell goats. Thus, GoatTokens have an annual velocity of $V = 10$.

3. Given this, the QTM tells us what the value of GoatToken must be. If 50,000 goats are sold for a total of 100,000 GoatTokens each year, then it must be true that each goat sold for two tokens. In other words, each GoatToken must be worth half of a goat ($P = 0.5$). Nothing else is possible.

Looking at the equation again we find: $10{,}000 \times 10 = 50{,}000/P$. In other words, if we know the first three variables, we also know the token's value. Therefore, our focus should be on understanding and estimating these three variables.

Volatility and inflation are the traditional problems that central bankers have faced in making monetary policy. Cryptocurrencies suffer from the same problems, perhaps to an even greater extent. The tokenomics of any project are just as important as the monetary policy of a country when it comes to establishing confidence in a currency. Let's look more deeply at the first three elements of the QTM equation.

Transaction Demand: In the real-world, transaction demand is the value of the goods that people decide to buy and sell using a given currency. In the cryptoworld, this is the dollar value people wish to exchange using a given cryptocurrency. For example, people chose to exchange about $350 billion in Bitcoin in 2022[40], which was about equal to its average marketcap. The level and value of economic activity in a country, or on a platform, is a result of decisions made by self-interested agents in the economy. It is not under the direct control of the platform or other monetary authority. However, the greater the quantity and value of things traded on the platform, the greater T will be, and so, the greater the value of the currency, P, all else equal.

Velocity: How quickly tokens move (or equivalently, how long they are held on average) is also determined by user choices. However, platforms can be designed to slow down velocity. For example, if tokens are used for staking, are put into escrow so that they can be transacted on side-chains, need to be locked up in smart contracts to do something valuable on a platform, or if transactions are slow or costly to complete, then velocity is lower, and token value higher. Velocity is also slower when speculators **Hodl** (slang for holding a token in hopes that its value will eventually go up). Since speculators can sell their tokens very quickly as well, low velocity driven by hodling is unstable and likely to lead to large token value fluctuations.

Money Supply: The number of tokens issued, and how they are issued is completely under the control of the platform. Double the tokens, and you halve their value, all else equal. If money or

[40] $1.4 trillion of Bitcoin have been traded in 2023 as of the end of July, which is about 2.5 its average marketcap. Projecting this linearly suggests a velocity of about 4 in 2023, as compared to about 1 in 2022.

tokens were created in a fixed quantity, and could never change, then their value would depend on T and V alone.

Most blockchain platform launches issue a fixed number of premined coins.[41] In some cases, more are issued on a fixed schedule, when benchmarks are achieved, or as a result of mining, forging, or other user activities. Some platforms even require holders to lock up their tokens and take them out of circulation for a set period of time in order to get certain rights or benefits. Sounds good, right? Startups are pledged not to dilute their monetary base, and speculation is discouraged. What could possibly go wrong?

What we learn here is that an essential requirement for a token to have value is that it have a significant use case on its platform. Unless the token is used for large transfers of value on a continuing basis, it is not likely to have much real-world value. Token price is also likely to be larger if platform elements, such as staking, structurally reduce velocity. Finally, fewer tokens imply higher token value.

Section 13.2. Efficient Market Theory

The QTM is always true, but it only takes us so far. Transactions demand and velocity depend on the choices of token holders and platform users. But what guides these choices? The short answer is beliefs and expectations.

Economics and economists are of very little value in estimating what people actually think about the future, or more importantly, predicting what they will think about the future, *in the future*. This is the main reasons that economists (and every other type of pundit) are so frequently wrong. Economics does, however, provide at least one important tool that helps us understand how what people believe affects prices.

Efficient Market Theory (EMT): A theory which says that the best predictor of tomorrow's price is today's price. More formally:

$$P_t = E(P_{t+1}).$$

Suppose that the price of a token today was lower than its expected price tomorrow. Then you could buy it today, hold it, and sell it tomorrow at a profit. Since this arbitrage is available to everyone, today's price would necessarily be driven up to equal tomorrow's expected price. If a token's value is expected to fall in the future, these future expectations would be transmitted to the present as owners sell the token until its price falls to its future expected value.

[41] One significant problem is that in many token launches, the company's founders retain a significant share of the tokens. This fact has caused considerable anxiety in the Ripple community, and has led to disputes and legal action between Ripple Labs and Jed McCaleb, its cofounder, and owner of about 9 billion XRP (about 9% of the total supply). In May of 2017, Ripple Labs pledged to lock up 88% of its own holdings of XRP. This may have led to more recent unwelcome attention from the SEC.

It is important to understand that EMT has nothing to do with the correctness of these predictions. EMT also applies to stocks, bonds, fiat currencies, and every other type of asset class. It may very well be that the current price of a stock has nothing to do with the underlying value of the company.

The "efficient" price reflects the average estimate of an asset's future price, but this may turn out to be completely wrong. If you are smarter, or have more knowledge than the rest of the market (especially inside information), you can profit by buying or shorting the asset and waiting for expectations to catch up to its real future price.

This brings us to the essence of the problem of estimating the value of a platform's token. There is nothing fundamental that pins down expectations of future prices at any moment in time. In part, these are driven by expectations of future transactions demand and velocity, but nothing pins those down either.

Historical Ethereum/USD Price					
Jan 1 2016	Jan 1 2017	Jan 1 2018	Jan 9 2018	Jan 1 2019	Jan 1 2020
$.90	$7.90	$776	$1300	$140	$130
Jan 1 2021	Jan 20 2021	Jan 1 2022	July 1 2022	Jan 1 2023	Jan 1 2024
$742	$1250	$3700	$1200	$1200	$2275

The table above shows the dollar price of Ethereum, on or about, January 1 over several years. Clearly Ethereum is extremely volatile. What does this mean in light of EMT? Clearly, if you had known in 2017 what the price would be in 2018, you would have invested every nickle you could beg, borrow, or steal. Unfortunately, you did not know, and neither did the market as a whole. So, was the price of $7.90 in 2017 wrong, or was the $776.00 in 2018 wrong? Maybe 2019 and 2020 got it wrong since the price jumped back up to the 2018 price at the beginning of 2021. There must be a mistake somewhere!

The point that EMT makes is the ALL of these prices were right, if right is understood correctly. Suppose that the average expectation was that ETH price would stick at $200 for the foreseeable future. Then why would anyone buy it at any price higher than this? On the other hand, if people expected that by the end of the year ETH would be worth $800, they would start buying it now and continue until the price got to $800 (or close to it). In other words, $200 and $800 are both equilibrium prices.

What price we actually see depends entirely on expectations of future value. In other words, there are multiple equilibria, in fact there are an infinity of them. You can see why predicting the future prices is so difficult.

Section 13.3. Stablecoins

If it were possible to create a **Stablecoin** that had a fixed value with respect to dollars or other fiat currencies, it would relieve a great deal of the public's concern and anxiety about using cryptocurrencies. There are three main approaches to stablecoins. Only one works, except, it doesn't.

Non-Collateralized Monetary Policies: Coins backed by the selling of bonds to reduce money supply when token value drops, or a policy that taxes or burns outstanding tokens held by users to reduce money supply, that use seigniorage shares, and other forms of derivatives, to prop-up token value, and so on.

Crypto-Collateralized Stablecoins: Coins that are issued only when ETH, or some other cryptocurrency, is put into escrow in a smart contract to serve as collateral.

Fiat-Collateralized Stablecoins: Coins that can be redeemed for dollars, euros, or even gold, on demand by some mechanism.

Subsection 13.3.1. Non-Collateralized Stablecoins

Non-collateralized stablecoins are an economic impossibility. Effectively, the value of the coin depends upon trust in the value of the coin. For example, suppose I issued a stablecoin that I wanted to peg at a value of exactly $1. Unfortunately, the market price of my coin on exchanges drops to $.90. In response, I offer to sell a bond for one coin that will pay you 1.1 coins when the coin's value goes back up to $1. Whoopee! you cry. Free money!

All you would have to do buy is a bond today with a coin worth $.90, and you will get back 1.1 coins later worth $1.10, a pure profit of $.20 for each bond you buy. If enough people believe this, then enough tokens are taken out of circulation (since the platform holds the tokens used to buy these bonds) and the wondrous forces of supply and demand drive the price of the stablecoin back up to $1.

Notice what I slipped in there: "if enough people believe it". If they don't, then not enough bonds are purchased, the price stays below $1, and the bonds can never be paid off. A stablecoin that has broken its value promise is not one that inspires confidence. It is likely to fall even further in value, and quickly enter a death spiral. No matter what approach is used, in the end, it always comes down to some version of this story: the value of the token depends on people believing in the value of the token.

Subsection 13.3.2. Crypto-Collateralized Stablecoins

Crypto-collateralized stablecoins would certainly work if you wanted to peg the value of one token to the value of another token, but what would be the point in that? You could simply use the original token.

Using cryptocurrency to collateralize a peg of a token's value to a fiat currency, on the other hand, is quite tricky. To begin with, there are a raft of practical concerns such as whether the smart contract used is bug free, and the cryptocurrency collateral store is secure. There is also a more fundamental problem resulting from the volatility of whatever cryptocurrency is being used to back the value of the stablecoin.

To understand this, suppose I put $100 worth of ETH in escrow in exchange for $100 worth of a stablecoin. Obviously, if the value of ETH falls, there is not enough value in escrow to redeem the stablecoins that were issued at their stated value. Over-collateralization is needed to protect against this possibility.

Suppose instead that in January of 2018, I deposited one ETH worth $800 to be held in reserve against $400 worth of a stablecoin. As it turns out, the ETH/USD price dropped from $800 in January to $200 in September. As a result, only half of the required collateral would be on hand to redeem the stablecoin. Of course one could require a four to one, or a ten to one, deposit of ETH for each stablecoin, but this does not solve the fundamental problem of volatility in the value of the currency used for collateral.

An even more serious problem is that stablecoins do not protect the original purchaser from volatility in ETH price. Let me make this clearer: a stablecoin purchaser who deposits one ETH in May gives up something worth $800 in exchange for a stablecoin which is only worth $200 in September. By buying the stablecoin in May and holding it until September instead of cashing out into dollar bills (which are remarkably stable with respect to the value of a dollar), the purchaser loses $600.

In effect, the original purchaser who funds the escrow is taking all the volatility risk upon himself to make a coin that has a stable value for subsequent users (always provided the value of the escrow does not fall too much). This is a very generous act, but it is hard to see it as a foundation for sustainable monetary system. Again, there are many variations on this basic approach, but all are subject to similar problems.

Subsection 13.3.3. Fiat-Collateralized Tokens

This brings us to fiat-collateralized tokens. The good news is that these could actually work, at least in theory. The bad news is that it is extremely difficult and costly to do so in practice. Let's begin by looking at some real-world examples.

When you deposit money into your checking account, what you get in exchange is an addition to your balance in the bank's database. In effect, this is an electronic IOU recorded in a private ledger. Why should you trust that the bank will actually give you back your dollars when you ask for them? If you think the bank manager looks like an embezzler, or that the bank might be insolvent, you would be well advised to go to the nearest ATM and withdraw all you can before the bank is out of money. (This is called a **Bank Run**.)

As is well known, banks do not keep one dollar in a vault for each dollar on deposit. Deposits are only fractionally collateralized. The rest is lent out, and so is illiquid. Some loans may even have gone bad, and the bank may not have enough assets, liquid or illiquid, to pay back all its depositors.

To prevent the bank runs that might result, the **Federal Deposit Insurance Corporation** (**FDIC**) was set up to serve as a guarantor of bank deposits. (They are from the government, and are here to help.) The FDIC promises to give you dollars in exchange for electronic bank balances on demand if the bank cannot. As long as we believe this, our deposits are fully collateralized by the printing presses of the US Federal Reserve.

This is an example of a fully fiat-collateralized token (electronic bank balances) that works. A dollar in our accounts can always be redeemed for a dollar in fiat no matter what the condition of the bank. If the FDIC or Federal Reserve happen to fail, it will most likely be because of a zombie apocalypse, or some larger catastrophe. Pieces of paper with dead presidents on them will probably not be high on our list of concerns in this event.

A somewhat less successful set of examples are gold (or other commodity) standards that attempt to peg dollars or other fiat currencies to a fixed amount of metal. If a 100% gold reserve were kept on hand, and everyone trusted the government to honor its redemption promise, this would work just fine.

The problem is, some wiseguy, usually an economist, points out that all that gold is just sitting there doing nothing. Why not lend it out, build some roads, feed the hungry, pay for economic think tanks, and fund other key governmental functions? After all, how likely is it that everyone is going to want to get their gold back at once? Let's just keep half of it on reserve, I mean a third, or perhaps a tenth …

The inevitable result is that someone mounts a speculative attack, sells dollars short, and demands gold, until the government gives up or runs out. The value of the dollar plummets, and the currency speculator walks away with a nice profit from his short position.

An even less successful example was England's attempt to maintain a 2.7 mark/pound exchange rate as part of its effort to support the European Exchange Rate Mechanism in the early 1990s. George Soros and other currency speculators shorted the pound, forcing the Bank of England to raise interest rates, and commit large parts of its foreign exchange reserves to buying back the pound on the open market.

This became increasingly difficult as the Bank of England's reserves dwindled. Ultimately, England was forced to give up and let the exchange rate float. There are too many other examples of failed attempts to support under-collateralized fixed pegs to count. They litter economic history like empty kombucha bottles after Burning Man.

The underlying economics here is that the one and only way to support a fixed exchange rate is to have a 100% reserve of the other currency. For example, suppose Venezuela issued 100 billion Bolivars and wished to maintain a ten to one exchange rate with the US dollar. The Venezuelan central bank would need to have 10 billion dollars on reserve in order to credibly claim that it would be able to defend this exchange rate, come what may. The same thing is true of commodity backed currencies, as we pointed out above. There simply is no free lunch.

Subsection 13.3.4. The Demand for Stablecoins

There have been many attempts to create non-collateralized stablecoins. Almost all have died and none have significant use. There also exist a much smaller number of fiat-collateralized stablecoins, the most prominent of which is **Tether's USDT**.

The reason that Tether is valued is that it provides a liquid way of moving into, out of, and between, other cryptocurrencies. Tether also offers the advantage of allowing users to keep dollar denominated wealth on the blockchain instead of banks. This reduces the visibility of transactions to tax and other authorities.

Tether claims to have a dollar on deposit in various banks for each USDT it issues. It is difficult to fully verify this claim, and even if it were true, what would Tether do if everything started to collapse? Would they take the remaining money and run? Nothing prevents this from a practical standpoint. What if the local banking, or other authorities confiscated, or locked, Tether's accounts? What if a court made Tether pay off a judgment out of these reserve accounts? The point is there is a significant amount of trust needed for even fully fiat backed stabletokens to work, and they expose users to significant risk.

Nevertheless, USDT has almost always traded within a few cents of $1 since its inception in early 2015. For the most part, the price has stayed in the $.98 to $1.02 band, which is explained by the transactions cost of getting in and out of Tether. As of August 11, 2023, approximately $85 billion worth of Tether was in existence. The table below gives some comparisons to put this in context:

Currency Market Cap and Velocity Projected as of August 11, 2023		
Currency	Market Capitalization (Billion USD)	% Annual Velocity (Market Cap) ÷ (Annual Volume)
U.S. Dollars (M0)	5,500	1.4
Tether	85	26
Bitcoin	570	4
Ethereum	220	3

The annual velocities of Bitcoin, Ethereum, and most other cryptocurrencies, are highly volatile. They usually range from .3 to 4, while Tether generally ranges from 10 to 25.

Velocity for the USD is not as volatile, but shows a clear secular trend. In 2010 there were about $850 billion of M0 in circulation, and velocity was about 10. The Federal Reserve Bank has been printing new dollars at a furious rate since then, and as the table shows, $5,500 billion were in circulation as of 2023. If velocity had remained the same, the quantity theory of money equation would have required a significant drop in the value of the dollar. As it turned out, however, velocity dropped to about 1.4 instead. Thus, while the monetary base went up by a factor of 6.5, velocity dropped by factor of about 7, keeping the equation is balance.

Section 13.4. Blockchain, Financial Economics and the Law

One of the most difficult questions in blockchain has nothing to do with the underlying technology. The legal status of cryptocurrencies is completely unclear despite having existed for more than a full decade at this point. This section will briefly discuss the major issues.

Subsection 13.4.1. What are Cryptocurrencies?

The most fundamental question is what exactly is a token? Is it a currency, a security, a commodity, voucher, or a meaningless invented digital object? This makes a great deal of difference regarding how it regulated and used.

Currency: Bitcoin was intended to be a decentralized currency that allowed people to trade value in a trustless environment. Like any other currency, it is only worth what people agree it is worth. Owning bitcoins does not give you the right any profits from a company or to vote over company policies. In fact, there is no company at all behind Bitcoin, and that is kind of the point. Bitcoin is probably the purest example of a token that is transactional currency, but many others follow Bitcoin's model closely.

Security: A security is an investment contract. This includes equities, corporate bonds, futures, and derivatives, among other things. In US law, the **Howey Test** says that a transaction an investment contract (and therefore a security) if:

- There is a money investment
- There is an expected profit associated with the investment
- The money investment is a common enterprise
- Profit comes from the efforts of a third-party or promoter

How crypto fits here is unclear. If a purchaser of a token does so with the primary expectation of making profits from an increase in the token's value, then this satisfies part of the Howey test. However, if tokens are purchased to pay for services on a platform, then these conditions fail. The more decentralized a platform is, the less it can be viewed as a common enterprise. and the less are any profits due to the platform's creator or to a promoter. This test aside, tokens that offer shares of profits from the platform, or voting rights to token holders, come closer looking like a standard stock or equity instrument.

Commodity: One could view tokens as a digital commodity, manufactured by a platform. which the buyer thinks is useful for some reason. For example, Magic cards or Pokémon cards are physical commodities that are manufactured and sold for prices that far exceed their costs because people value them. The makers do not promise any share of the profits or voting power, nor that the monetary value of the cards will increase. People nevertheless buy them in part because they expect that the value will appreciate, and the company is aware of this. In other words, both tokens and cards may neither be money, nor securities, although both have value primarily determined by their users.

Voucher: One could also think of tokens as being like poker chips or vouchers, that can be exchanged for services. Companies issue both of these, and sell them to consumers. The value they have is determined by the goods or services they can be exchanged for. Poker chips are like a stable token that is backed by the full faith and credit of the casino, and is the only currency that can be used on the platform (that is, to gamble at the tables). Vouchers are similar but may change value depending on the demand for the services they can be exchanged for. Many cryptocurrencies fit this.

Meaningless Invented Digital Object: Some blockchain startups have taken the path of starting a non-profit foundation, usually in Switzerland. The foundation accepts donations and gives away tokens (presumed to be invented, and intrinsically valueless) as a premium to its supporters. The foundation then uses the proceeds to build the blockchain platform itself.

Subsection 13.4.2. Why does this Matter?

If cryptocurrencies are simply a currency, then there are a wide array of regulations that are supposed to be followed. The most important of these are KYC/AML. In addition, platforms may be considered **Money Transmitters** which requires special licenses, and includes strict reporting and operating requirements. KYC/AML is an expensive process, and more to the point, it deanonymizes account holders.

This is certainly contrary to the vision of blockchain as digital cash that protects the financial privacy of users. It is also contrary to the vision of blockchain as a decentralized platform that allows people who don't know or trust one another to create and exchange value. Nevertheless, these regulations exist. It is unclear how they could ever be imposed or enforced in the case of Bitcoin, Ethereum, or other truly decentralized platforms.

If a token is a form of security, then the **Securities and Exchange Commission (SEC)** requires that issuers follow very strict rules. This involves registering the token as a security and selling it on regulated exchange (both very expensive processes). Few, if any, platforms that have issued tokens have been more than partially compliant with these regulations. The fixed and transactions costs of compliance would destroy any value a token might have as currency on a platform.

The SEC has legitimate concerns. The reasons for these regulations is to protect unsophisticated investors from being scammed. There are exceptions carved out for wealthy **Accredited Investors** to buy unregistered securities because they are considered sophisticated enough to take care of themselves. On the other hand, this has lead to complaints that ordinary people are thereby precluded from investing in new technologies and startups, and that this is unfair.

Tokens created a new way for startups in the blockchain space to raise capital. Instead of an **Initial Public Offering (IPO)**, blockchain platforms can hold **Initial Coin Offerings (ICO)**. IPOs are expensive and heavily regulated. Although ICOs in many ways look like IPOs, it is not clear what regulations apply since it is not clear what is being offered, a currency, a security, a voucher, etc.

In 2017 about 1000 companies launched ICOs and raised about $6 billion. Somewhat more was raised in the first part of 2018, but then the market largely dried up. This was because most of the 2000 or so projects that were funded were ill-conceived, failed to deliver, or were actually fraudulent. In many ways, this echoed the dot com bubble of the year 2000.

Creating and selling commodities does not require much regulatory compliance. If tokens are commodities that cost nothing to produce, however, then all the revenue raised may be corporate income, and subject to tax. Instead of being able to invest the full amount raised (as with equity sales), only about 70% would actually make it into constructing the platform.

Thinking of tokens as vouchers or poker chips makes a lot of sense on many platforms. These are often called **Utility Tokens** to distinguish them from **Security Tokens**. When a platform provides services such as distributed data storage, brokering the sale of unused CPU cycles, or keeping chain of custody records for logistics and provenance, then the token is needed to pay for these services on the blockchain.

Dollars can't be used because they don't exist on the blockchain, and high transaction costs makes out-of-band payments unpractical. Platform users therefore buy tokens for cash, give them to nodes and other users on the blockchain in exchange for goods and services, who then sell them back to other platform users who want to pay for services. This keeps the platform and its ecosystem running. To the extent a cryptocurrency is a utility token, the regulatory regime is relatively light and compliance is fairly easy.

Subsection 13.4.3. Conclusion

The legal discussion above focuses on US regulations. There is a whole world out there, however, with its own set of ideas regarding blockchain. Gibraltar, Malta, Switzerland, Estonia, and a few other jurisdiction are very open to blockchain and have laws that help facilitate its use. Others completely forbid their citizens from holding tokens or using blockchains.

This creates a confusing patchwork that makes it impossible to know what is legal and compliant in this space. This is similar to the case of cloud computing where users might come from any jurisdiction, the data might pass through any country, and the platform might or might not have a legal existence in each place. Even if the laws were clear in every detail, it still would be unclear which should be followed in any given case.

Chapter 14. Security and Privacy

This chapter deals with security threats and what you can do to protect yourself. In addition, it discusses privacy and identity, where the dangers lie, and how you can minimize your exposure.

Preventing malware infections is a necessary step to protecting a user's data from exposure and corruption. Of course, users still need to send data in and out of their networks, allow their computer to run programs, and to respond to instructions coming from external sources. The key is to find a balance that allows in and out only what the user wants and prevents undesired actions without unduly hindering desired ones. The first line of defense is called a firewall.

Firewall: Hardware or software that implements a set of rules to allow or disallow traffic going into or out of a network. Firewalls try to determine if external requests are valid using strategies such as inspecting each packet entering or leaving a network, authenticating and establishing secure connections between internal and external nodes before traffic is allowed, and restricting access based on the internal application that is involved in the traffic.

Section 14.1. Malware and Hackers

Malware is a general term for software which is deliberately designed to act in a way that is contrary to the requirements of the computer user. It does not include software that causes unintentional harm due to a bugs or flaws (this is just bad software). Examples include all the animals below.

Spyware: Software that reports the actions of, or shares data belonging to, a user without his permission and/or knowledge. Secretly uploading a user's files to a server is an obvious example. Sometimes this is even done by apps or applications that the user has intentionally installed.

Spyware can record **Clickstreams** (a record of where on the screen a user clicked when using apps, or a browser), **Heatmaps** (a record showing where on the screen the user rests his mouse, or where a user's eyes are focused), or **Keystroke Logs** (a record of everything you type, including passwords, account numbers, and other private information).

Operating system and hardware report all kinds of telemetry to their producers including user address books, emails, text communications, files backed up to the cloud, media files on the user’s disk, and presence or absence of associated licenses, directory listings, browser history and so on.

Virus: A type of malware that replicates by inserting copies of itself (possibly modified) into other computer programs, data files, or the boot sector of the hard drive. To do so, a virus must attach itself to a host program. It is unable to execute and replicate on its own. If a virus does attach itself to a program and replicate, then the computer, or the affected data sectors, are said to be **Infected**.

The great majority of viruses target systems running Microsoft Windows. Unix-like systems (including Linux and OSX) have more robust security architectures which are more difficult, though not impossible, to attack. The motives for creating viruses include seeking a profit, sending a political message, personal amusement, demonstrating that vulnerabilities exists in software, hardware, or operating systems, sabotaging commercial rivals or political opponents, spying on users or stealing data, building botnets, and exploring artificial life and evolutionary algorithms, to name a few.

Although viruses can be relatively harmless, they generally waste computer resources, corrupt data, and even cause system failures. Billions of dollars are spent each year protecting against viruses and repairing the damage they do.

Worm: A type of malware that replicates itself without needing to be attached to another executable program. As they replicate, viruses must corrupt program files. Worms can replicate without altering files. Otherwise, worm are very similar to viruses and can be designed to do the same types of things.

Trojan: A Trojan is generally a non-self-replicating type of malware containing malicious code that typically causes loss or theft of data, and other damage when executed. The term is derived from the story of the wooden horse used to trick the defenders of Troy into taking concealed Greek warriors into their besieged city. Similarly, computer Trojans often present themselves as routine, useful, or interesting programs, in order to persuade victims to install them on their computers.

Ransomware: A type of malware encrypts a user's files and demands a payment (often in Bitcoins) to obtain the decryption key.

Man-in-the-Middle Attack: If some aspect of your network is insecure, a hacker can place himself between your computer and your router, or your router and your target URL. If a malicious agent hacks into your wireless network, taps into your wired network, breaks through your router’s firewall, or enters through an insecure IoT device, he becomes a trusted node within your LAN. Trusted nodes may be able to see, alter, delete, or initiate forged traffic on your network. They can access disks and attached IoT devices, or introduce worms and Trojans to your network. Malicious agents can also compromise DNS servers, intercept email and HTTP traffic passing through compromised backbone routers. There are even cases of bad actors setting up fake PCS towers the get between your phone and your provider.

Zombies and Botnets: Malicious Actors using viruses, Trojans, or other malware, and gain access to a computers hardware and resource, and use them for their own purposes. We call such infected computers Zombies, since they do things without the owners knowledge or permission, in response to external commands, usually not in the owner’s interests. Zombied computer can do such things as:

- Spy on a user's activities
- Steal data
- Mine cryptocurrencies
- Store and relay illegal content

Botnets are networks of zombied computers, controlled from a central server. Hundreds of thousands of computers infected with the same malware can be coordinated for all types of malicious activates. For example:

- **Distributed DoS (Denial-of-Service) Attacks:** Coordinated attacks on a target website using large numbers of computers to request pages all at once. This overwhelms the website's server, can crash it, or at least prevent it from serving content to legitimate users.
- **Click Fraud:** Simulating engagement with advertising, social media content, and search engines, to gain advertising revenue, bias results, or affect the profiles of users on various sites.
- **Cracking:** Botnets can bombard sites with authentication attempts in an effort to uncover passwords and credentials with a distributed brute-force attack.
- **Spamming:** Botnets can be rented to send masses of email out with malware for phishing attempts or advertising. This allows spammers to avoid detection, and reduces their bandwidth costs since the “owners” of the zombies pay for their own bandwidth.

How do these Animals get into a System?

Phishing: An attack aimed at getting a user to go to a fraudulent website and provide useful information. This might be done using an email seeming to be from his bank or employer asking him to log in, and provide an update or other information, getting the user to connect to a hacked or captured DNS server that causes a legitimate URL to be directed to another server, or providing links in webpages that go to the wrong place.

Spear Phishing: A special case of phishing where an acquaintance's social network or email account has been taken over. The target receives an email that is apparently from a friend asking him to join a network, or log in to look at new content.

Backdoors: This is an “entrance” to a computer system, cryptosystem, or application, that bypasses normal authentication. A backdoor may take the form of an installed program (Back Orifice, for example), a boot-sector virus, or a code intentionally written into closed source software.

Hackers: A hacker is someone who finds and exploits weaknesses in a computer system or computer network. Hackers may be motivated by profit, protest, challenge, or enjoyment. Recently, the term hacker is being reclaimed by computer programmers who argue that someone who breaks into computers, whether a computer criminal (black hats), or a computer security expert (white hats), is more appropriately called a **Cracker.** A hacker, in contrast, is simply a programmer who studies computer security. Black-hat hackers or crackers are the ones responsible for writing and distributing malware.

Stupid User Tricks: Users do all kinds of dangerous things, from opening attachments that have macro viruses, to installing programs that seem interesting, but are closed source, and unverified. Users turn off antivirus software, do not scan for malware, do not install security updates, use simple or default passwords, use closed source OS's and software, and attach their Wi-Fi to unknown ,and insecure wireless networks. **Social Engineering,** in which users are coaxed into giving access or revealing passwords, also falls into this category. Sometimes, people are their own worst enemies.

Section 14.2. Privacy Under the Law

There are many laws covering rights to digital privacy in the US, and many more worldwide. Catching up and keeping up with current technology is challenging. This section covers some of the basics.

The Sixth Amendment in the US Bill of Rights states the following:

> *The right of the people to be secure in their persons, houses, papers, and effects, against unreasonable searches and seizures, shall not be violated, and no Warrants shall issue, but upon probable cause, supported by Oath or affirmation, and particularly describing the place to be searched, and the persons or things to be seized.*

How exactly electronic records fit into this schema is not immediately clear. Is an email, or an electronic document, a “paper”? Certainly, email messages serve the same purpose as paper letters did when the bill of rights was passed, but in a literal sense, they are not. Suppose a document is stored on a cloud service. Does this mean it is not within a person's house? It is only used and accessed there, but does this make it part of a person's “effects”? A warrant is needed to open a first-class mail sent within the US. Should similar protections be extended to email? Several more recent laws and acts have addressed this.

The **Privacy Act** of 1974 sets a standard for data protection and disclosure by federal agencies. It states:

> *No agency shall disclose any record which is contained in a system of records by any means of communication to any person, or to another agency, except pursuant to a written request by, or with the prior written consent of, the individual to whom the record pertains.*

There are a number of exceptions to allowed:

- Routine uses within a U.S. government agency
- A valid **FOIA (Freedom Of Information Act)** request
- Disclosure to the National Archives and Records Administration for historical purposes
- Requests from law enforcement agencies and courts
- Requests from Congress
- Administrative purposes

The **ECPA (Electronic Communications Privacy Act)** of 1986 was meant to bring laws covering such things as wiretaps up-to-date. Several important amendments have been added including the **USA PATRIOT Act (Uniting and Strengthening America by Providing Appropriate Tools Required to Intercept and Obstruct Terrorists)** of 2001 and the **FISA Amendments Act** (**Foreign Intelligence Surveillance Act of 1978 Amendments Act)** of 2008.

The ECPA covers transmissions of all sorts using electromagnetic, wired, or optical methods. It only applies to communications that affect interstate or foreign commerce, but this does not restrict the scope very much in practice. There are several other exclusions, including tracking devices, electronic funds transfers, and cellphone location data, which are of more concern.

The government is required to get warrants or subpoenas to intercept communications transmissions, and to access stored data. These protections are considered weak in practice for several reasons. For example, emails stored for more than 180 days are considered “abandoned” and have less protection from search than newer emails. Agencies are allowed to seize and examine physical computers under weaker conditions than would apply to an electronic search of the data in many cases. Perhaps even more concerning is that various Internet, social network, and telecom companies, routinely give the government broad access to data and communications belonging to their users without the necessary warrants or subpoenas.

Other important data and US privacy laws include

- **FCRA (Fair Credit Reporting Act)** of 1970 as amended in 2002 allows people to view, correct, contest, and limit the uses of credit reports.
- **FDCPA (Fair Debt Collection Practices Act)** of 1977 as amended in 2006 forbids creditors or their agents from telling any third party about an individual's debt, calling a debtor at work, and certain other kinds of harassment.
- **HIPAA (Health Insurance Portability and Accountability Act)** of 1996 as amended in 2013 had two original objectives. The first was to protect health insurance coverage for workers and their families when they change or lose their jobs. HIPAA therefore extended and clarified the rights of workers to extend the insurance coverage under **COBRA (Consolidated Omnibus Budget Reconciliation Act)**, passed in 1985. The second was to

establish standardized mechanisms for **EDI (Electronic Data Interchange)** security, and confidentiality of all healthcare-related data.

- **FERPA (Family Educational Rights and Privacy Act)** of 1974 as amended in 2012 gives parents access to, and control over, disclosure of their minor child's education records, When a person turns 18, parents lose all these rights which then devolves on the student.

There are many other laws and acts at both the federal and state level that define the privacy rights of citizens. One characteristic of the US approach is that it tends to be enacted on a sector by sector basis (telecommunications, education, employment, health, etc.) instead of as an overarching law. The US is also focused on personal privacy more than data protection.

The **EU (European Union)** regulates in a more holistic way and focuses on data security more than personal privacy. The **DPA (Data Protection Act)** of 1998 is the main EU law that addresses the issue.

The DPA includes the provision that "Personal data shall not be transferred to countries with inadequate level of protections." Since the US is one of those countries with weaker data protection laws, this provision means that a US- based user could not have Facebook friends in France beause of the data exchanges that this would require. Also, Google, Amazon, and any other companies operating in Europe would have to maintain their data in places that meet the standards of the DPA.

To get around this, the US and the EU set up a **Safe Harbor** certification framework. Companies voluntarily agree to go beyond what US law requires and to meet the EU standard. If the certification is given, then data can be freely moved by the company even to non-compliant countries. This has been criticized as being something of a fig leaf. Although certified companies themselves may abide by these standards, they still must comply with orders issued by local courts and other authorities which are not bound by DPA or the Safe Harbor agreements. The safe harbor agreement was in place from 2000-2015 when the European Court of Justice invalidated the agreement.

In 2014, the European Court of Justice went even further. It found that there is a fundamental **Right to be Forgotten** under certain circumstances. It ordered Google to stop providing search results that link to personal information that is "inaccurate, inadequate, or no longer relevant".

The right to be forgotten exposes the fundamental tension between privacy and freedom of speech. It allows people to request the removal of certain information on the grounds that it is embarrassing, personal, incorrect, or irrelevant. It also puts companies like Google in the very difficult position of having to decide if these requests for removal are valid, and hundreds of thousands of such requests have been made.

This ruling has obvious practical problems as well. The Internet never forgets, so even a diligent effort to remove all search links does not remove the underlying data, or prevent the data from being republished in a different form and linked to. In addition, the EU does not have the authority to regulate what users outside its jurisdiction should be able to access. If we went down this path, any country could forbid worldwide access to content it deemed illegal. We certainly do not want North Korean or Iran dictating what can be seen on the Internet.

The EU therefore only requires that Google remove links from search results within the EU. Thus, searching a term on *Google.com* may give more and different results than searching the same term on *Google.fr* or *Google.uk.* Any EU user who wishes to see all the results simply has to go to **Google.com,** or perhaps use a US proxy server, if this is blocked, to find complete information. Although most users go to their local version of Google, this makes the right to be forgotten a fairly weak protection.

More recently, the European Parliament enacted the **General Data Protection Regulation (GDPR).** As of May 2018, any company that held data on European citizens was required to abide by very strict privacy, anonymity, data usage, and protection, regulations regardless of where the company is legally incorporated.

For example, if a Belgian happened contact a server located in the USA to order a laptop from Newegg.com, New Egg is obliged to take special care to recognize that the customer is a European, and comply with the GDPR. Failure to do so could result in fines of up to €20 million or 4% of the annual worldwide turnover IN the preceding financial year, which ever is greater.

The main result of the GDPR is that users are forced to explicitly chose and affirm their data usage preferences at almost every site they go to. It is not clear that this added cost really changes anything.

Section 14.3. Privacy in Reality

Subsection 14.3.1. Public Records

The US and many democratic countries have a long tradition of open public records. Citizens can go to courthouses, city halls, and federal agencies and get access to all kinds of information. For example, one can find out:

- Whether your doctor has a medical license and if it is current
- Whether a potential business partner has ever been sued
- Whether a potential renter has ever declared bankruptcy or has a criminal record
- Whether the property you are thinking about buying has a tax lien
- Whether your neighbor's house is being taxed at the same rate as yours
- What is said in government committee meetings and hearings
- What is said in the reports of various committees and agencies

- What the details are of government budgets and accounts
- How agencies responsible for policing, fire production, public health, education, road construction and repair, and so on, distribute their services to different neighborhoods and treat different types of citizens.

One can see why having this kind of information might be important. Citizens often have a reasonable need to know the legal status of someone or something before taking an action. One would not want to hire a nanny who was a sex offender, for example. Citizens should also be able to verify that the government is treating people fairly, everyone is paying their share, spending is being distributed equitably, the police are treating all citizens in the same way. Finally, citizens should be able to see how government works, how it makes its decisions, what those decisions are, and how they are carried out.

Robert Heinlein summed this up nicely when he wrote:

Secrecy is the beginning of tyranny.

It is vital that governments behave with transparency, so that any misbehavior is exposed, and extinguished.

Traditionally, examining government records required going to a government office in person and making a request. This took time and effort. Not many people did so unless the information they wanted was important to them. A journalist might be motivated to dig into the records, and would be able to widely broadcast any discovery of government corruption.

Few ordinary citizens would go digging for dirt on their neighbors, however, and even if something private and personal were to be discovered, it would be hard to publicize. Thus, the difficulty of accessing public records before the Internet meant that the loses to personal privacy that resulted were not very significant in practice.

Now, many courts and other government agencies are putting all kinds of records on the Internet. This makes them easily available, and searchable. Private information brokers also purchase access to government records, index, and integrate them. This makes it is possible search for outstanding warrants in many states at once, and even to buy an electronic biography of another person without leaving your desk.

Why should we worry about this? At first glance, this would only seem to matter for people who have something they wanted to hide. In fact, there are several reasons that ordinary citizens should be concerned as well:

Too Much Disclosure: Court records often include addresses of witnesses and victims, bank account and credit card numbers, social security numbers, false accusations made by angry business partners, or bitter ex-spouses, the names and ages of one's children, employment and income information, medical and mental health information, facts about sexual orientation or one's political views, and other details which may simply be private or embarrassing.

Inaccuracy: The records may be false. Information brokers may have mistakenly connected someone else's records to you. Records may have been poorly kept by the government agency that provided them. Records may not have been accurately updated or property expunged. When records were decentralized and difficult to access, this did not create much of a problem. Now that they are readily available, the lack of a mechanism or any legal requirement for those who distribute the records to correct them is of significant concern.

Serious but Untraceable Effects: The data in these records, accurate or not, can have seriously negative consequences. Background checks based on public records can result in credit being denied, jobs not being offered, apartments not being rented, admission not being granted, and so on. In these cases, the affected party has no legal right to know the reason for the negative outcome, nor to correct the record.

Long Memory: Public records can go back many decades. Unfortunate actions such as DUIs or bankruptcies in the distant past may disqualify a person from something by policy. A DUI in your teen years may prevent you from getting a job as a truck driver in your forties. There is no regulation to prevent this, and no statute of limitations. There are concerns that this could create a subclass of people who made a possibly serious error at some point in their lives, from which they will never be allowed to recover.

Subsection 14.3.2. Email

Under the ECPA, email messages stored on a server are no longer protected communications after 180 days. They become an ordinary database record, and government agencies can obtain access with a subpoena rather than a warrant. In any event, it seems that the NSA can read most email any way, and often does.

As a matter of fact, we make it very easy for government agencies, commercial rivals, hackers, and criminals, to access our email. When an email is sent, it goes through many servers and routers on the way to its destination. Recall that Internet protocols typically used for email, such as SSL and TLS, generate a secure tunnel between two nodes. This allows the nodes to send plaintext back and forth, which is encrypted as it enters the tunnel, and decrypted as it emerges.

What this means in practice is that every router in the chain that sends your email on to the next link sees the plaintext as it emerges from the tunnel. It may or may not establish a secure SSL/TLS tunnel to the next backbone router. The sender cannot choose the route a message takes, and cannot tell if the email was ever transmitted in the clear. In any event, any router on the path can store a plaintext copy of your message, including your own ISP's router,

The use of SSL/TLS also makes email subject to a kind of man-in-the-middle attack. Not only can any router the passes the message on read it, store it, and send copies to anyone it chooses, but it can also alter it. At least in theory, there is no assurance to that the message that is received is the same as the one that was sent. In practice, however, there are so many copies of any email sent scattered throughout the web, that any doubt could probably be resolved by seeing if the message is the same at all points on the route it took to its destination.

Email is subject to discovery in civil suits, and publicly traded companies and government agencies are required to keep complete searchable records of any emails entering or leaving their servers in case of legal action.

Many people are unaware that the law allows employers to access and read any email using a company account. Employees (including government employees) are generally required by policy to use their employer's email account for official purposes only. A company or agency could be held liable if an employee releases data in violation of one of the federal data privacy acts, might be damaged if embarrassing or secret information is emailed by an employee, and in any event, would prefer that their bandwidth, and the services of their computers, not be wasted or appropriated by employees for personal use.

You might wonder why email is not sent via the same kind of secure tunnel used in the HTTPS protocol, or why SSH is not used. The reason is that when you contact a web server, you have a clear target with an IP address that is likely to have an SSL certificate in the PKI.

In contrast, email addresses lead to recipients who live behind an SMTP server, have no IP address, and may or may not have identifiable SSL certificates. If the email recipient does happen to have an SSL certificate that can credibly provide the sender his public key, then standard encryption methods can be used, but otherwise, not. Email clients that facilitate encrypted email exist, but using them in practice is cumbersome.

Even if you managed to establish a secure encrypted link to the receiver, and you both maintain the message in an encrypted form at rest on your own server and computer, email is still extremely insecure. The recipient may accidentally or intentionally forward your email so someone without your permission. The email might be modified to make it appear you said something you did not. The email may be subpoenaed by a court. Finally, email may be legally read by an employer if it went to or from a company account.

In short, when you send email, you have no reason to think that your message did not transit the Internet in plaintext at some point and was intercepted, stored and read by anyone from the NSA to your stalker from high school. You should expect that copies are stored in many places unknown to you and which you cannot erase. Once an email leaves your computer, you should think of it as permanently available public information.

Subsection 14.3.3. Browsers, Cookies, Tracking, and Search

Browsers are natively **Stateless** in that each request for a webpage is an independent transaction that is not affected by any previous request. As a result, your browser has no way of proving that it has previously provided login credentials when requesting a new password-protected page. It also has no way of transferring information between different pages on a website. For example, the contents of a shopping cart, or the text that a user writes into data fields, cannot be transferred when the user moves to a new page.

The solution is to allow HTTP servers to send a small piece of data called a **Cookie** to the client. You can receive cookies from any site you visit, and your browser sends back any cookies placed there by a site whenever you request content from the server of origin. This creates a kind of **Statefulness**. The data in the cookie can inform the server that the client requesting the page has successfully logged in, allows the server to look up the things the client previous put into a shopping cart, or information he wrote into data fields on a previous page.

There are many benign and useful things that cookies facilitate, but there are also some less desirable ones. While cookies cannot install viruses or malware, they can be used as a kind spyware.

When you visit a website, the content on page usually comes from a collection of HTTP servers. This allows webpages to aggregate and present content such as weather reports, news, and audio or video content from other platforms. Much of this external content, however, is advertising from a small set of companies such as Google, Adblade, Yahoo!, and Microsoft. When you request a page from some site like foobar.com (the first party), you (the second party) are also requesting the content from advertising companies (the third parties) that fill in the various banners, headers, and slots ,where ads appear.

In effect, you visit these advertising sites all the time through different webpages, whether you want to or not. Each time you visit a site that serves ads from Google, for example, Google is allowed to put a new cookie in your jar with information about the time, date, URL, your IP address, a fingerprint of your computer, and so. It is also allowed to rifle through your cookie jar and grab copies of any cookies it placed there previously. These **Third-party Cookies** allow Google to build a details record of your browsing activities.

This is a good reason to use browser plugins such as Adblock Plus to prevent known advertising servers playing “whose got the cookie” with you. One of the best things you can do to increase your privacy is to not accept third party cookies by default, and to clear your cookies automatically when you close your browser (you will find setting in your browser's preferences that allow you to do this).

Subsection 14.3.4. Smartphones, Apps, and Platforms

Smartphones are the most invasive, privacy destroying, spy devices ever conceived. Consider the following:

Payment apps (Google Pay, PayPal, Apple Pay, Venmo, etc.) log your purchases. They know how, when, and where, you spend your money. Do you have a big social life, or are you a loner. Do you buy a lot of alcohol, got to concerts, donate money to certain organizations, live month to month, and so on. Give some thought to what sort of picture of who you are as a person can be gleaned from a full look at all your purchases and other financial interactions.

Backup Services prevent data loss, they also allow the provider access to your files, pictures, texts, contacts, social media posts, and more. When your data goes to the cloud for backup, it is read and analyzed by the providers in almost all cases. They may promise to do this only with “anonymized” data, but then you must depend on the integrity and competence of the provider. In any event, if a government agency asks to see the data, platforms usually comply. Again, what would a machine learn about you from analyzing every communication you received or sent, and every piece of content you produced or accessed?

Microsoft 365, Google Teams, Google Docs, iWork, and other cloud based productivity software, include backup and sharing by default. This allows not only the privacy breaches outlined above, but also a more integrated look at the context and connections surrounding your content and communications.

Speech-to-text, text-to-speech, grammar assistants, translation services, smart assistants (I’m looking at you Alexa), Swype keyboards, predictive search bar completions, and almost all applications that require natural language interpretation, can’t work within the confines of the processing power and storage available on user devices. They must send your input to the cloud for analysis using large language models and other AI driven approaches. In other words, if you use any of these seemingly benign utilities, you are agreeing to allow Google to capture all the corresponding data.

Every time you Swype
Every text you type
Every use of mikes
Everything you like
They’ll be watching you

Subsection 14.3.5. GPS and Location Services

Many applications, especially social networking and media sites, but also browsers, search engines, and hardware devices, report real time locations to servers somewhere in the cloud. The leading technologies that allow this are the following:

GPS (Global Positioning System): This uses a set of low-earth orbit satellites that broadcast a signal which allows a GPS receiver to triangulate its location. These devices can determine not only your longitude and latitude, but also your altitude (are you on a plane or the 27th floor of a building?). GPS receivers are inexpensive and are found in most cell phones, tablets, laptops, and wearable technology. Although GPS receivers themselves do not transmit your location, they are often attached to connected devices that do.

Wi-Fi Location: Wi-Fi networks broadcast **SSID (Service Set IDentifier)** and **MAC (Media Access Control)** data. The strength of the signal coming from the Wi-Fi routers to your phone or laptop is able to detect at any given moment can be used to estimate your location if the locations of the routers being detected are known. Google and other companies can find these router locations with the help of users and their GPS receivers. As millions of people go about their daily business, the SSID, MAC address, and signal strength of visible Wi-Fi networks are noted by your device if you have activated location services (sometimes, even if you have not). This is tied to your current GPS determined location and then sent to a server, From this, the location of the Wi-Fi routers can be calculated.

Cellphone Location: This works in a way similar to GPS and Wi-Fi. The difference is that the tower locations are known to your service provider already. Measuring the signal strength you receive from several of these known locations allows the cell phone company to get a fairly accurate estimate of your location. This data is stored by providers, and made available to law enforcement. It can even be subpoenaed in civil cases in some circumstances.

CC Cameras: Closed circuit cameras are becoming more common every day. They are installed on public streets, highways, at key intersections, in airports, and in public buildings. The images they collect can be combined in real time with automatic facial recognition systems or license plate readers. If the network is dense enough, this can allow authorities to track where you drive, how long it takes you to get there, who you are with, where you walk, and where you are now.

People traditionally have had an expectation of anonymity in public spaces as they go about their lawful occasions. The cost of surveilling and tracking even a single individual without technology is extremely high (three full time people at least to cover a 24 watch). Now, most humans obliging carry devices that emit radio signals with identifying information, and can be visually identified and tracked by networks of inexpensive sensors. People have never had a legal expectation of privacy in public spaces, and new technology exploits this to the make any expectation of anonymity a thing of the past.

Subsection 14.3.6. Identity Theft

Identity Theft is a term that suggests taking over another person's identity. A more accurate term might be impersonation, or identity fraud, since the victim ends up sharing his identity with a criminal rather than losing it. Such frauds go back to the beginning of history, but modern ICT has substantially changed their flavor and scope.

Large scale data breaches involving millions of medical histories, credit card and social security numbers, and so on, are a common occurrence. This creates the potential for identity thieves to do such things as apply for loans and credit cards, make charges on existing credit card accounts, and even charge medical services to insurance in another person's name. Of course the problem is that victims of identity theft end up being held responsible for such actions unless they can prove that they were not involved.

The motivations to pretend to be another person extend beyond engaging in various types of financial fraud. For example, medical fraud not only permits the identity thief to get free medical care, but may also allow him access to drugs. Of even greater concern is that an identity thief may claim to be another person when he is arrested, or even convicted of a crime. In the most extreme cases, identity thieves may pretend to be another person in everyday life, perhaps to escape something in his past.

Victims of identity theft may not notice that anything is wrong until it is too late. Money may be transferred out of accounts for weeks before the account is checked. Doctors may give patients inappropriate treatments if their medical records contain things that relate to another person, and may even suspect that the victim is a drug seeker. Months or years can pass between doctor's visits. A victim might be called for jury duty, try to renew his driver's license, or be pulled over, only to be arrested on outstanding warrants attributable to the identity thief.

Background checks when applying for jobs may turn up this criminal record or various financial improprieties and can result is a victim not being offered employment. The victim may not even be told the reason he was turned down. Identity thieves find that impersonating recently deceased people to be especially profitable. The person in charge of clearing up the estate is often an amateur, and unfamiliar with the financial details of the departed. He may not notice anything is wrong for a very long time.

Surprisingly, children have their identities stolen as very high rates. This is largely because they have mostly blank histories. This is ideal for someone who needs a social security number to get a job, an illegal alien, or someone with a criminal history, for example. As a bonus, neither children nor their parents are likely to be on the lookout for signs of identity theft. Children do not apply for credit cards, try to find jobs, or get pulled over for speeding, as a rule of thumb.

Even if victim does notice a problem, fixing it is quite difficult. First, how do you prove that you are who you say you are? The reality is that most electronic systems identify people through the numbers they provide (credit cards, banks accounts and SSNs) or through passwords and security questions.

Identity thieves can know all of these things. Who is to say that the person asking to have a credit card account canceled without paying off the balance is a victim of identity theft, and not simply a freeloader? Similarly, a court is unlikely to cancel a warrant, or expunge a record, simply because someone calls up and claims that some other guy pretending to be him did the crime.

Even if the victim succeeds in tracking down and fixing all the threads of damage inflicted by the thief, and somehow locks everything down so no new damage is done, he is still not off the hook. Local courts, medical providers, banks, and credit card companies may be slow or inaccurate in fixing or expunging their records. They may even forget to follow through completely. Even if all this goes well, Data brokers may have purchased records before they were fixed and may never update them. Copies of the erroneous records may exist in other places as yet undiscovered. Victims of identity theft typically spend hundreds of hours and thousands of dollars in the attempt to get their lives back to normal.

When we lived is smaller groups, and travel and communication was expensive and difficult, people knew one another. Few really had to worry about identity theft. A person's name and reputation were valued and protected. Now, a lot of who we are to other people, and the quality of our reputation, depend on the electronic records associated with us.

Behaving in an honest, generous, and upright way, is no longer a guaranteed path to a good name. You need to be clever and careful as well. Certainly, never post social security or financial data in any public place. It is risky even posting information about your first dog, girl or boyfriend, or the name of your mother. Shred or otherwise destroy mail containing any personal data and make sure that any electronic files holding this information are secure and encrypted. Use good passwords, and do not fall for phishing or social hacking scams. No, you don't have an uncle in Nigeria, and no, the Irish Lottery Agency does not need your credit card number in order to FedEx the check for your winnings.

Never **give information in response to emails or phone calls!**

Always look up the number or address yourself, and initiate contact from you side. There is no other way of knowing who you are dealing with. Destroy any hard drives, USB keys, and other storage media, before throwing away electronic equipment. You may still fall victim to identity theft due to someone else's carelessness, but at least you can try not to contribute to the problem.