# Security Engineering
# Third Edition

Published by John Wiley & Sons, Inc., Indianapolis, Indiana
Published simultaneously in Canada

ISBN: 978-1-119-64278-7
ISBN: 978-1-119-64283-1 (ebk)
ISBN: 978-1-119-64281-7 (ebk)

Manufactured in the United States of America

**Limit of Liability/Disclaimer of Warranty:** The publisher and the author make no representations or warranties with respect to the accuracy or completeness of the contents of this work and specifically disclaim all warranties, including without limitation warranties of fitness for a particular purpose. No warranty may be created or extended by sales or promotional materials. The advice and strategies contained herein may not be suitable for every situation. This work is sold with the understanding that the publisher is not engaged in rendering legal, accounting, or other professional services. If professional assistance is required, the services of a competent professional person should be sought. Neither the publisher nor the author shall be liable for damages arising herefrom. The fact that an organization or Web site is referred to in this work as a citation and/or a potential source of further information does not mean that the author or the publisher endorses the information the organization or website may provide or recommendations it may make. Further, readers should be aware that Internet websites listed in this work may have changed or disappeared between when this work was written and when it is read.

For general information on our other products and services please contact our Customer Care Department within the United States at (877) 762-2974, outside the United States at (317) 572-3993 or fax (317) 572-4002.

Wiley publishes in a variety of print and electronic formats and by print-on-demand. Some material included with standard print versions of this book may not be included in e-books or in print-on-demand. If this book refers to media such as a CD or DVD that is not included in the version you purchased, you may download this material at `booksupport.wiley.com`. For more information about Wiley products, visit `www.wiley.com`.

**Library of Congress Control Number:** 2020948679

# Contents at a Glance

Part II

Part III

# Contents

# Preface to the Third Edition

The first edition of *Security Engineering* was published in 2001 and the second in 2008. Since then there have been huge changes.

The most obvious is that the smartphone has displaced the PC and laptop. Most of the world's population now walk around with a computer that's also a phone, a camera and a satnav; and the apps that run on these magic devices have displaced many of the things we were building ten years ago. Taxi rides are now charged by ride-hailing apps rather than by taxi meters. Banking has largely gone online, with phones starting to displace credit cards. Energy saving is no longer about your meter talking to your heating system but about both talking to your phone. Social networking has taken over many people's lives, driving everything from advertising to politics.

A related but less visible change is the move to large server farms. Sensitive data have moved from servers in schools, doctors' offices and law firms to cloud service providers. Many people no longer do their writing on word processing software on their laptop but on Google Docs or Office365 (I'm writing this book on Overleaf). This has consequences. Security breaches can happen at a scale no-one would have imagined twenty years ago. Compromises of tens of millions of passwords, or credit cards, have become almost routine. And in 2013, we discovered that fifteen years' worth of UK hospital medical records had been sold to 1200 organisations worldwide without the consent of the patients (who were still identifable via their postcodes and dates of birth).

A real game-changer of the last decade was the Snowden revelations, also in 2013, when over 50,000 Top Secret documents about the NSA's signals intelligence activities were leaked to the press. The scale and intrusiveness of government surveillance surprised even cynical security engineers. It followed on from Stuxnet, where America attacked Iran's nuclear weapons program using malware, and was followed by NotPetya, where a Russian cyberweapon, deployed against the Ukraine, inflicted hundreds of millions of dollars' worth of collateral damage on firms elsewhere. This brings us to the third big change, which is a much better understanding of nation-state security threats. In addition to understanding the capabilities and priorities of western intelligence agencies, we have a reasonably good idea of what the Chinese, the Russians and even the Syrians get up to.

And where the money is, the crooks follow too. The last decade has also seen the emergence of a cyber-crime ecosystem, with malware writers providing the tools to subvert millions of machines, many of which are used as criminal infrastructure while others are subverted in various ways into defrauding their users. We have a team at Cambridge that studies this, and so do dozens of other research groups worldwide. The rise of cybercrime is changing policing, and other state activity too: cryptocurrencies are not just making it easier to write ransomware, but undermining financial regulation. And then there are non-financial threats from cyber-bullying up through hate speech to election manipulation and videos of rape and murder.

So online harms now engage all sorts of people from teachers and the police to banks and the military. It is ever more important to measure the costs of these harms, and the effectiveness of the measures we deploy to mitigate them.

Some of the changes would have really surprised someone who read my book ten years ago and then spent a decade in solitary confinement. For example, the multilevel security industry is moribund, despite being the beneficiary of billions of dollars of US government funding over forty years; the Pentagon's entire information security philosophy – of mandating architectures to stop information flowing downward from Top Secret to Secret to Confidential to Unclassified – has been abandoned as unworkable. While architecture still matters, the emphasis has shifted to ecosystems. Given that bugs are ubiquitous and exploits inevitable, we had better be good at detecting exploits, fixing bugs and recovering from attacks. The game is no longer trusted systems but coordinated disclosure, DevSecOps and resilience.

What might the future hold? A likely game-changer is that as we put software into safety-critical systems like cars and medical devices, and connect them to the Internet, safety and security engineering are converging. This is leading to real strains; while security engineers fix bugs quickly, safety engineers like to test systems rigorously against standards that change slowly if at all. A wicked problem is how we will patch durable goods. At present, you might get security patches for your phone for three years and your laptop for five; you're expected to buy a new one after that. But cars last for fifteen years on average and if we're suddenly asked to scrap them after five the environmental costs won't be acceptable. So tell me, if you're writing navigation software today in 2020 for a car that will launch in 2023, how will you ensure that you can keep on shipping security patches in 2033, 2043 and 2053? What tools will you choose today?

Finally, there has been a sea change in the political environment. After decades in which political leaders considered technology policy to be for men in anoraks, and generally took the line of least resistance, the reports of Russian interference in the Brexit referendum and the Trump election got their attention. The prospect of losing your job can concentrate the mind wonderfully. The close attention of lawmakers is changing the game, first with tighter general rules such as Europe's General Data Protection Regulation; and second as products that are already regulated for safety, from cars and railway signals to children's toys acquire software and online connectivity, which has led to rules in Europe about how long software has to be maintained.

The questions the security engineer has to ask today are just the same as a decade ago: what are we seeking to prevent, and will the proposed mechanisms actually work? However, the canvas on which we work is now much broader. Almost all human life is there.

Ross Anderson
Cambridge, October 2020

safety in applications from cars through utilities to electronic healthcare. The security engineer needs to understand not just crypto and operating systems, but economics and human factors as well.

And the ubiquity of digital devices means that ‘computer security’ is no longer just a problem for a few systems specialists. Almost all white-collar crime (and much crime of the serious violent sort) now involves computers or mobile phones, so a detective needs to understand computer forensics just as she needs to know how to drive. More and more lawyers, accountants, managers and other people with no formal engineering training are going to have to understand system security in order to do their jobs well.

The rapid growth of online services, from Google and Facebook to massively multiplayer games, has also changed the world. Bugs in online applications can be fixed rapidly once they’re noticed, but the applications get ever more complex and their side-effects harder to predict. We may have a reasonably good idea what it means for an operating system or even a banking service to be secure, but we can’t make any such claims for online lifestyles that evolve all the time. We’re entering a novel world of evolving socio-technical systems, and that raises profound questions about how the evolution is driven and who is in control.

The largest changes, however, may be those driven by the tragic events of September 2001 and by our reaction to them. These have altered perceptions and priorities in many ways, and changed the shape of the security industry. Terrorism is not just about risk, but about the perception of risk, and about the manipulation of perception. This adds psychology and politics to the mix. Security engineers also have a duty to contribute to the political debate. Where inappropriate reactions to terrorist crimes have led to major waste of resources and unforced policy errors, we have to keep on educating people to ask a few simple questions: what are we seeking to prevent, and will the proposed mechanisms actually work?

Ross Anderson
Cambridge, January 2008

# Preface to the First Edition

For generations, people have defined and protected their property and their privacy using locks, fences, signatures, seals, account books, and meters. These have been supported by a host of social constructs ranging from international treaties through national laws to manners and customs.

This is changing, and quickly. Most records are now electronic, from bank accounts to registers of real property; and transactions are increasingly electronic, as shopping moves to the Internet. Just as important, but less obvious, are the many everyday systems that have been quietly automated. Burglar alarms no longer wake up the neighborhood, but send silent messages to the police; students no longer fill their dormitory washers and dryers with coins, but credit them using a smartcard they recharge at the college bookstore; locks are no longer simple mechanical affairs, but are operated by electronic remote controls or swipe cards; and instead of renting videocassettes, millions of people get their movies from satellite or cable channels. Even the humble banknote is no longer just ink on paper, but may contain digital watermarks that enable many forgeries to be detected by machine.

How good is all this new security technology? Unfortunately, the honest answer is 'nowhere near as good as it should be.' New systems are often rapidly broken, and the same elementary mistakes are repeated in one application after another. It often takes four or five attempts to get a security design right, and that is far too many.

The media regularly report security breaches on the Internet; banks fight their customers over 'phantom withdrawals' from cash machines; VISA reports huge increases in the number of disputed Internet credit card transactions; satellite TV companies hound pirates who copy their smartcards; and law enforcement agencies try to stake out territory in cyberspace with laws controlling the use of encryption. Worse still, features interact. A mobile phone that calls the last number again if one of the keys is pressed by accident may be just a minor nuisance – until someone invents a machine that dispenses a can of soft drink every time its phone number is called. When all of a sudden you find 50 cans of Coke on your phone bill, who is responsible, the phone company, the handset manufacturer, or the vending machine operator? Once almost every electronic device that affects your life is connected to the Internet – which Microsoft expects to happen by 2010 – what does 'Internet security' mean to you, and how do you cope with it?

As well as the systems that fail, many systems just don't work well enough. Medical record systems don't let doctors share personal health information as they would like, but still don't protect it against inquisitive private eyes. Zillion-dollar military systems prevent anyone without a "top secret" clearance from getting at intelligence data, but are often designed so that almost everyone needs this clearance to do any work. Passenger ticket systems are designed to prevent customers cheating, but when trustbusters break up the railroad, they cannot stop the new rail companies cheating each other. Many of these failures could have been foreseen if designers had just a little bit more knowledge of what had been tried, and had failed, elsewhere.

Security engineering is the new discipline that is starting to emerge out of all this chaos.

Although most of the underlying technologies (cryptology, software reliability, tamper resistance, security printing, auditing, etc.) are relatively well understood, the knowledge and experience of how to apply them effectively is much scarcer. And since the move from mechanical to digital mechanisms is happening everywhere at once, there just has not been time for the lessons learned to percolate through the engineering community. Time and again, we see the same old square wheels being reinvented.

The industries that have managed the transition most capably are often those that have been able to borrow an appropriate technology from another discipline. Examples include the reuse of technology designed for military identify-friend-or-foe equipment in bank cash machines and even prepayment gas meters. So even if a security designer has serious expertise in some particular speciality – whether as a mathematician working with ciphers or a chemist developing banknote inks – it is still prudent to have an overview of the whole subject. The essence of good security engineering is understanding the potential threats to a system, then applying an appropriate mix of protective measures – both technological and organizational – to control them. Knowing what has worked, and more importantly what has failed, in other applications is a great help in developing judgment. It can also save a lot of money.

The purpose of this book is to give a solid introduction to security engineering, as we understand it at the beginning of the twenty-first century. My goal is that it works at four different levels:

1. as a textbook that you can read from one end to the other over a few days as an introduction to the subject. The book is to be used mainly by the working IT professional who needs to learn about the subject, but it can also be used in a one-semester course in a university;
2. as a reference book to which you can come for an overview of the workings of some particular type of system (such as cash machines, taxi meters, radar jammers, anonymous medical record databases or whatever);
3. as an introduction to the underlying technologies, such as crypto, access control, inference control, tamper resistance, and seals. Space prevents me from going into great depth; but I provide a basic road map for each subject, plus a reading list for the curious (and a list of open research problems for the prospective graduate student);
4. as an original scientific contribution in which I have tried to draw out the common principles that underlie security engineering, and the lessons that people building one kind of system should have learned from others. In the many years I have been working in security, I keep coming across these. For example, a simple attack on stream ciphers wasn't known to the people who designed a common anti-aircraft fire control radar so it was easy to jam; while a trick well known to the radar community wasn't understood by banknote printers and people who design copyright marking schemes, which led to a quite general attack on most digital watermarks.

I have tried to keep this book resolutely mid-Atlantic. A security engineering book has to be, as many of the fundamental technologies are American, while many of the interesting applications are European. (This isn't surprising given the better funding of US universities and research labs, and the greater diversity of nations and markets in Europe.) What's more, many of the successful European innovations – from the smartcard to the GSM mobile phone to the pay-per-view TV service – have crossed the Atlantic and now thrive in the Americas. Both the science, and the case studies, are necessary.

This book grew out of the security engineering courses I teach at Cambridge University, but I have rewritten my notes to make them self-contained and added at least as much material again. It should be useful to the established professional security manager or consultant as a first-line reference; to the computer science professor doing research in cryptology; to the working police detective trying to figure out the latest computer scam; and to policy wonks struggling with the conflicts involved in regulating cryptography and anonymity. Above all, it is aimed at Dilbert. My main audience is the working programmer or engineer who is trying to design real systems that will keep on working despite the best efforts of customers, managers, and everybody else.

This book is divided into three parts.

- The first looks at basic concepts, starting with the central concept of a security protocol, and going on to the human-computer interface, access controls, cryptology and distributed system issues. It does not assume any particular technical background other than basic computer literacy. It is based on an 'Introduction to Security' course which we teach to second year undergraduates.
- The second part looks in much more detail at a number of important applications such as military communications, medical record systems, cash machines, mobile phones and pay-TV. These are used to introduce more of the advanced technologies and concepts. It also considers information security from the viewpoint of a number of different interest groups such as companies, consumers, criminals, the police and spies. This material is drawn from my senior course on security, from research work, and from experience consulting.
- The third part looks at the organizational and policy issues: how computer security interacts with law, with evidence, and with corporate politics; how we can gain confidence that a system will perform as intended; and how the whole business of security engineering can best be managed.

I believe that building systems which continue to perform robustly in the face of malice is one of the most important, interesting, and difficult tasks facing engineers in the twenty-first century.

Ross Anderson
Cambridge, January 2001

# Foreword

In a paper he wrote with Roger Needham, Ross Anderson coined the phrase 'programming Satan's computer' to describe the problems faced by computer-security engineers. It's the sort of evocative image I've come to expect from Ross, and a phrase I've used ever since.

Programming a computer is straightforward: keep hammering away at the problem until the computer does what it's supposed to do. Large application programs and operating systems are a lot more complicated, but the methodology is basically the same. Writing a reliable computer program is much harder, because the program needs to work even in the face of random errors and mistakes: Murphy's computer, if you will. Significant research has gone into reliable software design, and there are many mission-critical software applications that are designed to withstand Murphy's Law.

Writing a secure computer program is another matter entirely. Security involves making sure things work, not in the presence of random faults, but in the face of an intelligent and malicious adversary trying to ensure that things fail in the worst possible way at the worst possible time … again and again. It truly is programming Satan's computer.

Security engineering is different from any other kind of programming. It's a point I made over and over again: in my own book, Secrets and Lies, in my monthly newsletter Crypto-Gram, and in my other writings. And it's a point Ross makes in every chapter of this book. This is why, if you're doing any security engineering … if you're even thinking of doing any security engineering, you need to read this book. It's the first, and only, end-to-end modern security design and engineering book ever written.

And it comes just in time. You can divide the history of the Internet into three waves. The first wave centered around mainframes and terminals. Computers were expensive and rare. The second wave, from about 1992 until now, centered around personal computers, browsers, and large application programs. And the third, starting now, will see the connection of all sorts of devices that are currently in proprietary networks, standalone, and non-computerized. By 2003, there will be more mobile phones connected to the Internet than computers. Within a few years we'll see many of the world's refrigerators, heart monitors, bus and train ticket dispensers, burglar alarms, and electricity meters talking IP. Personal computers will be a minority player on the Internet.

Security engineering, especially in this third wave, requires you to think differently. You need to figure out not how something works, but how something can be made to not work. You have to imagine an intelligent and malicious adversary inside your system (remember Satan's computer), constantly trying new ways to subvert it. You have to consider all the ways your system can fail, most of them having nothing to do with the design itself. You have to look at everything backwards, upside down, and sideways. You have to think like an alien.

As the late great science fiction editor John W. Campbell, said: "An alien thinks as well as a human, but not like a human." Computer security is a lot like that. Ross is one of those rare people who can think like an alien, and then explain that thinking to humans. Have fun reading.

Bruce Schneier
January 2001

CHAPTER

1

# What Is Security Engineering?

*Out of the crooked timber of humanity, no straight thing was ever made.*

– IMMANUEL KANT

*The world is never going to be perfect, either on- or offline; so let's not set impossibly high standards for online.*

– ESTHER DYSON

## 1.1 Introduction

Security engineering is about building systems to remain dependable in the face of malice, error, or mischance. As a discipline, it focuses on the tools, processes, and methods needed to design, implement, and test complete systems, and to adapt existing systems as their environment evolves.

Security engineering requires cross-disciplinary expertise, ranging from cryptography and computer security through hardware tamper-resistance to a knowledge of economics, applied psychology, organisations and the law. System engineering skills, from business process analysis through software engineering to evaluation and testing, are also important; but they are not sufficient, as they deal only with error and mischance rather than malice. The security engineer also needs some skill at adversarial thinking, just like a chess player; you need to have studied lots of attacks that worked in the past, from their openings through their development to the outcomes.

Many systems have critical assurance requirements. Their failure may endanger human life and the environment (as with nuclear safety and control systems), do serious damage to major economic infrastructure (cash machines and online payment systems), endanger personal privacy (medical record systems), undermine the viability of whole business sectors (prepayment utility meters), and facilitate crime (burglar and car alarms). Security and safety are becoming ever more intertwined as we get software in everything.

Even the perception that a system is more vulnerable or less reliable than it really is can have real social costs.

The conventional view is that while software engineering is about ensuring that certain things happen ("John can read this file"), security is about ensuring that they don't ("The Chinese government can't read this file"). Reality is much more complex. Security requirements differ greatly from one system to another. You typically need some combination of user authentication, transaction integrity and accountability, fault-tolerance, message secrecy, and covertness. But many systems fail because their designers protect the wrong things, or protect the right things but in the wrong way.

Getting protection right thus depends on several different types of process. You have to figure out what needs protecting, and how to do it. You also need to ensure that the people who will guard the system and maintain it are properly motivated. In the next section, I'll set out a framework for thinking about this. Then, in order to illustrate the range of different things that security and safety systems have to do, I will take a quick look at four application areas: a bank, a military base, a hospital, and the home. Once we've given concrete examples of the stuff that security engineers have to understand and build, we will be in a position to attempt some definitions.

## 1.2 A framework

To build really dependable systems, you need four things to come together. There's policy: what you're supposed to achieve. There's mechanism: the ciphers, access controls, hardware tamper-resistance and other machinery that you use to implement the policy. There's assurance: the amount of reliance you can place on each particular mechanism, and how well they work together. Finally, there's incentive: the motive that the people guarding and maintaining the system have to do their job properly, and also the motive that the attackers have to try to defeat your policy. All of these interact (see Figure 1.1).

As an example, let's think of the 9/11 terrorist attacks. The hijackers' success in getting knives through airport security was not a mechanism failure but a policy one; the screeners did their job of keeping out guns and explosives, but at that time, knives with blades up to three inches were permitted. Policy changed quickly: first to prohibit all knives, then most weapons (baseball bats are now forbidden but whiskey bottles are OK); it's flip-flopped on many details (butane lighters forbidden then allowed again). Mechanism is weak, because of things like composite knives and explosives that don't contain nitrogen. Assurance is always poor; many tons of harmless passengers' possessions are consigned to the trash each month, while less than half of all the real weapons taken through screening (whether accidentally or for test purposes) are spotted and confiscated.

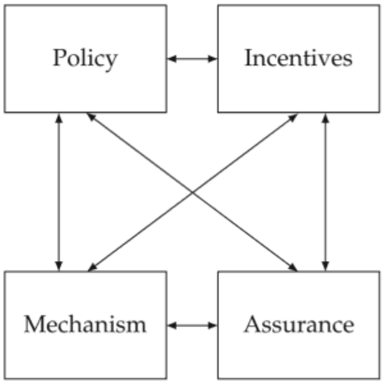


**Figure 1.1:** – Security Engineering Analysis Framework

Most governments have prioritised visible measures over effective ones. For example, the TSA has spent billions on passenger screening, which is fairly ineffective, while the $100m spent on reinforcing cockpit doors removed most of the risk [1526]. The President of the Airline Pilots Security Alliance noted that most ground staff aren't screened, and almost no care is taken to guard aircraft parked on the ground overnight. As most airliners don't have door locks, there's not much to stop a bad guy wheeling steps up to a plane and placing a bomb on board; if he had piloting skills and a bit of chutzpah, he could file a flight plan and make off with it [1204]. Yet screening staff and guarding planes are just not a priority.

Why are such policy choices made? Quite simply, the incentives on the decision makers favour visible controls over effective ones. The result is what Bruce Schneier calls 'security theatre' – measures designed to produce a feeling of security rather than the reality. Most players also have an incentive to exaggerate the threat from terrorism: politicians to 'scare up the vote' (as President Obama put it), journalists to sell more papers, companies to sell more equipment, government officials to build their empires, and security academics to get grants. The upshot is that most of the damage done by terrorists to democratic countries comes from the overreaction. Fortunately, electorates figure this out over time, and now – nineteen years after 9/11 – less money is wasted. Of course, we now know that much more of our society's resilience budget should have been spent on preparing for pandemic disease. It was at the top of Britain's risk register, but terrorism was politically more sexy. The countries that managed their priorities more rationally got much better outcomes.

Security engineers need to understand all this; we need to be able to put risks and threats in context, make realistic assessments of what might go wrong, and give our clients good advice. That depends on a wide understanding of what has gone wrong over time with various systems; what sort of attacks have worked, what their consequences were, and how they were stopped (if it was

2. Starting in the 1940s, governments spent a lot of money on electronic warfare systems. The arms race of trying to jam enemy radars while preventing the enemy from jamming yours has led to many sophisticated deception tricks, countermeasures, and counter-countermeasures – with a depth, subtlety and range of strategies that are still not found elsewhere. Spoofing and service-denial attacks were a reality there long before blackmailers started targeting the websites of bankers, bookmakers and gamers.
3. Military organisations need to hold some information close, such as intelligence sources and plans for future operations. These are typically labeled 'Top Secret' and handled on separate systems; they may be further restricted in compartments, so that the most sensitive information is known to only a handful of people. For years, attempts were made to enforce information flow rules, so you could copy a file from a *Secret* stores system to a *Top Secret* command system, but not vice versa. Managing multiple systems with information flow restrictions is a hard problem, and the billions that were spent on attempting to automate military security helped develop the access-control technology you now have in your mobile phone and laptop.
4. The problems of protecting nuclear weapons led to the invention of a lot of cool security technology, ranging from provably-secure authentication systems, through optical-fibre alarm sensors, to methods of identifying people using biometrics – including the iris patterns now used to identify all citizens of India.

The security engineer can still learn a lot from this. For example, the military was until recently one of the few customers for software systems that had to be maintained for decades. Now that software and Internet connectivity are finding their way into safety-critical consumer goods such as cars, software sustainability is becoming a much wider concern. In 2019, the European Union passed a law demanding that if you sell goods with digital components, you must maintain those components for two years, or for longer if that's a reasonable expectation of the customer – which will mean ten years for cars and white goods. If you're writing software for a car or fridge that will be on sale for seven years, you'll have to maintain it for almost twenty years. What tools should you use?

## 1.5 Example 3 – a hospital

From bankers and soldiers we move on to healthcare. Hospitals have a number of interesting protection requirements – mostly to do with patient safety and privacy.

1. Safety usability is important for medical equipment, and is by no means a solved problem. Safety usability failures are estimated to kill about as many people as road traffic accidents – a few tens of thousands a year in the USA, for example, and a few thousand in the UK. The biggest single problem is with the infusion pumps used to drip-feed patients with drugs; a typical hospital might have half-a-dozen makes, all with somewhat different controls, making fatal errors more likely. Safety usability interacts with security: unsafe devices that are also found to be hackable are much more likely to have product recalls ordered as regulators know that the public's appetite for risk is lower when hostile action becomes a possibility. So as more and more medical devices acquire not just software but radio communications, security sensitivities may lead to better safety.
2. Patient record systems should not let all the staff see every patient's record, or privacy violations can be expected. In fact, since the second edition of this book, the European Court has ruled that patients have a right to restrict their personal health information to the clinical staff involved in their care. That means that systems have to implement rules such as "nurses can see the records of any patient who has been cared for in their department at any time during the previous 90 days". This can be harder than it looks. (The US HIPAA legislation sets easier standards for compliance but is still a driver of information security investment.)
3. Patient records are often anonymized for use in research, but this is hard to do well. Simply encrypting patient names is not enough: an enquiry such as "show me all males born in 1953 who were treated for atrial fibrillation on October 19th 2003" should be enough to target former Prime Minister Tony Blair, who was rushed to hospital that day to be treated for an irregular heartbeat. Figuring out what data can be anonymized effectively is hard, and it's also a moving target as we get more and more social and contextual data – not to mention the genetic data of relatives near and far.
4. New technology can introduce poorly-understood risks. Hospital administrators understand the need for backup procedures to deal with outages of power; hospitals are supposed to be able to deal with casualties even if their mains electricity and water supplies fail. But after several hospitals in Britain had machines infected by the Wannacry malware in May 2017, they closed down their networks to limit further infection, and then found that they had to close their accident and emergency departments – as X-rays no longer travel from the X-ray machine to the operating theatre in an envelope, but via a server in a distant town. So a network failure can stop doctors operating when a power failure would not. There were standby generators, but no standby

network. Cloud services can make things more reliable on average, but the failures can be bigger, more complex, and correlated. An issue surfaced by the coronavirus pandemic is accessory control: some medical devices authenticate their spare parts, just as printers authenticate ink cartridges. Although the vendors claim this is for safety, it's actually so they can charge more money for spares. But it introduces fragility: when the supply chain gets interrupted, things are a lot harder to fix.

We'll look at medical system security (and safety too) in more detail later. This is a younger field than banking IT or military systems, but as healthcare accounts for a larger proportion of GNP than either of them in all developed countries, its importance is growing. It's also consistently the largest source of privacy breaches in countries with mandatory reporting.

## 1.6 Example 4 – the home

You might not think that the typical family operates any secure systems. But just stop and think.

1. You probably use some of the systems I've already described. You may use a web-based electronic banking system to pay bills, and you may have online access to your doctor's surgery so you can order repeat prescriptions. If you're diabetic then your insulin pump may communicate with a docking station at your bedside. Your home burglar alarm may send an encrypted 'all's well' signal to the security company every few minutes, rather than waking up the neighborhood when something happens.
2. Your car probably has an electronic immobilizer. If it was made before about 2015, the car unlocks when you press a button on the key, which sends an encrypted unlock command. If it's a more recent model, where you don't have to press any buttons but just have the key in your pocket, the car sends an encrypted challenge to the key and waits for the right response. But eliminating the button press meant that if you leave your key near the front door, a thief might use a radio relay to steal your car. Car thefts have shot up since this technology was introduced.
3. Your mobile phone authenticates itself to the network by a cryptographic challenge-response protocol similar to the ones used in car door locks and immobilizers, but the police can use a false base station (known in Europe as an IMSI-catcher, and in America as a Stingray) to listen in. And, as I mentioned above, many phone companies are relaxed about selling new SIM cards to people who claim their phones have been stolen; so a crook might steal your phone number and use this to raid your bank account.

4. In over 100 countries, households can get prepayment meters for electricity and gas, which they top up using a 20-digit code that they buy from an ATM or an online service. It even works off-grid; in Kenyan villages, people who can't afford $200 to buy a solar panel can get one for $2 a week and unlock the electricity it generates using codes they buy with their mobile phones.
5. Above all, the home provides a haven of physical security and seclusion. This is changing in a number of ways. Burglars aren't worried by locks as much as by occupants, so alarms and monitoring systems can help; but monitoring is also becoming pervasive, with many households buying systems like Alexa and Google Home that listen to what people say. All sorts of other gadgets now have microphones and cameras as voice and gesture interfaces become common, and the speech processing is typically done in the cloud to save battery life. By 2015, President Obama's council of advisers on science and technology was predicting that pretty soon every inhabited space on earth would have microphones that were connected to a small number of cloud service providers. (The USA and Europe have quite different views on how privacy law should deal with this.) One way or another, the security of your home may come to depend on remote systems over which you have little control.

Over the next few years, the number of such systems is going to increase rapidly. On past experience, many of them will be badly designed. For example, in 2019, Europe banned a children's watch that used unencrypted communications to the vendor's cloud service; a wiretapper could download any child's location history and cause their watch to phone any number in the world. When this was discovered, the EU ordered the immediate safety recall of all watches [903].

This book aims to help you avoid such outcomes. To design systems that are safe and secure, an engineer needs to know about what systems there are, how they work, and – at least as important – how they have failed in the past. Civil engineers learn far more from the one bridge that falls down than from the hundred that stay up; exactly the same holds in security engineering.

## 1.7 Definitions

Many of the terms used in security engineering are straightforward, but some are misleading or even controversial. There are more detailed definitions of technical terms in the relevant chapters, which you can find using the index. In this section, I'll try to point out where the main problems lie.

The first thing we need to clarify is what we mean by *system*. In practice, this can denote:

1. a product or component, such as a cryptographic protocol, a smartcard, or the hardware of a phone, a laptop or server;
2. one or more of the above plus an operating system, communications and other infrastructure;
3. the above plus one or more applications (banking app, health app, media player, browser, accounts/payroll package, and so on – including both client and cloud components);
4. any or all of the above plus IT staff;
5. any or all of the above plus internal users and management;
6. any or all of the above plus customers and other external users.

Confusion between the above definitions is a fertile source of errors and vulnerabilities. Broadly speaking, the vendor and evaluator communities focus on the first and (occasionally) the second of them, while a business will focus on the sixth (and occasionally the fifth). We will come across many examples of systems that were advertised or even certified as secure because the hardware was, but that broke badly when a particular application was run, or when the equipment was used in a way the designers didn't anticipate. Ignoring the human components, and thus neglecting usability issues, is one of the largest causes of security failure. So we will generally use definition 6; when we take a more restrictive view, it should be clear from the context.

The next set of problems comes from lack of clarity about who the players are and what they're trying to prove. In the literature on security and cryptology, it's a convention that principals in security protocols are identified by names chosen with (usually) successive initial letters – much like hurricanes, except that we use alternating genders. So we see lots of statements such as "Alice authenticates herself to Bob". This makes things much more readable, but can come at the expense of precision. Do we mean that Alice proves to Bob that her name actually is Alice, or that she proves she's got a particular credential? Do we mean that the authentication is done by Alice the human being, or by a smartcard or software tool acting as Alice's agent? In that case, are we sure it's Alice, and not perhaps Carol to whom Alice lent her card, or David who stole her phone, or Eve who hacked her laptop?

By a *subject* I will mean a physical person in any role including that of an operator, principal or victim. By a *person*, I will mean either a physical person or a legal person such as a company or government[1].

[1] The law around companies may come in handy when we start having to develop rules around AI. A company, like a robot, may be immortal and have some functional intelligence – but without consciousness. You can't jail a company but you can fine it.

placing it in an evidence bag. (The meaning of integrity has changed in the new context to include not just the signature but any fingerprints.)

The things we don't want are often described as hacking. I'll follow Bruce Schneier and define a *hack* as something a system's rules permit, but which was unanticipated and unwanted by its designers [1682]. For example, tax attorneys study the tax code to find loopholes which they develop into tax avoidance strategies; in exactly the same way, black hats study software code to find loopholes which they develop into exploits. Hacks can target not just the tax system and computer systems, but the market economy, our systems for electing leaders and even our cognitive systems. They can happen at multiple layers: lawyers can hack the tax code, or move up the stack and hack the legislature, or even the media. In the same way, you might try to hack a cryptosystem by finding a mathematical weakness in the encryption algorithm, or you can go down a level and measure the power drawn by a device that implements it in order to work out the key, or up a level and deceive the device's custodian into using it when they shouldn't. This book contains many examples. In the broader context, hacking is sometimes a source of significant innovation. If a hack becomes popular, the rules may be changed to stop it; but it may also become normalised (examples range from libraries through the filibuster to search engines and social media).

The last matter I'll clarify here is the terminology that describes what we're trying to achieve. A *vulnerability* is a property of a system or its environment which, in conjunction with an internal or external *threat*, can lead to a *security failure*, which is a breach of the system's security policy. By *security policy* I will mean a succinct statement of a system's protection strategy (for example, "in each transaction, sums of credits and debits are equal, and all transactions over $1,000,000 must be authorized by two managers"). A *security target* is a more detailed specification which sets out the means by which a security policy will be implemented in a particular product – encryption and digital signature mechanisms, access controls, audit logs and so on – and which will be used as the yardstick to evaluate whether the engineers have done a proper job. Between these two levels you may find a *protection profile* which is like a security target, except written in a sufficiently device-independent way to allow comparative evaluations among different products and different versions of the same product. I'll elaborate on security policies, security targets and protection profiles in Part 3. In general, the word *protection* will mean a property such as confidentiality or integrity, defined in a sufficiently abstract way for us to reason about it in the context of general systems rather than specific implementations.

This somewhat mirrors the terminology we use for safety-critical systems, and as we are going to have to engineer security and safety together in ever more applications it is useful to keep thinking of the two side by side.

In the safety world, a *critical* system or component is one whose failure could lead to an accident, given a *hazard* – a set of internal conditions or external circumstances. *Danger* is the probability that a hazard will lead to an accident, and *risk* is the overall probability of an accident. Risk is thus hazard level combined with danger and *latency* – the hazard exposure and duration. *Uncertainty* is where the risk is not quantifiable, while *safety* is freedom from accidents. We then have a *safety policy* which gives us a succinct statement of how risks will be kept below an acceptable threshold (and this might range from succinct, such as "don't put explosives and detonators in the same truck", to the much more complex policies used in medicine and aviation); at the next level down, we might find a *safety case* having to be made for a particular component such as an aircraft, an aircraft engine or even the control software for an aircraft engine.

## 1.8 Summary

'Security' is a terribly overloaded word, which often means quite incompatible things to different people. To a corporation, it might mean the ability to monitor all employees' email and web browsing; to the employees, it might mean being able to use email and the web without being monitored.

As time goes on, and security mechanisms are used more and more by the people who control a system's design to gain some commercial advantage over the other people who use it, we can expect conflicts, confusion and the deceptive use of language to increase.

One is reminded of a passage from Lewis Carroll:

> *"When I use a word," Humpty Dumpty said, in a rather scornful tone, "it means just what I choose it to mean – neither more nor less." "The question is," said Alice, "whether you can make words mean so many different things." "The question is," said Humpty Dumpty, "which is to be master – that's all."*

The security engineer must be sensitive to the different nuances of meaning that words acquire in different applications, and be able to formalize what the security policy and target actually are. That may sometimes be inconvenient for clients who wish to get away with something, but, in general, robust security design requires that the protection goals are made explicit.

# CHAPTER 2

# Who Is the Opponent?

*Going all the way back to early time-sharing systems we systems people regarded the users, and any code they wrote, as the mortal enemies of us and each other. We were like the police force in a violent slum.*

– ROGER NEEDHAM

*False face must hide what the false heart doth know.*

– MACBETH

## 2.1 Introduction

Ideologues may deal with the world as they would wish it to be, but engineers deal with the world as it is. If you're going to defend systems against attack, you first need to know who your enemies are.

In the early days of computing, we mostly didn't have real enemies; while banks and the military had to protect their systems, most other people didn't really bother. The first computer systems were isolated, serving a single company or university. Students might try to hack the system to get more resources and sysadmins would try to stop them, but it was mostly a game. When dial-up connections started to appear, pranksters occasionally guessed passwords and left joke messages, as they'd done at university. The early Internet was a friendly place, inhabited by academics, engineers at tech companies, and a few hobbyists. We knew that malware was possible but almost nobody took it seriously until the late 1980s when PC viruses appeared, followed by the Internet worm in 1988. (Even that was a student experiment that escaped from the lab; I tell the story in section 21.3.2.)

Things changed once everyone started to get online. The mid-1990s saw the first spam, the late 1990s brought the first distributed denial-of-service attack, and the explosion of mail-order business in the dotcom boom introduced credit card fraud. To begin with, online fraud was a cottage industry; the same person would steal credit card numbers and use them to buy goods which he'd then sell, or make up forged cards to use in a store. Things changed in the mid-2000s with the emergence of underground markets. These let the bad guys specialise – one gang could write malware, another could harvest bank credentials, and yet others could devise ways of cashing out. This enabled them to get good at their jobs, to scale up and to globalise, just as manufacturing did in the late eighteenth century. The 2000s also saw the world's governments putting in the effort to 'Master the Internet' (as the NSA put it) – working out how to collect data at scale and index it, just as Google does, to make it available to analysts. It also saw the emergence of social networks, so that everyone could have a home online – not just geeks with the skills to create their own handcrafted web pages. And of course, once everyone is online, that includes not just spies and crooks but also jerks, creeps, racists and bullies.

Over the past decade, this threat landscape has stabilised. We also know quite a lot about it. Thanks to Ed Snowden and other whistleblowers, we know a lot about the capabilities and methods of Western intelligence services; we've also learned a lot about China, Russia and other nation-state threat actors. We know a lot about cybercrime; online crime now makes up about half of all crime, by volume and by value. There's a substantial criminal infrastructure based on malware and botnets with which we are constantly struggling; there's also a large ecosystem of scams. Many traditional crimes have gone online, and a typical firm has to worry not just about external fraudsters but also about dishonest insiders. Some firms have to worry about hostile governments, some about other firms, and some about activists. Many people have to deal with online hostility, from kids suffering cyber-bullying at school through harassment of elected politicians to people who are stalked by former partners. And our politics may become more polarised because of the dynamics of online extremism.

One of the first things the security engineer needs to do when tackling a new problem is to identify the likely opponents. Although you can design some specific system components (such as cryptography) to resist all reasonable adversaries, the same is much less true for a complex real-world system. You can't protect it against all possible threats and still expect it to do useful work at a reasonable cost. So what sort of capabilities will the adversaries have, and what motivation? How certain are you of this assessment, and how might it change over the system's lifetime? In this chapter I will classify online and electronic threats depending on motive. First, I'll discuss surveillance, intrusion and manipulation done by governments for reasons of state, ranging from cyber-intelligence to cyber-conflict operations. Second, I'll deal with criminals whose motive is mainly money. Third will be researchers who find vulnerabilities for fun or for money, or who report them out of social conscience – compelling firms to patch their software and clean up their operations. Finally, I'll discuss bad actors whose reasons are personal and who mainly commit crimes against the person, from cyber-bullies to stalkers.

The big service firms, such as Microsoft, Google and Facebook, have to worry about all four classes of threat. Most firms and most private individuals will only be concerned with some of them. But it's important for a security engineer to understand the big picture so you can help clients work out what their own threat model should be, and what sort of attacks they should plan to forestall.

## 2.2 Spies

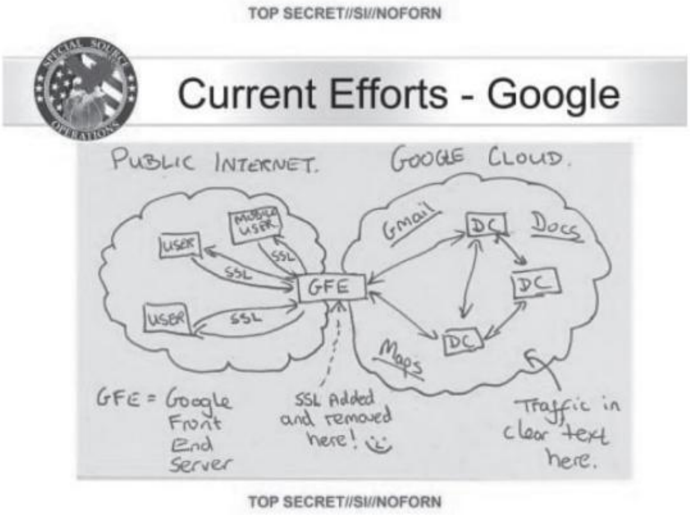
Governments have a range of tools for both passive surveillance of networks and active attacks on computer systems. Hundreds of firms sell equipment for wiretapping, for radio intercept, and for using various vulnerabilities to take over computers, phones and other digital devices. However, there are significant differences among governments in scale, objectives and capabilities. We'll discuss four representative categories – the USA and its allies, China, Russia and the Arab world – from the viewpoint of potential opponents. Even if spies aren't in your threat model today, the tools they use will quite often end up in the hands of the crooks too, sooner or later.

### 2.2.1 The Five Eyes

Just as everyone in a certain age range remembers where they were when John Lennon was shot, everyone who's been in our trade since 2013 remembers where they were when they learned of the Snowden revelations on Friday 7th June of that year.

#### *2.2.1.1 Prism*

I was in a hotel in Palo Alto, California, reading the Guardian online before a scheduled visit to Google where I'd been as a scientific visitor in 2011, helping develop contactless payments for Android phones. The headline was 'NSA Prism program taps in to user data of Apple, Google and others'; the article, written by Glenn Greenwald and Ewen MacAskill, describes a system called Prism that collects the Gmail and other data of users who are not US citizens or permanent residents, and is carried out under an order from the FISA court [818]. After breakfast I drove to the Googleplex, and found that my former colleagues were just as perplexed as I was. They knew nothing about Prism. Neither did the mail team. How could such a wiretap have been built? Had an order been served on Eric Schmidt, and if so how could he have implemented it without the mail and security teams knowing? As the day went on, people stopped talking.



TOP SECRET//SI//NOFORN

Current Efforts - Google

PUBLIC INTERNET.
GOOGLE CLOUD.
USER
MOBILE USER
SSL
SSL
GFE
USER
SSL
Gmail
DC
Docs
DC
DC
Maps
GFE = Google Front End Server
SSL Added and removed here! ☺
Traffic in clear text here.

TOP SECRET//SI//NOFORN

**Figure 2.1:** Muscular – the slide

### *2.2.1.4 Special collection*

The NSA and CIA jointly operate the Special Collection Service (SCS) whose most visible activity may be the plastic panels near the roofs of US and allied embassies worldwide; these hide antennas for hoovering up cellular communication (a program known as 'Stateroom'). Beyond this, SCS implants collection equipment in foreign telcos, Internet exchanges and government facilities. This can involve classical spy tradecraft, from placing bugs that monitor speech or electronic communications, through recruiting moles in target organisations, to the covert deployment of antennas in target countries to tap internal microwave links. Such techniques are not restricted to state targets: Mexican drug cartel leader 'El Chapo' Guzman was caught after US agents suborned his system administrator.

Close-access operations include Tempest monitoring: the collection of information leaked by the electromagnetic emissions from computer monitors and other equipment, described in 19.3.2. The Snowden leaks disclose the collection of computer screen data and other electromagnetic emanations from a number of countries' embassies and UN missions including those of India, Japan, Slovakia and the EU[2].

### *2.2.1.5 Bullrun and Edgehill*

Special collection increasingly involves supply-chain tampering. SCS routinely intercepts equipment such as routers being exported from the USA,

[2] If the NSA needs to use high-tech collection against you as they can't get a software implant into your computer, that may be a compliment!

adds surveillance implants, repackages them with factory seals and sends them onward to customers. And an extreme form of supply-chain tampering was when the NSA covertly bought Crypto AG, a Swiss firm that was the main supplier of cryptographic equipment to non-aligned countries during the Cold War; I tell the story in more detail later in section 26.2.7.1.

Bullrun is the NSA codename, and Edgehill the GCHQ one, for 'crypto enabling', a $100m-a-year program of tampering with supplies and suppliers at all levels of the stack. This starts off with attempts to direct, or misdirect, academic research[3]; it continued with placing trusted people on standards committees, and using NIST's influence to get weak standards adopted. One spectacular incident was the `Dual_EC_DRBG` debacle, where NIST standardised a random number generator based on elliptic curves that turned out to contain an NSA backdoor. Most of the actual damage, though, was done by restrictions on cryptographic key length, dovetailed with diplomatic pressure on allies to enforce export controls, so that firms needing export licenses could have their arms twisted to use an 'appropriate' standard, and was entangled with the Crypto Wars (which I discuss in section 26.2.7). The result was that many of the systems in use today were compelled to use weak cryptography, leading to vulnerabilities in everything from hotel and car door locks to VPNs. In addition to that, supply-chain attacks introduce covert vulnerabilities into widely-used software; many nation states play this game, along with some private actors [892]. We'll see vulnerabilities that result from surveillance and cryptography policies in one chapter after another, and return in Part 3 of the book to discuss the policy history in more detail.

### *2.2.1.6 Xkeyscore*

With such a vast collection of data, you need good tools to search it. The Five Eyes search computer data using Xkeyscore, a distributed database that enables an analyst to search collected data remotely and assemble the results. Exposed on July 31 2013, NSA documents describe it as its "widest-reaching" system for developing intelligence; it enables an analyst to search emails, SMSes, chats, address book entries and browsing histories [816]. Examples in a 2008 training deck include "my target speaks German but is in Pakistan. How can I find him?" "Show me all the encrypted Word documents from Iran" and "Show me all PGP usage in Iran". By searching for anomalous behaviour, the analyst can find suspects and identify strong selectors (such

[3] In the 1990s, when I bid to run a research program in coding theory, cryptography and computer security at the Isaac Newton Institute at Cambridge University, a senior official from GCHQ offered the institute a £50,000 donation not to go ahead, saying "There's nothing interesting happening in cryptography, and Her Majesty's Government would like this state of affairs to continue". He was shown the door and my program went ahead.

as email addresses, phone numbers or IP addresses) for more conventional collection.

Xkeyscore is a federated system, where one query scans all sites. Its components buffer information at collection points – in 2008, 700 servers at 150 sites. Some appear to be hacked systems overseas from which the NSA malware can exfiltrate data matching a submitted query. The only judicial approval required is a prompt for the analyst to enter a reason why they believe that one of the parties to the conversation is not resident in the USA. The volumes are such that traffic data are kept for 30 days but content for only 3–5 days. Tasked items are extracted and sent on to whoever requested them, and there's a notification system (Trafficthief) for tipping off analysts when their targets do anything of interest. Extraction is based either on fingerprints or plugins – the latter allow analysts to respond quickly with detectors for new challenges like steganography and homebrew encryption.

Xkeyscore can also be used for target discovery: one of the training queries is "Show me all the exploitable machines in country X" (machine fingerprints are compiled by a crawler called Mugshot). For example, it came out in 2015 that GCHQ and the NSA hacked the world's leading provider of SIM cards, the Franco-Dutch company Gemalto, to compromise the keys needed to intercept (and if need be spoof) the traffic from hundreds of millions of mobile phones [1661]. The hack used Xkeyscore to identify the firm's sysadmins, who were then phished; agents were also able to compromise billing servers to suppress SMS billing and authentication servers to steal keys; another technique was to harvest keys in transit from Gemalto to mobile service providers. According to an interview with Snowden in 2014, Xkeyscore also lets an analyst build a fingerprint of any target's online activity so that they can be followed automatically round the world. The successes of this system are claimed to include the capture of over 300 terrorists; in one case, Al-Qaida's Sheikh Atiyatallah blew his cover by googling himself, his various aliases, an associate and the name of his book [1661].

There's a collection of decks on Xkeyscore with a survey by Morgan Marquis-Boire, Glenn Greenwald and Micah Lee [1232]; a careful reading of the decks can be a good starting point for exploring the Snowden hoard[4].

#### *2.2.1.7 Longhaul*

Bulk key theft and supply-chain tampering are not the only ways to defeat cryptography. The Xkeyscore training deck gives an example: "Show me all the VPN startups in country X, and give me the data so I can decrypt and discover the users". VPNs appear to be easily defeated; a decryption service

[4] There's also a search engine for the collection at `https://www.edwardsnowden.com`.

called Longhaul ingests ciphertext and returns plaintext. The detailed description of cryptanalytic techniques is held as *Extremely Compartmented Information* (ECI) and is not found in the Snowden papers, but some of them talk of recent breakthroughs in cryptanalysis. What might these be?

The leaks do show diligent collection of the protocol messages used to set up VPN encryption, so some cryptographers suggested in 2015 that some variant of the "Logjam attack" is feasible for a nation-state attacker against the 1024-bit prime used by most VPNs and many TLS connections with Diffie-Hellman key exchange [26]. Others pointed to the involvement of NSA cryptographers in the relevant standard, and a protocol flaw discovered later; yet others pointed out that even with advances in number theory or protocol exploits, the NSA has enough money to simply break 1024-bit Diffie-Hellman by brute force, and this would be easily justified if many people used the same small number of prime moduli – which they do [854]. I'll discuss cryptanalysis in more detail in Chapter 5.

#### *2.2.1.8 Quantum*

There is a long history of attacks on protocols, which can be spoofed, replayed and manipulated in various ways. (We'll discuss this topic in detail in Chapter 4.) The best-documented NSA attack on Internet traffic goes under the codename of Quantum and involves the dynamic exploitation of one of the communication end-points. Thus, to tap an encrypted SSL/TLS session to a webmail provider, the Quantum system fires a 'shot' that exploits the browser. There are various flavours; in 'Quantuminsert', an injected packet redirects the browser to a 'Foxacid' attack server. Other variants attack software updates and the advertising networks whose code runs in mobile phone apps [1999].

#### *2.2.1.9 CNE*

Computer and Network Exploitation (CNE) is the generic NSA term for hacking, and it can be used for more than just key theft or TLS session hijacking; it can be used to acquire access to traffic too. Operation Socialist was the GCHQ codename for a hack of Belgium's main telco Belgacom[5] in 2010–11. GCHQ attackers used Xkeyscore to identify three key Belgacom technical staff, then used Quantuminsert to take over their PCs when they visited sites like LinkedIn. The attackers then used their sysadmin privileges to install malware on dozens of servers, including authentication servers to leverage further access, billing servers so they could cover their tracks, and the company's core Cisco routers [734]. This gave them access to large quantities of mobile

[5] It is now called Proximus.

roaming traffic, as Belgacom provides service to many foreign providers when their subscribers roam in Europe. The idea that one NATO and EU member state would conduct a cyber-attack on the critical infrastructure of another took many by surprise. The attack also gave GCHQ access to the phone system in the European Commission and other European institutions. Given that these institutions make many of the laws for the UK and other member states, this was almost as if a US state governor had got his state troopers to hack AT&T so he could wiretap Congress and the White House.

Belgacom engineers started to suspect something was wrong in 2012, and realised they'd been hacked in the spring of 2013; an anti-virus company found sophisticated malware masquerading as Windows files. The story went public in September 2013, and the German news magazine Der Spiegel published Snowden documents showing that GCHQ was responsible. After the Belgian prosecutor reported in February 2018, we learned that the attack must have been authorised by then UK Foreign Secretary William Hague, but there was not enough evidence to prosecute anyone; the investigation had been hampered in all sorts of ways both technical and political; the software started deleting itself within minutes of discovery, and institutions such as Europol (whose head was British) refused to help. The Belgian minister responsible for telecomms, Alexander de Croo, even suggested that Belgium's own intelligence service might have informally given the operation a green light [735]. Europol later adopted a policy that it will help investigate hacks of 'suspected criminal origin'; it has nothing to say about hacks by governments.

A GCHQ slide deck on CNE explains that it's used to support conventional Sigint both by redirecting traffic and by "enabling" (breaking) cryptography; that it must always be "UK deniable"; and that it can also be used for "effects", such as degrading communications or "changing users' passwords on extremist website" [735]. Other papers show that the agencies frequently target admins of phone companies and ISPs in the Middle East, Africa and indeed worldwide – compromising a key technician is "generally the entry ticket to the network" [1141]. As one phone company executive explained, "The MNOs were clueless at the time about network security. Most networks were open to their suppliers for remote maintenance with an ID and password and the techie in China or India had no clue that their PC had been hacked".

The hacking tools and methods used by the NSA and its allies are now fairly well understood; some are shared with law enforcement. The Snowden papers reveal an internal store where analysts can get a variety of tools; a series of leaks in 2016–7 by the Shadow Brokers (thought to be Russian military intelligence, the GRU) disclosed a number of actual NSA malware samples, used by hackers at the NSA's Tailored Access Operations team to launch attacks [239]. (Some of these tools were repurposed by the Russians to launch the NotPetya worm and by the North Koreans in Wannacry, as I'll discuss later.) The best documentation of all is probably about a separate store of goodies used by the

Stuxnet acted as a wake-up call for other governments, which rushed to acquire 'cyber-weapons' and develop offensive cyber *doctrine* – a set of principles for what cyber warriors might do, developed with some thought given to rationale, strategy, tactics and legality. Oh, and the price of zero-day vulnerabilities rose sharply.

#### *2.2.1.12 Attack scaling*

Computer scientists know the importance of how algorithms scale, and exactly the same holds for attacks. Tapping a single mobile phone is hard. You have to drive around behind the suspect with radio and cryptanalysis gear in your car, risk being spotted, and hope that you manage to catch the suspect's signal as they roam from one cell to another. Or you can drive behind them with a false base station[7] and hope their phone will roam to it as the signal is louder than the genuine one; but then you risk electronic detection too. Both are highly skilled work and low-yield: you lose the signal maybe a quarter of the time. So if you want to wiretap someone in central Paris often enough, why not just wiretap everyone? Put antennas on your embassy roof, collect it all, write the decrypted calls and text messages into a database, and reconstruct the sessions electronically. If you want to hack everyone in France, hack the telco, perhaps by subverting the equipment it uses. At each stage the capital cost goes up but the marginal cost of each tap goes down. The Five Eyes strategy is essentially to collect everything in the world; it might cost billions to establish and maintain the infrastructure, but once it's there you have everything.

The same applies to offensive cyber operations, which are rather like sabotage. In wartime, you can send commandos to blow up an enemy radar station; but if you do it more than once or twice, your lads will start to run into a lot of sentries. So we scale kinetic attacks differently: by building hundreds of bomber aircraft, or artillery pieces, or (nowadays) thousands of drones. So how do you scale a cyber attack to take down not just one power station, but the opponent's whole power grid? The Five Eyes approach is this. Just as Google keeps a copy of the Internet on a few thousand servers, with all the content and links indexed, US Cyber Command keeps a copy of the Internet that indexes what version of software all the machines in the world are using – the Mugshot system mentioned above – so a Five Eyes cyber warrior can instantly see which targets can be taken over by which exploits.

A key question for competitor states, therefore, is not just to what extent they can create some electronic spaces that are generally off-limits to the Five Eyes. It's the extent to which they can scale up their own intelligence and offensive capabilities rather than having to rely on America. The number of scans and

[7] These devices are known in the USA as a Stingray and in Europe as an IMSI-catcher; they conduct a man-in-the-middle attack of the kind we'll discuss in detail in section 22.3.1.

probes that we see online indicates that the NSA are not alone in trying to build cyber weapons that scale. Not all of them might be nation states; some might simply be arms vendors or mercenaries. This raises a host of policy problems to which we'll return in Part 3. For now we'll continue to look at capabilities.

### 2.2.2 China

China is now the leading competitor to the USA, being second not just in terms of GDP but as a technology powerhouse. The Chinese lack the NSA's network of alliances and access to global infrastructure (although they're working hard at that). Within China itself, however, they demand unrestricted access to local data. Some US service firms used to operate there, but trouble followed. After Yahoo's systems were used to trap the dissident Wang Xiaoning in 2002, Alibaba took over Yahoo's China operation in 2005; but there was still a row when Wang's wife sued Yahoo in US courts in 2007, and showed that Yahoo had misled Congress over the matter [1764]. In 2008, it emerged that the version of Skype available in China had been modified so that messages were scanned for sensitive keywords and, if they were found, the user's texts were uploaded to a server in China [1963]. In December 2009, Google discovered a Chinese attack on its corporate infrastructure, which became known as Operation Aurora; Chinese agents had hacked into the Google systems used to do wiretaps for the FBI (see Prism above) in order to discover which of their own agents in the USA were under surveillance. Google had already suffered criticism for operating a censored version of their search engine for Chinese users, and a few months later, they pulled out of China. By this time, Facebook, Twitter and YouTube had already been blocked. A Chinese strategy was emerging of total domestic control, augmented by ever-more aggressive collection overseas.

From about 2002, there had been a series of hacking attacks on US and UK defence agencies and contractors, codenamed 'Titan Rain' and ascribed to the Chinese armed forces. According to a 2004 study by the US Foreign Military Studies Office (FMSO), Chinese military doctrine sees the country in a state of war with the West; we are continuing the Cold War by attacking China, trying to overthrow its communist regime by exporting subversive ideas to it over the Internet [1884]. Chinese leaders see US service firms, news websites and anonymity tools such as Tor (which the State Department funds so that Chinese and other people can defeat censorship) as being of one fabric with the US surveillance satellites and aircraft that observe their military defences. Yahoo and Google were thus seen as fair game, just like Lockheed Martin and BAe.

Our own group's first contact with the Chinese came in 2008. We were asked for help by the Dalai Lama, who had realised that the Chinese had hacked his office systems in the run-up to the Beijing Olympics that year. One of my research students, Shishir Nagaraja, happened to be in Delhi waiting for his UK visa to be renewed, so he volunteered to go up to the Tibetan HQ in Dharamsala and run some forensics. He found that about 35 of the 50 PCs in the office of the Tibetan government in exile had been hacked; information was being siphoned off to China, to IP addresses located near the three organs of Chinese state security charged with different aspects of Tibetan affairs. The attackers appear to have got in by sending one of the monks an email that seemed to come from a colleague; when he clicked on the attached PDF, it had a JavaScript buffer overflow that used a vulnerability in Adobe Reader to take over his machine. This technique is called *phishing*, as it works by offering a lure that someone bites on; when it's aimed at a specific individual (as in this case) it's called *spear phishing*. They then compromised the Tibetans' mail server, so that whenever one person in the office sent a .pdf file to another, it would arrive with an embedded attack. The mail server itself was in California.

This is pretty sobering, when you stop to think about it. You get an email from a colleague sitting ten feet away, you ask him if he just sent it – and when he says yes, you click on the attachment. And your machine is suddenly infected by a server that you rent ten thousand miles away in a friendly country. We wrote this up in a tech report on the 'Snooping Dragon' [1376]. After it came out, we had to deal for a while with attacks on our equipment, and heckling at conference talks by Chinese people who claimed we had no evidence to attribute the attacks to their government. Colleagues at the Open Net Initiative in Toronto followed through, and eventually found from analysis of the hacking tools' dashboard that the same espionage network had targeted 1,295 computers in 103 countries [1225] – ranging from the Indian embassy in Washington through Associated Press in New York to the ministries of foreign affairs in Thailand, Iran and Laos.

There followed a series of further reports of Chinese state hacking, from a complex dispute with Rio Tinto in 2009 over the price of iron ore and a hack of the Melbourne International Film festival in the same year when it showed a film about a Uighur leader [1902]. In 2011, the Chinese hacked the CIA's covert communications system, after the Iranians had traced it, and executed about 30 agents – though that did not become publicly known till later [578]. The first flashbulb moment was a leaked Pentagon report in 2013 that Chinese hackers had stolen some of the secrets of the F35 joint strike fighter, as well as a series of other weapon systems [1381]. Meanwhile China and Hong Kong were amounting for over 80% of all counterfeit goods seized at US ports. The Obama administration vowed to make investigations and prosecutions in the theft of trade secrets a top priority, and the following year five members of the People's Liberation Army were indicted in absentia.

The White House felt compelled to act once more after the June 2015 news that the Chinese had hacked the Office of Personnel Management (OPM), getting access to highly personal data on 22 million current and former federal employees, ranging from fingerprints to sensitive information from security clearance interviews. Staff applying for Top Secret clearances are ordered to divulge all information that could be used to blackmail them, from teenage drug use to closeted gay relationships. All sexual partners in the past five years have to be declared for a normal Top Secret clearance; for a Strap clearance (to deal with signals intelligence material) the candidate even has to report any foreigners they meet regularly at their church. So this leak affected more than just 22 million people. Officially, this invasive data collection is to mitigate the risk that intelligence agency staff can be blackmailed. (Cynics supposed it was also so that whistleblowers could be discredited.) Whatever the motives, putting all such information in one place was beyond stupid; it was a real 'database of ruin'. For the Chinese to get all the compromising information on every American with a sensitive government job was jaw-dropping. (Britain screwed up too; in 2008, a navy officer lost a laptop containing the personal data of 600,000 people who had joined the Royal Navy, or tried to [1074].) At a summit in September that year, Presidents Obama and Xi agreed to refrain from computer-enabled theft of intellectual property for commercial gain[8]. Nothing was said in public though about military secrets – or the sex lives of federal agents.

The Chinese attacks of the 2000s used smart people plus simple tools; the attacks on the Tibetans used Russian crimeware as the remote access Trojans. The state also co-opted groups of 'patriotic hackers', or perhaps used them for deniability; some analysts noted waves of naïve attacks on western firms that were correlated with Chinese university terms, and wondered whether students had been tasked to hack as coursework. The UK police and security service warned UK firms in 2007. By 2009, multiple Chinese probes had been reported on US electricity firms, and by 2010, Chinese spear-phishing attacks had been reported on government targets in the USA, Poland and Belgium [1306]. As with the Tibetan attacks, these typically used crude tools and had such poor operational security that it was fairly clear where they came from.

By 2020 the attacks had become more sophisticated, with a series of advanced persistent threats (APTs) tracked by threat intelligence firms. A campaign to hack the phones of Uighurs involved multiple zero-day attacks, even on iPhones, that were delivered via compromised Uighur websites [395]; this targeted not only Uighurs in China but the diaspora too. China also conducts industrial and commercial espionage, and Western agencies claim they exploit

[8] The Chinese have kept their promise; according to US firms doing business in China, IP is now sixth on the list of concerns, down from second in 2014 [704]. In any case, the phrase 'IP theft' was always a simplification, used to conflate the theft of classified information from defence contractors with the larger issue of compelled technology transfer by other firms who wanted access to Chinese markets and the side-issue of counterfeiting.

managed service providers[9]. Another approach was attacking software supply chains; a Chinese group variously called Wicked Panda or Barium compromised software updates from computer maker Asus, a PC cleanup tool and a Korean remote management tool, as well as three popular computer games, getting its malware installed on millions of machines; rather than launching banking trojans or ransomware, it was then used for spying [811]. Just as in GCHQ's Operation Socialist, such indirect strategies give a way to scale attacks in territory where you're not the sovereign. And China was also playing the Socialist game: it came out in 2019 that someone had hacked at least ten western mobile phone companies over the previous seven years and exfiltrated call data records – and that the perpetrators appeared to be the APT10 gang, linked to the Chinese military [2021].

Since 2018 there has been a political row over whether Chinese firms should be permitted to sell routers and 5G network hardware in NATO countries, with the Trump administration blacklisting Huawei in May 2019. There had been a previous spat over another Chinese firm, ZTE; in 2018 GCHQ warned that ZTE equipment "would present risk to UK national security that could not be mitigated effectively or practicably" [1477][10]. President Trump banned ZTE for breaking sanctions on North Korea and Iran, but relented and allowed its equipment back in the USA subject to security controls[11].

The security controls route had been tried with Huawei, which set up a centre in Oxfordshire in 2010 where GCHQ could study its software as a condition of the company's being allowed to sell in the UK. While the analysts did not find any backdoors, their 2019 report surfaced some scathing criticisms of Huawei's software engineering practices [933]. Huawei had copied a lot of code, couldn't patch what they didn't understand, and no progress was being made in tackling many problems despite years of promises. There was an unmanageable number of versions of OpenSSL, including versions that had known vulnerabilities and that were not supported: 70 full copies of 4 different OpenSSL versions, and 304 partial copies of 14 versions. Not only could the Chinese hack the Huawei systems; so could anybody. Their equipment had been excluded for some years from UK backbone routers and from systems used for wiretapping. The UK demanded "sustained evidence of improvement across multiple versions and multiple product ranges" before

[9] This became public in 2019 with the claim that they had hacked Wipro and used this to compromise their customers [1095]; but it later emerged that Wipro had been hacked by a crime gang operating for profit.

[10] The only router vendor to have actually been caught with a malicious backdoor in its code is the US company Juniper, which not only used the NSA's Dual-EC backdoor to make VPN traffic exploitable, but did it in such a clumsy way that others could exploit it too – and at least one other party did so [415].

[11] This was done as a favour to President Xi, according to former National Security Adviser John Bolton, who declared himself 'appalled' that the president would interfere in a criminal prosecution [157].

Estonia and Georgia were little more than warm-ups for the Ukraine invasion. Following demonstrations in Maidan Square in Kiev against pro-Russian President Yanukovich, and an intervention in February 2014 by Russian mercenaries who shot about a hundred demonstrators, Yanukovich fled. The Russians invaded Ukraine on February 24th, annexing Crimea and setting up two puppet states in the Donbass area of eastern Ukraine. Their tactics combined Russian special forces in plain uniforms, a welter of propaganda claims of an insurgency by Russian-speaking Ukrainians or of Russia helping defend the population against Ukrainian fascists or of defending Russian purity against homosexuals and Jews; all of this coordinated with a variety of cyber-attacks. For example, in May the Russians hacked the website of the Ukrainian election commission and rigged it to display a message that a nationalist who'd received less than 1% of the vote had won; this was spotted and blocked, but Russian media announced the bogus result anyway [1802].

The following year, as the conflict dragged on, Russia took down 30 electricity substations on three different distribution systems within half an hour of each other, leaving 230,000 people without electricity for several hours. They involved multiple different attack vectors that had been implanted over a period of months, and since they followed a Ukrainian attack on power distribution in Crimea – and switched equipment off when they could have destroyed it instead – seemed to have been intended as a warning [2070]. This attack was still tiny compared with the other effects of the conflict, which included the shooting down of a Malaysian Airlines airliner with the loss of all on board; but it was the first cyber-attack to disrupt mains electricity. Finally on June 27 2017 came the NotPetya attack – by far the most damaging cyber-attack to date [814].

The NotPetya worm was initially distributed using the update service for MeDoc, the accounting software used by the great majority of Ukrainian businesses. It then spread laterally in organisations across Windows file-shares using the EternalBlue vulnerability, an NSA exploit with an interesting history. From March 2016, a Chinese gang started using it against targets in Vietnam, Hong Kong and the Philippines, perhaps as a result of finding and reverse engineering it (it's said that you don't launch a cyberweapon; you share it). It was leaked by a gang called the 'Shadow Brokers' in April 2017, along with other NSA software that the Chinese didn't deploy, and then used by the Russians in June. The NotPetya worm used EternalBlue together with the Mimikatz tool that recovers passwords from Windows memory. The worm's payload pretended to be ransomware; it encrypted the infected computer's hard disk and demanded a ransom of $300 in bitcoin. But there was no mechanism to decrypt the files of computer owners who paid the ransom, so it was really a destructive service-denial worm. The only way to deal with it was to re-install the operating system and restore files from backup.

The NotPetya attack took down banks, telcos and even the radiation monitoring systems at the former Chernobyl nuclear plant. What's more, it spread from Ukraine to international firms who had offices there. The world's largest container shipping company, Maersk, had to replace most of its computers and compensate customers for late shipments, at a cost of $300m; FedEx also lost $300m, and Mondelez $100m. Mondelez' insurers refused to pay out on the ground that it was an 'Act of War', as the governments of Ukraine, the USA and the UK all attributed NotPetya to Russian military intelligence, the GRU [1234].

2016 was marked by the Brexit referendum in the UK and the election of President Trump in the USA, in both of which there was substantial Russian interference. In the former, the main intervention was financial support for the leave campaigns, which were later found to have broken the law by spending too much [1267]; this was backed by intensive campaigning on social media [365]. In the latter, Russian interference was denounced by President Obama during the campaign, leading to renewed economic sanctions, and by the US intelligence community afterwards. An inquiry by former FBI director Robert Mueller found that Russia interfered very widely via the disinformation and social media campaigns run by its Internet Research Agency 'troll farm', and by the GRU which hacked the emails of the Democratic national and campaign committees, most notably those of the Clinton campaign chair John Podesta. Some Trump associates went to jail for various offences.

As I'll discuss in section 26.4.2, it's hard to assess the effects of such interventions. On the one hand, a report to the US Senate's Committee on Foreign Relations sets out a story of a persistent Russian policy, since Putin came to power, to undermine the influence of democratic states and the rules-based international order, promoting authoritarian governments of both left and right, and causing trouble where it can. It notes that European countries use broad defensive measures including bipartisan agreements on electoral conduct and raising media literacy among voters; it recommends that these be adopted in the USA as well [387]. On the other hand, Yochai Benkler cautions Democrats against believing that Trump's election was all Russia's fault; the roots of popular disaffection with the political elite are much older and deeper [228]. Russia's information war with the West predates Putin; it continues the old USSR's strategy of weakening the West by fomenting conflict via a variety of national liberation movements and terrorist groups (I discuss the information-warfare aspects in section 23.8.3). Timothy Snyder places this all in the context of modern Russian history and politics [1802]; his analysis also outlines the playbook for disruptive information warfare against a democracy. It's not just about hacking substations, but about hacking voters' minds; about undermining trust in institutions and even in facts, exploiting social media and recasting politics as showbusiness. Putin is a judo player; judo's about using an opponent's strength and momentum to trip them up.

### 2.2.4 The rest

The rest of the world's governments have quite a range of cyber capabilities, but common themes, including the nature and source of their tools. Middle Eastern governments were badly shaken by the Arab Spring uprisings, and some even turned off the Internet for a while, such as Libya in April–July 2010, when rebels were using Google maps to generate target files for US, UK and French warplanes. Since then, Arab states have developed strategies that combine spyware and hacking against high-profile targets, through troll farms pumping out abusive comments in public fora, with physical coercion.

The operations of the United Arab Emirates were described in 2019 by a whistleblower, Lori Stroud [248]. An NSA analyst – and Ed Snowden's former boss – she was headhunted by a Maryland contractor in 2014 to work in Dubai as a mercenary, but left after the UAE's operations started to target Americans. The UAE's main technique was spear-phishing with Windows malware, but their most effective tool, called Karma, enabled them to hack the iPhones of foreign statesmen and local dissidents. They also targeted foreigners critical of the regime. In one case they social-engineered a UK grad student into installing spyware on his PC on the pretext that it would make his communications hard to trace. The intelligence team consisted of several dozen people, both mercenaries and Emiratis, in a large villa in Dubai. The use of iPhone malware by the UAE government was documented by independent observers [1221].

In 2018, the government of Saudi Arabia murdered the Washington Post journalist Jamal Khashoggi in its consulate in Istanbul. The Post campaigned to expose Saudi crown prince Mohammed bin Salman as the man who gave the order, and in January 2019 the National Enquirer published a special edition containing texts showing that the Post's owner Jeff Bezos was having an affair. Bezos pre-empted the Enquirer by announcing that he and his wife were divorcing, and hired an investigator to find the source of the leak. The Enquirer had attempted to blackmail Bezos over some photos it had also obtained; it wanted both him and the investigator to declare that the paper hadn't relied upon 'any form of electronic eavesdropping or hacking in their news-gathering process'. Bezos went public instead. According to the investigator, his iPhone had been hacked by the Saudi Arabian government [200]; the malicious WhatsApp message that did the damage was sent from the phone of the Crown Prince himself [1055]. The US Justice Department later charged two former Twitter employees with spying, by disclosing to the Saudis personal account information of people who criticised their government [1502].

An even more unpleasant example is Syria, where the industrialisation of brutality is a third approach to scaling information collection. Malware attacks on dissidents were reported from 2012, and initially used a variety of spear-phishing lures. As the civil war got underway, police who were arresting suspects would threaten female family members with rape on the spot unless the suspect disclosed his passwords for mail and social media. They would then spear-phish all his contacts while he was being taken away in the van to the torture chamber. This victim-based approach to attack scaling resulted in the compromise of many machines not just in Syria but in America and Europe. The campaigns became steadily more sophisticated as the war evolved, with false-flag attacks, yet retained a brutal edge with some tools displaying beheading videos [737].

Thanks to John Scott-Railton and colleagues at Toronto, we have many further documented examples of online surveillance, computer malware and phone exploits being used to target dissidents; many in Middle Eastern and African countries but also in Mexico and indeed in Hungary [1221]. The real issue here is the ecosystem of companies, mostly in the USA, Europe and Israel, that supply hacking tools to unsavoury states. These tools range from phone malware, through mass-surveillance tools you use on your own network against your own dissidents, to tools that enable you to track and eavesdrop on phones overseas by abusing the signaling system [489]. These tools are used by dictators to track and monitor their enemies in the USA and Europe.

NGOs have made attempts to push back on this cyber arms trade. In one case NGOs argued that the Syrian government's ability to purchase mass-surveillance equipment from the German subsidiary of a UK company should be subject to export control, but the UK authorities were unwilling to block it. GCHQ was determined that if there were going to be bulk surveillance devices on President Assad's network, they should be British devices rather than Ukrainian ones. (I describe this in more detail later in section 26.2.8.) So the ethical issues around conventional arms sales persist in the age of cyber; indeed they can be worse because these tools are used against Americans, Brits and others who are sitting at home but who are unlucky enough to be on the contact list of someone an unpleasant government doesn't like. In the old days, selling weapons to a far-off dictator didn't put your own residents in harm's way; but cyber weapons can have global effects.

Having been isolated for years by sanctions, Iran has developed an indigenous cyber capability, drawing on local hacker forums. Like Syria, its main focus is on intelligence operations, particularly against dissident Iranians, both at home and overseas. It has also been the target of US and other attacks of which the best known was Stuxnet, after which it traced the CIA's covert communications network and rounded up a number of agents [578]. It has launched both espionage operations and attacks of its own overseas. An example of the former was its hack of the Diginotar CA in the Netherlands which enabled it to monitor dissidents' Gmail; while its Shamoon malware damaged thousands of PCs at Aramco, Saudi Arabia's national oil company. The history of Iranian cyber capabilities is told by Collin Anderson and Karim Sadjadpour [50]. Most recently, it attacked Israeli water treatment plants in

April 2020; Israel responded the following month with an attack on the Iranian port of Bandar Abbas [230].

Finally, it's worth mentioning North Korea. In 2014, after Sony Pictures started working on a comedy about a plot to assassinate the North Korean leader, a hacker group trashed much of Sony's infrastructure, released embarrassing emails that caused its top film executive Amy Pascal to resign, and leaked some unreleased films. This was followed by threats of terrorist attacks on movie theatres if the comedy were put on general release. The company put the film on limited release, but when President Obama criticised them for giving in to North Korean blackmail, they put it on full release instead.

In 2017, North Korea again came to attention after their Wannacry worm infected over 200,000 computers worldwide, encrypting data and demanding a bitcoin ransom – though like NotPetya it didn't have a means of selective decryption, so was really just a destructive worm. It used the NSA EternalBlue vulnerability, like NotPetya, but was stopped when a malware researcher discovered a kill switch. In the meantime it had disrupted production at carmakers Nissan and Renault and at the Taiwanese chip foundry TSMC, and also caused several hospitals in Britain's National Health Service to close their accident and emergency units. In 2018, the US Department of Justice unsealed an indictment of a North Korean government hacker for both incidents, and also for a series of electronic bank robberies, including of $81m from the Bank of Bangladesh [1656]. In 2019, North Korean agents were further blamed, in a leaked United Nations report, for the theft of over $1bn from cryptocurrency exchanges [348].

### 2.2.5 Attribution

It's often said that cyber is different, because attribution is hard. As a general proposition this is untrue; anonymity online is much harder than you think. Even smart people make mistakes in operational security that give them away, and threat intelligence companies have compiled a lot of data that enable them to attribute even false-flag operations with reasonable probability in many cases [181]. Yet sometimes it may be true, and people still point to the Climategate affair. Several weeks before the 2009 Copenhagen summit on climate change, someone published over a thousand emails, mostly sent to or from four climate scientists at the University of East Anglia, England. Climate sceptics seized on some of them, which discussed how to best present evidence of global warming, as evidence of a global conspiracy. Official inquiries later established that the emails had been quoted out of context, but the damage had been done. People wonder whether the perpetrator could have been the Russians or the Saudis or even an energy company. However one of the more convincing analyses suggests that it was an internal leak, or even an accident;

machines in hospitals to conduct a SYN flood attack [370]. The next use was spam, and by 2000 the Earthlink spammer sent over a million phishing emails; its author was sued by Earthlink. Once cyber-criminals started to get organised, there was a significant scale-up. We started to see professionally built and maintained botnets that could be rented out by bad guys, whether spammers, phishermen or others; by 2007 the Cutwail botnet was sending over 50 million spams a minute from over a million infected machines [1836]. Bots would initially contact a command-and-control server for instructions; these would be taken down, or taken over by threat intelligence companies for use as sinkholes to monitor infected machines, and to feed lists of them to ISPs and corporates.

The spammers' first response was peer-to-peer botnets. In 2007 Storm suddenly grew to account for 8% of all Windows malware; it infected machines mostly by malware in email attachments and had them use the eDonkey peer-to-peer network to find other infected machines. It was used not just for spam but for DDoS, for pump-and-dump stock scams and for harvesting bank credentials. Defenders got lots of peers to join this network to harvest lists of bot addresses, so the bots could be cleaned up, and by late 2008 Storm had been cut to a tenth of the size. It was followed by Kelihos, a similar botnet that also stole bitcoins; its creator, a Russian national, was arrested while on holiday in Spain in 2017 and extradited to the USA where he pled guilty in 2018 [661].

The next criminal innovation arrived with the Conficker botnet: the domain generation algorithm (DGA). Conficker was a worm that spread by exploiting a Windows network service vulnerability; it generated 250 domain names every day, and infected machines would try them all out in the hope that the botmaster had managed to rent one of them. Defenders started out by simply buying up the domains, but a later variant generated 50,000 domains a day and an industry working group made agreements with registrars that these domains would simply be put beyond use. By 2009 Conficker had grown so large, with maybe ten million machines, that it was felt to pose a threat to the largest websites and perhaps even to nation states. As with Storm, its use of randomisation proved to be a two-edged sword; defenders could sit on a subset of the domains and harvest feeds of infected machines. By 2015 the number of infected machines had fallen to under a million.

Regardless of whether something can be done to take out the command-and-control system, whether by arresting the botmaster or by technical tricks, the universal fix for botnet infections is to clean up infected machines. But this raises many issues of scale and incentives. While AV companies make tools available, and Microsoft supplies patches, many people don't use them. So long as your infected PC is merely sending occasional spam but works well enough otherwise, why should you go to the trouble of doing anything? But bandwidth costs ISPs money, so the next step was that some ISPs, particularly the cable companies like Comcast, would identify infected machines and confine their users to a 'walled garden' until they promised to clean up. By 2019 that has become less common as people now have all sorts of devices on their wifi, many of which have no user interface; communicating with human users has become harder.

In 2020, we find many botnets with a few tens of thousands of machines that are too small for most defenders to care about, plus some large ones that tend to be multilayer – typically with peer-to-peer mechanisms at the bottom that enable the footsoldier bots to communicate with a few control nodes, which in turn use a domain generation algorithm to find the botmaster. Fragmenting the footsoldiers into a number of small botnets makes it hard for defenders to infiltrate all of them, while the control nodes may be located in places that are hard for defenders to get at. The big money for such botnets in 2020 appears to be in clickfraud.

The latest innovation is Mirai, a family of botnets that exploit IoT devices. The first Mirai worm infected CCTV cameras that had been manufactured by Xiaomi and that had a known factory default password that couldn't be changed. Mirai botnets scan the Internet's IPv4 address space for other vulnerable devices which typically get infected within minutes of being powered up. The first major attack was on DynDNS and took down Twitter for six hours on the US eastern seaboard in October 2016. Since then there have been over a thousand variants, which researchers study to determine what's changed and to work out what countermeasures might be used.

At any one time, there may be half a dozen large botnet herders. The Mirai operators, for example, seem to be two or three groups that might have involved a few dozen people.

#### *2.3.1.2 Malware devs*

In addition to the several hundred software engineers who write malware for the world's intelligence agencies and their contractors, there may be hundreds of people writing malware for the criminal market; nobody really knows (though we can monitor traffic on hacker forums to guess the order of magnitude).

Within this community there are specialists. Some concentrate on turning vulnerabilities into exploits, a nontrivial task for modern operating systems that use stack canaries, ASLR and other techniques we'll discuss later in section 6.4.1. Others specialise in the remote access Trojans that the exploits install; others build the peer-to-peer and DGA software for resilient command-and-control communications; yet others design specialised payloads for bank fraud. The highest-value operations seem to be platforms that are maintained with constant upgrades to cope with the latest countermeasures from the anti-virus companies. Within each specialist market segment there are typically a handful of operators, so that when we arrest one of them it makes a difference for a while. Some of the providers are based in jurisdictions that don't extradite their nationals, like Russia, and Russian crimeware is used not just by Russian state actors but by others too.

As Android has taken over from Windows as the most frequently used operating system we've seen a rise in Android malware. In China and in countries with a lot of second-hand and older phones, this may be software that uses an unpatched vulnerability to root an Android phone; the USA and Europe have lots of unpatched phones (as many OEMs stop offering patches once a phone is no longer on sale) but it's often just apps that do bad things, such as stealing SMSes used to authenticate banking transactions.

#### 2.3.1.3 Spam senders

Spamming arrived on a small scale when the Internet opened to the public in the mid-1990s, and by 2000 we saw the Earthlink spammer making millions from sending phishing lures. By 2010 spam was costing the world's ISPs and tech companies about $1bn a year in countermeasures, but it earned its operators perhaps one percent of that. The main beneficiaries may have been webmail services such as Yahoo, Hotmail and Gmail, which can operate better spam filters because of scale; during the 2010s, hundreds of millions of people switched to using their services.

Spam is now a highly specialised business, as getting past modern spam filters requires a whole toolbox of constantly-changing tricks. If you want to use spam to install ransomware, you're better off paying an existing service than trying to learn it all from scratch. Some spam involves industrial-scale email compromise, which can be expensive for the victim; some $350m was knocked off the $4.8bn price at which Yahoo was sold to Verizon after a bulk compromise [772].

#### 2.3.1.4 Bulk account compromise

Some botnets are constantly trying to break into email and other online accounts by trying to guess passwords and password recovery questions. A large email service provider might be recovering several tens of thousands of accounts every day. There are peaks, typically when hackers compromise millions of email addresses and passwords at one website and then try them out at all the others. In 2019, this *credential stuffing* still accounts for the largest number of attempted account compromises by volume [1885]. Compromised accounts are sold on to people who exploit them in various ways. Primary email accounts often have recovery information for other accounts, including bank accounts if the attacker is lucky. They can also be used for scams such as the stranded traveler, where the victim emails all their friends saying they've been robbed in some foreign city and asking for urgent financial help to pay the hotel bill. If all else fails, compromised email accounts can be used to send spam.

A variant on the theme is the pay-per-install service, which implants malware on phones or PCs to order and at scale. This can involve a range of phishing lures in a variety of contexts, from free porn sites that ask you to install a special viewer, to sports paraphernalia offers and news about topical events. It can also use more technical means such as drive-by downloads. Such services are often offered by botnets which need them to maintain their own numbers; they might charge third party customers $10-15 per thousand machines infected in the USA and Europe, and perhaps $3 for Asia.

#### *2.3.1.5 Targeted attackers*

We've seen the emergence of hack-for-hire operators who will try to compromise a specific target account for a fee, of typically $750 [1885]. They will investigate the target, make multiple spear-phishing attempts, try password recovery procedures, and see if they can break in through related accounts. This continues a tradition of private eyes who traditionally helped in divorce cases and also stalked celebrities on behalf of red-top newspapers – though with even fewer ethical constraints now that services can be purchased anonymously online. John Scott-Railton and colleagues exposed the workings of Dark Basin, a hack-for-hire company that had targeted critics of ExxonMobil, and also net neutrality advocates, and traced it to a company in India [1695].

In recent years, targeted attacks have also been used at scale against small business owners and the finance staff of larger firms in order to carry out various kinds of payment fraud, as I'll discuss below in 2.3.2.

#### *2.3.1.6 Cashout gangs*

Back in the twentieth century, people who stole credit card numbers would have to go to the trouble of shopping for goods and then selling them to get money out. Nowadays there are specialists who buy compromised bank credentials on underground markets and exploit them. The prices reveal where the real value lies in the criminal chain; a combination of credit card number and expiry date sells for under a dollar, and to get into the single dollars you need a CVV, the cardholder's name and address, and more.

Cashout techniques change every few years, as paths are discovered through the world's money-laundering controls, and the regulations get tweaked to block them. Some cashout firms organise armies of *mules* to whom they transfer some of the risk. Back in the mid-2000s, mules could be drug users who would go to stores and buy goods with stolen credit cards; then there was a period when unwitting mules were recruited by ads promising large earnings to 'agents' to represent foreign companies but who were used to remit stolen funds through their personal bank accounts. The laundrymen next used Russian banks in Latvia, to which Russian mules would turn up to withdraw cash. Then Liberty Reserve, an unlicensed digital currency based in Costa Rica, was all the rage until it was closed down and its founder arrested in 2013. Bitcoin took over for a while but its popularity with the cybercrime community tailed off as its price became more volatile, as the US Department of the Treasury started arm-twisting bitcoin exchanges into identifying their customers.

As with spam, cashout is a constantly evolving attack-defence game. We monitor it and analyse the trends using CrimeBB, a database we've assembled of tens of millions of posts in underground hacker forums where cybercriminals buy and sell services including cashout [1501]. It also appears to favour gangs who can scale up, until they get big enough to attract serious law-enforcement attention: in 2020, one Sergey Medvedev pleaded guilty to inflicting more than $568 million in actual losses over the period 2010–15 [1932].

#### *2.3.1.7 Ransomware*

One reason for the decline in cryptocurrency may have been the growth of ransomware, and as the gangs involved in this switched to payment methods that are easier for victims to use. By 2016–17, 42% of ransomware encountered by US victims demanded prepaid vouchers such as Amazon gift cards; 14% demanded wire transfers and only 12% demanded cryptocurrency; a lot of the low-end ransomware aimed at consumers is now really scareware as it doesn't actually encrypt files at all [1746]. Since 2017, we've seen ransomware-as-a-service platforms; the operators who use these platforms are often amateurs and can't decrypt even if you're willing to pay.

Meanwhile a number of more professional gangs penetrate systems, install ransomware, wait until several days or weeks of backup data have been encrypted and demand substantial sums of bitcoin. This has grown rapidly over 2019–20, with the most high-profile ransomware victims in the USA being public-sector bodies; several hundred local government bodies and a handful of hospitals have suffered service failures [356]. During the pandemic, more hospitals have been targeted; the medical school at UCSF paid over $1m [1482]. It's an international phenomenon, though, and many private-sector firms fall victim too. Ransomware operators have also been threatening large-scale leaks of personal data to bully victims into paying.

### 2.3.2 Attacks on banking and payment systems

Attacks on card payment systems started with lost and stolen cards, with forgery at scale arriving in the 1980s; the dotcom boom ramped things up further in the 1990s as many businesses started selling online with little idea exactly the same game with aftermarket vendors. The use of cryptography for accessory control is now pervasive, being found even on water filter cartridges in fridges [1073]. Many customers find this annoying and try to circumvent the controls. The US courts decided in the Lexmark v SCC case that this was fine: the printer vendor Lexmark sued SCC, a company that sold clones of its security chips to independent ink vendors, but lost. So the incumbent can now hire the best cryptographers they can find to lock their products, while the challenger can hire the best cryptanalysts they can find to unlock them – and customers can hack them any way they can. Here, the conflict is legal and open. As with state actors, corporates sometimes assemble teams with multiple PhDs, millions of dollars in funding, and capital assets such as electron microscopes[13]. We discuss this in greater detail later in section 24.6.

Not all corporate attacks are conducted as openly. Perhaps the best-known covert hack was by Volkswagen on the EU and US emissions testing schemes; diesel engines sold in cars were programmed to run cleanly if they detected the standard emission test conditions, and efficiently otherwise. For this, the CEO of VW was fired and indicted in the USA (to which Germany won't extradite him), while the CEO of Audi was fired and jailed in Germany [1086]. VW has set aside €25bn to cover criminal and civil fines and compensation. Other carmakers were cheating too; Daimler was fined €860m in Europe in 2019 [1468], and in 2020 reached a US settlement consisting of a fine of $1.5bn from four government agencies plus a class action of $700m [1859]. Settlements for other manufacturers and other countries are in the pipeline.

Sometimes products are designed to break whole classes of protection system, an example being the overlay SIM cards described later in Chapter 12. These are SIM cards with two sides and only 160 microns thick, which you stick on top of the SIM card in your phone to provide a second root of trust; they were designed to enable people in China to defeat the high roaming charges of the early 2010s. The overlay SIM essentially does a man-in-the-middle attack on the real SIM, and can be programmed in Javacard. A side-effect is that such SIMs make it really easy to do some types of bank fraud.

So when putting together the threat model for your system, stop and think what capable motivated opponents you might have among your competitors, or among firms competing with suppliers on which products you depend. The obvious attacks include industrial espionage, but nowadays it's much more complex than that.

### 2.3.6 Whistleblowers

Intelligence agencies, and secretive firms, can get obsessive about 'the insider threat'. But in 2018, Barclays Bank's CEO was fined £642,000 and ordered to

[13] Full disclosure: both our hardware lab and our NGO activities have on occasion received funding from such actors.

repay £500,000 of his bonus for attempting to trace a whistleblower in the bank [698]. So let's turn it round and look at it from the other perspective – that of the whistleblower. Many are trying to do the right thing, often at a fairly mundane level such as reporting a manager who's getting bribes from suppliers or who is sexually harassing staff. In regulated industries such as banking they may have a legal duty to report wrongdoing and legal immunity against claims of breach of confidence by their employer. Even then, they often lose because of the power imbalance; they get fired and the problem goes on. Many security engineers think the right countermeasure to leakers is technical, such as data loss prevention systems, but robust mechanisms for staff to report wrongdoing are usually more important. Some organisations, such as banks, police forces and online services, have mechanisms for reporting crimes by staff but no effective process for raising ethical concerns about management decisions[14].

But even basic whistleblowing mechanisms are often an afterthought; they typically lead the complainant to HR rather than to the board's audit committee. External mechanisms may be little better. One big service firm ran a "Whistle-blowing hotline" for its clients in 2019; but the web page code has trackers from LinkedIn, Facebook and Google, who could thus identify unhappy staff members, and also JavaScript from CDNs, littered with cookies and referrers from yet more IT companies. No technically savvy leaker would use such a service. At the top end of the ecosystem, some newspapers offer ways for whistleblowers to make contact using encrypted email. But the mechanisms tend to be clunky and the web pages that promote them do not always educate potential leakers about either the surveillance risks, or the operational security measures that might counter them. I discuss the usability and support issues around whistleblowing in more detail in section 25.4.

This is mostly a policy problem rather than a technical one. It's difficult to design a technical mechanism whereby honest staff can blow the whistle on abuses that have become ingrained in an organisation's culture, such as pervasive sexual harassment or financial misconduct. In most cases, it's immediately clear who the whistleblower is, so the critical factor is whether the whistleblower will get external support. For example, will they ever get another job? This isn't just a matter of formal legal protection but also of culture. For example, the rape conviction of Harvey Weinstein empowered many women to protest about sexual harassment and discrimination; hopefully the Black Lives Matter protests will similarly empower people of colour [32].

An example where anonymity did help, though, was the UK parliamentary expenses scandal of 2008–9. During a long court case about whether the public could get access to the expense claims of members of parliament, someone

[14] Google staff ended up going on strike in 2018 about the handling of sexual harassment scandals.

went to the PC where the records were kept, copied them to a DVD and sold the lot to the Daily Telegraph. The paper published the juicy bits in instalments all through May and June, when MPs gave up and published the lot on Parliament's website. Half-a-dozen ministers resigned; seven MPs and peers went to prison; dozens of MPs stood down or lost their seats at the following election; and there was both mirth and outrage at some of the things charged to the taxpayer. The whistleblower may have technically committed a crime, but their action was clearly in the public interest; now all parliamentary expenses are public, as they should have been all along. If a nation's lawmakers have their hands in the till, what else will clean up the system?

Even in the case of Ed Snowden, there should have been a robust way for him to report unlawful conduct by the NSA to the appropriate arm of government, probably a Congressional committee. But he knew that a previous whistleblower, Bill Binney, had been arrested and harassed after trying to do that. In hindsight, that aggressive approach was unwise, as President Obama's NSA review group eventually conceded. At the less exalted level of a commercial firm, if one of your staff is stealing your money, and another wants to tell you about it, you'd better make that work.

## 2.4 Geeks

Our third category of attacker are the people like me – researchers who investigate vulnerabilities and report them so they can be fixed. Academics look for new attacks out of curiosity, and get rewarded with professional acclaim – which can lead to promotion for professors and jobs for the students who help us. Researchers working for security companies also look for newsworthy exploits; publicity at conferences such as Black Hat can win new customers. Hobby hackers break into stuff as a challenge, just as people climb mountains or play chess; hacktivists do it to annoy companies they consider to be wicked. Whether on the right side of the law or not, we tend to be curious introverts who need to feel in control, but accept challenges and look for the 'rush'. Our reward is often fame – whether via academic publications, by winning customers for a security consulting business, by winning medals from academic societies or government agencies, or even on social media. Sometimes we break stuff out of irritation, so we can circumvent something that stops us fixing something we own; and sometimes there's an element of altruism. For example, people have come to us in the past complaining that their bank cards had been stolen and used to buy stuff, and the banks wouldn't give them a refund, saying their PIN must have been used, when it hadn't. We looked into some of these cases and discovered the No-PIN and preplay attacks on chip and PIN systems, which I'll describe in the chapter on banking (the bad guys had actually discovered these attacks, but we replicated them and got justice for some of the victims).

Security researchers who discovered and reported vulnerabilities to a software vendor or system operator used to risk legal threats, as companies sometimes thought this would be cheaper than fixing things. So some researchers took to disclosing bugs anonymously on mailing lists; but this meant that the bad guys could use them at once. By the early 2000s, the IT industry had evolved practices of responsible disclosure whereby researchers disclose the bug to the maintainer some months in advance of disclosure. Many firms operate bug-bounty programs that offer rewards for vulnerabilities; as a result, independent researchers can now make serious money selling vulnerabilities, and more than one assiduous researcher has now earned over $1m doing this. Since the Stuxnet worm, governments have raced to stockpile vulnerabilities, and we now see some firms that buy vulnerabilities from researchers in order to weaponise them, and sell them to cyber-arms suppliers. Once they're used, they spread, are eventually reverse-engineered and patched. I'll discuss this ecosystem in more detail in the chapters on economics and assurance.

Some more traditional sectors still haven't adopted responsible disclosure. Volkswagen sued researchers in the universities of Birmingham and Nijmegen who reverse-engineered some online car theft tools and documented how poor their remote key entry system was. The company lost, making fools of themselves and publicising the insecurity of their vehicles (I'll discuss the technical details in section 4.3.1 and the policy in section 27.5.7.2). Eventually, as software permeates everything, software industry ways of working will become more widespread too. In the meantime, we can expect turbulence. Firms that cover up problems that harm their customers will have to reckon with the possibility that either an internal whistleblower, or an external security researcher, will figure out what's going on, and when that happens there will often be an established responsible disclosure process to invoke. This will impose costs on firms that fail to align their business models with it.

## 2.5 The swamp

Our fourth category is abuse, by which we usually mean offences against the person rather than against property. These range from cyber-bullying at schools all the way to state-sponsored Facebook advertising campaigns that get people to swamp legislators with death threats. I'll deal first with offences that scale, including political harassment and child sex abuse material, and then with offences that don't, ranging from school bullying to intimate partner abuse.

### 2.5.1 Hacktivism and hate campaigns

Propaganda and protest evolved as technology did. Ancient societies had to make do with epic poetry; cities enabled people to communicate with hundreds of others directly, by making speeches in the forum; and the invention of writing enabled a further scale-up. The spread of printing in the sixteenth century led to wars of religion in the seventeenth, daily newspapers in the eighteenth and mass-market newspapers in the nineteenth. Activists learned to compete for attention in the mass media, and honed their skills as radio and then TV came along.

Activism in the Internet age started off with using online media to mobilise people to do conventional lobbying, such as writing to legislators; organisations such as Indymedia and Avaaz developed expertise at this during the 2000s. In 2011, activists such as Wael Ghonim used social media to trigger the Arab Spring, which we discuss in more detail in section 26.4.1. Since then, governments have started to crack down, and activism has spread into online hate campaigns and radicalisation. Many hate campaigns are covertly funded by governments or opposition parties, but by no means all: single-issue campaign groups are also players. If you can motivate hundreds of people to send angry emails or tweets, then a company or individual on the receiving end can have a real problem. Denial-of-service attacks can interrupt operations while doxxing can do real brand damage as well as causing distress to executives and staff.

Activists vary in their goals, in their organisational coherence and in the extent to which they'll break the law. There's a whole spectrum, from the completely law-abiding NGOs who get their supporters to email legislators to the slightly edgy, who may manipulate news by getting bots to click on news stories, to game the media analytics and make editors pay more attention to their issue. Then there are whistleblowers who go to respectable newspapers, political partisans who harass people behind the mild anonymity of Twitter accounts, hackers who break into target firms and vandalise their websites or even doxx them. The Climategate scandal, described in 2.2.5 above, may be an example of doxxing by a hacktivist. At the top end, there are the hard-core types who end up in jail for terrorist offences.

During the 1990s, I happily used email and usenet to mobilise people against surveillance bills going through the UK parliament, as I'll describe later in section 26.2.7. I found myself on the receiving end of hacktivism in 2003 when the Animal Liberation Front targeted my university because of plans to build a monkey house, for primates to be used in research. The online component consisted of thousands of emails sent to staff members with distressing images of monkeys with wires in their brains; this was an early example of 'brigading', where hundreds of people gang up on one target online. We dealt with that online attack easily enough by getting their email accounts closed down. But they persisted with physical demonstrations and media harassment; our kids can set up other kids and denounce them. This leads to general issues of bullying and more specific issues of intimate partner abuse.

### 2.5.3 School and workplace bullying

Online harassment and bullying are a fact of life in modern societies, not just in schools but in workplaces too, as people jostle for rank, mates and resources. From the media stories of teens who kill themselves following online abuse, you might think that cyber-bullying now accounts for most of the problem – at least at school – but the figures show that it's less than half. An annual UK survey discloses that about a quarter of children and young people are constantly bullied (13% verbal, 5% cyber and 3% physical) while about half are bullied sometimes (24%, 8% and 9% respectively) [565]. The only national survey of all ages of which I'm aware is the French national victimisation survey, which since 2007 has collected data not just on physical crimes such as burglary and online crimes such as fraud, but on harassment too [1460]. This is based on face-to-face interviews with 16,000 households and the 2017 survey reported two million cases of threatening behaviour, 7% were made on social networks and a further 9% by phone. But have social media made this worse? Research suggests that the effects of social media use on adolescent well-being are nuanced, small at best, and contingent on analytic methods [1475].

Yet there is talk in the media of a rise in teen suicide which some commentators link to social media use. Thankfully, the OECD mortality statistics show that this is also untrue: suicides among 15–19 year olds have declined slightly from about 8 to about 7 cases per 100,000 over the period 1990–2015 [1479].

### 2.5.4 Intimate relationship abuse

Just as I ended the last section by discussing whistleblowers – the insider threat to companies – I'll end this section with intimate relationship abuse, the insider threat to families and individuals. Gamergate may have been a flashbulb example, but protection from former intimate partners and other family members is a real problem that exists at scale – with about half of all marriages ending in divorce, and not all breakups being amicable. Intimate partner abuse has been suffered by 27% of women and 11% of men. Stalking is not of course limited to former partners. Celebrities in particular can be stalked by people they've never met – with occasional tragic outcomes, as in the case of John Lennon. But former partners account for most of it, and law enforcement in most countries have historically been reluctant to do anything effective about them. Technology has made the victims' plight worse.

One subproblem is the publication of non-consensual intimate imagery (NCII), once called 'revenge porn' – until California Attorney General Kamala

Harris objected that this is cyber-exploitation and a crime. Her message got through to the big service firms who since 2015 have been taking down such material on demand from the victims [1693]. This followed an earlier report in 2012 where Harris documented the increasing use of smartphones, online marketplaces and social media in forcing vulnerable people into unregulated work including prostitution – raising broader questions about how technology can be used to connect with, and assist, crime victims [867].

The problems faced by a woman leaving an abusive and controlling husband are among the hardest in the universe of information security. All the usual advice is the wrong way round: your opponent knows not just your passwords but has such deep contextual knowledge that he can answer all your password recovery questions. There are typically three phases: a physical control phase where the abuser has access to your device and may install malware, or even destroy devices; a high-risk escape phase as you try to find a new home, a job and so on; and a life-apart phase when you might want to shield location, email address and phone numbers to escape harassment, and may have lifelong concerns. It takes seven escape attempts on average to get to life apart, and disconnecting from online services can cause other abuse to escalate. After escape, you may have to restrict childrens' online activities and sever mutual relationships; letting your child post anything can leak the school location and lead to the abuser turning up. You may have to change career as it can be impossible to work as a self-employed professional if you can no longer advertise.

To support such users, responsible designers should think hard about usability during times of high stress and high risk; they should allow users to have multiple accounts; they should design things so that someone reviewing your history should not be able to tell you deleted anything; they should push two-factor authentication, unusual activity notifications, and incognito mode. They should also think about how a survivor can capture evidence for use in divorce and custody cases and possibly in criminal prosecution, while minimising the trauma [1250]. But that's not what we find in real life. Many banks don't really want to know about disputes or financial exploitation within families. A big problem in some countries is stalkerware – apps designed to monitor partners, ex-partners, children or employees. A report from Citizen Lab spells out the poor information security practices of these apps, how they are marketed explicitly to abusive men, and how they break the law in Europe and Canada; as for the USA and Australia, over half of abusers tracked women using stalkerware [1497]. And then there's the Absher app, which enables men in Saudi Arabia to control their women in ways unacceptable in developed countries; its availability in app stores has led to protests against Apple and Google elsewhere in the world, but as of 2020 it's still there.

Intimate abuse is hard for designers and others to deal with as it's entangled with normal human caregiving between partners, between friends and colleagues, between parents and young children, and later between children and elderly parents. Many relationships are largely beneficent but with some abusive aspects, and participants often don't agree on which aspects. The best analysis I know, by Karen Levy and Bruce Schneier, discusses the combination of multiple motivations, copresence which leads to technical vulnerabilities, and power dynamics leading to relational vulnerabilities [1156]. Technology facilitates multiple privacy invasions in relationships, ranging from casual annoyance to serious crime; designers need to be aware that households are not units, devices are not personal, and the purchaser of a device is not the only user. I expect that concerns about intimate abuse will expand in the next few years to concerns about victims of abuse by friends, teachers and parents, and will be made ever more complex by new forms of home and school automation.

## 2.6 Summary

The systems you build or operate can be attacked by a wide range of opponents. It's important to work out who might attack you and how, and it's also important to be able to figure out how you were attacked and by whom. Your systems can also be used to attack others, and if you don't think about this in advance you may find yourself in serious legal or political trouble.

In this chapter I've grouped adversaries under four general themes: spies, crooks, hackers and bullies. Not all threat actors are bad: many hackers report bugs responsibly and many whistleblowers are public-spirited. ('Our' spies are of course considered good while 'theirs' are bad; moral valence depends on the public and private interests in play.) Intelligence and law enforcement agencies may use a mix of traffic data analysis and content sampling when hunting, and targeted collection for gathering; collection methods range from legal coercion via malware to deception. Both spies and crooks use malware to establish botnets as infrastructure. Crooks typically use opportunistic collection for mass attacks, while for targeted work, spear-phishing is the weapon of choice; the agencies may have fancier tools but use the same basic methods. There are also cybercrime ecosystems attached to specific business sectors; crime will evolve where it can scale. As for the swamp, the weapon of choice is the angry mob, wielded nowadays by states, activist groups and even individual orators. There are many ways in which abuse can scale, and when designing a system you need to work out how crimes against it, or abuse using it, might scale. It's not enough to think about usability; you need to think about abusability too.

Personal abuse matters too. Every police officer knows that the person who assaults you or murders you isn't usually a stranger, but someone you know – maybe another boy in your school class, or your stepfather. This has been ignored by the security research community, perhaps because we're mostly clever white or Asian boys from stable families in good neighbourhoods.

If you're defending a company of any size, you'll see enough machines on your network getting infected, and you need to know whether they're just zombies on a botnet or part of a targeted attack. So it's not enough to rely on patching and antivirus. You need to watch your network and keep good enough logs that when an infected machine is spotted you can tell whether it's a kid building a botnet or a targeted attacker who responds to loss of a viewpoint with a scramble to develop another one. You need to make plans to respond to incidents, so you know who to call for forensics – and so your CEO isn't left gasping like a landed fish in front of the TV cameras. You need to think systematically about your essential controls: backup to recover from ransomware, payment procedures to block business email compromise, and so on. If you're advising a large company they should have much of this already, and if it's a small company you need to help them figure out how to do enough of it.

The rest of this book will fill in the details.

## Research problems

Until recently, research on cybercrime wasn't really scientific. Someone would get some data – often under NDA from an anti-virus company – work out some statistics, write up their thesis, and then go get a job. The data were never available to anyone else who wanted to check their results or try a new type of analysis. Since 2015 we've been trying to fix that by setting up the Cambridge Cybercrime Centre, where we collect masses of data on spam, phish, botnets and malware as a shared resource for researchers. We're delighted for other academics to use it. If you want to do research on cybercrime, call us.

We also need something similar for espionage and cyber warfare. People trying to implant malware into control systems and other operational technology are quite likely to be either state actors, or cyber-arms vendors who sell to states. The criticisms made by President Eisenhower of the 'military-industrial complex' apply here in spades. Yet not one of the legacy think-tanks seems interested in tracking what's going on. As a result, nations are more likely to make strategic miscalculations, which could lead not just to cyber-conflict but the real kinetic variety, too.

As for research into cyber abuse, there is now some research, but the technologists, the psychologists, the criminologists and the political scientists aren't talking to each other enough. There are many issues, from the welfare and rights of children and young people, through the issues facing families separated by prison, to our ability to hold fair and free elections. We need to engage more technologists with public-policy issues and educate more policy people about the realities of technology. We also need to get more women involved, and people from poor and marginalised communities in both developed and less developed countries, so we have a less narrow perspective on what the real problems are.

## Further reading

There's an enormous literature on the topics discussed in this chapter but it's rather fragmented. A starting point for the Snowden revelations might be Glenn Greenwald's book *'No Place to Hide'* [817]; for an account of Russian strategy and tactics, see the 2018 report to the US Senate's Committee on Foreign Relations [387]; and for a great introduction to the history of propaganda see Tim Wu's *'The Attention Merchants'* [2052]. For surveys of cybercrime, see our 2012 paper "Measuring the Cost of Cybercrime" [91] and our 2019 follow-up "Measuring the Changing Cost of Cybercrime" [92]. Criminologists such as Bill Chambliss have studied state-organised crime, from piracy and slavery in previous centuries through the more recent smuggling of drugs and weapons by intelligence agencies to torture and assassination; this gives the broader context within which to assess unlawful surveillance. The story of Gamergate is told in Zoë Quinn's *'Crash Override'* [1570]. Finally, the tale of Marcus Hutchins, the malware expert who stopped Wannacry, is at [812].

would be easier than bad use. We have many examples in the physical world: a potato peeler is easier to use for peeling potatoes than a knife is, but a lot harder to use for murder. But we've not always got this right for computer systems yet. Much of the asymmetry between good and bad on which we rely in our daily business doesn't just depend on formal exchanges – which can be automated easily – but on some combination of physical objects, judgment of people, and the supporting social protocols. So, as our relationships with employers, banks and government become more formalised via online communication, and we lose both physical and human context, the forgery of these communications becomes more of a risk.

Deception, of various kinds, is now the principal mechanism used to defeat online security. It can be used to get passwords, to compromise confidential information or to manipulate financial transactions directly. Hoaxes and frauds have always happened, but the Internet makes some of them easier, and lets others be repackaged in ways that may bypass our existing controls (be they personal intuitions, company procedures or even laws).

Another driver for the surge in attacks based on social engineering is that people are getting better at technology. As designers learn how to forestall the easier technical attacks, psychological manipulation of system users or operators becomes ever more attractive. So the security engineer absolutely must understand basic psychology, as a prerequisite for dealing competently with everything from passwords to CAPTCHAs and from phishing to social engineering in general; a working appreciation of risk misperception and scaremongering is also necessary to understand the mechanisms underlying angry online mobs and the societal response to emergencies from terrorism to pandemic disease. So just as research in security economics led to a real shift in perspective between the first and second editions of this book, research in security psychology has made much of the difference to how we view the world between the second edition and this one.

In the rest of this chapter, I'll first survey relevant research in psychology, then work through how we apply the principles to make password authentication mechanisms more robust against attack, to security usability more generally, and beyond that to good design.

## 3.2 Insights from psychology research

Psychology is a huge subject, ranging from neuroscience through to clinical topics, and spilling over into cognate disciplines from philosophy through artificial intelligence to sociology. Although it has been studied for much longer than computer science, our understanding of the mind is much less complete: the brain is so much more complex. There's one central problem – the nature of consciousness – that we just don't understand at all. We know that 'the mind is what the brain does', yet the mechanisms that underlie our sense of self and of personal history remain obscure.

Nonetheless a huge amount is known about the functioning of the mind and the brain, and we're learning interesting new things all the time. In what follows I can only offer a helicopter tour of three of the themes in psychology research that are very relevant to our trade: cognitive psychology, which studies topics such as how we remember and what sort of mistakes we make; social psychology, which deals with how we relate to others in groups and to authority; and behavioral economics, which studies the heuristics and biases that lead us to make decisions that are consistently irrational in measurable and exploitable ways.

### 3.2.1 Cognitive psychology

Cognitive psychology is the classical approach to the subject – building on early empirical work in the nineteenth century. It deals with how we think, remember, make decisions and even daydream. Twentieth-century pioneers such as Ulric Neisser discovered that human memory doesn't work like a video recorder: our memories are stored in networks across the brain, from which they are reconstructed, so they change over time and can be manipulated [1429]. There are many well-known results. For example, it's easier to memorise things that are repeated frequently, and it's easier to store things in context. Many of these insights are used by marketers and scammers, but misunderstood or just ignored by most system developers.

For example, most of us have heard of George Miller's result that human short-term memory can cope with about seven (plus or minus two) simultaneous choices [1319] and, as a result, many designers limit menu choices to about five. But this is not the right conclusion. People search for information first by recalling where to look, and then by scanning; once you've found the relevant menu, scanning ten items is only twice as hard as scanning five. The real limits on menu size are screen size, which might give you ten choices, and with spoken menus, where the average user has difficulty dealing with more than three or four [1547]. Here, too, Miller's insight is misused because spatio-structural memory is a different faculty from echoic memory. This illustrates why a broad idea like 7+/-2 can be hazardous; you need to look at the detail.

In recent years, the centre of gravity in this field has been shifting from applied cognitive psychology to the human-computer interaction (HCI) research community, because of the huge amount of empirical know-how gained not just from lab experiments, but from the iterative improvement of fielded systems. As a result, HCI researchers not only model and measure human performance, including perception, motor control, memory and problem-solving; they have also developed an understanding of how users' mental models of systems work, how they differ from developers' mental models, and of the techniques (such as task analysis and cognitive walkthrough) that we can use to explore how people learn to use and understand systems.

Security researchers need to find ways of turning these ploughshares into swords (the bad guys are already working on it). There are some low-hanging fruit; for example, the safety research community has put a lot of effort into studying the errors people make when operating equipment [1592]. It's said that 'to err is human' and error research confirms this: the predictable varieties of human error are rooted in the very nature of cognition. The schemata, or mental models, that enable us to recognise people, sounds and concepts so much better than computers, also make us vulnerable when the wrong model gets activated.

Human errors made while operating equipment fall into broadly three categories, depending on where they occur in the 'stack': slips and lapses at the level of skill, mistakes at the level of rules, and misconceptions at the cognitive level.

- Actions performed often become a matter of skill, but we can slip when a manual skill fails – for example, pressing the wrong button – and we can also have a lapse where we use the wrong skill. For example, when you intend to go to the supermarket on the way home from work you may take the road home by mistake, if that's what you do most days (this is also known as a *capture error*). Slips are exploited by typosquatters, who register domains similar to popular ones, and harvest people who make typing errors; other attacks exploit the fact that people are trained to click 'OK' to pop-up boxes to get their work done. So when designing a system you need to ensure that dangerous actions, such as installing software, require action sequences that are quite different from routine ones. Errors also commonly follow interruptions and perceptual confusion. One example is the *post-completion error*: once they've accomplished their immediate goal, people are easily distracted from tidying-up actions. More people leave cards behind in ATMs that give them the money first and the card back second.
- Actions that people take by following rules are open to errors when they follow the wrong rule. Various circumstances – such as information overload – can cause people to follow the strongest rule they know, or the most general rule, rather than the best one. Phishermen use many tricks to get people to follow the wrong rule, ranging from using `https` (because 'it's secure') to starting URLs with the impersonated bank's name, as `www.citibank.secureauthentication.com` – for most people, looking for a name is a stronger rule than parsing its position.
- The third category of mistakes are those made by people for cognitive reasons – either they simply don't understand the problem, or pretend

> that they do, and ignore advice in order to get their work done. The seminal paper on security usability, Alma Whitten and Doug Tygar's "Why Johnny Can't Encrypt", demonstrated that the encryption program PGP was simply too hard for most college students to use as they didn't understand the subtleties of private versus public keys, encryption and signatures [2022]. And there's growing realisation that many security bugs occur because most programmers can't use security mechanisms either. Both access control mechanisms and security APIs are hard to understand and fiddly to use; security testing tools are often not much better. Programs often appear to work even when protection mechanisms are used in quite mistaken ways. Engineers then copy code from each other, and from online code-sharing sites, so misconceptions and errors are propagated widely [11]. They often know this is bad, but there's just not the time to do better.

There is some important science behind all this, and here are just two examples. James Gibson developed the concept of action possibilities or *affordances*: the physical environment may be climbable or fall-off-able or get-under-able for an animal, and similarly a seat is sit-on-able. People have developed great skill at creating environments that induce others to behave in certain ways: we build stairways and doorways, we make objects portable or graspable; we make pens and swords [763]. Often perceptions are made up of affordances, which can be more fundamental than value or meaning. In exactly the same way, we design software artefacts to train and condition our users' choices, so the affordances of the systems we use can affect how we think in all sorts of ways. We can also design traps for the unwary: an animal that mistakes a pitfall for solid ground is in trouble.

Gibson also came up with the idea of optical flows, further developed by Christopher Longuet-Higgins [1187]. As our eyes move relative to the environment, the resulting *optical flow field* lets us interpret the image, understanding the size, distance and motion of objects in it. There is an elegant mathematical theory of optical parallax, but our eyes deal with it differently: they contain receptors for specific aspects of this flow field which assume that objects in it are rigid, which then enables us to resolve rotational and translational components. Optical flows enable us to understand the shapes of objects around us, independently of binocular vision. We use them for some critical tasks such as landing an aeroplane and driving a car.

In short, cognitive science gives useful insights into how to design system interfaces so as to make certain courses of action easy, hard or impossible. It is increasingly tied up with research into computer human interaction. You can make mistakes more or less likely by making them easy or difficult; in section 28.2.2 I give real examples of usability failures causing serious accidents involving both medical devices and aircraft. Yet security can be even harder than safety if we have a sentient attacker who can provoke exploitable errors.

What can the defender expect attackers to do? They will use errors whose effect is predictable, such as capture errors; they will exploit perverse affordances; they will disrupt the flows on which safe operation relies; and they will look for, or create, exploitable dissonances between users' mental models of a system and its actual logic. To look for these, you should try a cognitive walkthrough aimed at identifying attack points, just as a code walkthough can be used to search for software vulnerabilities. Attackers also learn by experiment and share techniques with each other, and develop tools to look efficiently for known attacks. So it's important to be aware of the attacks that have already worked. (That's one of the functions of this book.)

### 3.2.2 Gender, diversity and interpersonal variation

Many women die because medical tests and technology assume that patients are men, or because engineers use male crash-test dummies when designing cars; protective equipment, from sportswear through stab-vests to spacesuits, gets tailored for men by default [498]. So do we have problems with information systems too? They are designed by men, and young geeky men at that, yet over half their users may be women. This realisation has led to research on *gender HCI* – on how software should be designed so that women can also use it effectively. Early experiments started from the study of behaviour: experiments showed that women use peripheral vision more, and it duly turned out that larger displays reduce gender bias. Work on American female programmers suggested that they tinker less than males, but more effectively [203]. But how much is nature, and how much nurture? Societal factors matter, and US women who program appear to be more thoughtful, but lower self-esteem and higher risk-aversion leads them to use fewer features.

Gender has become a controversial topic in psychology research. In the early 2000s, discussion of male aptitude for computer science was sometimes in terms of an analysis by Simon Baron-Cohen which gives people separate scores as systemisers (good at geometry and some kinds of symbolic reasoning) and as empathisers (good at intuiting the emotions of others and social intelligence generally) [177]. Most men score higher at systematising, while most women do better at empathising. The correspondence isn't exact; a minority of men are better at empathising while a minority of women are better at systematising. Baron-Cohen's research is in Asperger's and autism spectrum disorder, which he sees as an extreme form of male brain. This theory gained some traction among geeks who saw an explanation of why we're often introverted with more aptitude for understanding things than for understanding people. If we're born that way, it's not out fault. It also suggests an explanation for why geek couples often have kids on the spectrum.

the self-esteem we get by comparing ourselves with others. The results that put it on the map were three early papers that laid the groundwork for understanding the abuse of authority and its relevance to propaganda, interrogation and aggression. They were closely followed by work on the bystander effect which is also highly relevant to crime and security.

#### *3.2.3.1 Authority and its abuse*

In 1951, Solomon Asch showed that people could be induced to deny the evidence of their own eyes in order to conform to a group. Subjects judged the lengths of lines after hearing wrong opinions from other group members, who were actually the experimenter's stooges. Most subjects gave in and conformed, with only 29% resisting the bogus majority [136].

Stanley Milgram was inspired by the 1961 trial of Nazi war criminal Adolf Eichmann to investigate how many experimental subjects were prepared to administer severe electric shocks to an actor playing the role of a 'learner' at the behest of an experimenter while the subject played the role of the 'teacher' – even when the 'learner' appeared to be in severe pain and begged the subject to stop. This experiment was designed to measure what proportion of people will obey an authority rather than their conscience. Most did – Milgram found that consistently over 60% of subjects would do downright immoral things if they were told to [1314]. This experiment is now controversial but had real influence on the development of the subject.

The third was the Stanford Prisoner Experiment which showed that normal people can behave wickedly even in the absence of orders. In 1971, experimenter Philip Zimbardo set up a 'prison' at Stanford where 24 students were assigned at random to the roles of 12 warders and 12 inmates. The aim of the experiment was to discover whether prison abuses occurred because warders (and possibly prisoners) were self-selecting. However, the students playing the role of warders rapidly became sadistic authoritarians, and the experiment was halted after six days on ethical grounds [2076]. This experiment is also controversial now and it's unlikely that a repeat would get ethical approval today. But abuse of authority, whether real or ostensible, is a real issue if you are designing operational security measures for a business.

During the period 1995–2005, a telephone hoaxer calling himself 'Officer Scott' ordered the managers of over 68 US stores and restaurants in 32 US states (including at least 17 McDonald's stores) to detain some young employee on suspicion of theft and strip-search them. Various other degradations were ordered, including beatings and sexual assaults [2036]. A former prison guard was tried for impersonating a police officer but acquitted. At least 13 people who obeyed the caller and did searches were charged with crimes, and seven were convicted. McDonald's got sued for not training its store managers properly, even years after the pattern of hoax calls was established; and in October 2007, a jury ordered them to pay $6.1 million dollars to one of the victims, who had been strip-searched when she was an 18-year-old employee. It was a nasty case, as she was left by the store manager in the custody of her boyfriend, who then committed a further indecent assault on her. The boyfriend got five years, and the manager pleaded guilty to unlawfully detaining her. McDonald's argued that she was responsible for whatever damages she suffered for not realizing it was a hoax, and that the store manager had failed to apply common sense. A Kentucky jury didn't buy this and ordered McDonald's to pay up. The store manager also sued, claiming to be another victim of the firm's negligence to warn her of the hoax, and got $1.1 million [1090]. So US employers now risk heavy damages if they fail to train their staff to resist the abuse of authority.

#### *3.2.3.2 The bystander effect*

On March 13, 1964, a young lady called Kitty Genovese was stabbed to death in the street outside her apartment in Queens, New York. The press reported that thirty-eight separate witnesses had failed to help or even to call the police, although the assault lasted almost half an hour. Although these reports were later found to be exaggerated, the crime led to the nationwide 911 emergency number, and also to research on why bystanders often don't get involved.

John Darley and Bibb Latané reported experiments in 1968 on what factors modulated the probability of a bystander helping someone who appeared to be having an epileptic fit. They found that a lone bystander would help 85% of the time, while someone who thought that four other people could see the victim would help only 31% of the time; group size dominated all other effects. Whether another bystander was male, female or even medically qualified made essentially no difference [513]. The diffusion of responsibility has visible effects in many other contexts. If you want something done, you'll email one person to ask, not three people. Of course, security is usually seen as something that other people deal with.

However, if you ever find yourself in danger, the real question is whether at least one of the bystanders will help, and here the recent research is much more positive. Lasse Liebst, Mark Levine and others have surveyed CCTV footage of a number of public conflicts in several countries over the last ten years, finding that in 9 out of 10 cases, one or more bystanders intervened to de-escalate a fight, and that the more bystanders intervene, the more successful they are [1166]. So it would be wrong to assume that bystanders generally pass by on the other side; so the bystander effect's name is rather misleading.

### 3.2.4 The social-brain theory of deception

Our second big theme, which also fits into social psychology, is the growing body of research into deception. How does deception work, how can we detect and measure it, and how can we deter it?

The modern approach started in 1976 with the social intelligence hypothesis. Until then, anthropologists had assumed that we evolved larger brains in order to make better tools. But the archaeological evidence doesn't support this. All through the paleolithic period, while our brains evolved from chimp size to human size, we used the same simple stone axes. They only became more sophisticated in the neolithic period, by which time our ancestors were anatomically modern homo sapiens. So why, asked Nick Humphrey, did we evolve large brains if we didn't need them yet? Inspired by observing the behaviour of both caged and wild primates, his hypothesis was that the primary function of the intellect was social. Our ancestors didn't evolve bigger brains to make better tools, but to use other primates better as tools [936]. This is now supported by a growing body of evidence, and has transformed psychology as a discipline. Social psychology had been a poor country cousin until then and was not seen as rigorous; since then, people have realised it was probably the driving force of cognitive evolution. Almost all intelligent species developed in a social context. (One exception is the octopus, but even it has to understand how predators and prey react.)

The primatologist Andy Whiten then collected much of the early evidence on tactical deception, and recast social intelligence as the Machiavellian brain hypothesis: we became smart in order to deceive others, and to detect deception too [362]. Not everyone agrees completely with this characterisation, as the positive aspects of socialisation, such as empathy, also matter. But Hugo Mercier and Dan Sperber have recently collected masses of evidence that the modern human brain is more a machine for arguing than anything else [1296]. Our goal is persuasion rather than truth; rhetoric comes first, and logic second.

The second thread coming from the social intellect hypothesis is theory of mind, an idea due to David Premack and Guy Woodruff in 1978 but developed by Heinz Wimmer and Josef Perner in a classic 1983 experiment to determine when children are first able to tell that someone has been deceived [2032]. In this experiment, the Sally-Anne test, a child sees a sweet hidden under a cup by Sally while Anne and the child watch. Anne then leaves the room and Sally switches the sweet to be under a different cup. Anne then comes back and the child is asked where Anne thinks the sweet is. Normal children get the right answer from about age five; this is when they acquire the ability to discern others' beliefs and intentions. Simon Baron-Cohen, Alan Leslie and Uta Frith then showed that children on the Aspergers / autism spectrum acquire this ability significantly later [178].

Many computer scientists and engineers appear to be on the spectrum to some extent, and we're generally not as good at deception as neurotypical people are. This has all sorts of implications! We're under-represented in politics, among senior executives and in marketing. Oh, and there was a lot less cybercrime before underground markets brought together geeks who could write wicked code with criminals who could use it for wicked purposes. Geeks are also more likely to be whistleblowers; we're less likely to keep quiet about an uncomfortable truth just to please others, as we place less value on their opinions. But this is a complex field. Some well-known online miscreants who are on the spectrum were hapless more than anything else; Gary McKinnon claimed to have hacked the Pentagon to discover the truth about flying saucers and didn't anticipate the ferocity of the FBI's response. And other kinds of empathic deficit are involved in many crimes. Other people with dispositional empathy deficits include psychopaths who disregard the feelings of others but understand them well enough to manipulate them, while there are many people whose deficits are situational, ranging from Nigerian scammers who think that any white person who falls for their lure must think Africans are stupid, so they deserve it, right through to soldiers and terrorists who consider their opponents to be less than human or to be morally deserving of death. I'll discuss radicalisation in more detail later in section 26.4.2.

The third thread is self-deception. Robert Trivers argues that we've evolved the ability to deceive ourselves in order to better deceive others: "If deceit is fundamental in animal communication, then there must be strong selection to spot deception and this ought, in turn, to select for a degree of self-deception, rendering some facts and motives unconscious so as to not betray – by the subtle signs of self-knowledge – the deception being practiced" [906]. We forget inconvenient truths and rationalise things we want to believe. There may well be a range of self-deception abilities from honest geeks through to the great salesmen who have a magic ability to believe completely in their product. But it's controversial, and at a number of levels. For example, if Tony Blair really believed that Iraq had weapons of mass destruction when he persuaded Britain to go to war in 2003, was it actually a lie? How do you define sincerity? How can you measure it? And would you even elect a national leader if you expected that they'd be unable to lie to you? There is a lengthy discussion in [906], and the debate is linked to other work on motivated reasoning. Russell Golman, David Hagman and George Loewenstein survey research on how people avoid information, even when it is free and could lead to better decision-making: people at risk of illness avoid medical tests, managers avoid information that might show they made bad decisions, and investors look at their portfolios less when markets are down [782]. This strand of research goes all the way back to Sigmund Freud, who described various aspects of the *denial* of unpleasant information, including the ways in which we try to minimise our feelings of guilt for the bad things we do, and to blame others for them.

It also links up with filter-bubble effects on social media. People prefer to listen to others who confirm their beliefs and biases, and this can be analysed in terms of the hedonic value of information. People think of themselves as honest and try to avoid the *ethical dissonance* that results from deviations [173]; criminologists use the term *neutralisation* to describe the strategies that rule-breakers use to minimise the guilt that they feel about their actions (there's an overlap with both filter effects and self-deception). A further link is to Hugo Mercier and Dan Sperber's work on the brain as a machine for argument, which I mentioned above.

The fourth thread is intent. The detection of hostile intent was a big deal in our ancestral evolutionary environment; in pre-state societies, perhaps a quarter of men and boys die of homicide, and further back many of our ancestors were killed by animal predators. So we appear to have evolved a sensitivity to sounds and movements that might signal the intent of a person, an animal or even a god. As a result, we now spend too much on defending against threats that involve hostile intent, such as terrorism, and not enough on defending against epidemic disease, which kills many more people – or climate change, which could kill even more.

There are other reasons why we might want to think about intent more carefully. In cryptography, we use logics of belief to analyse the security of authentication protocols, and to deal with statements such as 'Alice believes that Bob believes that Charlie controls the key *K*'; we'll come to this in the next chapter. And now we realise that people use theories of mind to understand each other, philosophers have got engaged too. Dan Dennett derived the intentional stance in philosophy, arguing that the propositional attitudes we use when reasoning – beliefs, desires and perceptions – come down to the intentions of people and animals.

A related matter is socially-motivated reasoning: people do logic much better if the problem is set in a social role. In the Wason test, subjects are told they have to inspect some cards with a letter grade on one side, and a numerical code on the other, and given a rule such as "If a student has a grade D on the front of their card, then the back must be marked with code 3". They are shown four cards displaying (say) D, F, 3 and 7 and then asked "Which cards do you have to turn over to check that all cards are marked correctly?" Most subjects get this wrong; in the original experiment, only 48% of 96 subjects got the right answer of D and 7. However the evolutionary psychologists Leda Cosmides and John Tooby found the same problem becomes easier if the rule is changed to 'If a person is drinking beer, he must be 20 years old' and the individuals are a beer drinker, a coke drinker, a 25-year-old and a 16-year old. Now three-quarters of subjects deduce that the bouncer should check the age of the beer drinker and the drink of the 16-year-old [483]. Cosmides and Tooby argue that our ability to do logic and perhaps arithmetic evolved as a means of policing social exchanges.

The misperception of risk underlies many other public-policy problems. The psychologist Daniel Gilbert, in an article provocatively entitled 'If only gay sex caused global warming', compares our fear of terrorism with our fear of climate change. First, we evolved to be much more wary of hostile intent than of nature; 100,000 years ago, a man with a club (or a hungry lion) was a much worse threat than a thunderstorm. Second, global warming doesn't violate anyone's moral sensibilities; third, it's a long-term threat rather than a clear and present danger; and fourth, we're sensitive to rapid changes in the environment rather than slow ones [765]. There are many more risk biases: we are less afraid when we're in control, such as when driving a car, as opposed to being a passenger in a car or airplane; and we are more afraid of uncertainty, that is, when the magnitude of the risk is unknown (even when it's small) [1674, 1678]. We also indulge in *satisficing* which means we go for an alternative that's 'good enough' rather than going to the trouble of trying to work out the odds perfectly, especially for small transactions. (The misperception here is not that of the risk taker, but of the economists who ignored the fact that real people include transaction costs in their calculations.)

So, starting out from the folk saying that a bird in the hand is worth two in the bush, we can develop quite a lot of machinery to help us understand and model people's attitudes towards risk.

#### 3.2.5.2 Present bias and hyperbolic discounting

Saint Augustine famously prayed 'Lord, make me chaste, but not yet.' We find a similar sentiment with applying security updates, where people may pay more attention to the costs as they're immediate and determinate in time, storage and bandwidth, than the unpredictable future benefits. This *present bias* causes many people to decline updates, which was the major source of technical vulnerability online for many years. One way software companies pushed back was by allowing people to delay updates: Windows has 'restart / pick a time / snooze'. Reminders cut the ignore rate from about 90% to about 34%, and may ultimately double overall compliance [726]. A better design is to make updates so painless that they can be made mandatory, or nearly so; this is the approach now followed by some web browsers, and by cloud-based services generally.

*Hyperbolic discounting* is a model used by decision scientists to quantify present bias. Intuitive reasoning may lead people to use utility functions that discount the future so deeply that immediate gratification seems to be the best course of action, even when it isn't. Such models have been applied to try to explain the *privacy paradox* – why people say in surveys that they care about privacy but act otherwise online. I discuss this in more detail in section 8.67: other factors, such as uncertainty about the risks and about the efficacy of privacy measures, play a part too. Taken together, the immediate and determinate positive utility of getting free stuff outweighs the random future costs of disclosing too much personal information, or disclosing it to dubious websites.

### *3.2.5.3 Defaults and nudges*

This leads to the importance of defaults. Many people usually take the easiest path and use the standard configuration of a system, as they assume it will be good enough. In 2009, Richard Thaler and Cass Sunstein wrote a bestseller '*Nudge*' exploring this, pointing out that governments can achieve many policy goals without infringing personal liberty simply by setting the right defaults [1879]. For example, if a firm's staff are enrolled in a pension plan by default, most will not bother to opt out, while if it's optional most will not bother to opt in. A second example is that many more organs are made available for transplant in Spain, where the law lets a dead person's organs be used unless they objected, than in Britain where donors have to consent actively. A third example is that tax evasion can be cut by having the taxpayer declare that the information in the form is true when they start to fill it out, rather than at the end. The set of choices people have to make, the order in which they make them, and the defaults if they do nothing, are called the *choice architecture*. Sunnstein got a job in the Obama administration implementing some of these ideas while Thaler won the 2017 economics Nobel prize.

Defaults matter in security too, but often they are set by an adversary so as to trip you up. For example, Facebook defaults to fairly open information sharing, and whenever enough people have figured out how to increase their privacy settings, the architecture is changed so you have to opt out all over again. This exploits not just hazardous defaults but also the *control paradox* – providing the illusion of control causes people to share more information. We like to feel in control; we feel more comfortable driving in our cars than letting someone else fly us in an airplane – even if the latter is an order of magnitude safer. "Privacy control settings give people more rope to hang themselves," as behavioral economist George Loewenstein puts it. "Facebook has figured this out, so they give you incredibly granular controls." [1536]

### *3.2.5.4 The default to intentionality*

Behavioral economists follow a long tradition in psychology of seeing the mind as composed of interacting rational and emotional components – 'heart' and 'head', or 'affective' and 'cognitive' systems. Studies of developmental biology have shown that, from an early age, we have different mental processing systems for social phenomena (such as recognising parents and siblings) and physical phenomena. Paul Bloom argues that the tension between them explains why many people believe that mind and body are basically different [269]. Children try to explain what they see using physics, but when their understanding falls short, they explain phenomena in terms of intentional action. This has survival value to the young, as it disposes them to get advice from parents or other adults about novel natural phenomena. Bloom suggests that it has an interesting side effect: it predisposes humans to believe that body and soul are different, and thus lays the ground for religious belief. This argument may not overwhelm the faithful (who will retort that Bloom simply stumbled across a mechanism created by the Intelligent Designer to cause us to have faith in Him). But it may have relevance for the security engineer.

First, it goes some way to explaining the *fundamental attribution error* – people often err by trying to explain things from intentionality rather than from context. Second, attempts to curb phishing by teaching users about the gory design details of the Internet – for example, by telling them to parse URLs in emails that seem to come from a bank – will be of limited value once they get bewildered. If the emotional is programmed to take over whenever the rational runs out, then engaging in a war of technical instruction and counter-instruction with the phishermen is unsound, as they'll be better at it. Safe defaults would be better.

#### 3.2.5.5 The affect heuristic

Nudging people to think in terms of intent rather than of mechanism can exploit the *affect heuristic*, explored by Paul Slovic and colleagues [1791]. The idea is that while the human brain can handle multiple threads of cognitive processing, our emotions remain resolutely single-threaded, and they are even less good at probability theory than the rational part of our brains. So by making emotion salient, a marketer or a fraudster can try to get you to answer questions using emotion rather than reason, and using heuristics rather than calculation. A common trick is to ask an emotional question (whether 'How many dates did you have last month?' or even 'What do you think of President Trump?') to make people insensitive to probability.

So it should not surprise anyone that porn websites have been used to install a lot of malware – as have church websites, which are often poorly maintained and easy to hack. Similarly, events that evoke a feeling of dread – from cancer to terrorism – not only scare people more than the naked probabilities justify, but also make those probabilities harder to calculate, and deter people from even making the effort.

Other factors that can reinforce our tendency to explain things by intent include cognitive overload, where the rational part of the brain simply gets tired. Our capacity for self-control is also liable to fatigue, both physical and mental; some mental arithmetic will increase the probability that we'll pick up a chocolate rather than an apple. So a bank that builds a busy website may be able to sell more life insurance, but it's also likely to make its customers more vulnerable to phishing.

#### *3.2.5.6 Cognitive dissonance*

Another interesting offshoot of social psychology is cognitive dissonance theory. People are uncomfortable when they hold conflicting views; they seek out information that confirms their existing views of the world and of themselves, and try to reject information that conflicts with their views or might undermine their self-esteem. One practical consequence is that people are remarkably able to persist in wrong courses of action in the face of mounting evidence that things have gone wrong [1866]. Admitting to yourself or to others that you were duped can be painful; hustlers know this and exploit it. A security professional should 'feel the hustle' – that is, be alert for a situation in which recently established social cues and expectations place you under pressure to 'just do' something about which you'd normally have reservations. That's the time to step back and ask yourself whether you're being had. But training people to perceive this is hard enough, and getting the average person to break the social flow and say 'stop!' is hard. There have been some experiments, for example with training health-service staff to not give out health information on the phone, and training people in women's self-defence classes to resist demands for extra personal information. The problem with mainstreaming such training is that the money available for it is orders of magnitude less than the marketing budgets of the firms whose business model is to hustle their customers.

#### *3.2.5.7 The risk thermostat*

Some interesting empirical work has been done on how people manage their exposure to risk. John Adams studied mandatory seat belt laws, and established that they don't actually save lives: they just transfer casualties from vehicle occupants to pedestrians and cyclists [20]. Seat belts make drivers feel safer, so they drive faster in order to bring their perceived risk back up to its previous level. He calls this a *risk thermostat* and the model is borne out in other applications too [19]. The lesson is that testing needs to have ecological validity: you need to evaluate the effect of a proposed intervention in as realistic a setting as possible.

## 3.3 Deception in practice

This takes us from the theory to the practice. Deception often involves an abuse of the techniques developed by *compliance professionals* – those people whose job it is to get other people to do things. While a sales executive might dazzle you with an offer of a finance plan for a holiday apartment, a police officer might nudge you by their presence to drive more carefully, a park ranger might tell you to extinguish campfires carefully and not feed the bears, and a corporate lawyer might threaten you into taking down something from your website.

The behavioural economics pioneer and apostle of ‘nudge’, Dick Thaler, refers to the selfish use of behavioural economics as ‘sludge’ [1878]. But it’s odd that economists ever thought that the altruistic use of such techniques would ever be more common than the selfish ones. Not only do marketers push the most profitable option rather than the best value, but they use every other available trick too. Stanford’s Persuasive Technology Lab has been at the forefront of developing techniques to keep people addicted to their screens, and one of their alumni, ex-Googler Tristan Harris, has become a vocal critic. Sometimes dubbed ‘Silicon valley’s conscience’, he explains how tech earns its money by manipulating not just defaults but choices, and asks how this can be done ethically [868]. Phones and other screens present menus and thus control choices, but there’s more to it than that. Two techniques that screens have made mainstream are the casino’s technique of using intermittent variable rewards to create addiction (we check our phones 150 times a day to see if someone has rewarded us with attention) and bottomless message feeds (to keep us consuming even when we aren’t hungry any more). But there are many older techniques that predate computers.

### 3.3.1 The salesman and the scamster

Deception is the twin brother of marketing, so one starting point is the huge literature about sales techniques. One eminent writer is Robert Cialdini, a psychology professor who took summer jobs selling everything from used cars to home improvements and life insurance in order to document the tricks of the trade. His book ‘*Influence: Science and Practice*’ is widely read by sales professionals and describes six main classes of technique used to influence people and close a sale [426].

These are:

1. Reciprocity: most people feel the need to return favours;
2. Commitment and consistency: people suffer cognitive dissonance if they feel they’re being inconsistent;
3. Social proof: most people want the approval of others. This means following others in a group of which they’re a member, and the smaller the group the stronger the pressure;
4. Liking: most people want to do what a good-looking or otherwise likeable person asks;

Investigative journalists, private detectives and fraudsters developed the false-pretext phone call into something between an industrial process and an art form in the latter half of the 20th century. An example of the industrial process was how private detectives tracked people in Britain. Given that the country has a National Health Service with which everyone's registered, the trick was to phone up someone with access to the administrative systems in the area you thought the target was, pretend to be someone else in the health service, and ask. Colleagues of mine did an experiment in England in 1996 where they trained the staff at a local health authority to identify and report such calls[1]. They detected about 30 false-pretext calls a week, which would scale to 6000 a week or 300,000 a year for the whole of Britain. That eventually got sort-of fixed but it took over a decade. The real fix wasn't the enforcement of privacy law, but that administrators simply stopped answering the phone.

Another old scam from the 20th century is to steal someone's ATM card and then phone them up pretending to be from the bank asking whether their card's been stolen. On hearing that it has, the conman says 'We thought so. Please just tell me your PIN now so I can go into the system and cancel your card.' The most rapidly growing recent variety is the 'authorised push payment', where the conman again pretends to be from the bank, and persuades the customer to make a transfer to another account, typically by confusing the customer about the bank's authentication procedures, which most customers find rather mysterious anyway[2].

As for art form, one of the most disturbing security books ever published is Kevin Mitnick's *'Art of Deception'*. Mitnick, who was arrested and convicted for breaking into US phone systems, related after his release from prison how almost all of his exploits had involved social engineering. His typical hack was to pretend to a phone company employee that he was a colleague, and solicit 'help' such as a password. Ways of getting past a company's switchboard and winning its people's trust are a staple of sales-training courses, and hackers apply these directly. A harassed system administrator is called once or twice on trivial matters by someone claiming to be the CEO's personal assistant; once this idea has been accepted, the caller demands a new password for the boss. Mitnick became an expert at using such tricks to defeat company security procedures, and his book recounts a fascinating range of exploits [1327].

Social engineering became world headline news in September 2006 when it emerged that Hewlett-Packard chairwoman Patricia Dunn had hired private investigators who used pretexting to obtain the phone records of other board members of whom she was suspicious, and of journalists she considered hostile. She was forced to resign. The detectives were convicted of fraudulent wire communications and sentenced to do community service [139]. In the same

[1] The story is told in detail in chapter 9 of the second edition of this book, available free online.

[2] Very occasionally, a customer can confuse the bank; a 2019 innovation was the 'callhammer' attack, where someone phones up repeatedly to 'correct' the spelling of 'his name' and changes it one character at a time into another one.

year, the UK privacy authorities prosecuted a private detective agency that did pretexting jobs for top law firms [1140].

Amid growing publicity about social engineering, there was an audit of the IRS in 2007 by the Treasury Inspector General for Tax Administration, whose staff called 102 IRS employees at all levels, asked for their user IDs, and told them to change their passwords to a known value; 62 did so. What's worse, this happened despite similar audit tests in 2001 and 2004 [1676]. Since then, a number of audit firms have offered social engineering as a service; they phish their audit clients to show how easy it is. Since the mid-2010s, opinion has shifted against this practice, as it causes a lot of distress to staff without changing behaviour very much.

Social engineering isn't limited to stealing private information. It can also be about getting people to believe bogus public information. The quote from Bruce Schneier at the head of this chapter appeared in a report of a stock scam, where a bogus press release said that a company's CEO had resigned and its earnings would be restated. Several wire services passed this on, and the stock dropped 61% until the hoax was exposed [1673]. Fake news of this kind has been around forever, but the Internet has made it easier to promote and social media seem to be making it ubiquitous. We'll revisit this issue when I discuss censorship in section 26.4.

### 3.3.3 Phishing

While phone-based social engineering was the favoured tactic of the 20th century, online phishing seems to have replaced it as the main tactic of the 21st. The operators include both criminals and intelligence agencies, while the targets are both your staff and your customers. It is difficult enough to train your staff; training the average customer is even harder. They'll assume you're trying to hustle them, ignore your warnings and just figure out the easiest way to get what they want from your system. And you can't design simply for the average. If your systems are not safe to use by people who don't speak English well, or who are dyslexic, or who have learning difficulties, you are asking for serious legal trouble. So the easiest way to use your system had better be the safest.

The word 'phishing' appeared in 1996 in the context of the theft of AOL passwords. By then, attempts to crack email accounts to send spam had become common enough for AOL to have a 'report password solicitation' button on its web page; and the first reference to 'password fishing' is in 1990, in the context of people altering terminal firmware to collect Unix logon passwords [445]. Also in 1996, Tony Greening reported a systematic experimental study: 336 computer science students at the University of Sydney were sent an email message asking them to supply their password on the pretext that it was required to 'validate' the password database after a suspected break-in. 138 of them returned a valid password. Some were suspicious: 30 returned a plausible looking but invalid password, while over 200 changed their passwords without official prompting. But very few of them reported the email to authority [813].

Phishing attacks against banks started seven years later in 2003, with half-a-dozen attempts reported [443]. The early attacks imitated bank websites, but were both crude and greedy; the attackers asked for all sorts of information such as ATM PINs, and their emails were also written in poor English. Most customers smelt a rat. By about 2008, the attackers learned to use better psychology; they often reused genuine bank emails, with just the URLs changed, or sent an email saying something like 'Thank you for adding a new email address to your PayPal account' to provoke the customer to log on to complain that they hadn't. Of course, customers who used the provided link rather than typing in `www.paypal.com` or using an existing bookmark would get their accounts emptied. By then phishing was being used by state actors too; I described in section 2.2.2 how Chinese intelligence compromised the Dalai Lama's private office during the 2008 Olympic games. They used crimeware tools that were originally used by Russian fraud gangs, which they seemed to think gave them some deniability afterwards.

Fraud losses grew rapidly but stabilised by about 2015. A number of countermeasures helped bring things under control, including more complex logon schemes (using two-factor authentication, or its low-cost cousin, the request for some random letters of your password); a move to webmail systems that filter spam better; and back-end fraud engines that look for cashout patterns. The competitive landscape was rough, in that the phishermen would hit the easiest targets at any time in each country, both in terms of stealing their customer credentials and using their accounts to launder stolen funds. Concentrated losses caused the targets to wake up and take action. Since then, we've seen large-scale attacks on non-financial firms like Amazon; in the late 2000s, the crook would change your email and street address, then use your credit card to order a wide-screen TV. Since about 2016, the action has been in gift vouchers.

As we noted in the last chapter, phishing is also used at scale by botmasters to recruit new machines to their botnets, and in targeted ways both by crooks aiming at specific people or firms, and by intelligence agencies. There's a big difference between attacks conducted at scale, where the economics dictate that the cost of recruiting a new machine to a botnet can be at most a few cents, and targeted attacks, where spies can spend years trying to hack the phone of a rival head of government, or a fraudster can spend weeks or months of effort stalking a chief financial officer in the hope of a large payout. The lures and techniques used are different, even if the crimeware installed on the target's laptop or phone comes from the same stable. Cormac Herley argues that this gulf between the economics of targeted crime and volume crime is one of the reasons why cybercrime isn't much worse than it is [889]. After all, given that we depend on computers, and that all computers are insecure, and that there are attacks all the time, how come civilisation hasn't collapsed? Cybercrime can't always be as easy as it looks.

Another factor is that it takes time for innovations to be developed and disseminated. We noted that it took seven years for the bad guys to catch up with Tony Greening's 1995 phishing work. As another example, a 2007 paper by Tom Jagatic and colleagues showed how to make phishing much more effective by automatically personalising each phish using context mined from the target's social network [973]. I cited that in the second edition of this book, and in 2016 we saw it in the wild: a gang sent hundreds of thousands of phish with US and Australian banking Trojans to individuals working in finance departments of companies, with their names and job titles apparently scraped from LinkedIn [1299]. This seems to have been crude and hasn't really caught on, but once the bad guys figure it out we may see spear-phishing at scale in the future, and it's interesting to think of how we might respond. The other personalised bulk scams we see are blackmail attempts where the victims get email claiming that their personal information has been compromised and including a password or the last four digits of a credit card number as evidence, but the yield from such scams seems to be low.

As I write, crime gangs have been making ever more use of spear-phishing in targeted attacks on companies where they install ransomware, steal gift coupons and launch other scams. In 2020, a group of young men hacked Twitter, where over a thousand employees had access to internal tools that enabled them to take control of user accounts; the gang sent bitcoin scam tweets from the accounts of such well-known users as Bill Gates, Barack Obama and Elon Musk [1294]. They appear to have honed their spear-phishing skills on SIM swap fraud, which I'll discuss later in sections 3.4.1 and 12.7.4. The spread of such 'transferable skills' among crooks is similar in many ways to the adoption of mainstream technology.

### 3.3.4 Opsec

Getting your staff to resist attempts by outsiders to inveigle them into revealing secrets, whether over the phone or online, is known in military circles as *operational security* or opsec. Protecting really valuable secrets, such as unpublished financial data, not-yet-patented industrial research and military plans, depends on limiting the number of people with access, and also on doctrines about what may be discussed with whom and how. It's not enough for rules to exist; you have to train the staff who have access, explain the reasons behind the rules, and embed them socially in the organisation. In our medical privacy case, we educated health service staff about pretext calls and set up a strict callback policy: they would not discuss medical records on the phone unless they had called a number they had got from the health service internal phone book rather than from a caller. Once the staff have detected and defeated a few false-pretext calls, they talk about it and the message gets embedded in the way everybody works.

Another example comes from a large Silicon Valley service firm, which suffered intrusion attempts when outsiders tailgated staff into buildings on campus. Stopping this with airport-style ID checks, or even card-activated turnstiles, would have changed the ambience and clashed with the culture. The solution was to create and embed a social rule that when someone holds open a building door for you, you show them your badge. The critical factor, as with the bogus phone calls, is social embedding rather than just training. Often the hardest people to educate are the most senior; in my own experience in banking, the people you couldn't train were those who were paid more than you, such as traders in the dealing rooms. The service firm in question did better, as its CEO repeatedly stressed the need to stop tailgating at all-hands meetings.

Some opsec measures are common sense, such as not throwing sensitive papers in the trash, or leaving them on desks overnight. (One bank at which I worked had the cleaners move all such papers to the departmental manager's desk.) Less obvious is the need to train the people you trust. A leak of embarrassing emails that appeared to come from the office of UK Prime Minister Tony Blair and was initially blamed on 'hackers' turned out to have been fished out of the trash at his personal pollster's home by a private detective [1210].

People operate systems however they have to, and this usually means breaking some of the rules in order to get their work done. Research shows that company staff have only so much *compliance budget*, that is, they're only prepared to put so many hours a year into tasks that are not obviously helping them achieve their goals [197]. You need to figure out what this budget is, and use it wisely. If there's some information you don't want your staff to be tricked into disclosing, it's safer to design systems so that they just can't disclose it, or at least so that disclosures involve talking to other staff members or jumping through other hoops.

But what about a firm's customers? There is a lot of scope for phishermen to simply order bank customers to reveal their security data, and this happens at scale, against both retail and business customers. There are also the many small scams that customers try on when they find vulnerabilities in your business processes. I'll discuss both types of fraud further in the chapter on banking and bookkeeping.

### 3.3.5 Deception research

Finally, a word on deception research. Since 9/11, huge amounts of money have been spent by governments trying to find better lie detectors, and deception

Google knows I'm in Scotland, they send an SMS to my phone to check, and a small website can't do that. The main cause of attempted password abuse is when one firm gets hacked, disclosing millions of email addresses and passwords, which the bad guys try out elsewhere; big firms spot this quickly while small ones don't. The big firms also help their customers maintain situational awareness, by alerting you to logons from new devices or from strange places. Again, it's hard to do that if you're a small website or one that people visit infrequently.

As for syncing passwords between devices, only the device vendors can really do that well; and the protocol mechanisms for encrypting passwords in transit to a server that verifies them will be discussed in the next chapter. That brings us to password recovery.

### 3.4.1 Password recovery

The experience of the 2010s, as the large service firms scaled up and people moved en masse to smartphones, is that password recovery is often the hardest aspect of authentication. If people you know, such as your staff, forget their passwords, you can get them to interact with an administrator or manager who knows them. But for people you don't know such as your online customers it's harder. And as a large service firm will be recovering tens of thousands of accounts every day, you need some way of doing it without human intervention in the vast majority of cases.

During the 1990s and 2000s, many websites did password recovery using 'security questions' such as asking for your favourite team, the name of your pet or even that old chestnut, your mother's maiden name. Such near-public information is often easy to guess so it gave an easier way to break into accounts than guessing the password itself. This was made even worse by everyone asking the same questions. In the case of celebrities – or abuse by a former intimate partner – there may be no usable secrets. This was brought home to the public in 2008, when a student hacked the Yahoo email account of US Vice-Presidential candidate Sarah Palin via the password recovery questions – her date of birth and the name of her first school. Both of these were public information. Since then, crooks have learned to use security questions to loot accounts when they can; at the US Social Security Administration, a common fraud was to open an online account for a pensioner who's dealt with their pension by snail mail in the past, and redirect the payments to a different bank account. This peaked in 2013; the countermeasure that fixed it was to always notify beneficiaries of account changes by snail mail.

In 2015, five Google engineers published a thorough analysis of security questions, and many turned out to be extremely weak. For example, an attacker could get a 19.7% success rate against 'Favourite food?' in English.

Some 37% of people provided wrong answers, in some cases to make them stronger, but sometimes not. Fully 16% of people's answers were public. In addition to being insecure, the 'security questions' turned out to be hard to use: 40% of English-speaking US users were unable to recall the answers when needed, while twice as many could recover accounts using an SMS reset code [292].

Given these problems with security and memorability, most websites now let you recover your password by an email to the address with which you first registered. But if someone compromises that email account, they can get all your dependent accounts too. Email recovery may be adequate for websites where a compromise is of little consequence, but for important accounts – such as banking and email itself – standard practice is now to use a second factor. This is typically a code sent to your phone by SMS, or better still using an app that can encrypt the code and tie it to a specific handset. Many service providers that allow email recovery are nudging people towards using such a code instead where possible. Google research shows that SMSs stop all bulk password guessing by bots, 96% of bulk phishing and 76% of targeted attacks [574].

But this depends on phone companies taking care over who can get a replacement SIM card, and many don't. The problem in 2020 is rapid growth in attacks based on intercepting SMS authentication codes, which mostly seem to involve SIM swap, where the attacker pretends to be you to your mobile phone company and gets a replacement SIM card for your account. SIM-swap attacks started in South Africa in 2007, became the main form of bank fraud in Nigeria, then caught on in America – initially as a means of taking over valuable Instagram accounts, then to loot people's accounts at bitcoin exchanges, then for bank fraud more generally [1094]. I will discuss SIM-swap attacks in more detail in section 12.7.4.

Attackers have also exploited the SS7 signalling protocol to wiretap targets' mobile phones remotely and steal codes [485]. I'll discuss such attacks in more detail in the chapters on phones and on banking. The next step in the arms race will be moving customers from SMS messages for authentication and account recovery to an app; the same Google research shows that this improves these last two figures to 99% for bulk phishing and 90% for targeted attacks [574]. As for the targeted attacks, other research by Ariana Mirian along with colleagues from UCSD and Google approached gangs who advertised 'hack-for-hire' services online and asked them to phish Gmail passwords. Three of the gangs succeeded, defeating SMS-based 2fa with a middleperson attack; forensics then revealed 372 other attacks on Gmail users from the same IP addresses during March to October 2018 [1324]. This is still an immature criminal market, but to stop such attacks an app or authentication token is the way to go. It also raises further questions about account recovery. If I use a hardware security key on my Gmail, do I need a second one in a safe as a recovery mechanism? (Probably.) If I use one app on my phone to do banking and another as an authenticator, do I comply with rules on two-factor authentication? (See section 12.7.4 in the chapter on banking.)

Email notification is the default for telling people not just of suspicious login attempts, but of logins to new devices that succeeded with the help of a code. That way, if someone plants malware on your phone, you have some chance of detecting it. How a victim recovers then is the next question. If all else fails, a service provider may eventually let them speak to a real person. But when designing such a system, never forget that it's only as strong as the weakest fallback mechanism – be it a recovery email loop with an email provider you don't control, a phone code that's vulnerable to SIM swapping or mobile malware, or a human who's open to social engineering.

### 3.4.2 Password choice

Many accounts are compromised by guessing PINs or passwords. There are botnets constantly breaking into online accounts by guessing passwords and password-recovery questions, as I described in 2.3.1.4, in order to use email accounts to send spam and to recruit machines to botnets. And as people invent new services and put passwords on them, the password guessers find new targets. A recent example is cryptocurrency wallets: an anonymous 'bitcoin bandit' managed to steal $50m by trying lots of weak passwords for ethereum wallets [810]. Meanwhile, billions of dollars' worth of cryptocurrency has been lost because passwords were forgotten. So passwords matter, and there are basically three broad concerns, in ascending order of importance and difficulty:

1. Will the user enter the password correctly with a high enough probability?
2. Will the user remember the password, or will they have to either write it down or choose one that's easy for the attacker to guess?
3. Will the user break the system security by disclosing the password to a third party, whether accidentally, on purpose, or as a result of deception?

### 3.4.3 Difficulties with reliable password entry

The first human-factors issue is that if a password is too long or complex, users might have difficulty entering it correctly. If the operation they're trying to perform is urgent, this might have safety implications. If customers have difficulty entering software product activation codes, this can generate expensive calls to your support desk. And the move from laptops to smartphones during the 2010s has made password rules such as 'at least one lower-case letter, upper-case letter, number and special character' really fiddly and annoying.

This is one of the factors pushing people toward longer but simpler secrets, such as passphrases of three or four words. But will people be able to enter them without making too many errors?

An interesting study was done for the STS prepayment meters used to sell electricity in many less-developed countries. The customer hands some money to a sales agent, and gets a 20-digit number printed out on a receipt. They take this receipt home, enter the numbers at a keypad in the meter, and the lights come on. The STS designers worried that since a lot of the population was illiterate, and since people might get lost halfway through entering the number, the system might be unusable. But illiteracy was not a problem: even people who could not read had no difficulty with numbers ('everybody can use a phone', as one of the engineers said). The biggest problem was entry errors, and these were dealt with by printing the twenty digits in two rows, with three groups of four digits in the first row followed by two in the second [94]. I'll describe this in detail in section 14.2.

A quite different application is the firing codes for US nuclear weapons. These consist of only 12 decimal digits. If they are ever used, the operators will be under extreme stress, and possibly using improvised or obsolete communications channels. Experiments suggested that 12 digits was the maximum that could be conveyed reliably in such circumstances. I'll discuss how this evolved in section 15.2.

### 3.4.4 Difficulties with remembering the password

Our second psychological issue is that people often find passwords hard to remember [2079]. Twelve to twenty digits may be easy to copy from a telegram or a meter ticket, but when customers are expected to memorize passwords, they either choose values that are easy for attackers to guess, or write them down, or both. In fact, standard password advice has been summed up as: "Choose a password you can't remember, and don't write it down".

The problems are not limited to computer access. For example, one chain of cheap hotels in France introduced self service. You'd turn up at the hotel, swipe your credit card in the reception machine, and get a receipt with a numerical access code to unlock your room door. To keep costs down, the rooms did not have en-suite bathrooms. A common failure mode was that you'd get up in the middle of the night to go to the bathroom, forget your access code, and realise you hadn't taken the receipt with you. So you'd have to sleep on the bathroom floor until the staff arrived the following morning.

Password memorability can be discussed under five main headings: naïve choice, user abilities and training, design errors, operational failures and vulnerability to social-engineering attacks.

### *3.4.4.1 Naïve choice*

Since the mid-1980s, people have studied what sort of passwords people choose, and found they use spouses' names, single letters, or even just hit carriage return giving an empty string as their password. Cryptanalysis of tapes from a 1980 Unix system showed that of the pioneers, Dennis Ritchie used 'dmac' (his middle name was MacAlistair); the later Google chairman Eric Schmidt used 'wendy!!!' (his wife's name) and Brian Kernighan used '/.,/.,' [796]. Fred Grampp and Robert Morris's classic 1984 paper on Unix security [806] reports that after software became available which forced passwords to be at least six characters long and have at least one nonletter, they made a file of the 20 most common female names, each followed by a single digit. Of these 200 passwords, at least one was in use on each of several dozen machines they examined. At the time, Unix systems kept encrypted passwords in a file `/etc/passwd` that all system users could read, so any user could verify a guess of any other user's password. Other studies showed that requiring a non-letter simply changed the most popular password from 'password' to 'password1' [1675].

In 1990, Daniel Klein gathered 25,000 Unix passwords and found that 21–25% of passwords could be guessed depending on the amount of effort put in [1058]. Dictionary words accounted for 7.4%, common names for 4%, combinations of user and account name 2.7%, and so on down a list of less probable choices such as words from science fiction (0.4%) and sports terms (0.2%). Other password guesses used patterns, such as by taking an account *'klone'* belonging to the user 'Daniel V. Klein' and trying passwords such as klone, klone1, klone123, dvk, dvkdvk, leinad, neilk, DvkkvD, and so on. The following year, Alec Muffett released 'crack', software that would try to brute-force Unix passwords using dictionaries and patterns derived from them by a set of mangling rules.

The largest academic study of password choice of which I am aware is by Joe Bonneau, who in 2012 analysed tens of millions of passwords in leaked password files, and also interned at Yahoo where he instrumented the login system to collect live statistics on the choices of 70 million users. He also worked out the best metrics to use for password guessability, both in standalone systems and where attackers use passwords harvested from one system to crack accounts on another [290]. This work informed the design of password strength checkers and other current practices at the big service firms.

### *3.4.4.2 User abilities and training*

Sometimes you can train the users. Password checkers have trained them to use longer passwords with numbers as well as letters, and the effect spills over to websites that don't use them [446]. But you do not want to drive customers though nowadays it tends to be not a password but a password recovery question. You could always try to tell 'Yngstrom' to your bank, 'Jones' to the phone company, 'Geraghty' to the travel agent, and so on; but data are shared extensively between companies, so you could easily end up confusing their systems – not to mention yourself. And if you try to phone up your bank and tell them that you've decided to change your mother's maiden name from Yngstrom to `yGt5r4ad` – or even Smith – then good luck. In fact, given the large number of data breaches, you might as well assume that anyone who wants to can get all your common password recovery information – including your address, your date of birth, your first school and your social security number, as well as your mother's maiden name.

Some organisations use contextual security information. A bank I once used asks its business customers the value of the last check from their account that was cleared. In theory, this could be helpful: if someone overhears me doing a transaction on the telephone, then it's not a long-term compromise. The details bear some attention though. When this system was first introduced, I wondered whether a supplier, to whom I'd just written a check, might impersonate me, and concluded that asking for the last three checks' values would be safer. But the problem we actually had was unexpected. Having given the checkbook to our accountant for the annual audit, we couldn't talk to the bank. I also don't like the idea that someone who steals my physical post can also steal my money.

The sheer number of applications demanding a password nowadays exceeds the powers of human memory. A 2007 study by Dinei Florêncio and Cormac Herley of half a million web users over three months showed that the average user has 6.5 passwords, each shared across 3.9 different sites; has about 25 accounts that require passwords; and types an average of 8 passwords per day. Bonneau published more extensive statistics in 2012 [290] but since then the frequency of user password entry has fallen, thanks to smartphones. Modern web browsers also cache passwords; see the discussion of password managers at section 3.4.11 below. But many people use the same password for many different purposes and don't work out special processes to deal with their high-value logons such as to their bank, their social media accounts and their email. So you have to expect that the password chosen by the customer of the electronic banking system you've just designed, may be known to a Mafia-operated porn site as well. (There's even a website, `http://haveibeenpwned.com`, that will tell you which security breaches have leaked your email address and password.)

One of the most pervasive and persistent errors has been forcing users to change passwords regularly. When I first came across enforced monthly password changes in the 1980s, I observed that it led people to choose passwords such as 'julia03' for March, 'julia04' for April, and so on, and said as much in the first (2001) edition of this book (chapter 3, page 48). However, in 2003, Bill Burr of NIST wrote password guidelines recommending regular update [1098].

This was adopted by the Big Four auditors, who pushed it out to all their audit clients[3]. Meanwhile, security usability researchers conducted survey after survey showing that monthly change was suboptimal. The first systematic study by Yinqian Zhang, Fabian Monrose and Mike Reiter of the password transformation techniques users invented showed that in a system with forced expiration, over 40% of passwords could be guessed from previous ones, that forced change didn't do much to help people who chose weak passwords, and that the effort of regular password choice may also have diminished password quality [2073]. Finally a survey was written by usability guru Lorrie Cranor while she was Chief Technologist at the FTC [492], and backed up by an academic study [1507]. In 2017, NIST recanted; they now recommend long passphrases that are only changed on compromise[4]. Other governments' agencies such as Britain's GCHQ followed, and Microsoft finally announced the end of password-expiration policies in Windows 10 from April 2019. However, many firms are caught by the PCI standards set by the credit-card issuers, which haven't caught up and still dictate three-monthly changes; another problem is that the auditors dictate compliance to many companies, and will no doubt take time to catch up.

The current fashion, in 2020, is to invite users to select passphrases of three or more random dictionary words. This was promoted by a famous xkcd cartoon which suggested 'correct horse battery staple' as a password. Empirical research, however, shows that real users select multi-word passphrases with much less entropy than they'd get if they really did select at random from a dictionary; they tend to go for common noun bigrams, and moving to three or four words brings rapidly diminishing returns [297]. The Electronic Frontier Foundation now promotes using dice to pick words; they have a list of 7,776 words ($6^5$, so five dice rolls to pick a word) and note that a six-word phrase has 77 bits of entropy and is memorable [291].

### 3.4.4.4 Operational failures

The most pervasive operational error is failing to reset default passwords. This has been a chronic problem since the early dial access systems in the 1980s attracted attention from mischievous schoolkids. A particularly bad example is where systems have default passwords that can't be changed, checked by software that can't be patched. We see ever more such devices in the Internet of Things; they remain vulnerable for their operational lives. The Mirai botnets have emerged to recruit and exploit them, as I described in Chapter 2.

[3] Our university's auditors wrote in their annual report for three years in a row that we should have monthly enforced password change, but couldn't provide any evidence to support this and weren't even aware that their policy came ultimately from NIST. Unimpressed, we asked the chair of our Audit Committee to appoint a new lot of auditors, and eventually that happened.

[4] NIST SP 800-63-3

Passwords in plain sight are another long-running problem, whether on sticky notes or some electronic equivalent. A famous early case was R v Gold and Schifreen, where two young hackers saw a phone number for the development version of Prestel, an early public email service run by British Telecom, in a note stuck on a terminal at an exhibition. They dialed in later, and found the welcome screen had a maintenance password displayed on it. They tried this on the live system too, and it worked! They proceeded to hack into the Duke of Edinburgh's electronic mail account, and sent mail 'from' him to someone they didn't like, announcing the award of a knighthood. This heinous crime so shocked the establishment that when prosecutors failed to persuade the courts to convict the young men, Britain's parliament passed its first Computer Misuse Act.

A third operational issue is asking for passwords when they're not really needed, or wanted for dishonest reasons, as I discussed at the start of this section. Most of the passwords you're forced to set up on websites are there for marketing reasons – to get your email address or give you the feeling of belonging to a 'club' [295]. So it's perfectly rational for users who never plan to visit that site again to express their exasperation by entering '123456' or even ruder words in the password field.

A fourth is atrocious password management systems: some don't encrypt passwords at all, and there are reports from time to time of enterprising hackers smuggling back doors into password management libraries [429].

But perhaps the biggest operational issue is vulnerability to social-engineering attacks.

#### *3.4.4.5 Social-engineering attacks*

Careful organisations communicate security context in various ways to help staff avoid making mistakes. The NSA, for example, had different colored internal and external telephones, and when an external phone in a room is off-hook, classified material can't even be discussed in the room – let alone on the phone.

Yet while many banks and other businesses maintain some internal security context, they often train their customers to act in unsafe ways. Because of pervasive phishing, it's not prudent to try to log on to your bank by clicking on a link in an email, so you should always use a browser bookmark or type in the URL by hand. Yet bank marketing departments send out lots of emails containing clickable links. Indeed much of the marketing industry is devoted to getting people to click on links. Many email clients – including Apple's, Microsoft's, and Google's – make plaintext URLs clickable, so their users may never see a URL that isn't. Bank customers are well trained to do the wrong thing.

A prudent customer should also be cautious if a web service directs them somewhere else – yet bank systems use all sorts of strange URLs for their services. A spam from the Bank of America directed UK customers to `mynewcard.com` and got the certificate wrong (it was for `mynewcard.bankofamerica.com`). There are many more examples of major banks training their customers to practice unsafe computing – by disregarding domain names, ignoring certificate warnings, and merrily clicking links [582]. As a result, even security experts have difficulty telling bank spam from phish [445].

It's not prudent to give out security information over the phone to unidentified callers – yet we all get phoned by bank staff who demand security information. Banks also call us on our mobiles now and expect us to give out security information to a whole train carriage of strangers, rather than letting us text a response. (I've had a card blocked because a bank security team phoned me while I was driving; it would have been against the law to deal with the call other than in hands-free mode, and there was nowhere safe to stop.) It's also not prudent to put a bank card PIN into any device other than an ATM or a PIN entry device (PED) in a store; and Citibank even asks customers to disregard and report emails that ask for personal information, including PIN and account details. So what happened? You guessed it – it sent its Australian customers an email asking customers 'as part of a security upgrade' to log on to its website and authenticate themselves using a card number and an ATM PIN [1089]. And in one 2005 case, the Halifax sent a spam to the mother of a student of ours who contacted the bank's security department, which told her it was a phish. The student then contacted the ISP to report abuse, and found that the URL and the service were genuine [1243]. The Halifax disappeared during the crash of 2008, and given that their own security department couldn't tell spam from phish, perhaps that was justice (though it cost us taxpayers a shedload of money).

#### 3.4.4.6 Customer education

After phishing became a real threat to online banking in the mid-2000s, banks tried to train their customers to look for certain features in websites. This has been partly risk reduction, but partly risk dumping – seeing to it that customers who don't understand or can't follow instructions can be held responsible for the resulting loss. The general pattern has been that as soon as customers are trained to follow some particular rule, the phishermen exploit this, as the reasons for the rule are not adequately explained.

At the beginning, the advice was 'Check the English', so the bad guys either got someone who could write English, or simply started using the banks' own emails but with the URLs changed. Then it was 'Look for the lock symbol', so the phishing sites started to use SSL (or just forging it by putting graphics of lock symbols on their web pages). Some banks started putting the last four digits of the customer account number into emails; the phishermen responded by putting in the first four (which are constant for a given bank and card product). Next the advice was that it was OK to click on images, but not on URLs; the phishermen promptly put in links that appeared to be images but actually pointed at executables. The advice then was to check where a link would really go by hovering your mouse over it; the bad guys then either inserted a non-printing character into the URL to stop Internet Explorer from displaying the rest, or used an unmanageably long URL (as many banks also did).

This sort of arms race is most likely to benefit the attackers. The countermeasures become so complex and counterintuitive that they confuse more and more users – exactly what the phishermen need. The safety and usability communities have known for years that 'blame and train' is not the way to deal with unusable systems – the only real fix is to design for safe usability in the first place [1453].

#### *3.4.4.7 Phishing warnings*

Part of the solution is to give users better tools. Modern browsers alert you to wicked URLs, with a range of mechanisms under the hood. First, there are lists of bad URLs collated by the anti-virus and threat intelligence community. Second, there's logic to look for expired certificates and other compliance failures (as the majority of those alerts are false alarms).

There has been a lot of research, in both industry and academia, about how you get people to pay attention to warnings. We see so many of them, most are irrelevant, and many are designed to shift risk to us from someone else. So when do people pay attention? In our own work, we tried a number of things and found that people paid most attention when the warnings were not vague and general (*'Warning - visiting this web site may harm your computer!'*) but specific and concrete (*'The site you are about to visit has been confirmed to contain software that poses a significant risk to you, with no tangible benefit. It would try to infect your computer with malware designed to steal your bank account and credit card details in order to defraud you*) [1329]. Subsequent research by Adrienne Porter Felt and Google's usability team has tried many ideas including making warnings psychologically salient using faces (which doesn't work), simplifying the text (which helps) and making the safe defaults both attractive and prominent (which also helps). Optimising these factors improves compliance from about 35% to about 50% [675]. However, if you want to stop the great majority of people from clicking on known-bad URLs, then voluntary compliance isn't enough. You either have to block them at your firewall, or block them at the browser (as both Chrome and Firefox do for different types of certificate error – a matter to which we'll return in 21.6).

On the personal side, don't forget what we said about intimate partner abuse in 2.5.4: the passwords people choose are often easy for their spouses or partners to guess, and the same goes for password recovery questions: so some thought needs to be given to how abuse victims can recover their security.

On the system side, there are all sorts of passwords used for mutual authentication between subsystems, few mechanisms to enforce password quality in server-server environments, and many well-known issues (for example, the default password for the Java trusted keystore file is 'changeit'). Development teams often share passwords that end up in live systems, even 30 years after this practice led to the well-publicised hack of the Duke of Edinburgh's email described in section 3.4.4.4. Within a single big service firm you can lock stuff down by having named crypto keys and seeing to it that each name generates a call to an underlying hardware security module; or you can even use mechanisms like SGX to tie keys to known software. But that costs real money, and money isn't the only problem. Enterprise system components are often hosted at different service companies, which makes adoption of better practices a hard coordination problem too. As a result, server passwords often appear in scripts or other plaintext files, which can end up in Dropbox or Splunk. So it is vital to think of password practices beyond end users. In later chapters we'll look at protocols such as Kerberos and ssh; for now, recall Ed Snowden's remark that it was trivial to hack the typical large company: just spear-phish a sysadmin and then chain your way in. Much of this chapter is about the 'spear-phish a sysadmin' part; but don't neglect the 'chain your way in' part.

## 3.4.8 Attacks on password entry

Password entry is often poorly protected.

### *3.4.8.1 Interface design*

Thoughtless interface design is all too common. Some common makes of cash machine have a vertical keyboard at head height, making it simple for a pickpocket to watch a woman enter her PIN before lifting her purse from her handbag. The keyboards may have been at a reasonable height for the men who designed them, but women who are a few inches shorter are exposed.

When entering a card number or PIN in a public place, I usually cover my typing hand with my body or my other hand – but you can't assume that all your customers will. Many people are uncomfortable shielding a PIN as it's a signal of distrust, especially if they're in a supermarket queue and a friend is standing nearby. UK banks found that 20% of users never shield their PIN [128] – and then used this to blame customers whose PINs were compromised by an overhead CCTV camera, rather than designing better PIN entry devices.

#### 3.4.8.2 Trusted path, and bogus terminals

A *trusted path* is some means of being sure that you're logging into a genuine machine through a channel that isn't open to eavesdropping. False terminal attacks go back to the dawn of time-shared computing. A public terminal would be left running an attack program that looks just like the usual logon screen – asking for a user name and password. When an unsuspecting user did this, it would save the password, reply 'sorry, wrong password' and then vanish, invoking the genuine password program. The user assumed they'd made a typing error and just entered the password again. This is why Windows had a *secure attention sequence*; hitting `ctrl-alt-del` was guaranteed to take you to a genuine password prompt. But eventually, in Windows 10, this got removed to prepare the way for Windows tablets, and because almost nobody understood it.

ATM skimmers are devices that sit on an ATM's throat, copy card details, and have a camera to record the customer PIN. There are many variants on the theme. Fraudsters deploy bad PIN entry devices too, and have even been jailed for attaching password-stealing hardware to terminals in bank branches. I'll describe this world in much more detail in the chapter on banking and bookkeeping; the long-term solution has been to move from magnetic-strip cards that are easy to copy to chip cards that are much harder. In any case, if a terminal might contain malicious hardware or software, then passwords alone will not be enough.

#### 3.4.8.3 Technical defeats of password retry counters

Many kids find out that a bicycle combination lock can usually be broken in a few minutes by solving each ring in order of looseness. The same idea worked against a number of computer systems. The PDP-10 TENEX operating system checked passwords one character at a time, and stopped as soon as one of them was wrong. This opened up a *timing attack*: the attacker would repeatedly place a guessed password in memory at a suitable location, have it verified as part of a file access request, and wait to see how long it took to be rejected [1131]. An error in the first character would be reported almost at once, an error in the second character would take a little longer to report, and in the third character a little longer still, and so on. So you could guess the characters one after another, and instead of a password of $N$ characters drawn from an alphabet of $A$ characters taking $A^N/2$ guesses on average, it took $AN/2$. (Bear in mind that in thirty years' time, all that might remain of the system you're building today is the memory of its more newsworthy security failures.)

These same mistakes are being made all over again in the world of embedded systems. With one remote car locking device, as soon as a wrong byte was transmitted from the key fob, the red telltale light on the receiver came on. With some smartcards, it has been possible to determine the customer PIN by trying each possible input value and looking at the card's power consumption, then issuing a reset if the input was wrong. The reason was that a wrong PIN caused a PIN retry counter to be decremented, and writing to the EEPROM memory which held this counter caused a current surge of several milliamps – which could be detected in time to reset the card before the write was complete [1107]. These implementation details matter. Timing channels are a serious problem for people implementing cryptography, as we'll discuss at greater length in the next chapter.

A recent high-profile issue was the PIN retry counter in the iPhone. My colleague Sergei Skorobogatov noted that the iPhone keeps sensitive data encrypted in flash memory, and built an adapter that enabled him to save the encrypted memory contents and restore them to their original condition after several PIN attempts. This enabled him to try all 10,000 possible PINs rather than the ten PINs limit that Apple tried to impose [1781][5].

### 3.4.9 Attacks on password storage

Passwords have often been vulnerable where they are stored. In MIT's 'Compatible Time Sharing System' `ctss` – a 1960s predecessor of Multics – it once happened that one person was editing the message of the day, while another was editing the password file. Because of a software bug, the two editor temporary files got swapped, and everyone who logged on was greeted with a copy of the password file! [476].

Another horrible programming error struck a UK bank in the late 1980s, which issued all its customers with the same PIN by mistake [55]. As the procedures for handling PINs meant that no one in the bank got access to anyone's PIN other than their own, the bug wasn't spotted until after thousands of customer cards had been shipped. Big blunders continue: in 2019 the security company that does the Biostar and AEOS biometric lock system for building entry control and whose customers include banks and police forces in 83 countries left a database unprotected online with over a million people's IDs, plaintext passwords, fingerprints and facial recognition data; security researchers who discovered this from an Internet scan were able to add themselves as users [1867].

Auditing provides another hazard. When systems log failed password attempts, the log usually contains a large number of passwords, as users get the 'username, password' sequence out of phase. If the logs are not well protected then someone who sees an audit record of a failed login with a

[5] This was done to undermine an argument by then FBI Director James Comey that the iPhone was unhackable and so Apple should be ordered to produce an operating system upgrade that created a backdoor; see section 26.2.7.4.

non-existent user name of `e5gv*8yp` just has to try this as a password for all the valid user names.

#### 3.4.9.1 One-way encryption

Such incidents taught people to protect passwords by encrypting them using a one-way algorithm, an innovation due to Roger Needham and Mike Guy. The password, when entered, is passed through a one-way function and the user is logged on only if it matches a previously stored value. However, it's often implemented wrong. The right way to do it is to generate a random key, historically known in this context as a *salt*; combine the password with the salt using a slow, cryptographically strong one-way function; and store both the salt and the hash.

#### 3.4.9.2 Password cracking

Some systems that use an encrypted password file make it widely readable. Unix used to be the prime example – the password file `/etc/passwd` was readable by all users. So any user could fetch it and try to break passwords by encrypting all the passwords in a dictionary and comparing them with the encrypted values in the file. We already mentioned in 3.4.4.1 the 'Crack' software that people have used for years for this purpose.

Most modern operating systems have sort-of fixed this problem; in modern Linux distributions, for example, passwords are salted, hashed using 5000 rounds of SHA-512, and stored in a file that only the root user can read. But there are still password-recovery tools to help you if, for example, you've encrypted an Office document with a password you've forgotten [1677]. Such tools can also be used by a crook who has got root access, and there are still lots of badly designed systems out there where the password file is vulnerable in other ways.

There is also *credential stuffing*: when a system is hacked and passwords are cracked (or were even found unencrypted), they are then tried out on other systems to catch the many people who reused them. This remains a live problem. So password cracking is still worth some attention. One countermeasure worth considering is deception, which can work at all levels in the stack. You can have honeypot systems that alarm if anyone ever logs on to them, honeypot accounts on a system, or password canaries – bogus encrypted passwords for genuine accounts [998].

#### 3.4.9.3 Remote password checking

Many systems check passwords remotely, using cryptographic protocols to protect the password in transit, and the interaction between password security and network security can be complex. Local networks often use a protocol called Kerberos, where a server sends you a key encrypted under your password; if you know the password you can decrypt the key and use it to get tickets that give you access to resources. I'll discuss this in the next chapter, in section 4.7.4; it doesn't always protect weak passwords against an opponent who can wiretap encrypted traffic. Web servers mostly use a protocol called TLS to encrypt your traffic from the browser on your phone or laptop; I discuss TLS in the following chapter, in section 5.7.5. TLS does not protect you if the server gets hacked. However there is a new protocol called Simultaneous Authentication of Equals (SAE) which is designed to set up secure sessions even where the password is guessable, and which has been adopted from 2018 in the WPA3 standard for WiFi authentication. I'll discuss this later too.

And then there's OAuth, a protocol which allows access delegation, so you can grant one website the right to authenticate you using the mechanisms provided by another. Developed by Twitter from 2006, it's now used by the main service providers such as Google, Microsoft and Facebook to let you log on to media and other sites; an authorisation server issues access tokens for the purpose. We'll discuss the mechanisms later too. The concomitant risk is cross-site attacks; we are now (2019) seeing OAuth being used by state actors in authoritarian countries to phish local human-rights defenders. The technique is to create a malicious app with a plausible name (say 'Outlook Security Defender') and send an email, purportedly from Microsoft, asking for access. If the target responds they end up at a Microsoft web page where they're asked to authorise the app to have access to their data [47].

### 3.4.10 Absolute limits

If you have confidence in the cryptographic algorithms and operating-system security mechanisms that protect passwords, then the probability of a successful password guessing attack is a function of the entropy of passwords, if they are centrally assigned, and the psychology of users if they're allowed to choose them. Military sysadmins often prefer to issue random passwords, so the probability of password guessing attacks can be managed. For example, if $L$ is the maximum password lifetime, $R$ is login attempt rate, $S$ is the size of the password space, then the probability that a password can be guessed in its lifetime is $P = LR/S$, according to the US Department of Defense password management guideline [546].

There are issues with such a 'provable security' doctrine, starting with the attackers' goal. Do they want to crack a target account, or just any account? If an army has a million possible passwords and a million users, and the alarm goes off after three bad password attempts on any account, then the attacker

### 3.4.12 Will we ever get rid of passwords?

Passwords are annoying, so many people have discussed getting rid of them, and the move from laptops to phones gives us a chance. The proliferation of IoT devices that don't have keyboards will force us to do without them for some purposes. A handful of firms have tried to get rid of them completely. One example is the online bank Monzo, which operates exclusively via an app. They leave it up to the customer whether they protect their phone using a fingerprint, a pattern lock, a PIN or a password. However they still use email to prompt people to upgrade, and to authenticate people who buy a new phone, so account takeover involves either phone takeover, or guessing a password or a password recovery question. The most popular app that uses SMS to authenticate rather than a password may be WhatsApp. I expect that this will become more widespread; so we'll see more attacks based on phone takeover, from SIM swaps through Android malware, SS7 and RCS hacking, to simple physical theft. In such cases, recovery often means an email loop, making your email password more critical than ever – or phoning a call centre and telling them your mother's maiden name. So things may change less than they seem.

Joe Bonneau and colleagues analysed the options in 2012 [293]. There are many criteria against which an authentication system can be evaluated, and we've worked through them here: resilience to theft, to physical observation, to guessing, to malware and other internal compromise, to leaks from other verifiers, to phishing and to targeted impersonation. Other factors include ease of use, ease of learning, whether you need to carry something extra, error rate, ease of recovery, cost per user, and whether it's an open design that anyone can use. They concluded that most of the schemes involving net benefits were variants on single sign-on – and OpenID has indeed become widespread, with many people logging in to their newspaper using Google or Facebook, despite the obvious privacy cost[6]. Beyond that, any security improvements involve giving up one or more of the benefits of passwords, namely that they're easy, efficient and cheap.

Bonneau's survey gave high security ratings to physical authentication tokens such as the CAP reader, which enables people to use their bank cards to log on to online banking; bank regulators have already mandated two-factor

[6] Government attempts to set up single sign-on for public services have been less successful, with the UK 'Verify' program due to be shuttered in 2020 [1394]. There have been many problems around attempts to entrench government's role in identity assurance, which I'll discuss further in the chapter on biometrics, and which spill over into issues from online services to the security of elections. It was also hard for other private-sector firms to compete because of the network effects enjoyed by incumbents. However in 2019 Apple announced that it would provide a new, more privacy-friendly single sign-on mechanism, and use the market power of its app store to force websites to support it. Thus the quality and nature of privacy on offer is becoming a side-effect of battles fought for other motives. We'll analyse this in more depth in the chapter on economics.

authentication in a number of countries. Using something tied to a bank card gives a more traditional root of trust, at least with traditional high-street banks; a customer can walk into a branch and order a new card[7]. Firms that are targets of state-level attackers, such as Google and Microsoft, now give authentication tokens of some kind or another to all their staff.

Did the survey miss anything? Well, the old saying is 'something you have, something you know, or something you are' – or, as Simson Garfinkel engagingly puts it, 'something you had once, something you've forgotten, or something you once were'. The third option, biometrics, has started coming into wide use since high-end mobile phones started offering fingerprint readers. Some countries, like Germany, issue their citizens with ID cards containing a fingerprint, which may provide an alternate root of trust for when everything else goes wrong. We'll discuss biometrics in its own chapter later in the book.

Both tokens and biometrics are still mostly used with passwords, first as a backstop in case a device gets stolen, and second as part of the process of security recovery. So passwords remain the (shaky) foundation on which much of information security is built. What may change this is the growing number of devices that have no user interface at all, and so have to be authenticated using other mechanisms. One approach that's getting ever more common is trust on first use, also known as the 'resurrecting duckling' after the fact that a duckling bonds on the first moving animal it sees after it hatches. We'll discuss this in the next chapter, and also when we dive into specific applications such as security in vehicles.

Finally, you should think hard about how to authenticate customers or other people who exercise their right to demand copies of their personal information under data-protection law. In 2019, James Pavur sent out 150 such requests to companies, impersonating his fiancée [1890]. 86 firms admitted they had information about her, and many had the sense to demand her logon and password to authenticate her. But about a quarter were prepared to accept an email address or phone number as authentication; and a further 16 percent asked for easily forgeable ID. He collected full personal information about her, including her credit card number, her social security number and her mother's maiden name. A threat intelligence firm with which she'd never interacted sent a list of her accounts and passwords that had been compromised. Given that firms face big fines in the EU if they don't comply with such requests within 30 days, you'd better work out in advance how to cope with them, rather than leaving it to an assistant in your law office to improvise a procedure. If you abolish passwords, and a former customer claims their phone was stolen, what do you do then? And if you hold personal data on people who have never been your customers, how do you identify them?

[7] This doesn't work for branchless banks like Monzo; but they do take a video of you when you register so that their call centre can recognise you later.

## 3.5 CAPTCHAs

Can we have protection mechanisms that use the brain's strengths rather than its weaknesses? The most successful innovation in this field is probably the CAPTCHA – the 'Completely Automated Public Turing Test to Tell Computers and Humans Apart'. These are the little visual puzzles that you often have to solve to post to a blog, to register for a free online account, or to recover a password. The idea is that people can solve such problems easily, while computers find them hard.

CAPTCHAs first came into use in a big way in 2003 to stop spammers using scripts to open thousands of accounts on free email services, and to make it harder for attackers to try a few simple passwords with each of a large number of existing accounts. They were invented by Luis von Ahn and colleagues [1973], who were inspired by the test famously posed by Alan Turing as to whether a computer was intelligent: you put a computer in one room and a human in another, and invite a human to try to tell them apart. The test is turned round so that a computer can tell the difference between human and machine.

Early versions set out to use a known 'hard problem' in AI such as the recognition of distorted text against a noisy background. The idea is that breaking the CAPTCHA was equivalent to solving the AI problem, so an attacker would actually have to do the work by hand, or come up with a real innovation in computer science. Humans were good at reading distorted text, while programs were less good. It turned out to be harder than it seemed. A lot of the attacks on CAPTCHAs, even to this day, exploit the implementation details.

Many of the image recognition problems posed by early systems also turned out not to be too hard at all once smart people tried hard to solve them. There are also protocol-level attacks; von Ahn mentioned that in theory a spammer could get people to solve them as the price of access to free porn [1972]. This soon started to happen: spammers created a game in which you undress a woman by solving one CAPTCHA after another [192]. Within a few years, we saw commercial CAPTCHA-breaking tools arriving on the market [844]. Within a few more, generic attacks using signal-processing techniques inspired by the human visual system had become fairly efficient at solving at least a subset of most types of text CAPTCHA [746]. And security-economics research in underground markets has shown that by 2011 the action had moved to using humans; people in countries with incomes of a few dollars a day will solve CAPTCHAs for about 50c per 1000.

From 2014, the CAPTCHA has been superseded by the ReCAPTCHA, another of Luis von Ahn's inventions. Here the idea is to get a number of users to do some useful piece of work, and check their answers against each other. The service initially asked people to transcribe fragments of text from Google books that confused OCR software; more recently you get a puzzle with eight pictures asking 'click on all images containing a shop front', which helps Google train its vision-recognition AI systems[8]. It pushes back on the cheap-labour attack by putting up two or three multiple-choice puzzles and taking tens of seconds over it, rather than allowing rapid responses.

The implementation of CAPTCHAs is often thoughtless, with accessibility issues for users who are visually impaired. And try paying a road toll in Portugal where the website throws up a CAPTCHA asking you to identify pictures with an object, if you can't understand Portuguese well enough to figure out what you're supposed to look for!

## 3.6 Summary

Psychology matters to the security engineer, because of deception and because of usability. Most real attacks nowadays target the user. Various kinds of phishing are the main national-security threat, the principal means of developing and maintaining the cybercrime infrastructure, and one of the principal threats to online banking systems. Other forms of deception account for much of the rest of the cybercrime ecosystem, which is roughly equal to legacy crime in both volume and value.

Part of the remedy is security usability, yet research in this field was long neglected, being seen as less glamorous than cryptography or operating systems. That was a serious error on our part, and from the mid-2000s we have started to realise the importance of making it easier for ordinary people to use systems in safe ways. Since the mid-2010s we've also started to realise that we also have to make things easier for ordinary programmers; many of the security bugs that have broken real systems have been the result of tools that were just too hard to use, from cryptographic APIs that used unsafe defaults to the C programming language. Getting usability right also helps business directly: PayPal has built a $100bn business through being a safer and more convenient way to shop online[9].

In this chapter, we took a whistle-stop tour through psychology research relevant to deception and to the kinds of errors people make, and then tackled authentication as a case study. Much of the early work on security usability focused on password systems, which raise dozens of interesting questions. We now have more and more data not just on things we can measure in the lab such as guessability, memorability, and user trainability, but also on factors that can

[8] There's been pushback from users who see a ReCAPTCHA saying 'click on all images containing a helicopter' and don't want to help in military AI research. Google's own staff protested at this research too and the military program was discontinued. But other users still object to working for Google for free.

[9] Full disclosure: I consult for them.

only be observed in the field such as how real systems break, how real attacks scale and how the incentives facing different players lead to unsafe equilibria.

At the end of the first workshop on security and human behavior in 2008, the psychologist Nick Humphrey summed up a long discussion on risk. “We’re all agreed,” he said, “that people pay too much attention to terrorism and not enough to cybercrime. But to a psychologist this is obvious. If you want people to be more relaxed in airports, take away the tanks and guns, put in some nice sofas and Mozart in the loudspeakers, and people will relax soon enough. And if you want people to be more wary online, make everyone use Jaws as their screen saver. But that’s not going to happen as the computer industry goes out of its way to make computers seem a lot less scary than they used to be.” And of course governments want people to be anxious about terrorism, as it bids up the police budgets and helps politicians get re-elected. So we give people the wrong signals as well as spending our money on the wrong things. Understanding the many tensions between the demands of psychology, economics and engineering is essential to building robust systems at global scale.

## Research problems

Security psychology is one of the hot topics in 2020. In the second edition of this book, I noted that the whole field of security economics had sprung into life since the first edition in 2001, and wrote ‘We also need more fundamental thinking about the relationship between psychology and security’. Security usability has become a discipline too, with the annual Symposium on Usable Privacy and Security, and we’ve been running workshops to bring security engineers together with anthropologists, psychologists, philosophers and others who work on risk and how people cope with it.

My meta-algorithm for finding research topics is to look first at applications and then at neighbouring disciplines. An example of the first is safe usability: as safety-critical products from cars to medical devices acquire not just software and Internet connections, but complex interfaces and even their own apps, how can we design them so that they won’t harm people by accident, or as a result of malice?

An example of the second, and the theme of the Workshop on Security and Human Behaviour, is what we can learn from disciplines that study how people deal with risk, ranging from anthropology and psychology to sociology, history and philosophy. Our 2020 event is hosting leading criminologists. The pandemic now suggests that maybe we should work with architects too. They’re now working out how people can be physically distant but socially engaged, and their skill is understanding how form facilitates human experience and human interaction. There’s more to design than just hacking code.

now that people log on to websites over the Internet, it is much less obvious. Evaluating a protocol thus involves two questions: first, is the threat model realistic? Second, does the protocol deal with it?

Protocols may be very simple, such as swiping a badge through a reader to enter a building. They often involve interaction, and are not necessarily technical. For example, when we order a bottle of fine wine in a restaurant, the standard protocol is that the wine waiter offers us the menu (so that we see the prices but our guests don't); they bring the bottle, so we can check the label, the seal and the temperature; they open it so we can taste it; and then serve it. This has evolved to provide some privacy (our guests don't learn the price), some integrity (we can be sure we got the right bottle and that it wasn't refilled with cheap plonk) and non-repudiation (we can't complain afterwards that the wine was off). Matt Blaze gives other non-technical protocol examples from ticket inspection, aviation security and voting in [261]. Traditional protocols like these often evolved over decades or centuries to meet social expectations as well as technical threats.

At the technical end of things, protocols get a lot more complex, and they don't always get better. As the car industry moved from metal keys to electronic keys with buttons you press, theft fell, since the new keys were harder to copy. But the move to keyless entry has seen car crime rise again, as the bad guys figured out how to build relay devices that would make a key seem closer to the car than it actually was. Another security upgrade that's turned out to be tricky is the move from magnetic-strip cards to smartcards. Europe made this move in the late 2000s while the USA is only catching up in the late 2010s. Fraud against cards issued in Europe actually went up for several years; clones of European cards were used in magnetic-strip cash machines in the USA, as the two systems' protection mechanisms didn't quite mesh. And there was a protocol failure that let a thief use a stolen chipcard in a store even if he didn't know the PIN, which took the banks several years to fix.

So we need to look systematically at security protocols and how they fail.

## 4.2 Password eavesdropping risks

Passwords and PINs are still the foundation for much of computer security, as the main mechanism used to authenticate humans to machines. We discussed their usability in the last chapter; now let's consider the kinds of technical attack we have to block when designing protocols that operate between one machine and another.

Remote key entry is a good place to start. The early systems, such as the remote control used to open your garage or to unlock cars manufactured up to the mid-1990's, just broadcast a serial number. The attack that killed them was the 'grabber', a device that would record a code and replay it later. The first grabbers, seemingly from Taiwan, arrived on the market in about 1995; thieves would lurk in parking lots or outside a target's house, record the signal used to lock the car and then replay it once the owner had gone[1].

The first countermeasure was to use separate codes for lock and unlock. But the thief can lurk outside your house and record the unlock code before you drive away in the morning, and then come back at night and help himself. Second, sixteen-bit passwords are too short. Occasionally people found they could unlock the wrong car by mistake, or even set the alarm on a car whose owner didn't know he had one [309]. And by the mid-1990's, devices appeared that could try all possible codes one after the other. A code will be found on average after about $2^{15}$ tries, and at ten per second that takes under an hour. A thief operating in a parking lot with a hundred vehicles within range would be rewarded in less than a minute with a car helpfully flashing its lights.

The next countermeasure was to double the length of the password from 16 to 32 bits. The manufacturers proudly advertised 'over 4 billion codes'. But this only showed they hadn't really understood the problem. There were still only one or two codes for each car, and grabbers still worked fine.

Using a serial number as a password has a further vulnerability: lots of people have access to it. In the case of a car, this might mean all the dealer staff, and perhaps the state motor vehicle registration agency. Some burglar alarms have also used serial numbers as master passwords, and here it's even worse: when a bank buys a burglar alarm, the serial number may appear on the order, the delivery note and the invoice. And banks don't like sending someone out to buy something for cash.

Simple passwords are sometimes the appropriate technology. For example, a monthly season ticket for our local swimming pool simply has a barcode. I'm sure I could make a passable forgery, but as the turnstile attendants get to know the 'regulars', there's no need for anything more expensive. For things that are online, however, static passwords are hazardous; the Mirai botnet got going by recruiting wifi-connected CCTV cameras which had a password that couldn't be changed. And for things people want to steal, like cars, we also need something better. This brings us to cryptographic authentication protocols.

[1] With garage doors it's even worse. A common chip is the Princeton PT2262, which uses 12 tri-state pins to encode $3^{12}$ or 531,441 address codes. However implementers often don't read the data sheet carefully enough to understand tri-state inputs and treat them as binary instead, getting $2^{12}$. Many of them only use eight inputs, as the other four are on the other side of the chip. And as the chip has no retry-lockout logic, an attacker can cycle through the combinations quickly and open your garage door after $2^7$ attempts on average. Twelve years after I noted these problems in the second edition of this book, the chip has not been withdrawn. It's now also sold for home security systems and for the remote control of toys.

## 4.3 Who goes there? – simple authentication

A simple modern authentication device is the token that some multistorey parking garages give subscribers to raise the barrier. The token has a single button; when you press it, it first transmits its serial number and then sends an authentication block consisting of the same serial number, followed by a random number, all encrypted using a key unique to the device, and sent to the garage barrier (typically by radio at 434MHz, though infrared is also used). We will postpone discussion of how to encrypt data to the next chapter, and simply write $\{X\}_K$ for the message $X$ encrypted under the key $K$.

Then the protocol between the access token and the parking garage can be written as:

$$T \rightarrow G : T, \{T, N\}_{KT}$$

This is standard protocol notation, so we'll take it slowly.

The token $T$ sends a message to the garage $G$ consisting of its name $T$ followed by the encrypted value of $T$ concatenated with $N$, where $N$ stands for 'number used once', or *nonce*. Everything within the braces is encrypted, and the encryption binds $T$ and $N$ together as well as obscuring their values. The purpose of the nonce is to assure the recipient that the message is *fresh*, that is, it is not a replay of an old message. Verification is simple: the garage reads $T$, gets the corresponding key $KT$, deciphers the rest of the message, checks that the nonce $N$ has not been seen before, and finally that the plaintext contains $T$.

One reason many people get confused is that to the left of the colon, $T$ identifies one of the principals (the token that represents the subscriber) whereas to the right it means the name (that is, the unique device number) of the token. Another is that once we start discussing attacks on protocols, we may find that a message intended for one principal was intercepted and played back by another. So you might think of the $T \rightarrow G$ to the left of the colon as a hint as to what the protocol designer had in mind.

A *nonce* can be anything that guarantees the freshness of a message. It can be a random number, a counter, a random challenge received from a third party, or even a timestamp. There are subtle differences between them, such as in the level of resistance they offer to various kinds of replay attack, and the ways in which they increase system cost and complexity. In very low-cost systems, random numbers and counters predominate as it's cheaper to communicate in one direction only, and cheap devices usually don't have clocks.

Key management in such devices can be very simple. In a typical garage token product, each token's key is just its unique device number encrypted under a global master key $KM$ known to the garage:

$$KT = \{T\}_{KM}$$

This is known as *key diversification* or *key derivation*. It's a common way of implementing access tokens, and is widely used in smartcards too. The goal is that someone who compromises a token by drilling into it and extracting the key cannot masquerade as any other token; all he can do is make a copy of one particular subscriber's token. In order to do a complete break of the system, and extract the master key that would enable him to pretend to be any of the system's users, an attacker has to compromise the central server at the garage (which might protect this key in a tamper-resistant smartcard or hardware security module).

But there is still room for error. A common failure mode is for the serial numbers – whether unique device numbers or protocol counters – not to be long enough, so that someone occasionally finds that their remote control works for another car in the car park as well. This can be masked by cryptography. Having 128-bit keys doesn't help if the key is derived by encrypting a 16-bit device number, or by taking a 16-bit key and repeating it eight times. In either case, there are only $2^{16}$ possible keys, and that's unlikely to be enough even if they appear to be random[2].

Protocol vulnerabilities usually give rise to more, and simpler, attacks than cryptographic weaknesses do. An example comes from the world of prepayment utility meters. Over a million households in the UK, plus over 400 million in developing countries, have an electricity or gas meter that accepts encrypted tokens: the householder buys a magic number and types it into the meter, which then dispenses the purchased quantity of energy. One early meter that was widely used in South Africa checked only that the nonce was different from last time. So the customer could charge their meter indefinitely by buying two low-value power tickets and then feeding them in one after the other; given two valid codes *A* and *B*, the series *ABABAB*... was seen as valid [94].

So the question of whether to use a random number or a counter is not as easy as it looks. If you use random numbers, the lock has to remember a lot of past codes. There's the *valet attack*, where someone with temporary access, such as a valet parking attendant, records some access codes and replays them later to steal your car. In addition, someone might rent a car, record enough unlock codes, and then go back later to the rental lot to steal it. Providing enough nonvolatile memory to remember thousands of old codes might add a few cents to the cost of your lock.

If you opt for counters, the problem is synchronization. The key might be used for more than one lock; it may also be activated repeatedly by accident (I once took an experimental token home where it was gnawed by my dogs). So you need a way to recover after the counter has been incremented hundreds or possibly even thousands of times. One common product uses a sixteen bit

[2] We'll go into this in more detail in section 5.3.1.2 where we discuss the birthday theorem in probability theory.

counter, and allows access when the deciphered counter value is the last valid code incremented by no more than sixteen. To cope with cases where the token has been used more than sixteen times elsewhere (or gnawed by a family pet), the lock will open on a second press provided that the counter value has been incremented between 17 and 32,767 times since a valid code was entered (the counter rolls over so that 0 is the successor of 65,535). This is fine in many applications, but a thief who can get six well-chosen access codes – say for values 0, 1, 20,000, 20,001, 40,000 and 40,001 – can break the system completely. In your application, would you be worried about that?

So designing even a simple token authentication mechanism is not as easy as it looks, and if you assume that your product will only attract low-grade adversaries, this assumption might fail over time. An example is *accessory control*. Many printer companies embed authentication mechanisms in printers to ensure that genuine toner cartridges are used. If a competitor's product is loaded instead, the printer may quietly downgrade from 1200 dpi to 300 dpi, or simply refuse to work at all. All sorts of other industries are getting in on the act, from scientific instruments to games consoles. The cryptographic mechanisms used to support this started off in the 1990s being fairly rudimentary, as vendors thought that any competitor who circumvented them on an industrial scale could be sued or even jailed under copyright law. But then a judge found that while a vendor had the right to hire the best cryptographer they could find to lock their customers in, a competitor also had the right to hire the best cryptanalyst they could find to set them free to buy accessories from elsewhere. This set off a serious arms race, which we'll discuss in section 24.6. Here I'll just remark that security isn't always a good thing. Security mechanisms are used to support many business models, where they're typically stopping the device's owner doing things she wants to rather than protecting her from the bad guys. The effect may be contrary to public policy; one example is cellphone locking, which results in hundreds of millions of handsets ending up in landfills each year, with toxic heavy metals as well as the embedded carbon cost.

### 4.3.1 Challenge and response

Since 1995, all cars sold in Europe were required to have a 'cryptographically enabled immobiliser' and by 2010, most cars had remote-controlled door unlocking too, though most also have a fallback metal key so you can still get into your car even if the key fob battery is flat. The engine immobiliser is harder to bypass using physical means and uses a two-pass *challenge-response protocol* to authorise engine start. As the car key is inserted into the steering lock, the engine controller sends a challenge consisting of a random $n$-bit number to the key using short-range radio. The car key computes a response you keep the key in your pocket or handbag you don't have to worry about it; the car will unlock when you walk up to it, lock as you walk away, and start automatically when you touch the controls. What's not to like?

Well, now you don't have to press a button to unlock your car, it's easy for thieves to use devices that amplify or relay the signals. The thief sneaks up to your front door with one relay while leaving the other next to your car. If you left your keys on the table in the hall, the car door opens and away he goes. Even if the car is immobilised he can still steal your stuff. And after many years of falling car thefts, the statistics surged in 2017 with 56% more vehicles stolen in the UK, followed by a further 9% in 2018 [824][4].

The takeaway message is that the attempt since about 1990 to use cryptography to make cars harder to steal had some initial success, as immobilisers made cars harder to steal and insurance premiums fell. It has since backfired, as the politicians and then the marketing people got in the way. The politicians said it would be disastrous for law enforcement if people were allowed to use cryptography they couldn't crack, even for stopping car theft. Then the immobiliser vendors' marketing people wanted proprietary algorithms to lock in the car companies, whose own marketing people wanted passive keyless entry as it seemed cool.

What can we do? Well, at least two car makers have put an accelerometer in the key fob, so it won't work unless the key is moving. One of our friends left her key on the car seat while carrying her child indoors, and got locked out. The local police advise us to use old-fashioned metal steering-wheel locks; our residents' association recommends keeping keys in a biscuit tin. As for me, we bought such a car but found that the keyless entry was simply too flaky; my wife got stranded in a supermarket car park when it just wouldn't work at all. So we took that car back, and got a second-hand one with a proper push-button remote lock. There are now chips using AES from NXP, Atmel and TI – of which the Atmel is open source with an open protocol stack.

However crypto by itself can't fix relay attacks; the proper fix is a new radio protocol based on ultrawideband (UWB) with intrinsic ranging, which measures the distance from the key fob to the car with a precision of 10cm up to a range of 150m. This is fairly complex to do properly, and the design of the new 802.15.4z Enhanced Impulse Radio is described by Srdjan Capkun and colleagues [1768]; the first chip became available in 2019, and it will ship in cars from 2020. Such chips have the potential to replace both the Bluetooth and NFC protocols, but they might not all be compatible; there's a low-rate pulse (LRP) mode that has an open design, and a high-rate pulse (HRP) variant that's partly proprietary. Were I advising a car startup, LRP would be my starting point.

[4] To be fair this was not due solely to relay attacks, as about half of the high-value thefts seem to involve connecting a car theft kit to the onboard diagnostic port under the glove box. As it happens, the authentication protocols used on the CAN bus inside the vehicle are also vulnerable in a number of ways [893]. Updating these protocols will take many years because of the huge industry investment.

Locks are not the only application of challenge-response protocols. In HTTP Digest Authentication, a web server challenges a client or proxy, with whom it shares a password, by sending it a nonce. The response consists of the hash of the nonce, the password, and the requested URI [715]. This provides a mechanism that's not vulnerable to password snooping. It's used, for example, to authenticate clients and servers in SIP, the protocol for Voice-Over-IP (VOIP) telephony. It's much better than sending a password in the clear, but like keyless entry it suffers from middleperson attacks (the beneficiaries seem to be mostly intelligence agencies).

### 4.3.2 Two-factor authentication

The most visible use of challenge-response is probably in *two-factor authentication*. Many organizations issue their staff with password generators to let them log on to corporate computer systems, and many banks give similar devices to customers. They may look like little calculators (and some even work as such) but their main function is as follows. When you want to log in, you are presented with a random nonce of maybe seven digits. You key this into your password generator, together with a PIN of maybe four digits. The device encrypts these eleven digits using a secret key shared with the corporate security server, and displays the first seven digits of the result. You enter these seven digits as your password. This protocol is illustrated in Figure 4.1. If you had a password generator with the right secret key, and you entered the PIN right, and you typed in the result correctly, then you get in.

Formally, with $S$ for the server, $P$ for the password generator, $PIN$ for the user's Personal Identification Number, $U$ for the user and $N$ for the nonce:

$$\begin{array}{rl} S \rightarrow U: & N \\ U \rightarrow P: & N, PIN \\ P \rightarrow U: & \{N, PIN\}_K \\ U \rightarrow S: & \{N, PIN\}_K \end{array}$$

These devices appeared from the early 1980s and caught on first with phone companies, then in the 1990s with banks for use by staff. There are simplified versions that don't have a keyboard, but just generate new access codes by encrypting a counter or a clock. And they work; the US Defense Department announced in 2007 that an authentication system based on the DoD Common Access Card had cut network intrusions by 46% in the previous year [321].

This was just when crooks started phishing bank customers at scale, so many banks adopted the technology. One of my banks gives me a small calculator that generates a new code for each logon, and also allows me to authenticate new payees by using the last four digits of their account number in place of the challenge. My other bank uses the Chip Authentication Program (CAP), a calculator in which I can insert my bank card to do the crypto.

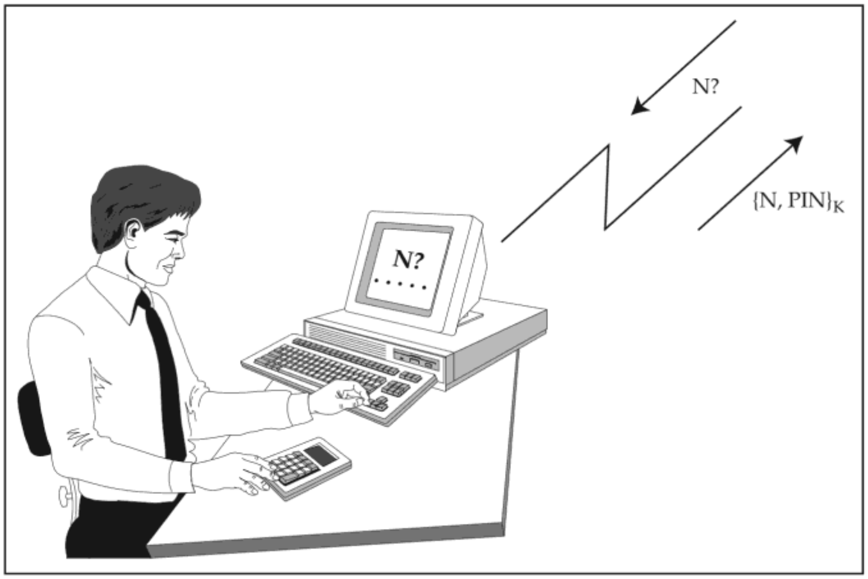


**Figure 4.1:** Password generator use

But this still isn't foolproof. In the second edition of this book, I noted 'someone who takes your bank card from you at knifepoint can now verify that you've told them the right PIN', and this now happens. I also noted that 'once lots of banks use one-time passwords, the phishermen will just rewrite their scripts to do real-time man-in-the-middle attacks' and this has also become widespread. To see how such attacks work, let's look at a military example.

### 4.3.3 The MIG-in-the-middle attack

The first use of challenge-response authentication protocols was probably in the military, with 'identify-friend-or-foe' (IFF) systems. The ever-increasing speeds of warplanes in the 1930s and 1940s, together with the invention of the jet engine, radar and rocketry, made it ever more difficult for air defence forces to tell their own craft apart from the enemy's. This led to a risk of pilots shooting down their colleagues by mistake and drove the development of automatic systems to prevent this. These were first fielded in World War II, and enabled an airplane illuminated by radar to broadcast an identifying number to signal friendly intent. In 1952, this system was adopted to identify civil aircraft to air traffic controllers and, worried about the loss of security once it became widely used, the US Air Force started a research program to incorporate cryptographic protection in the system. Nowadays, the typical air defense system sends random challenges with its radar signals, and friendly aircraft can identify themselves with correct responses.

It's tricky to design a good IFF system. One of the problems is illustrated by the following story, which I heard from an officer in the South African Air Force (SAAF). After it was published in the first edition of this book, the story was disputed – as I'll discuss below. Be that as it may, similar games have been played with other electronic warfare systems since World War 2. The 'MIG-in-the-middle' story has since become part of the folklore, and it nicely illustrates how attacks can be carried out in real time on challenge-response protocols.

In the late 1980's, South African troops were fighting a war in northern Namibia and southern Angola. Their goals were to keep Namibia under white rule, and impose a client government (UNITA) on Angola. Because the South African Defence Force consisted largely of conscripts from a small white population, it was important to limit casualties, so most South African soldiers remained in Namibia on policing duties while the fighting to the north was done by UNITA troops. The role of the SAAF was twofold: to provide tactical support to UNITA by bombing targets in Angola, and to ensure that the Angolans and their Cuban allies did not return the compliment in Namibia.

Suddenly, the Cubans broke through the South African air defenses and carried out a bombing raid on a South African camp in northern Namibia, killing a number of white conscripts. This proof that their air supremacy had been lost helped the Pretoria government decide to hand over Namibia to the insurgents –itself a huge step on the road to majority rule in South Africa several years later. The raid may also have been the last successful military operation ever carried out by Soviet bloc forces.

Some years afterwards, a SAAF officer told me how the Cubans had pulled it off. Several MIGs had loitered in southern Angola, just north of the South African air defense belt, until a flight of SAAF Impala bombers raided a target in Angola. Then the MIGs turned sharply and flew openly through the SAAF's air defenses, which sent IFF challenges. The MIGs relayed them to the Angolan air defense batteries, which transmitted them at a SAAF bomber; the responses were relayed back to the MIGs, who retransmitted them and were allowed through – as in Figure 4.2. According to my informant, this shocked the general staff in Pretoria. Being not only outfought by black opponents, but actually outsmarted, was not consistent with the world view they had held up till then.

After this tale was published in the first edition of my book, I was contacted by a former officer in SA Communications Security Agency who disputed the story's details. He said that their IFF equipment did not use cryptography yet at the time of the Angolan war, and was always switched off over enemy territory. Thus, he said, any electronic trickery must have been of a more primitive kind. However, others tell me that 'Mig-in-the-middle' tricks were significant in Korea, Vietnam and various Middle Eastern conflicts.

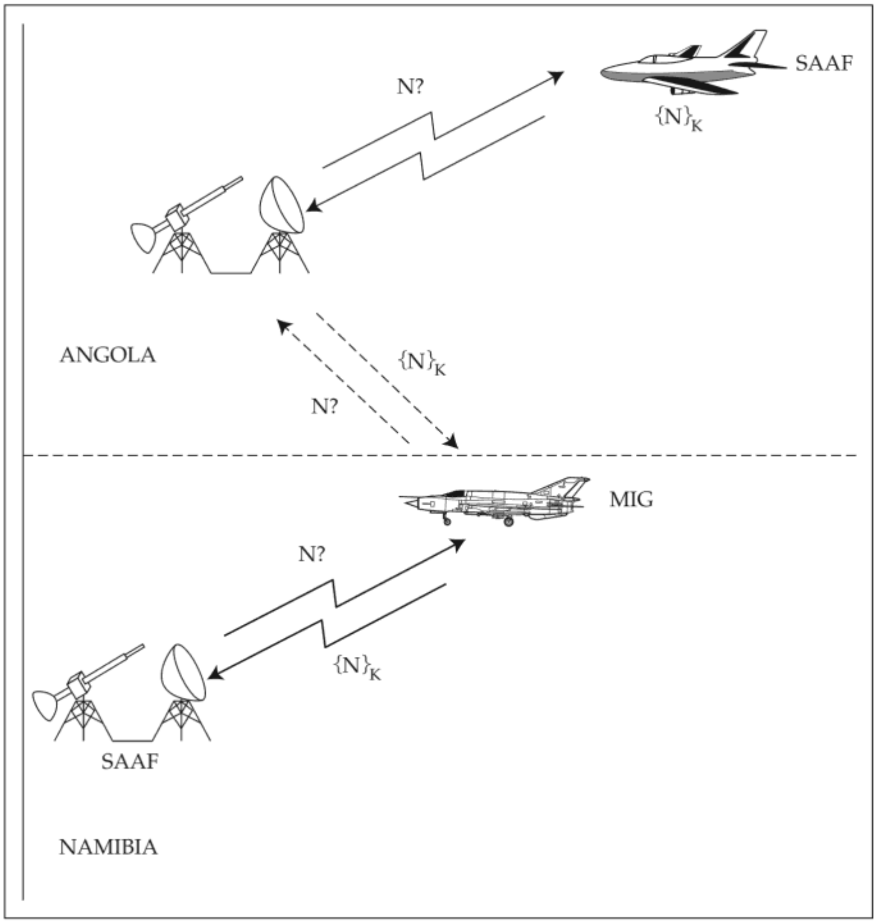


**Figure 4.2:** The MIG-in-the middle attack

In any case, the tale gives us another illustration of the man-in-the-middle attack. The relay attack against cars is another example. It also works against password calculators: the phishing site invites the mark to log on and simultaneously opens a logon session with his bank. The bank sends a challenge; the phisherman relays this to the mark, who uses his device to respond to it; the phisherman relays the response to the bank, and the bank now accepts the phisherman as the mark.

Stopping a middleperson attack is harder than it looks, and may involve multiple layers of defence. Banks typically look for a known machine, a password, a second factor such as an authentication code from a CAP reader, and a risk assessment of the transaction. For high-risk transactions, such as adding a new payee to an account, both my banks demand that I compute an authentication code on the payee account number. But they only authenticate the last four

One example we saw already is the prepayment meter that remembers only the last ticket it saw, so it can be recharged without limit by copying in the codes from two tickets *A* and *B* one after another: *ABABAB*.... Another is when dishonest cabbies insert pulse generators in the cable that connects their taximeter to a sensor in their taxi's gearbox. The sensor sends pulses as the prop shaft turns, which lets the meter work out how far the taxi has gone. A pirate device can insert extra pulses, making the taxi appear to have gone further. A truck driver who wants to drive faster or further than regulations allow can use a similar device to discard some pulses, so he seems to have been driving more slowly or not at all. We'll discuss such attacks in the chapter on 'Monitoring Systems', in section 14.3.

As well as monitoring systems, control systems often need to be hardened against message-manipulation attacks. The Intelsat satellites used for international telephone and data traffic have mechanisms to prevent a command being accepted twice – otherwise an attacker could replay control traffic and repeatedly order the same maneuver to be carried out until the satellite ran out of fuel [1529]. We will see lots of examples of protocol attacks involving message manipulation in later chapters on specific applications.

## 4.5 Changing the environment

A common cause of protocol failure is that the environment changes, so that the design assumptions no longer hold and the security protocols cannot cope with the new threats.

A nice example comes from the world of cash machine fraud. In 1993, Holland suffered an epidemic of 'phantom withdrawals'; there was much controversy in the press, with the banks claiming that their systems were secure while many people wrote in to the papers claiming to have been cheated. Eventually the banks noticed that many of the victims had used their bank cards at a certain filling station near Utrecht. This was staked out and one of the staff was arrested. It turned out that he had tapped the line from the card reader to the PC that controlled it; his tap recorded the magnetic stripe details from their cards while he used his eyeballs to capture their PINs [55]. Exactly the same fraud happened in the UK after the move to 'chip and PIN' smartcards in the mid-2000s; a gang wiretapped perhaps 200 filling stations, collected card data from the wire, observed the PINs using CCTV cameras, then made up thousands of magnetic-strip clone cards that were used in countries whose ATMs still used magnetic strip technology. At our local filling station, over 200 customers suddenly found that their cards had been used in ATMs in Thailand.

Why had the system been designed so badly, and why did the design error persist for over a decade through a major technology change? Well, when the standards for managing magnetic stripe cards and PINs were developed in the early 1980's by organizations such as IBM and VISA, the engineers had made two assumptions. The first was that the contents of the magnetic strip – the card number, version number and expiration date – were not secret, while the PIN was [1303]. (The analogy used was that the magnetic strip was your name and the PIN your password.) The second assumption was that bank card equipment would only be operated in trustworthy environments, such as in a physically robust automatic teller machine, or by a bank clerk at a teller station. So it was 'clearly' only necessary to encrypt the PIN, on its way from the PIN pad to the server; the magnetic strip data could be sent in clear from the card reader.

Both of these assumptions had changed by 1993. An epidemic of card forgery, mostly in the Far East in the late 1980's, drove banks to introduce authentication codes on the magnetic strips. Also, the commercial success of the bank card industry led banks in many countries to extend the use of debit cards from ATMs to terminals in all manner of shops. The combination of these two environmental changes destroyed the assumptions behind the original system architecture. Instead of putting a card whose magnetic strip contained no security data into a trusted machine, people were putting a card with clear security data into an untrusted machine. These changes had come about so gradually, and over such a long period, that the industry didn't see the problem coming.

## 4.6 Chosen protocol attacks

Governments keen to push ID cards have tried to get them used for many other transactions; some want a single card to be used for ID, banking and even transport ticketing. Singapore went so far as to experiment with a bank card that doubled as military ID. This introduced some interesting new risks: if a Navy captain tries to withdraw some cash from an ATM after a good dinner and forgets his PIN, will he be unable to take his ship to sea until Monday morning when they open the bank and give him his card back?

Some firms are pushing multifunction authentication devices that could be used in a wide range of transactions to save you having to carry around dozens of different cards and keys. A more realistic view of the future may be that people's phones will be used for most private-sector authentication functions.

But this too may not be as simple as it looks. The idea behind the 'Chosen Protocol Attack' is that given a target protocol, you design a new protocol that will attack it if the users can be inveigled into reusing the same token or crypto key. So how might the Mafia design a protocol to attack the authentication of bank transactions?

Here's one approach. It used to be common for people visiting a porn website to be asked for 'proof of age,' which usually involves giving a credit card number, whether to the site itself or to an age checking service. If smartphones are used to authenticate everything, it would be natural for the porn site to ask the customer to authenticate a random challenge as proof of age. A porn site might then mount a 'Mafia-in-the-middle' attack as shown in Figure 4.3. They wait until an unsuspecting customer visits their site, then order something resellable (such as gold coins) from a dealer, playing the role of the coin dealer's customer. When the coin dealer sends them the transaction data for authentication, they relay it through their porn site to the waiting customer. The poor man OKs it, the Mafia gets the gold coins, and when thousands of people suddenly complain about the huge charges to their cards at the end of the month, the porn site has vanished – along with the gold [1034].

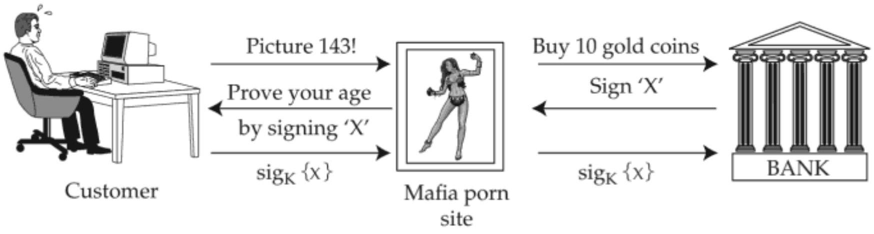


**Figure 4.3:** The Mafia-in-the-middle attack

In the 1990s a vulnerability of this kind found its way into international standards: the standards for digital signature and authentication could be run back-to-back in this way. It has since been shown that many protocols, though secure in themselves, can be broken if their users can be inveigled into reusing the same keys in other applications [1034]. This is why, if we're going to use our phones to authenticate everything, it will be really important to keep the banking apps and the porn apps separate. That will be the subject in Chapter 6 on Access Control.

In general, using crypto keys (or other authentication mechanisms) in more than one application is dangerous, while letting other people bootstrap their own application security off yours can be downright foolish. The classic case is where a bank relies for two-factor authentication on sending SMSes to customers as authentication codes. As I discussed in section 3.4.1, the bad guys have learned to attack that system by SIM-swap fraud – pretending to the phone company that they're the target, claiming to have lost their phone, and getting a replacement SIM card.

## 4.7 Managing encryption keys

The examples of security protocols that we've discussed so far are mostly about authenticating a principal's name, or application data such as the impulses driving a taximeter. There is one further class of authentication protocols that is very important – the protocols used to manage cryptographic keys.

### 4.7.1 The resurrecting duckling

In the Internet of Things, keys can sometimes be managed directly and physically, by local setup and a policy of *trust-on-first-use* or TOFU.

Vehicles provided an early example. I mentioned above that crooked taxi drivers used to put interruptors in the cable from their car's gearbox sensor to the taximeter, to add additional mileage. The same problem happened in reverse with tachographs, the devices used by trucks to monitor drivers' hours and speed. When tachographs went digital in the late 1990s, we decided to encrypt the pulse train from the sensor. But how could keys be managed? The solution was that whenever a new tachograph is powered up after a factory reset, it trusts the first crypto key it receives over the sensor cable. I'll discuss this further in section 14.3.

A second example is Homeplug AV, the standard used to encrypt data communications over domestic power lines, and widely used in LAN extenders. In the default, 'just-works' mode, a new Homeplug device trusts the first key it sees; and if your new wifi extender mates with the neighbour's wifi instead, you just press the reset button and try again. There is also a 'secure mode' where you open a browser to the network management node and manually enter a crypto key printed on the device packaging, but when we designed the Homeplug protocol we realised that most people have no reason to bother with that [1439].

The TOFU approach is also known as the 'resurrecting duckling' after an analysis that Frank Stajano and I did in the context of pairing medical devices [1822]. The idea is that when a baby duckling hatches, it imprints on the first thing it sees that moves and quacks, even if this is the farmer – who can end up being followed everywhere by a duck that thinks he's mummy. If such false imprinting happens with an electronic device, you need a way to kill it and resurrect it into a newborn state – which the reset button does in a device such as a LAN extender.

### 4.7.2 Remote key management

The more common, and interesting, case is the management of keys in remote devices. The basic technology was developed from the late 1970s to manage keys in distributed computer systems, with cash machines being an early application. In this section we'll discuss shared-key protocols such as Kerberos, leaving public-key protocols such as TLS and SSH until after we've discussed public-key cryptology in Chapter 5.

The basic idea behind key-distribution protocols is that where two principals want to communicate, they may use a trusted third party to introduce them. It's customary to give them human names in order to avoid getting lost in too much algebra. So we will call the two communicating principals 'Alice' and 'Bob', and the trusted third party 'Sam'. Alice, Bob and Sam are likely to be programs running on different devices. (For example, in a protocol to let a car