Wireless Personal Communications (2019) 106:85–117
https://doi.org/10.1007/s11277-019-06265-6

Why Cryptosystems Fail Revisited

Geir M. Køien[1]

Published online: 19 March 2019

Abstract
In the paper "Why Cryptosystems Fail", Ross Anderson ponders the question about why cryptosystems really fail. Obviously, there may be weak crypto-algorithms, too short key lengths and flawed crypto-protocols. However, these were not the main reason why cryptosystems failed. Anderson discovered that the problem had more to do with misplaced trust and misconceptions of the threats the systems faced. Now, more than 25 years later, it seems prudent to revisit the question of why cryptosystems fail. We investigate the original paper, and evaluate to what extent the situation is similar today.

Keywords Cryptosystems · Failure · Threat model · Mental model · Cryptography · Security controls · Trust · Assumptions · Human factors · Skills shortage

1 Introduction

"Why Cryptosystems Fail" [1] was a very timely paper when it was published. Moreover, it was an influential paper and it clearly demonstrated that there was more to security than advanced mathematics and cryptography. There was also a growing realization that while cryptography as a subject was beginning to mature, the security practices were not really showing the progress that one may had hoped for.

1.1 Background and Motivation

Our motivation for investigating this subject again is to find out what lessons we can learn today about "Why Cryptosystems Fail". Of course, many cryptosystems are actually very successful, so the overall picture is not that everything fails. There are many success stories, but our dependence on secure systems is a lot more pronounced today. There is therefore some urgency in trying to find out what can be done to prevent or mitigate future failures.

✉ Geir M. Køien
geir.koien@uia.no

1 University of Agder, Kristiansand, Norway

1.2 Cryptography in Context

This paper is not about cryptography, but cryptography and cryptographic systems is foundational to strong cyber security. Cryptography is generally not the weakest link in a cryptosystem, but when it is, then it tends to compromise the overall system.

1.2.1 Cryptosystem Versus Cryptographic Primitives

An important point to be made, both for the original paper and for this paper, is the distinction between cryptographic primitives and cryptosystems. A cryptographic primitive may or may not suffer from weaknesses, but even when using a perfectly secure cryptographic primitive, the surrounding cryptosystems may have severe flaws. So, the term *cryptosystem* implies a systems perspective, and that does include the overall system architecture, the various components, the security architecture, the individual security controls, etc.

A cryptographic primitive (crypto algorithm), on the other hand, is a single component, and a very specialized on at that. The expertise required for designing a cryptographic primitive is very high, and it is a skill that requires high mathematical sophistication and deep security insights. And, it is also the case that even the best experts do get it wrong quite frequently. Only after being scrutinized extensively by other cryptographic specialists, will one slowly being believing in the merits of a new cipher algorithm.

1.2.2 A Few Words About Bad Cryptography

A humours account of how to get cryptography wrong can be found in the paragraph "Beware of snake oil" in [2] (page 43) by Phil Zimmermann. It is a classical case of homegrown cryptography, which is almost always weak and insecure. Cryptographer Bruce Schneier wrote a memo about the problem [3], where he notes that:

> Anyone, from the most clueless amateur to the best cryptographer, can create an algorithm that he himself can't break. It's not even hard. What is hard is creating an algorithm that no one else can break, even after years of analysis. And the only way to prove that is to subject the algorithm to years of analysis by the best cryptographers around.

The two references above are fairly old (both 1998), but the problem resurfaces surprisingly often. One recent occasion (2017) concerned the cryptography for the cryptocurrency IOTA, where they devised their own hash function. It was no good [4]. We can sum this up nicely as "Cryptography is harder than it looks" [5]. With respect to this paper, we can also add that cryptography is the easy part. It is most often adequate and the real problems with cryptography is the usage. And even then, there are likely other flaws and vulnerabilities in the system that are far worse.

Bad cryptography may or may not be a common or even important problem, but it highlights the problem of how to distinguish between good and bad security. We shall return to this topic in Sect. 2.4.

1.3 Life-Cycle Phases

One typically distinguishes between the different life-cycle phases of a system. Classically, one has *design*, *implementation* and *operations*. However, many system are large and long-lived. They will thus go through iterations of the design-implementation-operations cycle, where functionality is added, removed and generally upgraded. To get the best possible security, one needs to be security-aware in all of these phases and adhere to best practices, etc.

- *Design* Carried out by systems experts and domain experts.
  Consists mainly of defining the system, the components, the interfaces, etc.
- *Implementation* Carried out by (topic) specialists.
  Implementation is the concrete manifestation of the design. This amounts to development of hardware, software, etc., and to the configuration and integration of these components.
- *Operations* Carried out by administrators, clerks/staff, and end-users.
  Operations consists of everything involved in the daily use of the systems. For critical systems, some level of operations and management (O&M) is expected to be in place. For systems with explicit security requirements, this also amounts to having security O&M capabilities in the organization that runs the system. It is in O&M where security policies really matters, and here is where organizational aspects such as security culture and attitude matters the most.

We note that there generally is little overlap between the groups that do *design*, *implementation* and *operations*. We also observe that the level of security and cryptography expertise is very different in the groups, and that communications between the groups will be hindered by different mental models of the system, by different levels of technical insight, and by different goals. When concerning weaknesses, and the propensity for failures, this is a game of conjunction. Every aspects must be done right; it does not suffice to have a brilliant design and impeccable implementation, if the operating procedures and security policies are inconsistent or incomplete, and actual operations carried out in an ad-hoc fashion.

1.4 Failure in Context

The term *failure* must be put in context. Different people use failure to mean quite different things. The same goes for "broken", etc.

What does it mean when a crypto-algorithm is broken*?*

It may mean that a significant theoretical flaws has been found. It may or may not be possible to exploit the flaw. An attack that reduces the guessing complexity of a cryptographic key by 100.000 would seem pretty significant. In reality, that would represent less that 17 bit reduction, and if the key was 128 bit to start with, it is still utterly impractical to brute force guessing the entire key. Of course, sometimes the algorithm is completely broken and practical attacks become almost trivial. The "Unsafe at any key size; an analysis of the WEP encapsulation" paper highlight this [6].

What does it mean when there is a serious flaw in a cryptosystem?

It may be very bad, but it may surprisingly also be inconsequential. One needs to take into account of the actual usage of the flawed component, the exposure and whether or not there are ways to mitigate the effects of the flaw. Flaws like the infamous OpenSSL Heartbleed bug [7] were very serious indeed. Other potentially devastating flaws were either easy to mitigate or had little exposure, and thus had only minor impacts. Sometimes, it is a matter of scalability. In 2003, Barkan et al, published an elegant and effective attack on the GSM system ("Instant ciphertext-only cryptanalysis of GSM encrypted communication", [8]). However, 15 years later, GSM cryptography, while surely broken, is still effective on an aggregate level.

What does it mean when penetration testing has "compromised" the system?

Penetration testing is a work method to exposure weaknesses and flaws. It is a branch of security testing, but here the goal is explicitly to break the security. Commonly, the pentest team will find serious flaws and omissions, but one should not be too alarmed by this. The problems are frequently fixable by carrying out proper security hardening and by implementing good security policies. Pentesting is thus a part of a quality and assurance process.

1.5 Related Work

There has been many papers written about security failures. These generally either focus on cryptography failures or generic security failures. We shall mentions some of these later in the paper. But, to the best knowledge of the author, there does not seem to be any paper revisiting the premises of the original "Why Cryptosystems Fail" paper.

1.6 Paper Layout

Section 1 *Introduction* sets the scene and defines the context. In Sect. 2 *Analysis of the Original Paper*, we take a look at what the original paper actually was about. Then, we provide commentary and observations, trying to illuminate whether or not the central claims are still valid today. This section contains the main body of the paper. Section 3 *Threat Landscape and Threat Models* highlights the current threat landscape and two threat modelling concepts. Section 4 *Discussion* provides a high-level discussion of the main findings. Section 5 *Conclusion* concludes the paper, with a call for action concerning solving the "skills shortage".

2 Analysis of the Original Paper

The analysis will consisted of a structured walk-through of the original paper. We highlight the main aspects, and provide discussion and commentary as we find appropriate. The examples in the original paper are all centered around the banking systems in the UK. We interpret the issues as generic and we also try to reframe the issues in a 2018 context when possible.

2.1 The Introduction

This is the analysis of the "Introduction" in [1].

2.1.1 How Does Systems Fail?

The paper starts out by noting that “Designers of cryptographic systems are at a disadvantage to most other engineers, in that information on how their systems fail is hard to get”. The context was that of government and military systems, in which the secrecy level is high.

One problem is that a cryptosystem may fail silently, and the affected party might not ever notice the failure. This stand in sharp contrast to, for instance, an aircraft crash, where it would be immediately obvious. In the airplane crash case, there would be a public inquiry and a through investigation of the accident. Root causes would be hunted for, and one would generally use the findings to improve future air safety. While not mentioned in [1], aircraft security is covered by the so-called Chicago Convention. Formally, it is now known as the “Convention on International Civil Aviation—Doc 7300” [9]. In particular, we have Annex 13, which covers “Aircraft Accidents and Incident Reporting”. This is why we have mandatory accident investigations and free sharing of any lesson learned in the aviation sector.

In 1993: 1 *Security designers lacked information about how systems failed.*

The situation concerning design feedback may be slightly better today, and policies of “full disclosure” are now not uncommon. Databases such as the US National Vulnerability Database (NVD, https://nvd.nist.gov/) and the Common Vulnerabilities and Exposures (CVE) reporting schemes, contribute significantly in the right direction.

In 2018: 1 *The problem of understanding how systems fail is still there, but one generally also have access to many tools that mitigate the problem.*

Lesson 1 *Understanding why cryptosystems fail is important.*

2.1.2 Wrong Threat Model

A central point in the paper is the claim that the systems were designed with the wrong threat model. We shall return to the “wrong threat model” issue later on, but suffice to say that this is still a major issue.

2.2 Automatic Teller Machines

This is the analysis of Sect. 2 “Automated Teller Machines” (ATM) in [1]. ATMs and the banking sector was the main technical context in the original paper. The ATMs dispense money, they communicate with central branches and they are distributed widely. The ATMs are a target for fraud, and as such its clear that they need security controls and cryptographic protection.

However, the requirements for security was heavily influenced by the legal context. The relevant legal context in the US generally protected the user. The system owners, the banks, are responsible for the ATMs and can be made liable for fraud. That is, unless the end user is shown to be particularly careless, the banks are held responsible. Thus, the banks will have the burden-of-proof concerning ATM frauds. It is therefore very much in their best interest to provide protection that takes this into account.

In 1993: 2 *The burden-of-proof led to improved security in US banks.*

The situation in the UK was very different. Here, the end-user had the burden of proof. There was no presumption of innocence concerning the end-user. The banking systems were deemed to be more or less infallible, and any error would then have to be due to external factors (the end-user). This caused a major asymmetry in power and control. The party with almost no power and almost no chance of affecting the system, is the party that has prove his/her innocence. The party with power and control wins any argument by default.

In 1993: 3 *It was untenable for the end-users to have the burden-of-proof.*

In economics there is the concept of *externality*. An externality is defined as a cost or benefit that affects a party who did not choose to incur that cost or benefit. In the UK banking context, the cost of security would fall to the UK banks, which would not benefit from the added security. To change the situation, one would need to *internalize* the externality. This internalization is exactly what the laws in the US brought about. Anderson did not explicitly make this point in his 1993 paper, but it is quite clear from the context.

In 1993: 4 *Inadequate internalization of security costs by the system owner.*

The above described asymmetry is certainly still there when end-users are buying or subscribing to services from large corporations, or when they are dealing with authorities and governments.

In 2018: 2 *Legal context: Burden-of-proof responsibly is important.*

The burden-of-proof is still widely left with the end-user, although many corporations have policies in place that mitigate the problem. Various legislations and regulations also comes into play.

In 2018: 3 *It is generally untenable for end-users to have the burden-of-proof.*

Corollary:

In 2018: 4 *The externality problem is a real problem.*

Legal and regulatory impacts change the power balance between the parties. This may affect burden-of-proof and may contribute to internalization of security costs. From this we can learn the following lessons:

Lesson 2 *Security costs should be internalized (if at all possible).*

The externality versus internality asymmetry is also about *accountability*. The book "Skin in the Game" [10] discusses such asymmetries in depth, and goes a long way to advocate accountability by having "skin in the game" (something to loose).

2.3 How ATM Fraud Took Place

Section 3 in [1] is concerned with how ATM frauds took place.

2.3.1 The Simple Examples—Insiders

The author outlines several concrete cases of ATM fraud. These were mostly caused by low-tech attacks. The attacks were mostly simple schemes that operated by attacking flawed accounting schemes and gullible operating procedures. In particular, many attacks involved system insiders, and there was not much to prevent these from carrying out the misdeeds.

In 1993: 5 *Several of the attacks were conducted or aided by insiders.*

The concrete cases are not very important today, but serves well to illustrate the insider problem. The insider problem was exacerbated by missing control mechanisms, which meant that there was little chance of being caught. In security parlance, there was little insider *accountability* and there was a lack *non-repudiation* controls for the parties. Schemes like event logging cannot prevent attacks, but their existence may deter known parties from being dishonest.

In 1993: 6 *There were no, or almost no, logging of transactions/events.*

Without protected audit trails, it was very hard to know what had happened. It seems reasonable to assume that the lack of event logging was a major contributing factor as to why the number insider attacks was so high.

We are generally better off today when it comes to accountability and non-repudiation and there are many excellent tools available. Of course, the tools must be deployed and there must be procedures in place to be able to act on what the tools report.

In 2018: 5 *The insider threat is still a real threat.*

In 2018: 6 *Insider attacks are not stopped by logging, but logging can have a preventive effect.*

Good logging facilities may make it possible to hold the culprit accountable. Of course, logging alone is not enough. One needs the whole gamut of detection and response capabilities in most systems. Additionally, one should provide means to enforce the principle of least privilege. This would in practice entail deploying authorization and access control mechanisms. That will limit what a dishonest insiders can do.

Lesson 3 *Security controls to provide accountability are important.*

Lesson 4 *Systems must have Detection and Response capabilities.*

Lesson 5 *The principle of least privilege should be enforced.*

2.3.2 The Simple Examples—Outsiders

The outsider attacks includes various skimming attacks and similar. It also included attacks on distribution of bank cards. The distribution was mostly carried out by the postal services, and theft of cards in postal transit was recognized as a major problem. Bank cards lost in postal transit is an example of a weakest link in an outsourced part of the system. It was also an avoidable weakest link.

In 1993: 7 *Outsourced services were an easy to exploit weakest link.*

The postal service did not advocate sending valuables in letters. For valuables, they had the registered post service. This would have been the appropriate service for distribution of bank cards.

The outsourcing problem is still with us and in full force. One example is the distribution of one-time codes, which often done by means of the short-message service (SMS). There is no end-to-end protection of SMS messages, and the service must be considered totally unfit for sensitive contents. Today the most prevalent outsourcing may perhaps be found in the cloud computing area. The cloud service provider may provide some basic security for free, but if one wants comprehensive security, then this must be explicitly provided. Outsourced services can be well protected, but this never be assumed to be the default.

In 2018: 7 *Outsourced services is still often a weakest link.*

Outsourcing must be done in a responsible way. The system owner must ensure that this is the case.

Lesson 6 *Outsourcing must be done in a responsible way.*

Handling of security may also be (partially) outsourced. Detection and Response is one part that may be successfully outsourced, and so too is incident analysis and recovery. Partial outsourcing is perhaps the best option. However, keep in mind that:

Lesson 7 *Security responsibility cannot be outsourced.*

Anderson also mentions cases with abuse of methods and procedures used during testing and verification. It is not uncommon for the designers and implementor to create small tools and utilities that bypass the normal working procedures. These tools can be very useful during implementation, but may be dangerous tools to have in a production system.

In 1993: 8 *Extraneous functionality in production systems was a liability.*

Proper release procedures should contain procedures that remove such functionality. So, this is a problem concerned with product release, and as such a quality assurance problem.

In 1993: 9 *Lacklustre system quality assurance was a problem.*

Extraneous functionality is a general problem, and it is bigger than the obvious problems associated with system-breaking test tools. That brings us over to system hardening, which is the process of removing all redundant parts of the installed functionality. It is all part of system quality assurance, and if it is lacking, then problems are very likely to surface.

In 2018: 8 *Extraneous functionality in production systems is a liability.*

In 2018: 9 *Lacklustre quality assurance is a problem.*

Anderson also mentions that some of the outsider attacks exploited implementation flaws in the systems. This is of course to be expected, and by itself nothing extraordinarily. The problem was almost certainly made worse by the fact that the banks did not have the proof-of-burden and that there was very little logging in place.

Lesson 8 *Minimizing the attack surface is vitally important.*

Lesson 9 *Strong quality assurance is a prerequisite for security.*

2.3.3 The Simple Examples—The Four Digit PIN

The paper provides an example of a scheme invented by one bank to make it easier for customers to remember their Personal Identification Number (PIN) code. The scheme was deeply flawed, going from a guessing likelihood of 1 in 3333 to 1 in 8.

In 1993: 10 *Bad advise had a significant adverse effect on security.*

The problems with users having to remember PIN codes and passwords remains with us. Today, password guessing attacks are quite effective. What seem good advise a few years ago may be bad advise today.

In 2018: 10 *Bad advise regarding PIN codes and passwords abound.*

Passwords and PINs are static and guessable (various attacks). When used alone, they can only provides weak security. Generally one wants to have two-factor authentication (2FA) schemes, or even three factor (3FA) schemes, for high-security access.

- 1FA: What you remember (password/PIN)
- 2FA: The above, and what you have (hardware crypto-token, etc.)
- 3FA: The above, and what you are (biometrics)

Lesson 10 *Misguided advise can be harmful (do not trust all advise).*

Lesson 11 *Passwords and PIN codes alone cannot provide strong security.*

Other problems included PINs with digits that were predictable. It even seemed that the need for unpredictable PIN was not fully appreciated. Secrets that are predictable is a cardinal sin in cryptography.

In 1993: 11 *Information that needed to be unpredictable was predictable.*

Cryptosystems are generally critically dependent on unpredictability. In particular, almost all cryptosystems depend on the use of pseudo-random numbers. If these are predictable to the outsider, then the cryptosystem is easily compromised. Abadi and Needham explains the need for being explicit about the unpredictability needs in [11]. A prominent example of a compromised pseudo-random number generator (PRNG) is the case of the `Dual_EC_DRBG` generator [12].

In 2018: 11 *Compromised unpredictability leads to compromised security.*

The paper by Checkoway et al [13] further illustrates the importance of pseudo-random numbers. It is a very instructive article, and it highlights both implementation problems, quality assurance problems and the seriousness of Advanced Persistent Threat (APT) actors.

We have then learned that:

Lesson 12 *Requirements for unpredictability must be explicit.*

Lesson 13 *Pseudo-random number generators must be verified.*

2.3.4 More Complex Attacks

The original paper mentions that there were comparatively few high-tech attacks or attacks that directly targeted the cryptography.

In 1993: 12 *High-tech attacks required sophisticated adversaries. There were relatively few known high-tech attacks.*

The situation today is probably quite similar in that low-tech attacks are common and widespread. However, there are some significant changes between 1993 and today. Notably, almost all systems are connected to the Internet, which allows for a whole new class of attacks. The fact that one can scale an attack to affect millon of end-points means that the "attack economy" has changed. The essentially friction-free scaling means that the delta cost in attacking yet another end-point is close to zero. The attack economy has also changed in another way. One may now buy sophisticated attacks (as a service) or one may by the attack software.

Organized crime in the digital arena is a fact, and they use whatever tools they may find. They are also cost–benefit oriented, and if a low-tech approach is cost-effective, then that is likely what they deploy.

Then we have the nation state actors. Since the intelligence agencies are not too concerned with costs, they can afford to spend a lot of money on breaking cryptographic primitives and other "hard" targets. They can therefore carry out APT attacks. These may be carried out with multiple attack vectors, over a prolonged period and on multiple platforms.

In 2018: 12 *Advanced attacks are not uncommon.*

- *Low-tech attacks still dominate. Scalability is key factor.*
- *Scalable attacks are far worse than single point attacks (system perspective).*
- *High-value systems/persons: the threat from high-tech attacks is real.*
- *Effective attacks on (weak) crypto exists, but they often scale poorly.*
- *Attacks on weak pseudo-random numbers exists.*
- *There is an attack marked, which lead to proliferation of high-tech attacks.*
- *Proliferation is hard to contain—cases of unintended outcomes and collateral damage are common.*

Back in 1993, the banks frequently had outdated equipment. Security updating was not a common occurrence.

In 1993: 13 *Security updates were very rare. Old equipment that could not be updated was a serious security risk.*

Today, operating systems and other systems software packages frequently provide security updates. Web browser commonly provide automated updates, and the user does not have to do anything to get the latest patches. But, most software will have some sort of best-before date, and they will not receive updates and security patches after that date. These best-before dates are commonly publish and should be well-known.

In 2018: 13 *Security updates are prevalent today. Old equipment and software will only receive security updates for a certain period.*

Products in the Internet-of-Things (IoT) category is probably the biggest current problem area in this respect. According to [14], security updating is a big issue with IoT and deployed hardware may become obsolete.

We have then have the following lessons:

Lesson 14 *Timely handling of security updates is absolutely essential.*

Lesson 15 *Old equipment that cannot be updated is a serious security risk.*

2.3.5 Problems with Encryption Products

Encryption products, then and now, may suffer from weaknesses. There was a fair amount of FUD (Fear, Uncertainty and Doubt) about. This affected the uptake of new products. The products also had to be from the "right" company. It was also noted that the "right" hardware tended to be very expensive, and so many banks choose to deploy security credentials in software. Software-only protection can never be very strong.

In 1993: 14 *Many crypto products had severe technical problems.*

In 1993: 15 *Software-only protection cannot provide strong security.*

In 1993: 16 *Misplaced trust was a significant problem.*

Generally speaking, there is a lot of brand trust. People tend to trust big corporations they recognize by name, and IBM was certainly trusted. The trust was there out of convenience, necessity and dependence, but also because brand owners have an incentive to protect their brand. However, appearances can be misleading. Ken Thompson, in his Turing Award Lecture, “Reflections on Trusting Trust”, brilliantly outlined why trust is so hard to ascertain [15].

In 2018: 14 *Some crypto products have severe problems.*

In 2018: 15 *Software-only protection cannot provide strong security.*

In 2018: 16 *Misplaced trust is a major problem.*

The lessons we can learn from this is:

Lesson 16 *Crypto products must not be blindly trusted.*

Lesson 17 *Hardware support is needed for strong security.*

Lesson 18 *Trust must be warranted and continually evaluated.*

Anderson also mentions problems with backdoors in the products. Strong crypto does not discriminate, and will protect villains as well as innocent end-users. There is therefore frequently calls to have backdoors installed in security products. This is a classical tradeoff question, protection of society at large versus the privacy of the individual.

In 1993: 17 *Backdoors tend to compromised security.*

However, if one chooses to deliberately weaken the security and/or insert deliberate backdoors, then one cannot expect the security to be very good and one cannot expect anyone to trust the security. If the weakness is there, it will be exploited, and we have no guarantee whatsoever that it is only used for “good” purposes or by the “good” authorities/people.

In 2018: 17 *The debate over the use of backdoors and deliberately weakened security products continues.*

Thus, the very reason for deploying security and cryptography is compromised by inclusion of backdoors.

Lesson 19 *Cryptographic backdoors must be considered harmful.*

2.3.6 Operating Procedures and Assumptions

Wrong assumptions is a serious source of problems. Initially, the ATMs were installed within a bank building. It therefore came to be assumed that ATM was located in a secured environment. However, the ATMs were very soon also deployed outside of bank buildings and in a fairly exposed environments.

In 1993: 18 *ATM: Exposed locations meant that assumptions about a protected environment was invalid.*

Threats due to exposed locations are still an issue. This problem is first and foremost a problem for widely distributed equipments, and for inexpensive equipment. This concerns many Internet-of-Things (IoT) and machine-to-machine (m2m) products and services. There obviously is cost considerations at play here, but bad design and bad engineering is also to blame. Again we have externalities at play. The cost associated with producing better hardward/software does not generally provide many benefits to the vendor. The service operator, one the other hand, will suffer from poor security.

In 2018: 18 *Exposed locations is a problem. Inadequate protection then becomes a major problem.*

We shall return to the topic of misguided assumptions in Sect. 2.4.

Lesson 20 *Physical exposure and inadequate protection is a problem.*

Anderson also noted that there were problems with insecure backups. This problem is generally caused by a lack of security architecture and security policies. Backups are useful and necessary, and the banks were actually good with data backups per se. However, there was also backups of the secure module software and data, but the backups were not protected.

In 1993: 19 *Insecure backups was a problem. The root cause was a lack of security architecture and inadequate security policies.*

Today most large systems have a security architecture. These are typically designed and implemented by the systems vendors, and they may draw heavily on established security standards. It is a very complex task to design and implement a security architecture. Security policies are also difficult to get right. These are heavily dependent on how the systems are used and the business model behind. This is hard to capture at the best of times.

In 2018: 19 *Incomplete and inconsistent security architectures and security policies is a large problem.*

A problem that was present in 1993, but that is much more pronounced today, is that the environment is highly dynamic. The technical solutions are continually improved and updated, and the systems are in constant flux (see life-cycle phases, etc. in Sect. 1.3). There is an endless stream of new threats, and consequently a lot of security patching. Patching and ad-hoc measures to prevent ongoing attacks puts pressure on the security architecture consistency. It also means that the security policies will soon be outdated. There is therefore a realization that security is a process.

There are several lessons here:

Lesson 21 *Systems need a complete and comprehensive security architecture.*

Lesson 22 *Systems need complete and comprehensive security policies.*

To be effective, the security policies must be enforceable and communicated clearly to the involved parties.

Lesson 23 *Security is a process. All parts have a best-before date.*

Anderson also mentions "sloppy operating procedures" as being a big problem. This has everything to do with security policies and with the security culture in the banks. The security culture is of course an aspects of human factors in security. Anderson mentions human factors several times, and stresses that many of the problems stem from problems associated with bad usability, etc.

In 1993: 20 *Human factors was seen as a crucial aspect of system security.*

Today, it is generally appreciated that most security failures involve human factors in one form or another. For instance, phising attacks almost always requires user interactions to succeed. On the other hand, human alertness and a proper security mindset, can improve the security and handle situations that are not well covered by a security policy. Security culture is amongst the factors, and as is indicated in the report on "The Norwegian Cyber-security Culture" [16], one may still have a long way to go. And, of course, there are many types of cultures and subcultures around. One should not make assumptions about the actual security culture in place.

In 2018: 20 *Human factors is a critical aspect of system security.*

Lesson 24 *Human factors must be taken into account.*

2.3.7 Key Management

Key management is explicitly highlighted as a problem area. The use of fixed and hard coded key material was quite common. Needless to say, but these practices invalidates the security of even the strongest crypto algorithm.

In 1993: 21 *Inadequate key management was a problem.*

The situation is still a difficult one, and it isn't uncommon to find key material embedded in software. We also have the problem that session keys, password, etc., often remain in memory after their primary use has ended. The tool Mimikatz, infamous for its inclusion in the NotPetya malware, operates by extracting passwords, PIN codes, etc., from memory. Mimikatz is available on GitHub (https://github.com/gentilkiwi/mimikatz).

In 2018: 21 *Inadequate management of security credentials is a real problem.*

There exists several guidelines and textbooks on how to implement security sensitive functions, including how to handle passwords and secret codes. An example is the CERT C coding standard [17]. It is a difficult problem, but is mostly an engineering problem.

Lesson 25 *Special care must be taken when handling security credentials.*

2.3.8 Lack of Proper Training and Documentation

Lack of proper training was seen as a problem back in 1993. This was related to operational procedures, and it coincided with a lack of documentation.

In 1993: 22 *Lack of training and documentation lead to incorrect configuration and usage of cryptographic equipment.*

The situation regarding documentation seems to have improved, but lack of training is still a huge problem.

In 2018: 22 *Lack of training implies an increased risk of incorrect configuration and usage of cryptographic security controls.*

We shall return to the problem of skills shortage in Sect. 2.3.11.

2.3.9 Cryptoanalysis and Key Length

Cryptoanalytical attacks was not seen as a big threat in 1993.

In 1993: 23 *Cryptoanalytical attacks were not a major issue.*

Comparatively, crypto breaking attacks may not be a big threat in 2018 either.

One crypto related threat that must be taken into account is that of the quantum computer threat to asymmetric cryptography. The theory behind the threat is quite old (1994) and due to Peter Shor [18]. There is intense research in this field today, and the creation of a practical quantum computer is generally perceived to be more of an engineering problem than a fundamental physics problem. Should there be a quantum computer able to run Shor's algorithm, then almost all asymmetric crypto-algorithms will be solvable in polynomial time. Asymmetric crypto is widely used in authentication and in key agrement protocols. It will be very costly and difficult to replace these with quantum-safe solutions. International standards bodies have recognized the problem, and work on quantum-safe asymmetric crypto primitives is ongoing [19, 20]. Symmetric key cryptography is comparatively safe from quantum attacks, but one will need to double the key length (state space) to remain safe. This is generally doable, but it would be expensive and complicated to orchestrate the roll out of revised algorithms.

In 2018: 23 *Cryptoanalytical attacks are not a major issue, but it must be considered.*

Lesson 26 *Cryptosystems must be designed to allow algorithm changes.*

In 1993, there was problems associated with key lengths. Anderson mentions specifically use of very short RSA keys, but the key length of single-DES encryption would also have been an issue.

In 1993: 24 *Insufficient key length was a serious cryptographic problem.*

Today, there is a general consensus about key lengths. The EU body ENISA has published a report on recommended algorithms and key sizes [21]. That of course, leaves us with the legacy problem. This problem is serious enough, but in the wider context of the overall threat landscape, even this threat is relatively minor. There is also commonly consensus about minimum standards, and the major web browser vendors often enforce such standards. For instance, Microsoft published a Security Advisory [22], that defined a deprecation deadline for use of the SHA-1 hash function in digital certificates. This has effectively phased out the use of SHA-1 in digital certificates.

In 2018: 24 *Insufficient key length and outdated algorithms is an issue in legacy systems.*

Generally speaking, the recommendations today are sound and quite future proof. Effective quantum machines, if or when they appear, will totally change that picture. As a minimum one must be able to double the key length for symmetric algorithm and hash algorithms. Longer key length must be accounted for in the key distribution/agreement protocols too.

Lesson 27 *Cryptosystems must be designed to allow extending the key lengths.*

2.3.10 Implication for Bankers

The banking systems were reported to be highly complex and the modules were tightly interlocked. Unneeded complexity is a system and security problem. We illustrate this thinking with a quote from the "Zen of Python" (PEP 20, [23]):

> Simple is better than complex.
> Complex is better than complicated.

The banks operated many system, and the banking systems were described as evolved, rather than designed. The quality of the design and implementations appears to have suffer from this.

In 1993: 25 *The systems were very complex, which left the system susceptible to implementation errors and to complicated operating procedures.*

Bad design or designs that evolves to become bad, is still with us. The delta cost of investing in a new design or to properly revise the design, is often seen as prohibitive in a business context. That is, even tough the long-term cost of patching an old system may far exceed the cost of a new design, the short-term cost of patching the system is almost always lower then redesigning the system. Patching a system, or adding features in an ad-hoc manner, will tend to increase the complexity of the system.

In 2018: 25 *Short-term cost optimization favor patching an existing design.*

While patching may be a necessary evil, it is prudent practice to ensure that even emergency patches is in line with an existing design. This minimizes complexity and reduces the "entropy" of the system.

Lesson 28 *Every effort should be made to keep complexity to a minimum.*

Another realization was that there would be "program bugs and operations blunders;". That is, there will inevitable be vulnerabilities in any one mechanism and security control.

In 1993: 26 *Vulnerabilities and exposure was inevitable.*

Even the best security architecture will have flaws. Systems are routinely exposed, and on the Internet the exposure is global by nature.

In 2018: 26 *Vulnerabilities and exposure is inevitable.*

It is important to have an architecture that permits defense-in-depth security controls. Defense-in-depth is a strategy and a way of structuring a layered security controls [24]. Defense-in-depth should be part of any security architecture, and properly applied it will prevent or mitigate single-point-of-failure cases.

Lesson 29 *Defense-in-depth is a recommended strategy.*

2.3.11 Implications for Equipment Vendors

In Section 3.3 in [1], Anderson returns to the issue of a skills shortage. One this he notes is that the vendors of the day probably overestimated the competence of their customers. This could lead to wrong procurements. He mentions that some of the products, for instance an IBM 4753 machine, was capable and flexible, but that the "..hidden cost of this flexibility was that almost all their customer lacked the skills to do a proper job..".

In 1993: 27 *Security and crypto skills shortage was prevalent.*

Today, the security field has grown both wide and deep, and there is a lot of skilled people doing all sorts of security-related work. Of course, the need for security skills have also grown. In fact, there is an increasing deficit in the number of ICT security professionals. Forbes magazine reported an expected deficit of 3.5 millon Cybersecurity professionals by 2021, and called the situation "An Industry Crisis" [25].

In 2018: 27 *The security and crypto skills shortage appears to worsen.*

Given that the skills shortage is so widespread and pervasive, it probably means that it cannot be solved by any quick fixes. Thus, there needs to be a long-term approach to solve the problem. It is probably also going to take many different approaches to solve this problem in a satisfactorily way. Many disciplines will have to be involved, and many security professionals will also need to have solid domain specific competence.

Lesson 30 *Security specialization studies needs increased capacity.*

This applies to all higher-education levels (BSc, MSc and PhD), but also to industry related professional courses and certificate programmes.

Lesson 31 *Security must be taught as part of all ICT educations.*

Security awareness and applied security (as a skill) must included. The non-ICT professions will also need to have security awareness and learn how to approach risk. This may be from a purely user perceptive, but also related to the specific professions.

Lesson 32 *Security and risk must be taught as part of all higher educations.*

Anderson also notes that there appears to have been an ad-hoc attitude to setting up the systems. Several of the cryptographic products did have various control mechanisms in place. For instance, Anderson mentions one product that had parity checking of the keys. But, he also notes that warnings were mostly ignored.

In 1993: 28 *Crypto-system warnings were silently ignored. That meant that one were left ignorant of a class of threats related to the cryptosystems.*

Now, it is no crime to ignore warnings, but then this should only be done explicitly. It is akin to risk management. One may decide to accept a risk, but one should never be ignorant about a risk.

In 2018: 28 *Warnings and errors are still silenced. This also happens for cryptosystems.*

It is difficult to assess the to which degree this takes place today and likewise the severity of the problem. But, ignorance is never acceptable. Again, it is tempting to refer to the Zen of Python [23]:

Errors should never pass silently.
Unless explicitly silenced.

Anderson was quite dissatisfied with the TCSEC and ITSEC standards as they appeared, or at least with how they were being used. However, he was solidly in favour of a more thorough approach to security and obviously accepts that there must be formal training. He also seem to advocate an engineering approach, which is to say an approach akin to that of a mature engineering profession. Anderson later wrote the book “Security Engineering” (1998, [26]), which may be seen as an effort in that direction.

2.4 Wider Implications—Why the Threat Model Was Wrong

Section 4 in [1] is entitled “The Wider Implications” and is mostly about the threat model. That is, it is about having a wrong threat model. In some sense, that is not a root cause. One may argue that behind a wrong model, one tend to find wrong assumptions. This is partly a psychological issue, and closely associated with the various biases we hold. In particular, assumptions are often formed around confirmation biases. We all too often only see what we are predisposed to see, and this goes for threat models too. In real life there will be evidence both for an against any particular threat model.

Being humans, we are likely to suffer from confirmation bias, and so it is to be expected that we dismiss or underestimate evidence against our model. At the same time, we probably will have too much faith in evidence that supports our model. To harbour the wrong assumptions will very easily lead us astray.

Anderson makes the claim that the banking sector had the wrong threat model. That claim is well documented, and it seems clear that the banking sector did not really analyse their systems and business model, but rather inherited a threat model. The perceived threats were mostly high-tech threats, while the real-world threats were mostly low-tech threats. To quote Anderson (section 4.1 in [1]):

> Designers concentrated on what could possibly happen rather than on what was likely to happen, and assumed that criminals would have the expertise, and use the techniques, of a government signals agency.

In short, the assumptions concerning the threats were wrong. Anderson lists two conception errors that may be have been at play:

1. Uncritical acceptance of conventional miliary wisdom of the 1970s.
   Cryptosystems were predominantly developed for miliary needs, so this misconception was likely inherent with the vendors.
2. Not properly accounting for human factors.
   Human factors also encompass organizational issues. Prominently, the fact that most banking organizations did not have a computer security team, and did not have a computer security culture.

In 1993: 29 *Wrong assumptions was a root cause behind the "Wrong Threat Model" problem.*

To hold an assumptions and to be susceptible to biases are fundamental human traits. Our only real antidote is to be explicitly aware of this and to be conscious about the psychology of decision making. The "assumption problem" is common enough to have been addressed in the Zen of Python too [23]:

> Explicit is better than implicit.
>
> ...
>
> In the face of ambiguity, refuse the temptation to guess.

With respect to cryptosystem design, the need for being explicit was succinctly captured in the paper "Prudent Engineering Practice for Cryptographic Protocols" by Abadi and Needham [11]. Both principles 1 and 2 is about being explicit. We have reason to assume (pun intended) that:

In 2018: 29 *Wrong assumptions is a major problem.*

Lesson 33 *Assumptions considered harmful. Explicitness is a virtue.*

We acknowledge the value of threat modeling, and shall discuss threat modeling further in Sect. 3.

Lesson 34 *Threat modeling should be done frequently.*

Anderson also noted that "However, telling good security from bad is notoriously difficult, ...". This point is well taken, and it is still very difficult to tell good security from bad or ineffective security.

In 1993: 30 *It was difficult to tell good security from bad security.*

Given that ineffective or bad security may appear to be good security, then it is possible to get away with bad security. It can even be quite profitable. In the security industry, this has become known as "snake oil" cryptography/security [2, 3]. It has also led to solutions, in which the very appearance of good security is the end-goal. That is, people, corporations and governments may deliberately deploy ineffective solutions, as long as those solutions appear to be effective. This is akin to what Richard Feynman called "Cargo Cult Science" [27], and what Bruce Schneier called "Security Theater" [28].

The end-user will have few means to verify the quality of the security, except from relying of brand association and trying to verify how reputable to system (or system operator) is. The situation is not all that much better for experts, but one can at least check recent history of incidents and perhaps even more important, the history of incident handling.

In 2018: 30 *It is still difficult to tell good security from bad security.*

Lesson 35 *One must take measures to ensure that one uses "good" security.*

This means that one must investigate products and verify assumptions. As such, this will be part of the security quality assurance process.

Corporate politics was mentioned as one factor that could have a negative effect on security. The problem was associated with wrong incentives, misplaced loyalty and fear of offending powerful colleagues.

In 1993: 31 *Corporate politics could have a negative effect on security.*

A case in point is the NotPetya/Maersk story [29] as reported in the Wired online magazine. The Maersk story provides ample evidence of how corporate politics and rewards (bonus) seriously hindered and prevented effective security. To quote the Wired article:

> The security revamp was green-lit and budgeted. But its success was never made a so-called key performance indicator for Maersks most senior IT overseers, so implementing it wouldnt contribute to their bonuses. They never carried the security makeover forward.

The Maersk story provides clear evidence that a bonus program actually stopped a well-planned and much need update. Maersk had a lot of old equipment, and the update would also mean replacing old Windows 2000 machines, etc. The extra cost would threaten the bonuses, and senior management thus halted the update effort.

In 2018: 31 *Corporate politics can have a negative effect on security.*

Lesson 36 *Company politics may affect security and may obscure the goals.*

2.5 A New Security Paradigm?

Section 5 in [1] is entitled "A New Security Paradigm?".

This section is mostly about trying to find appropriate threat models, and to propose a set of best practices. Anderson suggests a new model for security work, and it was roughly based on the one used in software engineering for safety critical systems. In safety engineering, one has realized that things go wrong and consequently one puts a lot of effort into dealing with this. All failure modes must be listed and accounted for. Then, these should be addressed if possible (prevention and mitigation). Recovery procedures must also be in place, and there must be assurance that the people involved would have the necessary competence. Finally, incidents and accidents must be monitored and described.

Today, the above is considered rather obvious and is captured in numerous guidelines, etc. But, it was not self-evident in 1993, and even today the above is not universally well known. That is to say, there are still many systems where one does not adhere to the above.

Anderson also mentions what he calls "The competing philosophies" problem. Essentially, the competing philosophies are about how to address safety, and the main stances are:

- The formal verification approach (*pro-active*)
- The constant monitoring and adjustments approach (*re-active*)

2.5.1 Formal Verification and Pro-active Security

Formal verification is often required in truly safety-critical systems. It is typically also applied to crypto protocols, etc. Similarly, crypto algorithms are subjected to rigours mathematical analysis. However, formal verification and mathematical analysis are no silver bullets for security. These may provide necessary tools and methods, but they are not sufficient. That is, one may prove certain aspects of a component or system, but one can in effect only prove a vanishingly small subset of all possible security attributes. Furthermore, formal methods and mathematical proofs work in a context. For formal verification, this context is the verification model. You can only prove aspects that are captured by the model, and even more important, what you then prove is an aspect of the model. The model may be too simplistic and the model may even be fundamentally wrong. The renowned security researcher Dieter Gollman declared that proofs is a non-goal for formal methods [30]. A mathematical analysis will also be given in a context, and consequently its applicability will be limited by the context. The famous mathematician and computer theorist, Donald Knuth, had this to say about proofs [31]:

> Beware of bugs in the above code;
> I have only proved it correct, not tried it.

Formal verification and mathematical proofs are also mostly concerned with the design of a system or component. This means that while the design may be sound, the implementation may still introduce vulnerabilities. We do not want to discourage the use of formal verification and mathematical proofs. Quite the opposite in fact, these are very important tools, but we must also recognize that no tool is suitable for all jobs. Only when we recognize the limitations of these tools, can we truly use them appropriately. Formal verification and mathematical analysis are best thought of as security design tools.

2.5.2 Reactive Security

The "constant monitoring and adjustments" is obviously a description of reactive handling of events. There is also reactive security, which normally is labelled "Detection and Response".

Since it is painfully clear that one cannot fully prevent security incidents by pro-active security, then it goes without saying that one must take measures for detecting and responding to incidents. Today, this is fully recognized and an integral part of best practices.

2.5.3 Competing Philosophies?

There really is no competing philosophies. Both approaches are needed.

Lesson 37 *Both pro-active and re-active security is needed.*

2.5.4 Security Policies

When companies and organizations need to interact, it is to be expected that these will have different perspectives and different security policies. Anderson points to this problem. The problem is deeply rooted in the value of assets, ownership and rights, and it entails all the problems of access control in general. It is a deep problem, and it involves economics and business models.

3 Threat Landscape and Threat Models

The threat landscape must be taken into account when designing secure systems. Threat models facilitates the process.

3.1 The Threat Landscape

The EU body ENISA annually publishes a report on the current threat landscape [32]. This report is well worth reading, and it does provide a fair amount of analysis. The security company Symantec annually publishes an "Internet Security Threat Report" [33]. The perspectives and approach is different from the ENISA reports, but this only adds to the value. Other efforts in this area also exits. All in all, there is no excuse for being ignorant of the real-world threat landscape.

3.2 A Mental Model for Threat Modeling

The standardization body ETSI has developed a method called the “Threat, Vulnerability and Risk Analysis (TVRA)” [34]. This is an integrated and comprehensive method. ETSI standardizes a host of different ICT systems and components, and the methodology is quite appropriate for this kind of work. It is noted in the standard that it has been tailored to apply to pre-production, and that it may be used for production too if one takes appropriate measures to adapt the methodology.

Microsoft was a driving force behind the STRIDE threat modeling methodology [35]. It is defined in the context of software development, but STRIDE is not exclusive to that context. The STRIDE approach is fairly simple and it does not pretend to be a complete solution. That is, the design goal for STRIDE was to be a practical solution, with real-world achievements.

It should be mentioned that Anderson’s problem with the “*what could possibly go wrong versus what is likely to go wrong*” cannot entirely be solved with approaches such as TVRA or STRIDE. But, both TVRA and STRIDE does include methodology to assess severity of the threats. Properly done, this should be sufficient to avoid wasting time on unrealistic and highly unlikely threats.

3.3 Scalability Issues

Some attacks, which are highly effective, does not scale too well. Case in point, the attack against GSM encryption mentioned earlier (Sect. 1.4), was highly effective. Since that attack was published, there have been many more attacks on the GSM cryptography. Some are based on so-called rainbow tables. These involve a once off brute force attack (very costly) to build the rainbow table, and then sophisticated memory-time trade-off at execution time. These rainbow-table attacks were technologically infeasible when GSM was designed, but then that was a time where the Intel 80386 processor where hot and a megabyte of memory was a lot. The rainbow-table attacks against GSM are by now fairly inexpensive and not very difficult to conduct [36, 37]. The newer GSM algorithm, A5/3 and its 3G cousin KASUMI, have also been successfully attacked [38].

Many of the attacks are very effective, and there can be no doubt whatsoever that the GSM cryptosystem is broken. And, yet, GSM is still in use, and there are literally billions of GSM compatible mobile phones out there. How can that be?

There are several factors at play here. One of them can be explained by the numbers involved. There will be many billion GSM call setups, etc. every day. The geographic spread covers more or less all inhabited parts of the world. An illicit intercept by the method used in [8], or some similar method, is effective, but not efficient in a mass surveillance context. Seen from the operator’s perspective, fraud and illicit activities are run-of-the-mill events. They routinely handle lost/stolen handsets, customers not paying, and other fraud attempts. While it is bothersome that GSM encryption is broken, it is not broken (to them) on a statistical scale. It is also likely that most customer are unaware of the problem. So, also long as the attacks don’t scale, GSM security is still somehow “good enough”.

Lesson 38 *Scalability issues affects both attacks and defenses.*

3.4 Attack Economy

Flaws, weaknesses and omissions does not always lead to failures. First, there needs to be exposure, but even with considerable exposure, there appears to be significantly fewer attacks that one would expect. This has been investigated in the paper “Where Do All the Attacks Go?” [39]. The authors found that for widespread undiscriminating attacks, it did not make sense to tailor the attacks. That is, the attacks were to a high degree dependent on default configurations. Even slight alterations to the default setup could foil some of these “mass market” attacks. There attacks were not very successful overall, but they were very cost efficient. Moreover, seen from there attackers point of view, the cost of adapting the attacks was not warranted.

Lesson 39 *Cost–benefit consideration affects both attacks and defenses.*

We note that scalability has a major impact on the attack economy, and that the cost model tends to favour simple attacks for mass distribution. The “simple” part is about how complex it is to orchestrate an attack, not about the exploit per se.

3.5 Human Factors: Perception and Threat Assessment

Humans design, implement, deploy and use the systems. Human factors are important for all these activities.

3.5.1 The Dunning–Kruger Effect

In the paper “Unskilled and Unaware of It: How Difficulties in Recognizing One’s Own Incompetence Lead to Inflated Self-Assessments” [40], Dunning and Kruger outlines how our perception of our own competencies may be seriously flawed. The main results were:

- *Unskilled students were unable to judge their own competency.*
  However, given some surface-skill knowledge, there was a tendency for these people to believe they understood the issues. Thus, they often (seriously) overestimated their competency level.
- *Average students fared better.*
  These students knew enough to know about their shortcomings. So, their assessment was more in accord with reality.
- *Highly skilled students underestimated they own competency.*
  These students expected a lot of themselves. They also knew that there was more to know, and they had a tendency to only compare themselves with other highly skilled students. Thus, in this context, they tended to underestimate their own competency level.

We may speculate that many end-users are in a similar situation to the “unskilled students”. They simply do not have sufficient knowledge to be able to accurately assess the threats.

Operative personnel may be in a similar situation to those of the “average students”. These people know a fair amount and they also tend to know about their own shortcomings. The assessment they have may therefore be reasonable accurate for normal operations.

People involved with systems design and specifically with the security of the systems, may be compared to the "highly skilled" group. They are highly competent, but they also know very well how much they don't know. Additionally, they will know of sophisticated threats and may consider low-tech threats to be mundane and uninteresting. This group may be susceptible to put too much emphasis on high-tech threats.

Lesson 40 *End-users cannot be relied upon concerning security decisions.*

Lesson 41 *Experts may sometimes overestimate threats/risks.*

3.5.2 Other Biases: A Preference for Exceptions Rather Than the Mundane

One effect we know about is the human tendency to lock onto threats that are exceptional. One well-known and well-researched threat is the fear of dying in a terrorist attack. In modern times, there has been a lot of research on the effects of terrorism on the human psyche. Terrorism, more than anything else, is a mind game. That is, terrorism, by its very definition, seeks to instill terror and fear in the targets. If an adversary can succeed in scaring an opposing part from taking measures, then he/she has won by default. There is a lot more to be said about the psychological effects of terror and terrorism, but suffice to say here that there usually is huge difference in the perceived and real threat.

To die in a traffic accident is not perceived as something extraordinary. In fact, for people not directly involved in the accident, it is mostly seen as just a run-of-the-mill event. Yet, in reality, a lot more people die or are seriously injured in traffic accidents than in terrorist attacks. So, how do we explain this discrepancy? According to psychologist Scott Plous [41]:

> In very general terms: (1) The more available an event is, the more frequent or probable it will seem; (2) the more vivid a piece of information is, the more easily recalled and convincing it will be; and (3) the more salient something is, the more likely it will be to appear causal.

The article "Drawing the wrong lessons from horrific events" [42] also highlight the "exceptional versus mundane" problem. The article was written in the context of the 2012 mass-shooting massacre in Aurora, Colorado, USA, but the exceptional versus mundane aspects are exactly the same as for terrorism.

Lesson 42 *Beware of the "exceptional versus mundane" problem.*

However, one must also take into account so-called fat tail events. These would be akin to Black Swan events [43]. If the exceptional turns out to be either more common than previously envisaged or extremely devastating, then one must prepare for the exceptional too.

Lesson 43 *Risk awareness of exceptional events is necessary.*

3.5.3 Security Cultures

Human factors, the psychology of fear and decision making, and how all this influences security in ICT systems, is a large subject. It is also outside scope for the present paper, although we note that one inevitably will need to asses human factors when designing/implementing/running successful systems.

Security culture aspects is also part of this wider vista of security, and here we refer to the previously mentioned report on Norwegian security culture [16]. It is not the final word on the topic, but at least it shows that these aspects *must* be taken into account when designing systems that interacts with end-users. Part of the problem is that there are many cultures and subcultures, and that many aspects of these will be in flux. The security cultures of British banking a la 1993, as exposed by Anderson, was a feature of the early 1990s. Britain's banking systems and the associated security cultures may have changed beyond recognition during the past 25 years. However, there is truth to the adage "the more things change the more they stay the same", and the true invariant is human nature. The context and circumstances may change, but the core aspects of human nature remains.

Lesson 44 *Security cultures can be part of the problem, but also the solution.*

Lesson 45 *Security cultures change, human nature remains.*

4 Discussion

This section provides a brief discussion of the findings and lessons learned.

4.1 The Main Findings of the Original Paper

The main findings of the original paper was that while cryptography could contain design flaws and implementation errors, cryptography and cryptographic systems where not the main culprit for most failures.

Anderson points to wrong threat model as the main problem. That is, one simply did not understand the security problem at hand. The paper also investigates how one could fail to understand the problem. There is no simple answer to this, but human factors plays a large role. Then there was misplaced trust assumptions, which is of course another aspects of human factors.

The paper concludes that there was a severe lack of competent people available. This was seen as an important explanation for the state of affairs.

4.2 Lessons Learned

It was not mentioned in the original paper, and it has not been mentioned here yet, but in the context of large and long lived systems, it should be clear that the security architecture must change with the system. This has a lot of repercussions for how one designs the security architecture.

Lesson 46 *Systems evolve. Everything may need to be replaced.*

The lessons themselves is listed in Annex. They cover many areas, and they are of varying levels of insight. The overall insights are:

1. Security is a weakest link game.
2. Security is a process.
3. Strong crypto and other technical solutions are absolutely essential.
4. Human factors and security cultures are essential too.
5. Sufficient security competence must be available.

The most important lesson of all, is probably the lesson about skills. In fact, the only way to ensure that one can address the other lessons, is to have a sufficient number of skilled people available.

To solve that problem would entail the development of security education on many different levels and on many different aspects of security. Academia, the universities and the colleges, will need to educate a lot more security professional on bachelor, master and PhD levels. These would have to span several fields, including Information and Communication Technologies (ICT) and Information Systems (IS). Other educations, like those within law school, business administration and economics, must have courses that teach security adapted to the context. To truly capture all human factors aspects, one must also educate the end users. This means that any field that uses ICT tools and systems, must be thought about how to be responsible users and be given a basic understanding of threats, risks and end-user related countermeasures.

Of course, one must begin teaching security earlier than that. Adapted for age, etc., kids must be given some information about how to behave when using smartphones, computer, etc.

Finally, learning about security is a process too. This means that on-the-job security training will be important. Not just to learn about specifics related to the job, but also to stay current in general.

4.3 Counterpoint Lessons

We have seen that flaws, weaknesses and omissions easily can lead to security failures. We have focused on failures in this paper, and tried to find root causes for the failures. However, there are a few counterpoint lessons here too.

4.3.1 Scalability

Scalability was briefly addressed in Sect. 3.3. There we saw that the GSM access security solutions were broken, but yet somewhat effective on an aggregate level. This is somewhat akin of how we secure our houses. Must houses are secured quite inadequately. The doors are not too strong, the locks can be picked and the windows can be broken. And, burglary is a fact of life. Yet, house burglary is a type of attack where the cost of attacking increases linearly with the number of houses. Our houses does not need perfect security to thwart burglary. On average, it just needs to sufficiently good to not being the softest target. This leads us to the following scalability factors:

- *Time* Attack that takes significant time will not scale well.
- *Space* Geographically bounded attacks will not scale well.
- *Uniquness* Attack that needs to be tailored will not scale well.

4.3.2 Attack Economy

As we saw in Sect. 3.4, exposure and flaws does not automatically lead to attacks. That is, for most untargeted attacks we have a “least common multiple” type of scenario. Even minor configuration differences may effectively mitigate or prevent an attack. The set of targets that satisfies all attacks criteria may be small compared to the total number of potential targets. However, the potential target population may be in the millions, and the relative cost of attacking concrete targets may be very low. Even with modest average returns, the attack may then be quite cost effective. Counterintuitively, even very soft and vulnerable targets, may be able to remain safe. Of course, for dedicated APT-style attacks, there targets would succumb almost immediately.

4.4 Antidote to System Failures

There exists a lot of paper on how to do security, and there is a long list of guidelines. The following is some of the more important high-level concepts and recommendations related to security.

4.4.1 Security-by-Default

There is a need for products to be secure as-is for end-customers. That is, products and services should always be in a secure configuration with respect to end-users. This concept is known as Security-by-Default, and is a recognized design principle. The UK National Cyber Security Centre (NCSC), defines the concept this way https://www.ncsc.gov.uk/articles/secure-default):

1. Security should be built into products from the beginning, it can’t be added in later;
2. Security should be added to treat the root cause of a problem, not its symptoms;
3. Security is never a goal in and of itself, it is a process and it must continue throughout the lifetime of the product;
4. Security should never compromise usability products need to be secure enough, then maximise usability;
5. Security should not require extensive configuration to work, and should just work reliably where implemented;
6. Security should constantly evolve to meet and defeat the latest threats new security features should take longer to defeat than they take to build;
7. Security through obscurity should be avoided;
8. security should not require specific technical understanding or non-obvious behaviour from the user.

4.4.2 Privacy-by-Design

Security-by-Default is often mentioned together with the Privacy-by-Design concept. Privacy-by-Design (PbD) as a concept is not directly related to security or cryptosystems, but it depends critically on strong security [44]. It is an important aspects of modern system design, and it has been included as a guiding principle for the EU directive

General Data Protection Regulation (GPDR) [45]. The foundational principles of Privacy-by-Design are:

1. Proactive not reactive;
2. Preventative not remedial;
3. Privacy as the default setting;
4. Privacy embedded into design;
5. Full functionality positive-sum, not zero-sum;
6. End-to-end security full lifecycle protection;
7. Visibility and transparency keep it open;
8. Respect for user privacy keep it user-centric.

Proper use of cryptosystems is necessary for realizing the PbD vision, and PbD is almost universally applicable to system designs.

4.4.3 The 4 and 10 Effective Countermeasures

The Norwegian National Security Authority (NSM) is responsible for security assurance on a national level in Norway, and they also run the Norwegian Computer Emergency Response Team (NorCERT) operation. In 2016 they published two short advisories (In Norwegian):

- An advisory with 4 essential action that should be taken [46]
- An advisory with 10 important action that should be taken [47]

The interesting part is that they claimed that one would be able to fend off 90% of all attacks by adhering to the 4 essential actions listed in the advisories. Given that they have extensive experience with live attacks and their modi operandi, it seems a credible claim. Of course, the claim is not valid for targets that are subject to APT type of attacks. So, then, what are those 4 countermeasures? In fact, the countermeasures are extremely trivial, which only serves to illustrate how remarkable it is that these actions are so effective.

1. Upgrade/update your software/firmware.
   Newer versions will generally be more secure.
2. Install security updates as fast as possible.
   Devastating malware like Wannacry, Petya and NotPetya was preventable by timely security updates.
3. Do not let end-user have administrative access (root privileges).
   This is the *least privilege* principle. Statistically, there will be compromised end-users, and this limits the damage/impact of a compromise.
4. Block unauthorized programs and apps (whitelisting).
   NSM recommend blocking programs contained on removable media (USB, CD-ROM, etc.) and they recommend blocking executable code received by email or other similar channels.

The advisory with 10 countermeasures is an extended version of the 4 countermeasures advisory. It contains additional system hardening measures.

5 Conclusions

Ross Anderson's 1993 paper "Why Cryptosystems Fail" was a very important paper, and it contributed to making it clear that one could not solve security entirely by means of technical solutions. Cryptography was, and is, fundamental to strong security, but strong security is a system property, and it does not follow from cryptography.

A systems perspective means capturing all relevant aspects of a system. One must acknowledge that security is a weakest link game, and that an attacker may chose what appears to him/her to be the weakest link. One must also take into account that security is a process, and what was good security yesterday may not be today. Strong crypto and other technical solutions are absolutely essential, and one cannot have security or privacy unless this is in place. Human factors and security cultures are essential too, and they can contribute to break systems that would otherwise be secure. We saw clearly that there were wrong assumptions behind many of the security and cryptosystems flaws, which means that these flaws ultimately must be attributed to human factors. Trust is another aspects that has a fundamental impact on security, and since trust by and large is influenced by human perceptions and social values, it is too an aspects that is crucially dependent on human factors.

Then we have the crux of all of our lessons: "Sufficient security competence must be available". This finding was there in the original paper and it certainly is here with use now. We therefore conclude this paper with our primary finding. There is a need for a massive effort in increasing awareness and in stepping up education concerning security. A comprehensive approach is needed, and both depth and breadth must be fully embraced. We have entered the cyber age, and it is time to address this. We therefore conclude the paper with a grand lesson:

The Security Skills Shortage Must Be Solved.

Annex: The list of Lessons

1. Understanding why cryptosystems fail is important.
2. Security costs should be internalized (if at all possible).
3. Security controls to provide accountability are important.
4. Systems must have Detection and Response capabilities.
5. The principle of least privilege should be enforced.
6. Outsourcing must be done in a responsible way.
7. Security responsibility cannot be outsourced.
8. Minimizing the attack surface is vitally important.
9. Strong quality assurance is a prerequisite for security.
10. Misguided advise can be harmful (do not trust all advise).
11. Passwords and PIN codes alone cannot provide strong security.
12. Requirements for unpredictability must be explicit.
13. Pseudo-random number generators must be verified.
14. Timely handling of security updates is absolutely essential.
15. Old equipment that cannot be updated is a serious security risk.
16. Crypto products must not be blindly trusted.
17. Hardware support is needed for strong security.

18. Trust must be warranted and continually evaluated.
19. Cryptographic backdoors must be considered harmful.
20. Physical exposure and inadequate protection is a problem.
21. Systems need a complete and comprehensive security architecture.
22. Systems need complete and comprehensive security policies.
23. Security is a process. All parts have a best-before date.
24. Human factors must be taken into account.
25. Special care must be taken when handling security credentials.
26. Cryptosystems must be designed to allow algorithm changes.
27. Cryptosystems must be designed to allow extending the key lengths.
28. Every effort should be made to keep complexity to a minimum.
29. Defense-in-depth is a recommended strategy.
30. Security specialization studies needs increased capacity.
31. Security must be taught as part of all ICT educations.
32. Security and risk must be taught as part of all higher educations.
33. Assumptions considered harmful. Explicitness is a virtue.
34. Threat modeling should be done frequently.
35. One must take measures to ensure that one uses “good” security.
36. Company politics may affect security and may obscure the goals.
37. Both pro-active and re-active security is needed.
38. Scalability issues affects both attacks and defenses.
39. Cost–benefit consideration affects both attacks and defenses.
40. End-users cannot be relied upon concerning security decisions.
41. Experts may sometimes overestimate threats/risks.
42. Beware of the “exceptional versus mundane” problem.
43. Risk awareness of exceptional events is necessary.
44. Security cultures can be part of the problem, but also the solution.
45. Security cultures change, human nature remains.
46. Systems evolve. Everything may need to be replaced.

References

1. Anderson, R. (1993). Why cryptosystems fail. In *Proceedings of the 1st ACM conference on computer and communications security* (pp. 215–227). ACM.
2. Zimmermann, P. (1998). *An introduction to cryptography. Documentation for pretty good privacy*. Network Associates: Santa Clara.
3. Schneier, B. Memo to the amateur cipher designer. Crypto-Gram Newsletter.
4. Heilman, E., Narula, N., Dryja, T., & Virza, M. Iota vulnerability report: Cryptanalysis of the curl hash function enabling practical signature forgery attacks on the iota cryptocurrency. Technical report, MIT Media Lab.
5. Schneier, B. (2016). Cryptography is harder than it looks. *IEEE Security & Privacy*, *14*(1), 87–88.
6. Walker, J., et al. (2000). Unsafe at any key size; An analysis of the wep encapsulation. *IEEE Document*, *802*(00), 362.
7. Carvalho, M., DeMott, J., Ford, R., & Wheeler, D. A. (2014). Heartbleed 101. *IEEE Security & Privacy*, *12*(4), 63–67.
8. Barkan, E., Biham, E., & Keller, N. (2003). Instant ciphertext-only cryptanalysis of gsm encrypted communication. In *Annual international cryptology conference* (pp. 600–616). Springer.

9. ICAO. Convention on International Civil Aviation. (2006). *Convention* (9th ed., Vol. 7300/9). Montreal: ICAO.
10. Taleb, N. N. (2018). *Skin in the game: Hidden asymmetries in daily life*. New York: Random House.
11. Abadi, M., & Needham, R. (1996). Prudent engineering practice for cryptographic protocols. *IEEE Transactions on Software Engineering*, *1*, 6–15.
12. Checkoway, S., Fredrikson, M., Niederhagen, R. F., Everspaugh, A., Green, M., Lange, T., Ristenpart, T., Bernstein, D. J., & Shacham, H., et al. (2014). On the practical exploitability of dual ec in tls implementations. In *Conference; 23rd USENIX security symposium; 2014-08-20; 2014-08-22*. Usenix Association.
13. Checkoway, S., Maskiewicz, J., Garman, C., Fried, J., Cohney, S., Green, M., et al. (2018). Where did i leave my keys? Lessons from the juniper dual ec incident. *Communications of the ACM*, *61*(11), 148–155.
14. Higginbotham, S. (2018). 6 ways IoT is vulnerable. *IEEE Spectrum*, *55*(7), 21.
15. Thompson, K. (1984). Reflections on trusting trust. *Communications of the ACM*, *27*(8), 761–763.
16. Malmedal, B. & Røislien, H. E. (2016). *The Norwegian cybersecurity culture*. NorSIS: Report.
17. Seacord, R. C. (2008). *The CERT C secure coding standard*. London: Pearson Education.
18. Shor, P. W. (1994). Algorithms for quantum computation: Discrete logarithms and factoring. In *35th Annual symposium on foundations of computer science, 1994 proceedings* (pp. 124–134). IEEE.
19. ETSI. (2011). Implementation security of quantum cryptography; Introduction, challenges, solutions. ETSI White Paper 27, ETSI, Sophia Antipolis, France.
20. ETSI. (2015). Quantum safe cryptography and security: An introduction, benefits, enablers and challenges. ETSI White Paper 8, ETSI, Sophia Antipolis, France.
21. Smart, N. P. (ed). (2014). Algorithms, key sizes and parameters report 2014. Technical report, ENISA.
22. Microsoft Corporation. Deprecation of SHA-1 for SSL/TLS certificates in microsoft edge and internet explorer 11. Technical report, Microsoft.
23. Peters, T. PEP 20—The Zen of Python (2004-08).
24. Industrial Control Systems Cyber Emergency Response Team (ICS-CERT). Recommended practice: Improving industrial control system cybersecurity with defense-in-depth strategies. Department of Homeland Security.
25. NeSmith, B.. The cybersecurity talent gap is an industry crisis. Forbes (online), 2018-08-09.
26. Anderson, R. (2008). *Security engineering*. Hoboken: Wiley.
27. Feynman, R. P. (1985). Cargo cult science. In W. W. Norton (Ed.), *In surely you're joking, Mr. Feynman* (1st ed.)., Originally a 1974 Caltech commencement address London: Vintage.
28. Schneier, B. (2003). *Beyond fear: Thinking sensibly about security in an uncertain world*. New York: Copernicus Book.
29. Greenberg, A. The untold story of NotPetya, the most devastating cyberattack in history. Wired (online), 22.08.2018.
30. Gollmann, D. (2003). Analysing security protocols. In *Formal aspects of security: First international conference, FASec 2002, London, UK, December 16–18, 2002, Revised papers* (vol. 1, p. 71). Springer.
31. Knuth, D. (1977). Notes on the van Emde Boas construction of priority deques: An instructive use of recursion. Memo/Letter.
32. ENISA. ENISA threat landscape report 2017. Technical report, ENISA.
33. Symantec Corporation. Internet security threat report. (2018). *Report*. Mountain View: Symantec Corporation.
34. ETSI Technical Committee Cyber Security. CYBER; methods and protocols; part 1: Method and pro forma for threat, vulnerability, risk analysis (TVRA). Technical Specification 102 165-1 V5.2.3, ETSI (2017).
35. Adam, S. (2014). *Threat modeling: Designing for security* (1st ed.). Hoboken: Wiley.
36. Kalenderi, M., Pnevmatikatos, D., Papaefstathiou, I., & Manifavas, C. (2012). Breaking the gsm a5/1 cryptography algorithm with rainbow tables and high-end fpgas. In *22nd International conference on field programmable logic and applications (FPL), 2012* (pp. 747–753). IEEE.
37. Nohl, K. (2010). Attacking phone privacy. *Black Hat USA*, pp. 1–6.
38. Dunkelman, O., Keller, N., & Shamir, A. (2010). A practical-time related-key attack on the kasumi cryptosystem used in gsm and 3g telephony. In *Annual cryptology conference* (pp. 393–410). Springer.
39. Florêncio, D., & Herley, C. (2013). Where do all the attacks go? In *Economics of information security and privacy III* (pp. 13–33). Springer.

40. Kruger, J., & Dunning, D. (1999). Unskilled and Unaware of it: How difficulties in recognizing one's own incompetence lead to inflated self-assessments. *Journal of Personality and Social Psychology*, *77*(6), 1121.
41. Plous, S. (1993). *The psychology of judgment and decision making*., McGraw-Hill series in social psychology New York: Mcgraw-Hill Book Company.
42. Schneier, B. Drawing the wrong lessons from horrific events. CNN.com.
43. Taleb, N. N. (2007). *The black swan: The impact of the highly improbable*. New York: Random House Publishing Group.
44. Cavoukian, A. (2009). *Privacy by design: The 7 foundational principles*. Ontario: Information and Privacy Commissioner of Ontario.
45. EU. (2016). Regulation (EU) 2016/679 (General data protection regulation). Regulations 679, EU, 04.
46. NSM. (2016). S-01 Fire effektive tiltak mot dataangrep.
47. NSM. (2016). S-02 Ti viktige tiltak mot dataangrep.

Geir M. Køien received his Ph.D. from Aalborg University in 2008. Before that he had worked for many years in industry, including LM Ericsson Norway and Telenor R&D. During these years he worked extensively with mobile systems and with security and privacy. He has also worked with the Norwegian Defence Research Establishment (FFI) and with Norwegian Communications Authority (NKom) on various security and communications related projects. Currently, he is a professor with the University of Agder, Norway.