Introduction to Information Security

[Last compilation: 5th October 2018]

$$\begin{aligned} \mu v_0 \exp(-\mu z^*)\lambda t = 1 \quad &\Rightarrow \quad \exp(-\mu z^*) = \frac{1}{\lambda \mu v_0 t} \\ &\Rightarrow \quad -\mu z^* = -\log(\lambda \mu v_0 t) \\ &\Rightarrow \quad z^* = \frac{1}{\mu}\log(\lambda \mu v_0 t) \end{aligned}$$

Illustration: Gordon–Loeb model for optimal risk coverage (cf. Section 1.2).

Rémi Géraud,
CentraleSupélec 2018–2019

Introduction

About this book

There are several obstacles to writing a book that claims to teach security. One reason is that any book is necessarily outdated as soon as it goes to press. Another is that security is more often learned than taught; through experience rather than reading.

Nevertheless, students need a reference, and some of what the community has gathered in the last decades is worthy of conceptualisation. It is the object of this book to give a *flavour* of security: What it means, why we care about it, and how in practice we try to guarantee it.

We cannot hope to deal with the topic in its whole breadth. Rather, starting from scratch, we will focus on the everyday world around us. A leading thread will thus be that there is already a lot we can do with whatever we have at hand. In the topics that we do investigate in some depth, on top of theory and examples, there will be many references for the readers interested to go further.

As a general rule, we will ask questions instead of providing answers. That being said, a few general principles will be highlighted:

- *There is no silver bullet.* There is no such thing as a solution to all the problems we will meet, that would work flawlessly and successfully all the time. Managing security is not so much providing strong guarantees as it is managing risks and sometimes even accepting them.
- *Security is a global question, and a changing one.* It must thus be addressed globally, at the very least by adapting to the changing conditions — be they technological, legal, etc.
- *A fundamental aspect of security is preparedness and resource management.* As such one must develop a sense of anticipation and acumen regarding events to come and adversaries to face.
- *Technology is an inherent part of the question, and a necessary one, but it is not the core concern.* Rather, at every step there are human interests and actions, so that in particular the goal of security is to enable — and not hamper — activity.
- *It is often necessary to adopt the adversary's viewpoint.* One cannot hope in particular to devise good defences without precise knowledge of how and why attacks work.
- *Laws should not be forgotten*; either to avoid engaging in harmful behaviour, or to protect oneself from abuse. As the legal framework is evolving fast, it is important to find one's way in the legalese corpus.

Version

This is version 3.1 of this book, last compiled 5th October 2018.

Acknowledgements

The author wishes to express his most grateful thanks to VIA Centrale Réseaux (p2010 onwards) and VIAREZO for its support and trust. Credit is also due to those who insisted that my introductory talks should be turned into a lecture, which happened thanks to Alexis Benoist, Julien Escribe, and Jean-Marie Détriché.

Conversations participated in many ways to make this course richer, hopefully more interesting and less incorrect. In particular, students from the previous years have provided encouraging and useful feedback.

Contents

Part I

Introduction

Chapter 1

What is information security?

In the last few years, it has become salient how much of the world stands in the throes of increasingly violent, obstinate, subtle, and professional cyberattacks. Such attacks are often known to the public at large thanks to extensive, if superficial, media coverage of some particularly striking events. The dire reality is that such attacks are only a sparkle from the iceberg, a glimpse at what is happening everyday in otherwise mundane circumstances: the ubiquitous rise in power of global adversaries targeting companies, administrations, and individuals.

It thence behooves us to understand their motivations and tools, and to set out to find appropriate protections. This is a very broad and blurry definition of security, albeit one which corresponds to the intuition of many.

In this chapter, we formalise the question of security more precisely, in terms of *risks* and risk *mitigation*. This enables us to effectively delineate security, and gives us tools to better understand the adversarial landscape.

1.1 Defining and measuring risks

We will define risk as the expected loss of something of value, usually money, over a given period of time. A common expression of risk is as the product

$$\text{risk} = \text{loss} \times \text{probability}.$$

which ought to be summed over all possible events. Such is the approach outlined in standards, including the ISO 27000 family of documents related to information security. As an illustration, if we know that a faulty lightbulb would cause a loss of 1000 €, and that the probability of such a failure is 0.03%/year, then the associated risk would be

$$1000\,€ \times 3.10^{-4} = 0.3\,€ \quad \text{(per year)}.$$

In reality, how much of this do we accurately know? In terms of probability, we may perform experiments, and maybe in some cases obtain a reasonably sound estimate; but in terms of losses the situation is not so straightforward.

Indeed, incidents never happen in a vacuum: on top of the immediate loss (e.g., the cost of replacing the broken lightbulb), there are indirect losses: additional personnel required for maintenance, lawsuits, loss of reputation or trust from investors, etc. It is not possible to measure all these ramifications in controlled conditions, in particular because they intertwine with business logic and value creation, but also because the outfall may be stretched across a long time and therefore hard to pin to a particular cause.

As a result any risk estimation is necessarily approximate, and incorrect to some degree. For this reason, we will keep in mind the necessity to *reevaluate regularly* our estimations.

It should be obvious that risks are always there, that they come with any activity; hopefully they are kept low. But to ensure this, we need to *know* what risks we undergo, and estimate them *methodologically*.

Information-related risks

In the context of information, we can rely on a traditional classification of risks in three large categories:

- *Availability*. This category includes all incidents that may prevent access to information; by extension, we include any kind of service interruption. Such incidents may be very costly, for instance on production lines or large websites.
- *Integrity*. This category includes all incidents that may alter information, rendering it either unusable or unreliable; by extension, we include any kind of attempt at hiding (or altering) the original source and path of information (authenticity and traceability).
- *Confidentiality*. This category includes all incidents that may allow information to be learned by unauthorised parties.

Most information-related risks can be placed into one of these categories. There are other possible categories, but these three are certainly fundamental. One mnemonic is *CIA*: confidentiality, integrity, availability.

This gives us a first tool to investigate risks: look at those three questions in order, understanding how they affects key organisational assets, and estimating their likelihood (based on either experience or insight). This also helps us read through reports of incidents, focusing on how informational assets are affected by cyberattacks and better understanding the attackers' ultimate motives. Finally, this will help us decide which countermeasures are most appropriate and most effective in a given situation.

Example: NotPetya 2017

> ***Note:*** *this section is illustrative and may be skipped at first.*

Starting in late June 2017, a series of powerful cyberattacks targeted websites of Ukrainian organisations, including banks, ministries, newspapers, and electricity firms. (To a lesser extent this also impacted France, Germany, Italy, Poland, Russia, United Kingdom, the United States and Australia.) Although the attack quickly halted (in about 24 hours) it was designed to cause maximum damage, with Ukraine being the main target. The attack came on the eve of the Ukrainian Constitution Day, celebrating the anniversary of the approval of the Constitution of Ukraine on 28 June 1996. Most government offices would be empty, allowing the cyberattack to spread without interference.

The cyberattack was based on a modified version of the EternalBlue exploit[1] to access computers, encrypt their files, and ask for USD 300 in Bitcoin in exchange for decryption — This is a lie: as in fact the malware did not encrypt but destroyed the files, in particular accounting data.

[1] This exploit was designed by the NSA, and leaked to the public in 2016 as part of the ShadowBroker's trove. Other variations on this exploit include the WannaCry ransomware.

As a result of this attack, the radiation monitoring system at Chernobyl nuclear power plant went offline; several ministries, banks, metro systems and state-owned enterprises (Boryspil International Airport, Ukrtelecom, Ukrposhta, State Savings Bank of Ukraine, Ukrainian Railways) were also affected.

The malware spread through a popular tax accounting software; it seems that there was a backdoor in this software as early as April 2017. The software editor is now facing criminal responsibility, despite claiming they are themselves victim of the attack.

Risk mitigation approaches

Facing risks, we may choose several *mitigation strategies*:

- *Avoidance*. By not engaging in the risky activity (or by ceasing it), we reduce the likelihood of an incident, and thereby the risk.
- *Transfer*. We may find someone willing to take on the risk on themselves, in exchange for a regular (i.e., risk-free) fee. This is typically how insurance works.
- *Control*. By understanding the inicident's nature, we can try to reduce its likelihood, or the impact it has when happening, thus reducing the associated risk.
- *Acceptance*. We may decide that the risk is worth taking, and not try to reduce it.

Choosing any of these strategies has a cost, and affects the risk profile. For instance, avoiding an activity reduces the risk, but also prevents us from reaping any benefits from the activity; and trying to prevent risks by using a control strategy might require an extensive and costly research phase.

1.2 Security as an economic question

We are now equipped to define security from *an economic angle*:

> *Security is the minimisation of (financial) losses.*

More precisely, security is achieved by adopting mitigation strategies in such a way that the total loss, due to risks and due to mitigation strategies, is minimal (see Figure 1.1).

How much confidence should we give to that informal model? This is a good question, especially when risk estimation is far from an exact science, and mitigation cost itself is tricky to measure.

In general, it is good practice to try and formalise security questions mathematically, using simplifying assumptions when they allow us to better understand the problem. Hopefully, the resulting solution is still reasonably realistic to shed some light on the actual problem at hand.

One should be careful to *take informed decisions* using mathematical models when they are appropriate, but neither be overly confident nor overly doubtful of their real-world efficacy.

The Gordon–Loeb model

> ***Note:*** *this section is illustrative and may be skipped at first.*

We consider the following simple model: there are three parameters

- $\lambda > 0$, la financial loss when an incident happens;

- $t \in [0, 1]$, the probability of an attack;
- $v \in [0, 1]$, the probability that an attack leads to an incident (i.e., a successful attack).

With these notations, the risk is $vt\lambda$. We will consider that, by investing in various protection mechanisms, we can make attackers less successful, i.e., make v lower. The probability of a successful attack after investing an amount z of money in protection is $v(z) \leq v = v(0)$ (we assume that it is never a bad idea to invest in protection). We consider for simplicity that $v(z)$ is differentiable.

We consider an additional assumption, that there are diminishing returns: mathematically, $'v(z) \leq 0$. However, by investing more, it is always possible to lower the risk, although this may require an infinite amount of money to achieve zero risk: $\lim_{z\to\infty} v(z) = 0$.

This abstract and somewhat general model of the economics of security already provides interesting insights. For instance, we may ask: what is the optimal amount z^* of investments that minimise total loss?

Compared to a situation where we do not protect ourselves at all, spending z on protections results in a gain

$$G(z) = (v(0) - v(z))\,\lambda t - z$$

This is a concave function because of the conditions imposed on $v(z)$. Therefore $G(z)$ has a maximum which we can find by cancelling its derivative:

$$G'(z) = 0 \qquad \Rightarrow \qquad -\frac{\mathrm{d}v_z}{\mathrm{d}z}(z^*)\lambda t = 1$$

The knowledge of $v(z)$ then enables us to compute z^*, the optimal investment.

What is remarkable is that in many cases, z^* is of the order of $\frac{1}{e}v(0)t\lambda$, i.e., about a third of the initial risk estimation. This generality may be optimistic in real scenarios, but it makes the Gordon–Loeb model interesting. While certainly a crude model, this shows there is untapped potential for even simple approaches to give insight about complex situations.

As an illustration, assuming $v(z) = v(0)\exp(-\mu z)$, for some constant μ, we have

$$\frac{\mathrm{d}v}{\mathrm{d}z}(z) = -\mu v(0)\exp(-\mu z).$$

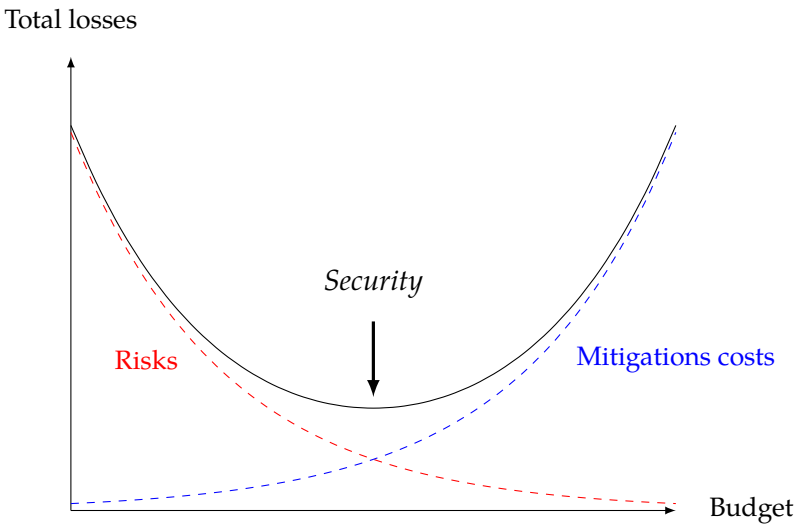

Figure 1.1: An informal description of what security is about.

Therefore

$$\begin{aligned} \mu v(0)\exp(-\mu z^*)\lambda t = 1 \quad &\Rightarrow \quad \exp(-\mu z^*) = \frac{1}{\lambda\mu v(0)t} \\ &\Rightarrow \quad -\mu z^* = -\log(\lambda\mu v(0)t) \\ &\Rightarrow \quad z^* = \frac{1}{\mu}\log(\lambda\mu v(0)t) \end{aligned}$$

Now we can see that $z^* \leq \frac{1}{e}v(0)t\lambda$. Indeed,

$$\begin{aligned} & z^* \leq \frac{1}{e}v(0)t\lambda \\ \Leftrightarrow \quad & \frac{1}{\mu}\log(\lambda\mu v(0)t) \leq \frac{1}{e}v(0)t\lambda \\ \Leftrightarrow \quad & \frac{\log(\lambda\mu v(0)t)}{\lambda\mu v(0)t} \leq \frac{1}{e} \\ \Leftrightarrow \quad & F(\lambda\mu v(0)t) \leq F(e) \end{aligned}$$

where $F(x) = \log(x)/x$, and the above inequality always holds, because F has only one maximum, at $x = e$. As a result, $F(\lambda\mu v_0 t) \leq F(e)$, with equality if and only if $\lambda\mu v_0 t = e$.

Consequences of the economic model

Let's reflect on the consequences of defining security through economics.

- *"Perfect security" makes no sense*. Specifically, we should not hope to reduce risks to zero, as this would require spending large amounts of money, and thus be counter-productive. In particular, security is not binary (secure/insecure), but rather a continuum of losses that (hopefully) can be kept to a minimum by adopting the right strategy and cunningly investing.

- *Security is not in the product*. It makes no sense to talk about a product or service that is "secure" in itself; security only makes sense for an organisation, facing risks and investing in countermeasures. Products play a role in this, by augmenting or reducing the likelihood or impact of incidents, but they are only components in a larger information system whose security has to be understood at a global level.

- *We do not know much: we must learn, try, and update*. Most of the parameters that would allow us to settle for the optimal stragegy are unknown to us: likelihood and actual cost of incidents, effectiveness of countermeasures, etc. A good security strategy is therefore *dynamic*: we learn from mistakes, update our estimation of risks and countermeasures, and do better next time.

- *The question is not 'if' but 'when'*. Security incidents *will happen*, and it would be irresponsible to think otherwise. Therefore we should make sure that *when they happen*, we are ready. Now there might be very unlikely incidents, and we may *choose to accept* the risk they pose, but never *ignore* it.

Although the economic outlook is interesting and enlightening, it only gives a coarse-grained and very global picture of security. To improve on that picture, we need to understand better three key aspects: *adversaries*, *security properties*, and *technology*. As you can already see, *security is a multi-faceted and interdisciplinary question*.

1.3 Further reading

- For further information about the history of risk and its (ongoing) conceptualisation, refer to Bernstein's book [Ber96].
- For further information on risk perception and psychology in the context of security, see Schneier's article [Sch08], or even Kahneman and Tversky's seminal paper on prospect theory [KT79].
- For a sociological perspective, the monograph by Lidskog and Sundqvist [LS12]. For a broader look into sociology, see the book by Smelser and Swedberg [SS10]. A different approach is discussed by Odlyzko [Odl03].
- The seminal paper by Gordon and Loeb goes into more details and explores more situations about the economics of security investment [GL02]. The interested reader can refer to Anderson and Moore's survey [AM07], and references therein.
- The ISO/IEC 27000 family, entitled *Information security management systems*, provides a standardised framework to information risk management. The documents are not free, but can be accessed for a fee on the ISO website.[2]
- For concrete examples of CIA risks, there is a reasonably recent survey by Androulidakis in the context of mobile phones [And16].

[2] https://www.iso.org/isoiec-27001-information-security.html

Chapter 2

Adversaries and threats

— 26 Sep 2017

In the previous chapter, we introduced the notion of risk. But risks do not appear out of thin air, they are not completely independent events unrelated to what we are doing. In fact, in security, there is always this *other player* opposing us: the *adversary*.

Adversaries are threats, in the sense that they increase risk. But most adversaries have strategies of their own: they are not mere statistical events, they are driven, active, and reactive. To understand this, we may want to look at what drives them.

2.1 What do adversaries want?

Of mice and men

Human desires are certainly complex, but some are much stronger than others. Some of the most well-known include:

- *Money*. Financial gain; While this might sound stereotypical, it is backed by statistics around the world.
- *Ideology*. Another form of power, ideology (mostly political or religious) motivates many actions.
- *Coercion/Control*. The power to force someone into doing something.
- *Ego/Excitation*. Last but not least, flattering or entertaining oneself.

One may remember the acronym: *MICE*. Naturally this typology is simplistic, and shouldn't be taken at face value. But it gives an overview and may help reading a bit into the adversary's mind. The list above applies very broadly, and subsumes many conflicts around the globe.

Let's try to make it more precise, by introducing a key concept for this course: *information*. We haven't defined information yet, at least not formally. Nevertheless,

- Companies make money in many ways; they may sell services, goods, intellectual property, etc. As such they either use or sell information. Should something happen to this information, it would result is a net loss for the victim company.
- Internet has replaced traditional media, transgressing national frontiers and political taboos. Social media favour broadcasting of individual, unfiltered opinions, videos, and other information.

- Industrial systems, and connected objects in general, use networks to communicate, provide status information, and receive orders. Such networks transport information that should not be tampered with, lest incidents may happen.

What should be visible from the above, is that *information is a prime target*, whatever form it takes. One the face of it, information is fundamental to any human activity. But since so much is at stake, criminals are attracted by the prospect or stealing (or otherwise destroying) others' fortunes, fame, or power — or acquiring these for themselves.

An example: Aldrich Ames

> ***Note:*** *this section is illustrative and may be skipped at first.*

The following example serves to show both the relevance and the limitations of the above model to capture the leverage of MICE on information.

In 1985, Aldrich Ames[1] had been working for the United State's Central Intelligence Agency for about 15 years. He just divorced from his wife Nancy after having an affair with a Colombian informant. He started drinking, and his mistress turned out to be a heavy spender: Ames quickly found himself in a difficult financial situation.

In April 1985, he walked into the Soviet embassy in Washington, D.C, with the stated intent of avoiding bankruptcy. He was ready to trade "useless" information with the Soviet union in exchange for USD 50,000 (about $114 400 in 2017, or 97,040 €). Ames gave the embassy's KGB chief of intelligence (Victor Cherkashin) a list of people the CIA suspected of being double agents. The money should be enough for Ames to cover his debts and he expected never to return to the Soviet embassy.

But Cherkashin had other plans.[2] Having received Ames' information, Cherkashin told him that this was very brave, but put Ames in danger. In order to protect him, Cherkashin would need to know who is most likely to put Ames at risk, in particular any CIA double agent that might divulge his collaboration.

> *[Ames] took out a notepad and paper and began writing down a list of names. He tore out the page and handed it to me. [...] That piece of paper contained more information about CIA espionage than had ever before been presented in a single communication. It was a catalog of virtually every CIA asset within the Soviet Union. Ames said nothing about whether the men he'd listed should be arrested or removed. "Just make sure these people don't find anything out about me," he said.*[CF08, pp. 27–29]

Ames continued to work for the KGB until he was arrested in 1994. It is estimated that the information he provided led to the compromise of at least a hundred U.S. intelligence operations and to the execution of at least ten U.S. sources

2.2 Who are the adversaries?

We drew above a simplistic picture of the situation. Reality is more nuanced.

First, the motivations we highlighted do not affect the sole "criminals" — companies make profit, advertise, and feel good about themselves, and there is usually nothing wrong about that. And companies compete: They want to rule over a market, they want to build an image. So what

[1] For more information about this story, see for instance Earley's book [Ear97].
[2] Cherkashin himself described this encounter in his biography [CF08].

happens in fact is that everyone fights the battle. Some people play fair, and some people cross the boundary defined by laws, when they exist or apply.

Typology: From $\bot$ to $\top$

We could thus organise adversaries according to how much they cross the line:

Null adversary: The 'null' adversary, represented by the symbol $\bot$, is accidental in nature. Events such as rain, earthquakes, lighting storms or cricket invasions fall into this category. Tripping on a wire, stroking a key with one's palm, losing balance also count as acts of the null adversary. Such an adversary is not 'intentionally' causing damage, and in most cases its actions are not illegal (or even reprehensible). This doesn't mean they don't have consequences. But since the null adversary does not *try* to cause damage, it is easily thwarted, and follows a reasonably predictable path.

Examples: Hot summer, rusty lock, flat tire, babies putting their fingers in the electric socket.

Typical size: 0 to 1.

Typical target: Random old equipment.

Typical actor: Random.

Weak adversary: A weak adversary relies on available tools and knowledge, but has neither special training nor many technical means. Opportunistic in nature, the weak adversary targets easy preys. Such an adversary can only make meager profit from this activity, and therefore only does so 'on the side'. They may act illegaly, but their limited knowledge and nuisance power means they do not cause immense loss.

Examples: Script kiddies, scammers, individual hackers.

Typical size: 1 to 3.

Typical target: Random people who respond to spam, and open the attached PDF.

Typical actor: Self-taught hobbyist.

Strong adversary: The strong adversary is organised in nature. Organisation means that this adversary can develop its own tools, become resilient, and design strategies on the mid to long term. It also means that enough profit can be made to sustain this activity, which becomes full-time. Organised crime is the most harshly punished form of criminality, but also one of the hardest to identify and stop.

Examples: Hacking Team, EAR IT, ...

Typical size: 1 to 50.

Typical target: Specific targets that haven't patched a recent vulnerability.

Typical actor: Professional/Engineer (in teams).

Strongest adversary: At the other end of the spectrum, the strongest adversary (represented by the symbol $\top$) is result-driven. Almost always nation-backed, such organisations engage in extremely targeted attacks, possess extensive technical and human means, nearly unlimited financial resources, and often an ironclad legal cover.

Examples: State agencies.

Typical size: 10 to 200+

Typical target: High-profile, attacked by coordinating zero-day exploits.

Typical actor: Professional/Engineer/Researcher (in teams).

Geography: Cold wars, Dark markets, Arab springs

A metal cold war

If we now turn our attention to geography, for instance by counting the number of victims per year and per country, then we see that the United States of America, the Russian Federation, and China are holding the top three positions. Perhaps surprisingly enough, when trying to identify *where* these attacks come from, we find that they almost always originate... from the very same countries.

In terms of volume at least, what we see is that information systems are used to wage an underground war between these superpowers. In recent years these countries have made clear their intention to use the Internet for military purposes, and all possess state agencies dedicated to this purpose.

Volume however does not distinguish between the different adversaries we identified earlier. It shows that the economic impact of information-related crime, or at least the bulk of it, is a first-world concern. But we lack precise information on what the stronger forms of adversaries do, and they may be much more active in areas of the world where information security is less of a priority.

The first event that may be counted as an act of Internet warfare is the 2007 distributed denial of service (DDoS) attack against Estonia. For three weeks during the spring season, the whole country was disconnected from the Internet and several key infrastructures were left unresponsive. But the point of no return was reached with an Israel–US coalition designed and used the Stuxnet attack in 2009 against Iran. This attack, a continuation of the US-driven Olympic Games programme (2006), targeted an uranium-enriching plant and caused partial destruction of strategic equipment — using nothing but a computer worm.

This raised awareness worldwide and more than a dozen extremely advanced spying malwares have since been discovered, each of which is probably the brainchild of a state agency. The breadth of such intrusions — made even more visible after Edward Snowden's publication of NSA material in 2013 — has yet to be measured in its entirety.

Example: Operation Aurora 2010.

> ***Note:*** *this section is illustrative and may be skipped at first.*

On January 12, 2010, Google announced it has been victim of an intrusion in December 2009. Twenty additional companies were thought to have been impacted at that time (in fact, there was more, including Adobe, Juniper Networks, Yahoo, Symantec, Northrop Grumman and Dow Chemical). The very sophisticated intrusion combined several zero-day vulnerabilities from Microsoft Internet Explorer and Adobe Flash (Microsoft has been aware of the vulnerability for several months before the attack took place). Attackers got access to confidential information, and altered the source code of several software components. They also downloaded and executed programs. The exploits used during this attack are now bundled with the `metasploit` toolkit.

Google claimed that this attack originated from China, and subsequently ceased all activities in that country. Chinese officials denied any implication and accused Google of partaking into a White House made conspiracy. When WikiLeaks made diplomatic cables public on December 4, 2010 (see also *The Guardian* et *The New York Times*), it was indeed confirmed that the Chinese government had a responsibility in the incident, codenamed *Operation Aurora*. According to *The*

Guardian, the attack was coordinated by a high-ranking official of the Politburo, who had felt offended when he tried to search his name only to find articles criticising him.

Analyses performed by McAfee, Symantec, Dell, Sophos, as well as several independent researchers, confirmed without doubt the Chinese origin of this attack, where the most likely actor is the PLA Unit 61398 (61398 部队). The Chinese government responded that it strongly opposed piracy. Without denying the existence of Unit 61398, the official stance is that China has nothing to do with operation Aurora.

Further reading about this topic include the Symantec report *The Elderwood Project*,[3] and the *Mandiant* report.[4]

Dark markets

In the same time, local police forces infiltrate and uncover black markets operated from the Internet. Such black markets naturally avoid drawing too much attention to them, and in particular try not to be referenced by well-known search engines — as a result they are often mediatised as constituting a 'dark web'. The URLs are often given in person or in very selective fora, which tends to keep these dark markets within the boundaries of a community or a country. The US dark market is the most visible and the most active — it is used by criminals and state agencies alike to buy tools and information, and is fairly open (anyone can come, buy or sell). The French or German underground markets are much more secretive, using reputation systems and 'anti-cop' measures, including rapidly-changing URLs. Markets also specialise — guns and drugs in France, tools and books in the US.

Internal conflicts.

In parallel, certain government adopt the position of an isolationist policy, trying to resist against what they claim is cultural invasion, political influence, deviance or extremism. This policy is implemented by setting up filtering and monitoring mecanisms, surveillance (including sometimes mass surveillance), censorship, and in some extreme cases some country simply cut the Internet cables to avoid communication with the outside world.

Victims of such abuse are often reporters and international observers, NGOs. Since around 2010 ('Cablegate' 2010, 'Arab spring' 2011, Snowden files 2012, SpyFiles 2013, DNC/DCCC 2016) the political role of such attempts at controlling information have become blatant, and the targets are very often populations from the least democratic countries.

The situation is of course evolving now, with some of the harshest countries now relaxing their grip (Egypt, Lybia); while other decided to strenghten theirs (United Kingdom, France).

2.3 Threat exposure

A natural notion is that we may face more risks than others, because we relate to threats in some specific way that others don't. This is informally captured by the concept of *threat exposure*: the closer, the more visible, the more symbolic we are, the higher the likelihood of attracting envy or scorn, and being attacked.

It is thus important to understand the neighbouring actors, both in terms of physical proximity and in terms of similar activities; to have a measure of one's visibility and exposure to the rest of the world; to understand what symbols we carry or what idea we embody — what matters in that question is how *others* see us, not how we see ourselves.

[3] http://www.symantec.com/content/en/us/enterprise/media/security_response/
[4] http://intelreport.mandiant.com/Mandiant_APT1_Report.pdf

For this reasons, financial institutions face more risks than some because they embody a form of capitalism or greed, because attacking them seems to promise large gains, and also because the financial world is wild with competition. But they face even more risks if they decide to build their headquarters in a relatively poor area, if they publicly disregard regulation or embrace their desire for money, or if they advertise. Conversely, some organisations are less exposed because their very existence is unknown to most.

An example: ÉDF vs. Greenpeace

> ***Note:*** *this section is illustrative and may be skipped at first.*

In 2008 the OCLCTIC[5] was leading an investigation after the national anti-doping laboratory detected a potential attack on the integrity of their systems.[6]

Investigators obtained a letter rogatory enabling them to arrest one of the participants, in Morocco. The forensic analysis of a hard drive[7], along with the person's testimony, revealed that he had in fact participated in several attacks, including against a law firm in Paris and against the Greenpeace association.

Both cases were known to the investigators, but they had not found enough evidence at the time to pursue. With this new information, they could connect the dots. In particular, in the Greenpeace case, data collected during the prosecution established that the Chief Security Officer from ÉDF had mandated a consulting firm[8], which contacted the Moroccan 'hacker'.

This man was self-taught, using information he could easily find on the Internet. He intruded[9] into Greenpeace's computers, in particular the e-mail inbox of the Program Director[10]. This information was then sent back to ÉDF.

When the police arrived at the man's home in Morocco, they found hundreds of CD-ROMs which the man kept as 'trophies', containing details about this operations. A copy of the Greenpeace *trophy* was also found in ÉDF's CSO's safe. As for the contract between ÉDF and the consulting firm, it never states the precise nature of the agreement and the judge finds therefore ÉDF complicit in the cyberattack..

During a first trial in 2011 (Tribunal correctionnel de Nanterre), ÉDF was sentenced *under the penal code* by virtue of articles 323-1 et 323-2 : one year of imprisonment for Pierre-Paul François and Pascal Durieux, ÉDF's CSOs; one year of imprisonment for Thierry Lorho, from the consulting firm; one year of imprisonment for Alain Quiros, the Moroccan amateur hacker; and a total of 2,200,000 € in fines and reparations, 1.5 millions being paid by ÉDF.

Appealing to that decision in 2013 (cour d'appel de Versailles), the sentence was maintained, except against the contre-amiral Pascal Durieux.[11]

---

[5] Office central de lutte contre la criminalité liée aux technologie de l'information, the French institution prosecuting cyberctime

[6] The investigation eventually revealed that there was indeed an attack; it had been comandeered by the coach training U.S. cyclist Floyd Landis, to try to hide that this athlete used illegal substances during the 2006 Tour de France.

[7] By virtue of the French Penal code procedure articles 230-1 to 230-5.

[8] Kargus Consultants, supposedly analysing open source data for technological watch.

[9] He only knew how to use the `Bifrost` trojan generator, which exploited the CVE-2005-4560 vulnerability in Windows systems.

[10] Yannick Jadot, today a Green European deputy.

[11] For more information about these events (in French): http://www.legalis.net/spip.php?page=jurisprudence-decision&id_article=3678.

2.4 Consequences of the adversarial landscape

Since security is about dealing with *adversaries* (and not, say, "natural" causes), we must be careful to include adversaries at the heart of our defence strategy. In particular, the following points can be made:

- *There is no such thing as "security", only security "against something"*. More specifically, we will always try to make precise both the adversaries' *goals* and their *means*; then we may say that we provide protections so that an adversary with these means cannot reach these goals in a reasonable amount of time.
- *Not all adversaries are the same*. Since we only have limited resources, our focus will be on understanding which categories of adversaries we can reasonably hope to repel.
- *Adversaries have objectives*. When these goals are economic in nature, we can reason in terms of economic rationality to thwart them off. This sheds light both on the means available to a defender to secure his infrastructure, and on the leverage lawmakers have to make crime less appealing.
- *Adversaries have strategies*. This means we cannot hope to get rid of them "easily" nor "once and for all" using off-the-shelf tools. That attackers will learn and adapt has always been true, but is increasingly visible.
- *Security must be by design*. This means that we cannot rely on the adversaries' ignorance; while it is always easy to build a system that *we* don't know how to attack, this is by no means a guarantee that no attack exists nor can be found by obstinate and specialised teams. Whenever possible, we will try to *design for security from the beginning*, having *proofs of security*
- *Security is a strategy game*. Technology is the chessboard, with its rules and limitations, but strategies and objectives are really (as of today!) decided by human beings, where creativity, experience, psychology, and politics can play a decisive role.

2.5 Further reading

- For an introduction to the geopolitical aspects of security, see (in French) Douzet [Dou14], and for more specific countries: Limonier [Lim14] or Harrel [Har13] (Russia), Douzet [Dou07] (China), Garrigues [GK12] (NATO). National agencies (DoD in the USA, ANSSI in France, etc.) also publish their official doctrines on the matter of cybersecurity.
- An example of how organised crime leverages technology can be found in Ferradi et al.'s forensic analysis [Fer+16].
- A thorough analysis of some aspects of the cybercriminal black market can be found in Thomas et al.'s paper [Tho+15], and in the RAND report on black markets.[12]

[12] See https://www.rand.org/pubs/research_reports/RR610.html

Chapter 3

A first look at the Internet

The Internet was initially designed to be a robust, fault-tolerant communication system. Since the vast deployment of the Internet in the early 1990s, increasingly many devices and institutions have been added to the network, and since the late 2000s, practically every aspect of modern life has a counterpart on the Internet.

What security properties are guaranteed by Internet communications? To properly address this question we must explore network communications in more details. But we can already say that the Internet was not designed against adversaries stronger than $\bot$, so that if any confidentiality, integrity, or availability property holds it probably is only by chance.

Since its inception, the Internet has no centralised governance, either in technological implementation or policies for access and usage. Only the overreaching definitions of the two principal name spaces in the Internet, the Internet Protocol address (IP address) space and the Domain Name System (DNS), are directed by a maintainer organisation: the Internet Corporation for Assigned Names and Numbers (ICANN).

3.1 Local networks

The smallest scale of a network consists in directly interconnected devices. For instance, they may all be connected to a switch or to a Wi-Fi hotspot. Such devices can already communicate with one another, they form a *local area network*. Devices on the same local network may receive (and in many cases, do receive) packets meant for another device. When that happens it is typical that one drops the corresponding packets, but it is a mere matter of correctly configuring one's network parameters to instead capture and keep these packets. While switches may adopt smarter behaviours and in certain cases be more specific when sending packets, there is no way for Wi-Fi emitters to avoid sending packets across their whole range.

Communicating over such a network is therefore not guaranteed to be confidential in any way. Furthermore, it is possible for any adversary to craft messages, or intercept and alter them; indeed nothing prevents this. Thus there is no integrity guarantee either. Finally, the physical layer itself may be disconnected, resulting in availability issues.

As we will later see, it is possible to some extent to integrate security mechanisms on top of this insecure channel, which is necessary if we are hoping to control information risks. It may sound surprising that such is not the case *already* and *by default*; in fact there are still obstacles ahead and some key mechanisms of what constitutes the Internet as we know it are still far from even *allowing* security mechanisms.

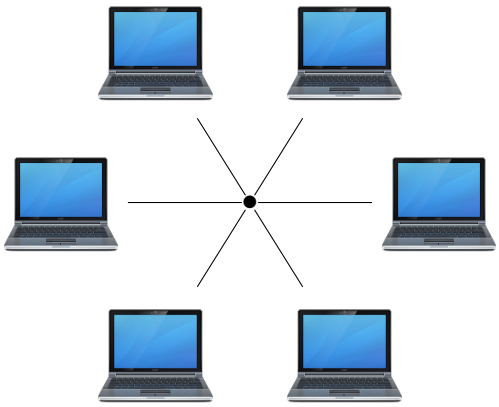

Figure 3.1: A local network consists in directly interconnected devices. This connection may be wired (as the above diagram implies) or wireless. Messages sent across this network can

3.2 Routing and the Internet

What gives the Internet its name is that it connects local networks together. This could be done in a very naive way by literally plugging them together (this would result in a world-wide local network), however doing so is very inefficient. Instead, the Internet relies on packet-switching and routing, to deliver messages from one local network to another, see Figure 3.2.

To achieve this, endpoints are given an IP address. When sending a packet, the destination IP is used by routers to find the shortest path and hand it, from hop to hop, until the appropriate local network is reached. In some case, a source IP may also be indicated, so that a response can be sent back.

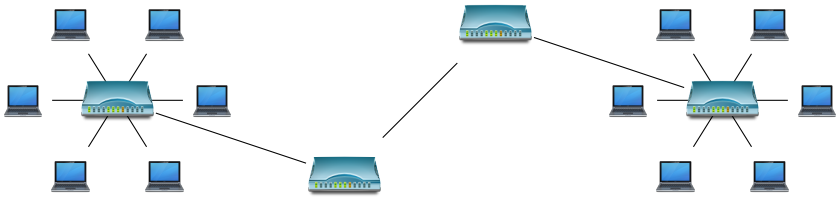

Figure 3.2: The network interconnection mechanism (inter-net) uses routers to find a path and transport packets to their destination.

While packets are not sent to *all* connected devices, routing is by no means a confidentiality guarantee. For starters, routers themselves could eavesdrop or alter all the packets they receive (and some do!), or simply drop packets. Thus there is no integrity guarantee here either. What routers do, however, is contribute a new means for availability to fail: if the route they choose is unavailable, incorrect, or too busy, then packets will not reach their destination.

In particular, a sender is free to specify *anything* for a source IP address. There are several situations in which this enables an adversary to learn information or even perform attacks, and in any case this means we should consider with caution any claim built from IP data. One way that this has been leveraged in recent years is to launch powerful *reflection attacks*: the attacker queries a fast and talkative server, with a forged source IP (the victim's); the victim then receives the full unsollicited response, which is typically orders of magnitude larger than the attacker's initial message. This attack, which can be distributed, amplifies the attacker's bandwidth and

Figure 3.3: Network congestion results in packets being dropped; alternatively, one may exhaust a computer's resources by having it wait for further information. One way to achieve this on older systems is to send many TCP `SYN` packets, as the target will open sockets, reply with a `SYN+ACK` packet, and wait for an `ACK` packet which will never arrive. In doing so, the target allocates resources, which cannot be used to respond to legitimate connections. In both scenarios, the result is a potential denial of service.

can quickly saturate their victims' network, preventing them from sending or receiving any information. This is an example of a (network-based, distributed) *denial of service* attack (DDoS).

Recent years have seen the rise in DDoS-as-a-service providers, who offer to slow down or cut a target's Internet access by launching such attacks in exchange for money. The methods may vary (more traditional connections, usage of botnets, reflection...) but the effect is always the same, with the victim facing packets coming from seemingly legitimate high-profile servers and eventually drowning under the sheer flow of unsollicited packets. In 2016, a record-breaking (botnet-powered) DDoS peaked at 1.5 Tb/s against the Dyn DNS provider, in effect disconnecting many of the major US websites for hours.

Another aspect of the lack of integrity guarantees of IP packets (and TCP/IP packets) is the ability for an adversary to alter or even hijack the exchanges between two devices. To be precise, TCP *does* have a checksum, but this only prevents against accidental noise (anyone can alter a packet, and recompute the correct checksum).

Finally, much information can be learned from the way a device reacts to specially crafted packets; indeed there is no reference implementation (and even if there were, implementers would probably deviate from it), resulting in shibboleths that an astute attacker can leverage. Such *fingerprinting* and *side-channel information* is essential to fine-tune attacks.

Port scanning and service identification

> ***Note:*** *this section is illustrative and may be skipped at first.*

TCP/IP is based on a client-server approach to communication. In that model, a "server" device is waiting for someone (a "client") to initiate communication. After an initial handshake (which merely tests that server and client can exchange messages), the connection is established and it is possible to send packets from one device to the other.

Now TCP/IP also introduces the *port* abstraction; since many applications may want to use the Internet connection for different reasons, using different higher-level protocols (e.g., video streaming, e-mails, web pages, etc.), it is handy to separate them. Thus every packet is attributed a *port number*, corresponding to one application. For instance, port number 80 is now usually associated with a HTTP server application, whereas port 23 would serve SSH, and 443 would be HTTPS. Although there is no strict association, most of the ports number have cemented around a de facto standard protocol, see Table 3.1.[1]

Combined together, the handshake and the port abstraction may reveal interesting information about a remote host. Indeed, assume we send a TCP `SYN` packet to some target, with some port

[1]Some of these ports are "officially" assigned by the IANA, but that doesn't prevent anyone from doing otherwise.

Table 3.1: Some very common port associations for TCP/IP.

Protocol	Shorthand	Typical port
File Transfer Protocol	FTP	20/21
Secure Shell	SSH	22
Telnet	Telnet	23
Simple Mail Transfer Protocol	SMTP	25
Hypertext Transfer Protocol	HTTP	80
Post Office Protocol	POP	110
Internet Message Access Protocol	IMAP	143
Border Gateway Protocol	BGP	179
Lightweight Directory Access Protocol	LDAP	389
HTTP over TLS	HTTPS	443
FTP over TLS	FTPS	989/990

number X. If there is an application waiting on this port, then a `SYN+ACK` reply will be sent, and we learn, at least, that there is an application waiting on this port. Sometimes, going through with the connection reveals even more information (e.g., the remote server sends us its name and version).

There are more clever ways to learn about which connections a remote host accepts (some of which we discussed during the lecture), which allow us to hide our origins or bypass firewall protections. But the bottom line is we learn something about what software is likely to run on our target (and possibly, which version). This information is valuable to an attacker, as part of a reconnaissance effort, and while looking for vulnerabilities. It may also help the adversary in learning new information, as for instance many protocols send passwords in the clear (`telnet`, `rsh`, `ftp`, `http`, ...).

In 2015, prior to the TV5Monde attack, a multimedia server used by journalists had its RDP port exposed to the Internet. Attackers noticed this during a port scan, and realised the RDP server was using default username and password. From this they could take control of this computer. (However, they quickly realised that this machine was not connected to the internal network, so they had to find another way).

The QUANTUM suite

> ***Note:*** *this section is illustrative and may be skipped at first.*

Starting around 2005, the U.S. National Security Agency (NSA) started using the QUANTUM suite of man-on-the-side capability. It relies on the ability to send a response faster than the legitimate server, impersonating it; this can be achieved in a number of ways, including having a compromised router that duplicates Internet traffic (typically HTTP requests) so that they go both to the intended target and to an NSA site.

This is often used in combination with other tools (such as NSA FOXACID or OLYMPUSFIRE) to insert malware in the response, or redirect the client to a malicious server.

In collaboration with the U.K. Government Communications Headquarters (GCHQ, program MUSCULAR), this allowed these agencies to intercept communications with many websites, including Hotmail, Linkedin, Facebook, Twitter, Yahoo, Gmail, and YouTube.

The QUANTUM suite more specifically provided the capabilities summarised in Table 3.2, all relying on the ability to race a legitimate reply.

Table 3.2: NSA QUANTUM suite capabilities, as of 2005; since the publication in 2013 of the NSA ANT catalog, many similar tools have been found to be used across the world.

QUANTUMINSERT	Hijack connection to redirect target to exploit server.
QUANTUMBOT	Hijack IRC bots and communications from bots.
QUANTUMBISQUIT	Facilitates attack against hard to reach targets.
QUANTUMDNS	DNS injection/redirection.
QUANTUMHAND	Targets Facebook users.
QUANTUMSKY	Denies access to a webpage by injecting/spoofing `RST` packets.
QUANTUMCOPPER	File download/upload disruption and corruption.

Quantum Insert has been used in the Middle East, but was also used in a controversial GCHQ/NSA operation against employees of the Belgian telecom Belgacom and against workers at OPEC, the Organization of Petroleum Exporting Countries. According to the NSA's internal documents, the "highly successful" technique allowed them to place 300 malicious implants on computers around the world in 2010.

Since this very simple attack relies on the lack of security guarantees at the TCP/IP level, anyone who can passively or actively monitor a network and send spoofed packets can perform QUANTUM-like attacks (in particular, any ISP can easily do this). As a result, since the 2013 publication of the NSA ANT catalog, describing the QUANTUM tool amongst others, researchers have identified many similar attacks being performed in the wild, as early as 1994. One of the way that the Great Firewall of China prevents access to websites is precisely by sending early `RST` packets to clients, similar to QUANTUMSKY.

The NSA is allegedly able to perform this attack on a large scale, which requires the capability to listen in on potentially high volumes of traffic.

3.3 Name and address resolution

For a vast majority of users however, Internet is not so much about IP addresses as it is about domain names: this is what they use in their browser to access websites, this is what they see when sending e-mails, etc. The task of matching a domain name, such as `www.guardian.co.uk` with an IP address is handled by a *domain name resolver*.

Because DNS is so central to most users, there are several copies of this information at different levels, so that one does not have to query a single server everytime one needs to access a website: this is a *cache hierarchy*. While such mechanisms improve latency somewhat, they also introduce leeway to attackers to learn information or interfere with normal operation. Indeed, the DNS protocol does not offer either confidentiality, integrity, or availability guarantees; some immediate consequences are:

- The possibility for an attacker to learn the contents (i.e., which websites are visited) of DNS requests, either directly or indirectly (using cache misses to learn that information);
- The possibility for an attacker to provide incorrect replies, possibly even before a request was made (*colorFireBrickcache poisoning*), thereby redirecting their target to any IP address; and the possibility to create forged requests, possibly redirecting the replies to a third party (this is a *reflection attack*).
- The possibility for an attacker to prevent access to a website by having the DNS requests be inaccurate or timed out, or simply by disconnecting the relevant DNS servers.

At the other end, a packet with some destination IP should be sent to a concrete device. This association is performed following the address resolution protocol (ARP), which also relies on a cache hierarchy. Cache poisoning therefore applies equally well and allows attackers, for instance, to redirect traffic in a local network.

An example: The Dyn DNS attack 2016

> ***Note:*** *this section is illustrative and may be skipped at first.*

On October 21, 2016, a targeted distributed attack against the Dyn DNS provider disconnected a large portion of popular websites, including Airbnb, Amazon, BBC, CNN, Comcast, Electronic Arts, Etsy, Fox News, The Guardian, GitHub, HBO, Heroku, Imgur, Netflix, The New York Times, PayPal, Pinterest, PlayStation Network, Quora, Reddit, Slack, Tumblr, Twitter, Visa, The Wall Street Journal, Wikia, Wired, Xbox Live, and Yelp.

In total, several thousand websites were unreachable. The attack started at 7:00 and laster for two hours and a half; a second attack was launched around 11:52, and a third attack took place after 16:00. The situation was not resolved before 18:11, at which point some services had been disconnected for about 11 hours.

The attack itself simply performed many DNS requests, too many for Dyn to handle, resulting in the denial of service. To send so many requests, the attacker assembled a *botnet*, a network of computers under their control, in that case consisting of a large number of Internet-connected devices such as printers, IP cameras, residential gateways and baby monitors. These devices are not considered by many users as computers, leading to most of them being left without protection and exposed to the Internet. The attacker took control of them using the Mirai open source malware.

The same technique and botnet was user earlier the same year to attack computer security journalist Brian Krebs's web site (20 September 2016), and participated in an attack on French web host OVH (4 October 2016). In November 2016, it temporarily but completely disconnected the country Liberia from the Internet.

This series of attacks currently holds the record for the largest distributed denial of service attack, peaking at an estimated $1.5\,\mathrm{Tb/s}$ of traffic. The attackers have not been identified.

3.4 Arresting botnets: Operation Tovar

> ***Note:*** *this section is illustrative and may be skipped at first.*

In some exceptional cases, international collaborations are formed to try and pin down botnet operators.

Operation Tovar was one of them, carried out by law enforcement agencies from multiple countries against the Gameover ZeuS botnet. This botnet was used in bank fraud and the distribution of the CryptoLocker ransomware. The collaboation included the U.S. Department of Justice, Europol, the FBI and the U.K. National Crime Agency, South African Police Service, Australian Federal Police, the National Police of the Netherlands' National High Tech Crime Unit, the European Cybercrime Centre (EC3), Germany's Bundeskriminalamt, France's Police Judiciaire, Italy's Polizia Postale e delle Comunicazioni, Japan's National Police Agency, Luxembourg's Police Grand Ducale, New Zealand Police, the Royal Canadian Mounted Police, Ukraine's Ministry of Internal Affairs' Division for Combating Cyber Crime, The Defense Criminal Investigative Service of the U.S. Department of Defense; together with a number of security

companies and academic researchers including Dell SecureWorks, Deloitte Cyber Risk Services, Microsoft Corporation, F-Secure, L3 Comms, McAfee, Neustar, Shadowserver, Anubisnetworks, Symantec, Heimdal Security, Sophos and Trend Micro, Carnegie Mellon University, Georgia Institute of Technology, VU University Amsterdam and Saarland University.

After a long investigation, in early June 2014, Operation Tovar had temporarily succeeded in cutting communication between Gameover ZeuS and its command-and-control servers. Soon after, Russian Evgeniy "lucky12345" Bogachev was charged by the FBI of being the ringleader. The arrest enabled investigators to definitively stop the botnet, to measure the full scale of the attacks perpetrated by ZeuS, and to realise (unexpectedly) that decryption of CryptoLocked files was possible.

About 1.3% of those infected had paid the ransom, resulting in a benefit of about 3 million dollars for the gang.

Chapter 4

Access control mechanisms

— 3 Oct 2017

An ubiquitous *response* to security concerns is the implementation of *access control mechanisms*. Concretely, a low-level *reference monitor* ensures that operations on the system are only allowed after three essential steps have been successfully fulfilled:

1. The person requesting the operation should be known of the system; usually this takes the form of a (declarative) *identification* step.
2. The person requesting the operation (having given a known identity) should provide proof that this identity is theirs; this is the *authentication* step.
3. The person requesting the operation (having given a known identity, and proven that it is theirs) should be allowed to perform the requested operation; this is the *authorisation* step.

All three steps are essential to ensure proper access control (AC).

4.1 Access control in theory

The code idea of access control is to monitor *users* of a system, so that any modification to the system could be traced back to a specific user's actions; or similarly to ensure that any confidentiality breach can be pinned down to a given individual's actions. In other terms, it assumes that any security risk can be blamed on the system's users.

Properties of the reference monitor

A necessary set of properties for this idea to make sense is that the reference monitor is non-bypassable (otherwise, users could alter data without being monitored), evaluable (otherwise, users may gain access despite illegitimate credentials), always invoked (otherwise, some operations may not be attached to a user), and tamper-proof (otherwise, an attacker may change the reference monitor's behaviour). A mnemonic for these properties is *NEAT*.

Ideal AC versus Real-world AC

In an ideal world, we may assume that access control is perfect, i.e., access is possible if and only if all three steps (identification, authentification, authorisation) have been successfully passed. In practice, we would have to consider how access is denied, and there might be situations in which it is possible, if hard, for an adversary to access resources despite access controls. It is therefore important to be explicit about the amount of protection offered by access control mechanisms, which we can then weight against adversary models. For example, a locked door can be broken

through, and depending on the door's material this may require more of less specialised tools and time.

An illustration: the Bell–LaPadula–Schell AC

> ***Note:*** *this section is illustrative and may be skipped at first.*

The Bell–LaPadula–Schell (BLP) access control model is interesting as it illustrates several appealing features of mathematical models, as well as remarkable limitations of access control mechanisms. Let's first define a lattice L, i.e., a partially ordered set with a maximum and a minimum. The typical example is

$$L = \{\text{unclassified} < \text{classified} < \text{secret} < \text{top secret}\}.$$

We then consider a set of users U, a set of resources or objects R, and a set of operations O, for instance

$$O = \{\text{read}, \text{write}\}.$$

To each user $u \in U$ we associate their *accreditation level* $\alpha(u) \in L$. To each resource $r \in R$ we associate their *classification level* $\gamma(r) \in L$. We can now specify the set of *authorised actions* as those actions that obey the following rules:

1. "No read up": a resource $r \in R$ cannot be read by an user $u \in U$ if $\alpha(u) < \gamma(r)$. All other reads are authorised.

2. "No write down": a resource $r \in R$ cannot be written by an user $u \in U$ if $\alpha(u) > \gamma(r)$. All other writes are authorised.

If a user writes on a resource, this resource's classification level rises to match the user's accredication level: $\gamma(r) \leftarrow \alpha(u)$.

One remarkable feature of the BLP model is that it can be represented as a finite automaton, and analysed completely from the premise that if we are in a "good state" to begin with, then any authorised action leads to a "good state". This enable us to check a system in a completely automated and exhaustive manner.

But we must now ask the question: how are security properties impacted by this approach?

It turns out this is not obvious at all. First, the BLP model does not cover any integrity concern. Furthermore, since there is no mechanism to have documents "fall down", they eventually all reach the highest level, which may be considered an availability concern. More worrying, is the fact that even confidentiality is not clearly captured.

The first obvious limitation is that the BLP model fails if individuals can change accreditation. A less obvious limitation is that it is vulnerable to *collusion*: assume that there is high-ranked official, Alice, with top secret clearance; and assume she is willing to collaborate with Bob, who only has unclassified accreditation. Then there is a way that Alice can exfiltrate top secret information to Bob, in spite of the BLP restrictions. Note that, as many attackers are insiders, this assumption is realistic. Here is one possible way that Alice and Bob can communicate:

1. Bob creates public documents

$$\text{a}, \text{abandon}, \text{ability}, \text{able}, \ldots, \text{zero}, \text{zone}, \text{ zoo}$$

   (this corresponds to all the words in a dictionary).

2. Alice can see all these documents. She cannot write in public documents, but if she tries, the unclassified documents become top secret, and Bob cannot read them anymore. Thus Alice decides to modify well-chosen files, e.g., the documents "Hello" and "world".

3. Bob tries to read all the files he created earlier. He will succeed, except for the documents "Hello" and "world", which are now top secret.

In that fashion, Alice has sent Bob the message "Hello world". By using a more clever technique (e.g., using binary or hexadecimal codes) it is possible to convey information more efficiently.

This communication channel between Alice and Bob is not provided by the BLP mechanism (and therefore not controlled by it), it is called a *covert channel*. Why does the BLP model allow such a channel? In part because it makes no clear difference between what it tries to achieve in terms of security (policy) and what means are used to make this happen (mechanism). Here the mechanism allows us to do something that a policy (had it been clearly stated) would not have allowed. This leads us to remember to important following design principle: *clearly separate mechanism from policy*.

4.2 Authentication in theory

We now turn our attention more precisely to authentication, one of the key steps in access control. Authentication requires a proof of identity, but "identity" is a tricky topic. Rather that identity, we will rely on an imperfect approach: we will assume that within the group of known users, every individual is endowed with a "secret". Authentication is then ensuring that the individual has the secret they claim.

What is a good secret?

Nevertheless, the secret is supposed to mimic identity, and in particular, should be unique (within that group), should not be copyable or guessable, and it should not be the case that someone gives their secret to someone else. Furthermore, it should not be the case that the secret disappears unexpectedly; but at the same time, it should be possible to replace a compromised secret by a fresh one, ideally in a way that the new secret cannot be derived from the previous one.

The above paragraph puts many constraints of what constitutes an acceptable secret. For instance, passwords fail to satisfy most of the desirable properties; and so do many biometric features. It is a very salient question in security: *what could be a realistic good secret* to use for authentication? Naturally, we may relax our expectations and accept a less-than-perfect secret, but that seems to cause an interesting paradox: indeed, if for instance we must protect our passwords, then it means *we need access control... to be able to use access control*!

Multi-factor authentication

Following this idea, a makeshift solution is *multi-factor authentication*; concretely, use several secrets. On the one hand, doing so reduces the likelihood that one secret's limitations suffice for an attacker to bypass authentication. On the other hand, it increases the likelihood that at least one secret is missing, and that legitimate users are blocked from accessing the system. In any case, multi-factor authentication requires the use of *independent secrets*. A bad example is to ask for both a password and a code sent by SMS — what if the person is using their phone to type the password? More generally,

> "two-factor authentication using SMS (...) isn't technically two-factor at all." — NIST 2016.

The above remark is motivated by the relative ease to prevent, intercept, or redirect SMS messages; but more pragmatically mobile phones have become the center of many users' virtual activity: *keep secrets physically separate*.

Properties of an authentication protocol

An authentication protocol checks that an individual indeed possesses the claimed secret. In that scenario we consider a *prover* (the user) trying to convince a *verifier* (the access control mechanism). Ideally, an authentication protocol satisfies the following properties:

1. *Correctness*: if the prover possesses the secret, then the authentication protocol succeeds.
2. *Significance*: if the authentication protocol succeeds, then the prover possesses the secret.
3. *Non-transferability*: if the authentication protocol succeeds, then the verifier cannot impersonate the prover to a third-party.

The first two properties are easy to achieve, for instance with passwords. However passwords are trivially transferable, since one has to transmit them to the verifier. Conversely, many biometric traits can be verified in a non-transferable way, but have some inevitable false positive and false negative probability.

As usual, we may relax our expectations and replace non-transferability by a weaker property, that of *eavesdropper resistance*: if the authentication protocol succeeds, then an eavesdropper (different from the verifier) cannot impersonate the prover. In other terms, we assume that the verifier is honest.

4.3 Zero-knowledge identification

> ***Note:*** *this section is illustrative and may be skipped at first.*

Using cryptographic techniques, it is possible to devise a non-transferable authentication protocol, that is correct and significant with high probability. However, the computations required for this protocol require the use of a special device, which means that what we authenticate is the device (and not the bearer of that device). We can even ensure that an adversary (who may be the verifier itself!) doesn't learn anything, except that the prover possesses the claimed secret, and for this reason such protocols are dubbed *zero-knowledge proofs*.

Here is an example. Let G_0, G_1 be two graphs. The prover wants to convince the verifier that they know the secret permutation π, such that $\pi(G_0) = G_1$ (i.e., π is a graph isomorphism). Of course, the prover may send π to the verifier, but that would be trivially transferable. Instead, they play the following game:

1. The prover chooses a random permutation σ, and a bit $b \in \{0, 1\}$; then it computes $H \leftarrow \sigma(G_b)$ and sends H to the verifier.
2. The verifier chooses a bit $b' \in \{0, 1\}$ and sends it to the prover.

3. The prover computes the permutation

$$\tau = \begin{cases} \sigma & \text{if } b = b' \\ \sigma \circ \pi^{-1} & \text{if } b \neq b' \text{ and } b = 0 \\ \sigma \circ \pi & \text{if } b \neq b' \text{ and } b = 1 \end{cases}$$

   then sends τ to the verifier.

4. The verifier accepts if and only if $H = \tau(G_{b'})$.

If π really is a graph isomorphism, then the verifier will always accept, and therefore this protocol is correct. If the verifier chooses b' randomly, and π is not an isomorphism, then $\tau(G_{b'}) = \sigma(G_b)$ if an only if $b = b'$; which happens with probability $1/2$. So the protocol is significant (or "sound") with probability $1/2$, which may seem small; in practice we would repeat the protocol several times, for instance 100 times, to bring the significance level to $1 - 2^{-100} \approx 1$. Therefore the verifier can be convinced beyond any doubt that the prover knows π.

How do we show that an adversary listening in on the conversation doesn't learn anything about π? The usual method is to exhibit a *simulator*: a program that doesn't know π and interacts with the verifier. If we can show that it is impossible to distinguish the simulator from the prover, then we have succeeded. (Think about this last sentence, let it sink it, make sure that you understand it before moving forward.)

Consider a verifier V, graphs G_0, G_1, and denote by b_V the bit chosen by the verifier in step 2. Construct a program S that does the following:

1. Choose a random permutation σ, and a bit $b \in \{0, 1\}$
2. Set $H \leftarrow \sigma(G_b)$, send it to V, and set $b' \leftarrow b_V$.
3. If $b' = b$, then return (b', H, σ); otherwise restart at step 1.

The bits b and b_V are chosen independently, hence are equal with probability $1/2$. As a result, S runs only twice in expectation. By design, the distribution of (b', H, σ) output from S is exactly the same as the distribution of (b', H, σ) that would happen between the prover and the verifier in a normal interaction. This means that, information-theoretically, S is indistinguishable from a prover-verifier transcript, and therefore is *perfectly zero-knowledge*.

Many protocols have the weaker property of being *computationally* zero-knowledge, which means that the difference between simulator and legitimate transcript is practically unusable (although the difference may exist); and some only provide *honest verifier* zero-knowledge, where we assume that the verifier follows the protocol (even it they try to learn our secrets).

4.4 Further reading

- A high-level introduction to access control is provided in the articles by Brose [Bro05] and Shekhar et al. [SXZ17].
- For a discussion on passwords and their limitations, see for instance Bonneau [Bon12][1] and Bonneau et al. [Bon+12; PB10; BP10].
- More information about the Bell–LaPadula model can be found in the original article [BL73] and in Bell's retrospective article [Bel05].

[1] This paper received the NSA Award for Best Scientific Cybersecurity Paper of 2012.

- A gentle introduction to zero-knowledge protocols was written by Quisquater et al. [Qui+89]. A more recent write-up is provided by one of the pioneers of zero-knowledge, Goldreich [Gol13]. The important fact that any **NP** statement can be proven in zero-knowledge is due to Goldreich et al. [GMW86] which also constitutes a nice introduction.

Part II

Ideal world vs. Real world

Chapter 5

Side and covert channels

Computers are not abstract machines. As a result, any algorithm, any running program, does strictly more than what the designers put into it. At the very least, computers receive and dissipate energy. The goal of this chapter is to illustrate how actual computation in the real world differs from the ideal model most programmers have in mind, and how this gap affects security properties. As we saw in the previous chapter, access control does not *per se* prevent the existence of covert channels (in the case we discussed, access control was *responsible* for introducing this channel).

5.1 What is a side channel?

A very general definition is that a *side channel* is any communication channel (including one-way channels) that occurs as a result of a system's operation, beyond its usual input and output. The existence of such channels is often an unforeseen consequence; in particular, most programming languages abstract away key hardware notions such as time, energy consumption, heat dissipation, noise, etc. which turn out to be valuable sources of side channel analysis, see Figure 5.1. Because it is unforeseen, a side channel may carry information that was supposed to be concealed, it is therefore usually a confidentiality hazard. In some cases, side channels can also change the system's internals (e.g., variables), in which case they can also affect integrity.

A *covert channel* is a communication channel between adversaries, through a medium in which we expected to control communication, see Figure 5.2. It therefore violates the system's policy and may again result in a security risk. Covert channels are especially powerful in bypassing access control limitations, since the colluding adversaries may undergo different restrictions: as an example, a weather application (accessing the Internet and knowing the user's location) could communicate with a photo application (having access to files and the camera)

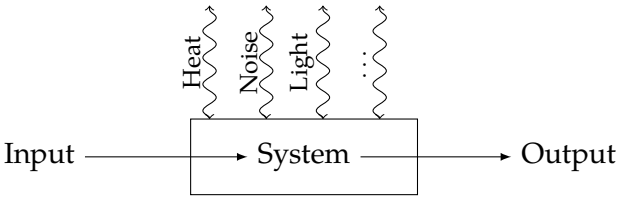

Figure 5.1: We can see a system taking an input and providing an output as a channel, our "main channel". But the system's actions result in other phenomena, some of which inform us about what the system is doing (or modifies the system's operation), of what inputs or outputs it is dealing with: these are the side channels.

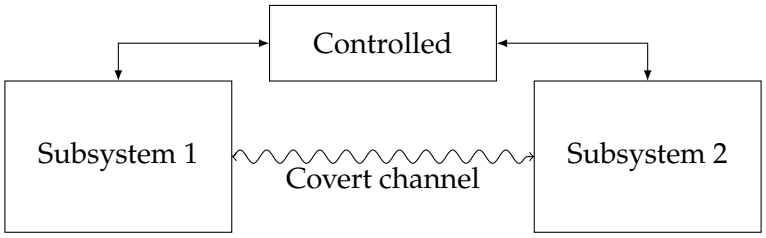

Figure 5.2: A covert channel allows subsystems to communicate without the system being able to control them. In particular, this means such channels are not subject to the system's access control policy.

through a covert channel to exfiltrate photos through the internet, something neither application could do alone.

A channel (main, side, or covert) is limited in the amount of information that it can reliably carry in a given amount of time: this is the channel's bandwidth. If the adversary is trying to learn information about the main channel by means of the side channel, then by the Shannon–Nyquist theorem the side channel's bandwidth should be large enough. This means we should either use high-bandwidth side channels; or target slow/slowed down main channels; or use a combination of sufficiently independent side channels.

5.2 A taxonomy of side and covert channels

Physical channels

A fair proportion of side and covert channel exist because computers are physical devices that interact with their environments, in way that are not prescribed or cannot be controlled by software. The most common of these can be classified as follows:

- *Electromagnetic fields*
  - Radiated fields, which are caused by the presence of electrical conductors carrying varying currents, in particular cables and buses, but also fast-switching components such as screens. We may further distinguish, with decreasing wavelengths:
    * Radio emissions, primarily due to cables;
    * Thermal emissions, resulting from heat dissipation;
    * Optical emissions, either deliberate (status LEDs, screens, etc.) or not (transistor switching photon emission).

    Conversely, electromagnetic fields can influence a device by induction or ionisation, which can be accidental (e.g., in space applications) or used to control the device's memory or behaviour by targeting for instance a laser.
  - Static fields, which are due the the presence of capacitive or magnetic effects, in particular
    * Impedance, between a system and the "ground" connection, measurable on the chassis, shields, cables, and wall sockets.
    * Remanence, for instance in DRAM or SRAM even after the system was shut down, and on the surface of magnetic hard drives even after erasure.

  - Electric power, used or transmitted by the device, which can be related to its operation and to the data being handled. Conversely, feeding a device with an inappropriate power source (intermittent, incorrect voltage, etc.) may cause it to malfunction in some way that may be exploited by an attacker.

  Electromagnetic fields are easy to measure and can sometimes be reliably acquired from long distances, due to radiation or to conduction.

- *Mechanical stresses*, in particular vibrations, which include motion (captured by mobile phones' gyroscopes and accelerometers), and acoustic emanations caused by coil whine, voltage regulators, and peripherals.

- *Timing*, as no operation is instantaneous, may reveal sensitive information about a device.

Microarchitectural channels

Another large family of side and covert channels result from hardware mechanisms that are often ignored by software programmers.

- *Memory access*, which is never constant-time and can reveal information about what a program is doing, in particular

  - Contention in data or instruction caches (HDD, DRAM, L3, L2, and L1 caches);
  - Paging mechanisms, including page faults and table lookaside buffers (TLB);
  - Other memory prefetching mechanisms, e.g., hard drive buffers.

- *CPU pipelines*, and in particular branch prediction.

- *Functional units*, and in particular ALUs, which fail to perform many operations in constant time.

Operating system channels

In some cases, the operating system (which may be a hypervisor) introduces additional channels that attackers can sometimes leverage, including:

- *Memory operations*, in particular

  - Virtual memory and swapping mechanisms;
  - Deduplication, which is sometimes directly exploitable and assists other attacks;
  - Logical erasure of data, in particular block remapping, which fails to physically erase information;
  - Memory prefetching and temporary files, backups.

- *Scheduling and multitasking*, which is sometimes directly exploitable and improves the temporal resolution of other attacks.

- *Global information*, such as the process table, network status, available memory.

- *Error handling*.

Miscellaneous

There are many more possible channels, which may be specific to a situation, and application, or a protocol (e.g., TCP/IP flags or sequence numbers), which may be local (e.g., plugging a non-input pin to an input) or non-local (e.g., Tor deanonymisation by traffic correlation).

5.3 An important example: Exponentiation

> ***Note:*** *this section is illustrative and may be skipped at first.*

Consider the simple but important task of computing x^n for given integers x and n. This operation is ubiquitous in computer science, and particularly in cryptography, where n is typically some secret that we do not want the adversary to learn. From a mathematical standpoint, there are many equivalent ways to compute x^n.

Maybe the most straightforward approach is to compute $x \times x \times \cdots \times x$, i.e., n multiplications. This corresponds to the following Python code:

```
def expo1(x,n):
  result = 1
  for i in range(n):
    result = result * x
  return result
```

This algorith is correct[1] and its analysis is immediate. However, it computes x^n by performing precisely n operations; an adversary can measure how long the algorithm take, and deduce n immediately: this is a *timing attack*. To do this, we would need to measure the algorithm's execution time precisely enough, which it usually not very challenging, and divide this by the time it takes for the computer to perform a single multiplication — this information is in general easy to find.

Let's compute x^n in another way. Write n in binary, and assume that it has a fixed length (we may pad with zeroes if necessary, so this is not really a strong assumption). Then we can compute successive powers of x in a more efficient way, as follows:

```
def expo2(x, n):
  result = 1
  for b in n:
    if b == 1:
      result = result * x
    result = result * result
  return result
```

This algorithm, known as the square-and-multiply algorithm, is much more efficient as it only performs of the order of $\log(n)$ operations; furthermore, the size of n is a fixed quantity, so we are not vulnerable to the previous timing attack.[2] However, if the attacker can measure the device's power consumption as this algorithm is run, then there would be a difference between the rounds where only a squaring is computed, and the rounds where both a multiplication and

[1] For small enough values of n and x, anyway.

[2] In fact, the presence of an `if` statement creates a measurable timing difference. But if n is large enough and chosen at random, there are impractically many possible candidates. Prove this to convince yourself.

a squaring are computed, manifest in the different power consumption of these operations.[3] This would allow the adversary to recover n using nothing more than an oscilloscope, as in Figure 5.3.

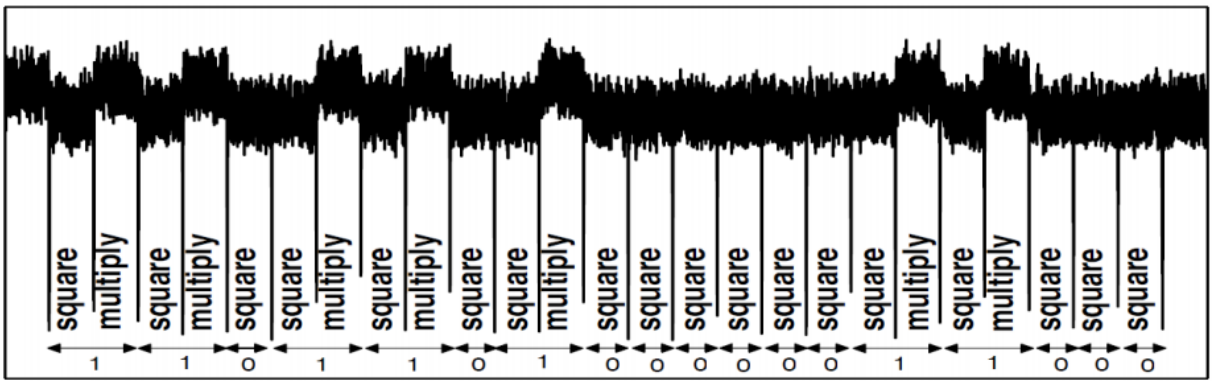

Figure 5.3: Simple power analysis of a square-and-multiply exponentiation. An adversary accessing power consumption information with a fine enough timescale can "read" the secret exponent's bits directly.

One idea is to perform *both* operations in any case. This is the intuition behind the "Montgomery ladder" algorithm:

```
def expo3(x, n):
  L = [1, x]
  for b in n.reverse():
    L[1 - b] = L[0] * L[1]
    L[b]     = L[b] * L[b]
  return L[0]
```

This algorithm performs a fixed number of operations depending on the size of n, so it not vulnerable to a simple timing attack. As it always does both a standard multiplication and a squaring, it is also a priori immune to power consumption attacks. But... *cache attacks* may create a measurable difference between access to `L[0]` and `L[1]`, which would allow an attacker to distinguish whether b is 0 or 1 at each round!

The cryptographic library OpenSSL, which implements exponentiation as part of its functionality, has been updated to patch its most severe side channel leakages. Following an efficient cache attack, the OpenSSL developers implemented another algorithm called scatter-and-gather, which was suggested by Intel (remember that caches belong to the CPU!). In 2017 however, Yarom et al. [YGH17] showed that scatter-and-gather is not constant time, and demonstrated a full attack. Computing x^n securely is not easy!

5.4 How is side channel information exploited?

In the exponentiation example, we saw that different sources of side channel information are used differently. In some cases, an adversary can directly read the main channel in the side channel (as in the power consumption case); in some other cases, they only get a distinguisher (as in the timing case). In general, one can compute the *correlation* between a set of measures and a set of simulations: if the adversary can simulate well enough the system's impact on a side

[3] In many cases, squaring and multiplication are often implemented differently in hardware fo efficiency reasons.

channel, a strong correlation would indicate that the system under attack corresponds to the simulation scenario.

Hardware side channels are especially interesting to adversaries, because they are often usable in spite of very restrictive software limitations. In particular, cache attacks are perfectly feasible between two different virtual machines, which are logically isolated by many layers (different users, operating systems, virtual managers, etc.) but run on a single device.

Note that some side channel attacks require a certain degree of physical access to the device being targeted; but one should not take this as an indication that these attacks are impractical, and even if physical access is deemed unlikely, it is important to know the consequences of a potential compromission.

An illustration: Non-invasive analysis

> ***Note:*** *this section is illustrative and may be skipped at first.*

In [Fer+16], Ferradi et al. performed the forensic analysis of payment cards seized from a criminal gang operating between France and Belgium. The cards had been stolen but somehow the criminals managed to use them without using the PIN code usually required for payment. To establish whether the cards had been modified, a judge demanded a scientific investigation. Initial observations increased suspicions, but were not sufficient in themselves to be decisive. When the card is used in a payment terminal, it passes all checks and seems to work, with one notable feature: any PIN code is accepted!

Suspecting something fishy, Ferradi et al. used power analysis to peek into the device's internal operation, see Figure 5.4. Indeeed, the command responsible for checking the PIN code is immediately answered, without any computation! But more interesting, every *other* command was echoed. They recognised in this behaviour the attitude of a "man-in-the-middle" interception, which was first described in the context of payment card only two years before the fact, by Murdoch et al. [Mur+10].

Murdoch et al.'s attack was described as "theoretical, at best" by the EMV group, responsible for payment card protocols. It is true that Murdoch et al.'s setup required almost a room full of specialised equipment. It worked by retransmitting all commands to a real card, except of course the check PIN command. But Ferradi et al. only had one card; or so it seemed. If their theory was correct, then another circuit was hiding somewhere. They took an X-ray of the device, which confirmed the presence of two different components *in the card's thickness*, see Figure 5.5.

At this point, external inspection, power analysis, and imaging were all consistent with a fraud, and more specifically with a state-of-the-art implementation of Murdoch et al.'s attack. Ferradi et al. were authorised to perform invasive analysis, i.e., dismantle the card to verify this hypothesis. This was done, and confirmed the suspected *modus operandi* beyond doubt.

An illustration: ALU side channels

> ***Note:*** *this section is illustrative and may be skipped at first.*

For many a programmer, operations on native types (8 to 64 bit integers, floats, etc.) seem instantaneous. This is a gross approximation, as even the simplest operations typically take several CPU cycles to complete. In fact, there are even operations that take a time which is a direct function of its operands, and thereby opens a timing channel.

Integer division in hardware is a relatively costly operation; implementations often use microcode, and will sometimes trigger a faster code path when the divisor or the dividend

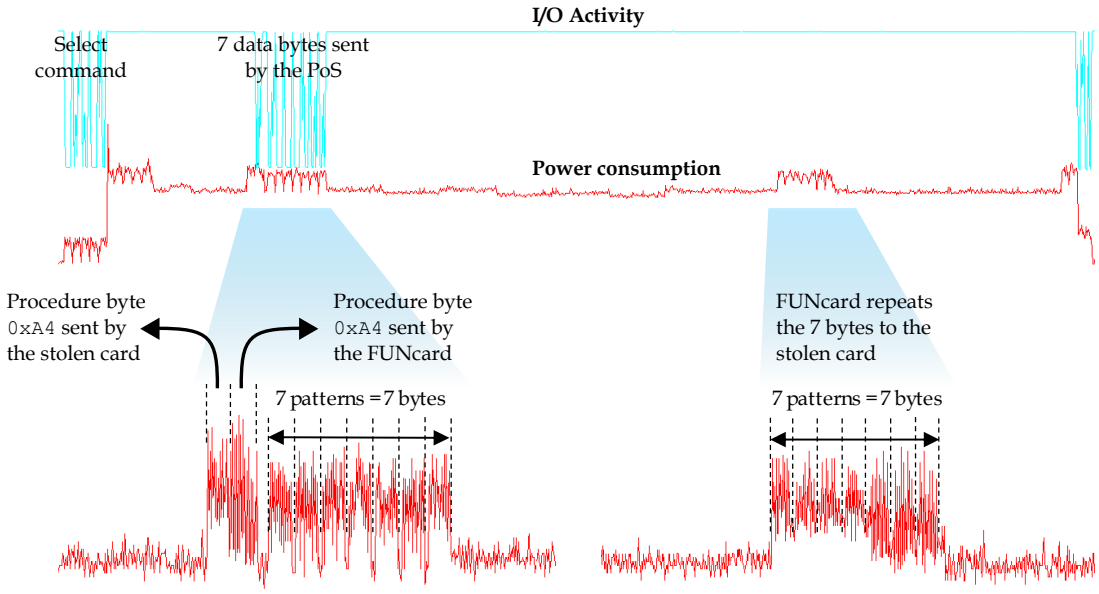

Figure 5.4: Power analysis of a suspect payment card (the "FUNcard"). Using a card reader, we can precisely time commands and responses; using an oscilloscope we can follow the internal computations and communications. Here we see how a typical command is echoed by the FUNcard to something else (which turns out to be a real, stolen payment chip).

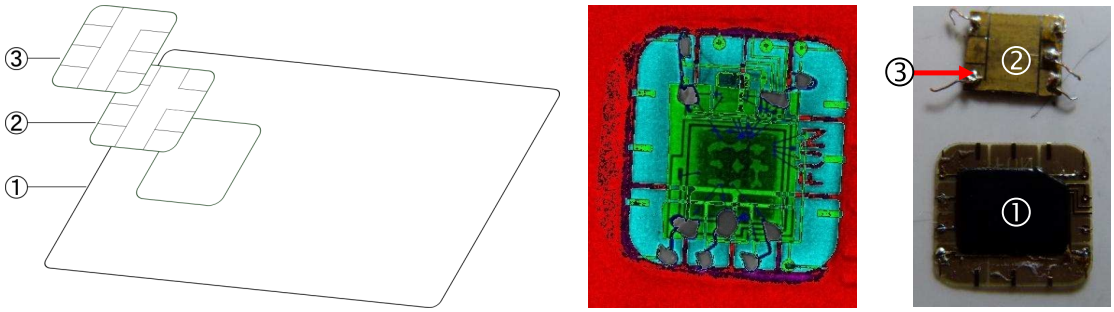

Figure 5.5: Ferradi et al. suspected that a real card was hidden underneath the visible connectors, and manipulated as in Murdoch's attack (left). This was consistent with X-ray imaging (middle) and later confirmed by destructive analysis (right).

is small. Some divisions can be optimised into shifts and masking, and dealing with signed integers may involve some special code with conditional jumps.

Shifts' and rotations' execution time often depends on the shift and rotation count respectively, unless the CPU features a barrel shifter. Early CPUs (for instance the Pentium IV) did not have barrel shifters, and many low-end microprocessors still don't.

Multiplications may also offer attackers with a timing channel. While all recent CPU designs guarantee constant-time multiplication, most older CPUs have shortcuts that make multiplications faster when operands are small. Here again, many lower-end CPUs, such as those used for phones (e.g., ARM9) or connected objects still have variable-time multiplication units.

5.5 Protecting against side channels

Many side channels cannot be completely eliminated; for instance, electro-magnetic emanations resulting from varying electric currents. But we may try to render their analysis impractical. In any case, the first step is to *properly identify side channels*, so as to provide adequate protections. One may use the above taxonomy, although we do not claim that it is complete. Sometimes, as in the exponentiation example, the choice of a particular algorithm, with careful implementation techniques (to avoid look-up tables, conditional branches, non-constant time algorithms...) is enough to tackle a particular leak. Often, however, this is not enough, and it becomes necessary to resort to *hardware* protections.

Hardware protections typically fall in three categories; with increasing complexity:

- *Shielding*: provide isolation between the leaky component and other components or the external world. For instance, an appropriate power supply could smooth out energy consumption variations; a metal plating and coaxial wires could capture electro-magnetic emanations, etc. A limitation of this approach is that an adversary with physical access to the system may remove these protections; another limitation is that while they mute one side channel, they create another: the power supply's switching behaviour causes coil whine, which is a sound channel, and the metal plating connects to the ground, providing an easy-to-measure signal.

- *Flooding*: add noise to the side channel so that it becomes difficult for the adversary to recover a useful signal. For instance, extra variations in clock speed, energy consumption, etc. could be produced, which would make the side channel analysis more difficult. In practice, the efficiency of such methods is limited by the adversary's capacity to perform statistical analysis, or to predict (and thus remove) the noise. At the same time, such mechanisms can make hardware design trickier, and interfere with the device's normal operation. In any case, this extra activity requires more energy, and produces by design more heat, noise, etc. than the original, unprotected device.

- *Masking*: performing basic operations in such a way that the side channel becomes independent of data. This is operation-dependent, and often requires access to a source of randomness, but is much more efficient than the other approaches. Indeed, side channel information is by design no longer useful to the adversary (to the extent that the operation is really masked, there are typically limitations), and masking schemes are very energy-efficient.

Note that while the above measures apply to side channels, covert channels are often harder both to detect and to prevent. As an example, cache-based covert channels can only be caught

when they are obvious, i.e., much more frequent that normal; and colluding parties may agree on a very discreet, low-bandwidth channel (or collection of channels).

An illustration: Masking scheme

> ***Note:*** *this section is illustrative and may be skipped at first.*

Masking consists in splitting information into shares, so that an adversary would need all of them to recover the secret. As a simple example, assume we want to compute the exclusive-OR operation between two bits, a and b, denoted $c = a \oplus b$. We can mask individual bits as follows: replace a by $m(a) = (a_1, a_2)$ so that $a = a_1 \oplus a_2$ and a_1 is a random bit; similarly split b as $m(b) = b_1 \oplus b_2$. Then if the adversary only measures one of a_1, a_2 they cannot learn a, and similarly they cannot learn b from only b_1 or b_2. We can however compute $m(a) \oplus m(b) = (a_1 \oplus b_1, a_2 \oplus b_2) = (c_1, c_2)$ which satisfies $c_1 \oplus c_2 = a \oplus b$. In other terms, we take a masked input and get a masked output, with a proof that if the adversary can only learn up to 3 input bits, then they cannot learn the value of c. Several other masked gates were proposed: AND, MUX, NAND, etc. and it is possible to design masking schemes that split inputs and outputs into d shares. As an example, here is a $d = 3$ masked AND gate: assuming that $a = a_1 \oplus a_2 \oplus a_3$ and $b = b_1 \oplus b_2 \oplus b_3$,

$$
\begin{aligned}
c_1 &= a_2 b_2 \oplus a_2 b_3 \oplus a_3 b_2 \\
c_2 &= a_1 b_3 \oplus a_3 b_1 \oplus a_3 b_3 \\
c_3 &= a_1 b_1 \oplus a_1 b_2 \oplus a_2 b_1,
\end{aligned}
$$

where the AND operation is written as a product.

However, it was realised in 2005 by Fischer et al. [FG05; MPG05] that glitches may render masking insecure. A glitch is a spurious transition of nodes in a combinational circuit within one clock cycle, resulting from different arrival times of the input signals; this happens often with regular CMOS circuits. In fact, Fischer et al. proved that there is no set of universal masked gates with 2 input bits and 1 output bit, split into 2 shares, that resist *glitches*.

Masking can be rendered ineffective if used carelessly: for instance, storing the shares in a register *one after the other*, or using a poor source of randomness, allow the adversary to reconstruct the masked bits. One very active area of research is the provably secure design of masking scehems that resist higher-order side channel analysis (leveraging not only the mean, but higher-order moments of the trace distribution).[4]

An illustration: Active attacks

> ***Note:*** *this section is illustrative and may be skipped at first.*

When side channel leakage fails to provide the attacker with usable information, they can bump up their game and start using more invasive approaches to squeeze secrets out of a system.

In a fault attack, the adversary puts the target system in conditions that cause it to malfunction in some way. This can be excessive heat or cold, underpowering the device, applying voltage spikes or strong magnetic fields, physically abusing the device, etc. One very popular approach is laser-induced faults, which can be very precise.

[4] In particular, for some masking schemes, there exist third order power analysis that can break them for any number d of shares... see for instance Coron et al. [CPR07].

Causing a precise fault, the attacker can change a the value of a bit (or more generally, a group of bits) in the device's memory. In some cases it is even possible to precisely control the new value. While this often gives the attacker large powers on the device (e.g., deactivate access control), it can also be used to learn information.

Here is a relatively simple example, which we try to describe without requiring much cryptography. The ECDSA signature algorithm is used as an authentication method by many systems (including for instance the Bitcoin protocol). It uses a secret key x to compute a signature of a message m. Here is a slightly simplified description of this algorithm: assume g is the generator of a cyclic group of large prime order q,

1. Choose a number k
2. $r \leftarrow g^k$
3. $s \leftarrow k^{-1}(m + rx) \bmod q$
4. Output the signature (m, r, s).

Note that the number k should be *unique*. Indeed, if two identical values of k are used to two different messages, we can use (m, r, s) and (m', r, s') to compute $(m - m')/(s - s') = k$, and from there $(ks - m)/r = x$. In other words, knowing or reusing k immediately reveals the secret key.

In 2010 the `fail0verflow` group announced recovery of the ECDSA private key used by Sony to sign software for the PlayStation 3. Indeed, the PS3 always used the same k.[5]

Using a timing attack, Brumley and Tuveri showe in 2011 how to remotely recover k, and therefore the private key, used in the TLS protocol by OpenSSL.[6]

RFC 6979 therefore suggests that k is generated in a deterministic way from the message, and that it is large enough to avoid collisions. Modern implementations should not subject to timing or cache attacks. What can the attacker do? They can inject a fault. Indeed, by faulting the value of k (for instance, force $k = \texttt{10...0}$) the attacker can make sure to find the private key.

5.6 Adjusting the paradigm

In the previous chapters we focused on access control as a way to guarantee security properties. In light of the discussion here, it should be clear that access control is *certainly not enough*. At the very least, it fails to capture important notions about how information can be leaked or inferred. *Access control is not information control*. In particular, we cannot rely on preventing something from happening, we must also have means to *detect that it happened*.

What this chapter also highlight is that *software security alone is not enough*, as even minute hardware details can ruin a system's confidentiality. Recall that all the mentioned attacks are performed despite access control policies being enforced.

Finally, and this will be a *leitmotiv* of this course, *theories, models, and metaphors should not be taken at face value*. They inform us, and provide us with understanding about problems, but they often do not represent faithfully what happens in a real system. Abstractions are our friends, but they don't tell the whole truth.

[5] In 2008, it was realised that the Debian project generated k in a predictable manner, resulting in completely insecure keys as well. The same mistake was found in Java in 2013, which resulted in much Bitcoin theft from the containing wallet on Android app implementations.

[6] The vulnerability was fixed in OpenSSL 1.0.0e.

5.7 Further reading

- An introduction to the history and methods of side channel analysis can be found in the book by Mangard et al. [MOP07]. The seminal paper on timing attacks is due to Kocher [Koc96].
- The seminal paper introducing fault analysis is due to Biham and Shamir [BS97]. Another early paper worth reading on that topic is the Boneh–DeMillo–Lipton attack on RSA-CRT [BDL01].
- For further information about PC and mobile exploitation of side channels, refer to the line of work by Genkin et al. [GVY17; Gen+16a; GPT15; Gen+16c; Gen+16b; Gen+15; GST14]. For server exploitation, see for instance Vaudenay [Vau02]. All these papers target some cryptographic scheme and therefore assume some basic knowledge of cryptography.
- The curious reader eager to know more about the trends and results in this field is invited to gaze through the proceedings of the CHES conference.

Bibliography

[12] *IEEE Symposium on Security and Privacy, SP 2012, 21-23 May 2012, San Francisco, California, USA*. IEEE Computer Society, 2012. ISBN: 978-0-7695-4681-0. URL: http://ieeexplore.ieee.org/xpl/mostRecentIssue.jsp?punumber=6233637.

[AM07] Ross J. Anderson and Tyler Moore. 'Information Security Economics - and Beyond'. In: *Advances in Cryptology - CRYPTO 2007, 27th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 19-23, 2007, Proceedings*. Ed. by Alfred Menezes. Vol. 4622. Lecture Notes in Computer Science. Springer, 2007, pp. 68–91. ISBN: 978-3-540-74142-8. URL: https://doi.org/10.1007/978-3-540-74143-5_5.

[And16] Iosif I. Androulidakis. 'Introduction: Confidentiality, Integrity, and Availability Threats in Mobile Phones'. In: *Mobile Phone Security and Forensics: A Practical Approach*. Cham: Springer International Publishing, 2016, pp. 1–14. ISBN: 978-3-319-29742-2. URL: https://doi.org/10.1007/978-3-319-29742-2_1.

[BDL01] Dan Boneh, Richard A. DeMillo and Richard J. Lipton. 'On the Importance of Eliminating Errors in Cryptographic Computations'. In: *J. Cryptology* 14.2 (2001), pp. 101–119. URL: https://doi.org/10.1007/s001450010016.

[Bel05] David Elliott Bell. 'Looking Back at the Bell-La Padula Model'. In: *21st Annual Computer Security Applications Conference (ACSAC 2005), 5-9 December 2005, Tucson, AZ, USA*. IEEE Computer Society, 2005, pp. 337–351. ISBN: 0-7695-2461-3. URL: https://doi.org/10.1109/CSAC.2005.37.

[Ber96] Peter L. Bernstein. *Against the gods: The remarkable story of risk*. Princeton University Press, 1996.

[BL73] David Elliott Bell and Leonard J. LaPadula. *Secure computer systems: Mathematical foundations*. Tech. rep. MITRE Corporation, 1973.

[Bon+12] Joseph Bonneau, Cormac Herley, Paul C. van Oorschot and Frank Stajano. 'The Quest to Replace Passwords: A Framework for Comparative Evaluation of Web Authentication Schemes'. In: *IEEE Symposium on Security and Privacy, SP 2012, 21-23 May 2012, San Francisco, California, USA*. IEEE Computer Society, 2012, pp. 553–567. ISBN: 978-0-7695-4681-0. URL: https://doi.org/10.1109/SP.2012.44.

[Bon12] Joseph Bonneau. 'The Science of Guessing: Analyzing an Anonymized Corpus of 70 Million Passwords'. In: *IEEE Symposium on Security and Privacy, SP 2012, 21-23 May 2012, San Francisco, California, USA*. IEEE Computer Society, 2012, pp. 538–552. ISBN: 978-0-7695-4681-0. URL: https://doi.org/10.1109/SP.2012.49.

[BP10] Joseph Bonneau and Sören Preibusch. 'The Password Thicket: Technical and Market Failures in Human Authentication on the Web'. In: *9th Annual Workshop on the Economics of Information Security, WEIS 2010, Harvard University, Cambridge, MA, USA, June 7-8, 2010*. 2010. URL: http://weis2010.econinfosec.org/papers/session3/weis2010_bonneau.pdf.

[Bro05] Gerald Brose. 'Access Control'. In: *Encyclopedia of Cryptography and Security*. Ed. by Henk C. A. van Tilborg. Springer, 2005. ISBN: 978-0-387-23473-1. URL: https://doi.org/10.1007/0-387-23483-7_3.

[BS97] Eli Biham and Adi Shamir. 'Differential Fault Analysis of Secret Key Cryptosystems'. In: *Advances in Cryptology - CRYPTO '97, 17th Annual International Cryptology Conference, Santa Barbara, California, USA, August 17-21, 1997, Proceedings*. Ed. by Burton S. Kaliski Jr. Vol. 1294. Lecture Notes in Computer Science. Springer, 1997, pp. 513–525. ISBN: 3-540-63384-7. URL: https://doi.org/10.1007/BFb0052259.

[CF08] Victor Cherkashin and Gregory Feifer. *Spy handler: memoir of a KGB officer: the true story of the man who recruited Robert Hanssen and Aldrich Ames*. Basic Books, 2008.

[CPR07] Jean-Sébastien Coron, Emmanuel Prouff and Matthieu Rivain. 'Side Channel Cryptanalysis of a Higher Order Masking Scheme'. In: *Cryptographic Hardware and Embedded Systems - CHES 2007, 9th International Workshop, Vienna, Austria, September 10-13, 2007, Proceedings*. Ed. by Pascal Paillier and Ingrid Verbauwhede. Vol. 4727. Lecture Notes in Computer Science. Springer, 2007, pp. 28–44. ISBN: 978-3-540-74734-5. URL: https://doi.org/10.1007/978-3-540-74735-2_3.

[Dou07] Frédérick Douzet. 'Les frontières chinoises de l'Internet'. French. In: *Hérodote* 2 (2007), pp. 127–142.

[Dou14] Frédérick Douzet. 'La géopolitique pour comprendre le cyberespace'. French. In: *Hérodote* 1 (2014), pp. 3–21.

[Ear97] Pete Earley. *Confessions of a spy: The real story of Aldrich Ames*. Berkley Books, 1997.

[Fer+16] Houda Ferradi, Rémi Géraud, David Naccache and Assia Tria. 'When organized crime applies academic results: a forensic analysis of an in-card listening device'. In: *J. Cryptographic Engineering* 6.1 (2016), pp. 49–59. URL: https://doi.org/10.1007/s13389-015-0112-3.

[FG05] Wieland Fischer and Berndt M. Gammel. 'Masking at Gate Level in the Presence of Glitches'. In: *Cryptographic Hardware and Embedded Systems - CHES 2005, 7th International Workshop, Edinburgh, UK, August 29 - September 1, 2005, Proceedings*. Ed. by Josyula R. Rao and Berk Sunar. Vol. 3659. Lecture Notes in Computer Science. Springer, 2005, pp. 187–200. ISBN: 3-540-28474-5. URL: https://doi.org/10.1007/11545262_14.

[Gen+15] Daniel Genkin, Lev Pachmanov, Itamar Pipman and Eran Tromer. 'Stealing Keys from PCs Using a Radio: Cheap Electromagnetic Attacks on Windowed Exponentiation'. In: *Cryptographic Hardware and Embedded Systems - CHES 2015 - 17th International Workshop, Saint-Malo, France, September 13-16, 2015, Proceedings*. Ed. by Tim Güneysu and Helena Handschuh. Vol. 9293. Lecture Notes in Computer Science. Springer, 2015, pp. 207–228. ISBN: 978-3-662-48323-7. URL: https://doi.org/10.1007/978-3-662-48324-4_11.

[Gen+16a] Daniel Genkin, Lev Pachmanov, Itamar Pipman, Adi Shamir and Eran Tromer. 'Physical key extraction attacks on PCs'. In: *Commun. ACM* 59.6 (2016), pp. 70–79. URL: http://doi.acm.org/10.1145/2851486.

[Gen+16b] Daniel Genkin, Lev Pachmanov, Itamar Pipman and Eran Tromer. 'ECDH Key-Extraction via Low-Bandwidth Electromagnetic Attacks on PCs'. In: *Topics in Cryptology - CT-RSA 2016 - The Cryptographers' Track at the RSA Conference 2016, San Francisco, CA, USA, February 29 - March 4, 2016, Proceedings*. Ed. by Kazue Sako. Vol. 9610. Lecture Notes in Computer Science. Springer, 2016, pp. 219–235. ISBN: 978-3-319-29484-1. URL: https://doi.org/10.1007/978-3-319-29485-8_13.

[Gen+16c] Daniel Genkin, Lev Pachmanov, Itamar Pipman, Eran Tromer and Yuval Yarom. 'ECDSA Key Extraction from Mobile Devices via Nonintrusive Physical Side Channels'. In: *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security, Vienna, Austria, October 24-28, 2016*. Ed. by Edgar R. Weippl, Stefan Katzenbeisser, Christopher Kruegel, Andrew C. Myers and Shai Halevi. ACM, 2016, pp. 1626–1638. ISBN: 978-1-4503-4139-4. URL: http://doi.acm.org/10.1145/2976749.2978353.

[GK12] Arnaud Garrigues and Olivier Kempf. 'L'OTAN et la cyberdéfense'. French. In: *Sécurité globale* 1 (2012), pp. 133–142.

[GL02] Lawrence A. Gordon and Martin P. Loeb. 'The economics of information security investment'. In: *ACM Transactions on Information and System Security (TISSEC)* 5.4 (2002), pp. 438–457. URL: http://doi.acm.org/10.1145/581271.581274.

[GMW86] Oded Goldreich, Silvio Micali and Avi Wigderson. 'How to Prove all NP-Statements in Zero-Knowledge, and a Methodology of Cryptographic Protocol Design'. In: *Advances in Cryptology - CRYPTO '86, Santa Barbara, California, USA, 1986, Proceedings*. Ed. by Andrew M. Odlyzko. Vol. 263. Lecture Notes in Computer Science. Springer, 1986, pp. 171–185. URL: https://doi.org/10.1007/3-540-47721-7_11.

[Gol13] Oded Goldreich. 'A Short Tutorial of Zero-Knowledge'. In: *Secure Multi-Party Computation*. Ed. by Manoj Prabhakaran and Amit Sahai. Vol. 10. Cryptology and Information Security Series. IOS Press, 2013, pp. 28–60. ISBN: 978-1-61499-168-7. URL: https://doi.org/10.3233/978-1-61499-169-4-28.

[GPT15] Daniel Genkin, Itamar Pipman and Eran Tromer. 'Get your hands off my laptop: physical side-channel key-extraction attacks on PCs - Extended version'. In: *J. Cryptographic Engineering* 5.2 (2015), pp. 95–112. URL: https://doi.org/10.1007/s13389-015-0100-7.

[GST14] Daniel Genkin, Adi Shamir and Eran Tromer. 'RSA Key Extraction via Low-Bandwidth Acoustic Cryptanalysis'. In: *Advances in Cryptology - CRYPTO 2014 - 34th Annual Cryptology Conference, Santa Barbara, CA, USA, August 17-21, 2014, Proceedings, Part I*. Ed. by Juan A. Garay and Rosario Gennaro. Vol. 8616. Lecture Notes in Computer Science. Springer, 2014, pp. 444–461. ISBN: 978-3-662-44370-5. URL: https://doi.org/10.1007/978-3-662-44371-2_25.

[GVY17] Daniel Genkin, Luke Valenta and Yuval Yarom. 'May the Fourth Be With You: A Microarchitectural Side Channel Attack on Several Real-World Applications of Curve25519'. In: *ACM Conference on Computer and Communications Security (CCS)*. 2017.

[Har13] Yannick Harrel. *La cyberstratégie russe*. French. Nuvis, 2013.

[Koc96] Paul C. Kocher. 'Timing Attacks on Implementations of Diffie-Hellman, RSA, DSS, and Other Systems'. In: *Advances in Cryptology - CRYPTO '96, 16th Annual International Cryptology Conference, Santa Barbara, California, USA, August 18-22, 1996, Proceedings*. Ed. by Neal Koblitz. Vol. 1109. Lecture Notes in Computer Science. Springer, 1996, pp. 104–113. ISBN: 3-540-61512-1. URL: https://doi.org/10.1007/3-540-68697-5_9.

[KT79] Daniel Kahneman and Amos Tversky. 'Prospect theory: An analysis of decision under risk'. In: *Econometrica: Journal of the econometric society* (1979), pp. 263–291.

[Lim14] Kevin Limonier. 'La Russie dans le cyberespace: représentations et enjeux'. French. In: *Hérodote* 1 (2014), pp. 140–160.

[LS12] Rolf Lidskog and Göran Sundqvist. 'Sociology of Risk'. In: *Handbook of Risk Theory: Epistemology, Decision Theory, Ethics, and Social Implications of Risk*. Ed. by Sabine Roeser, Rafaela Hillerbrand, Per Sandin and Martin Peterson. Dordrecht: Springer Netherlands, 2012, pp. 1001–1027. ISBN: 978-94-007-1433-5. URL: https://doi.org/10.1007/978-94-007-1433-5_40.

[MOP07] Stefan Mangard, Elisabeth Oswald and Thomas Popp. *Power analysis attacks - revealing the secrets of smart cards*. Springer, 2007. ISBN: 978-0-387-30857-9.

[MPG05] Stefan Mangard, Thomas Popp and Berndt M. Gammel. 'Side-Channel Leakage of Masked CMOS Gates'. In: *Topics in Cryptology - CT-RSA 2005, The Cryptographers' Track at the RSA Conference 2005, San Francisco, CA, USA, February 14-18, 2005, Proceedings*. Ed. by Alfred Menezes. Vol. 3376. Lecture Notes in Computer Science. Springer, 2005, pp. 351–365. ISBN: 3-540-24399-2. URL: https://doi.org/10.1007/978-3-540-30574-3_24.

[Mur+10] Steven J. Murdoch, Saar Drimer, Ross J. Anderson and Mike Bond. 'Chip and PIN is Broken'. In: *31st IEEE Symposium on Security and Privacy, S&P 2010, 16-19 May 2010, Berleley/Oakland, California, USA*. IEEE Computer Society, 2010, pp. 433–446. ISBN: 978-0-7695-4035-1. URL: https://doi.org/10.1109/SP.2010.33.

[Odl03] Andrew Odlyzko. 'Economics, psychology, and sociology of security'. In: *Computer Aided Verification*. Springer. 2003, pp. 182–189.

[PB10] Sören Preibusch and Joseph Bonneau. 'The Password Game: Negative Externalities from Weak Password Practices'. In: *Decision and Game Theory for Security - First International Conference, GameSec 2010, Berlin, Germany, November 22-23, 2010. Proceedings*. Ed. by Tansu Alpcan, Levente Buttyán and John S. Baras. Vol. 6442. Lecture Notes in Computer Science. Springer, 2010, pp. 192–207. ISBN: 978-3-642-17196-3. URL: https://doi.org/10.1007/978-3-642-17197-0_13.

[Qui+89] Jean-Jacques Quisquater, Myriam Quisquater, Muriel Quisquater, Michaëll Quisquater, Louis C. Guillou, Marie Annick Guillou, Gaïd Guillou, Anna Guillou, Gwenolé Guillou, Soazig Guillou and Thomas A. Berson. 'How to Explain Zero-Knowledge Protocols to Your Children'. In: *Advances in Cryptology - CRYPTO '89, 9th Annual International Cryptology Conference, Santa Barbara, California, USA, August 20-24, 1989, Proceedings*. Ed. by Gilles Brassard. Vol. 435. Lecture Notes in Computer Science. Springer, 1989, pp. 628–631. ISBN: 3-540-97317-6. URL: https://doi.org/10.1007/0-387-34805-0_60.

[Sch08] Bruce Schneier. 'The Psychology of Security'. In: *Progress in Cryptology - AFRICACRYPT 2008, First International Conference on Cryptology in Africa, Casablanca, Morocco, June 11-14, 2008. Proceedings*. Ed. by Serge Vaudenay. Vol. 5023. Lecture Notes in Computer Science. Springer, 2008, pp. 50–79. ISBN: 978-3-540-68159-5. URL: https://doi.org/10.1007/978-3-540-68164-9_5.

[SS10] Neil J. Smelser and Richard Swedberg. *The handbook of economic sociology*. Princeton University Press, 2010.

[SXZ17] 'Access Control'. In: *Encyclopedia of GIS*. Ed. by Shashi Shekhar, Hui Xiong and Xun Zhou. Springer, 2017, p. 39. ISBN: 978-3-319-17884-4. URL: https://doi.org/10.1007/978-3-319-17885-1_100020.

[Tho+15] Kurt Thomas, Danny Yuxing Huang, David Y. Wang, Elie Bursztein, Chris Grier, Tom Holt, Christopher Kruegel, Damon McCoy, Stefan Savage and Giovanni Vigna. 'Framing Dependencies Introduced by Underground Commoditization'. In: *14th Annual Workshop on the Economics of Information Security, WEIS 2015, Delft, The Netherlands, 22-23 June, 2015*. 2015. URL: http://www.econinfosec.org/archive/weis2015/papers/WEIS_2015_thomas.pdf.

[Vau02] Serge Vaudenay. 'Security Flaws Induced by CBC Padding - Applications to SSL, IPSEC, WTLS ...' In: *Advances in Cryptology - EUROCRYPT 2002, International Conference on the Theory and Applications of Cryptographic Techniques, Amsterdam, The Netherlands, April 28 - May 2, 2002, Proceedings*. Ed. by Lars R. Knudsen. Vol. 2332. Lecture Notes in Computer Science. Springer, 2002, pp. 534–546. ISBN: 3-540-43553-0. URL: https://doi.org/10.1007/3-540-46035-7_35.

[YGH17] Yuval Yarom, Daniel Genkin and Nadia Heninger. 'CacheBleed: a timing attack on OpenSSL constant-time RSA'. In: *J. Cryptographic Engineering* 7.2 (2017), pp. 99–112. URL: https://doi.org/10.1007/s13389-017-0152-y.