Faculty of Informatics
Masaryk University

Examining and exploiting randomness for cryptography

Habilitation thesis
(Collection of articles)

Petr Švenda

October 2018
Brno, Czech Republic

Abstract

Random data are widely used in cryptography, whether to generate cryptographic keys, provide unpredictable message paddings or to create unique authentication challenges. Frequently, a random bit sequence is not used directly, but serves as a partial input of some algorithm, e.g., to generate the large prime numbers or initialize an internal state of a pseudorandom generator. However, a design or implementation error of a random number generator may result in a significant advantages for an attacker including the predictable keys allowing to decrypt or forge messages. Therefore, the output of generators shall be thoroughly and (preferably) continuously tested. However, since the expected output is an arbitrary random value by design, the testing itself is not straightforward.

One possible approach is to systematically search for a dependence between individual bits of output data, which are rare to find in truly random data. If found, the tested sequence is rejected. If happen for the pseudo-random generators, their design is additionally scrutinized as the outputs of modern cryptographic primitives like ciphers, or hash algorithms shall be statistically indistinguishable from the truly random data.

The aim of this thesis is to acquaint the reader with our research results in the field of statistical randomness testing and its use for analysis of existing implementations of cryptographic algorithms. The examples of real-world impact are given, including a significant key generation vulnerability discovered in cryptographic smartcards and utilization of small, but widespread key generation bias to directly measure the popularity of cryptographic libraries – previously estimated only indirectly. At the same time, the thesis deals with the issue of cryptographic keys generation in a partially compromised environment with a partially exposed randomness state – arguably an increasingly common situation nowadays, due to the increasingly complicated interconnection of software and hardware components.

The thesis is presented in the form of a collection of selected scientific publications supplemented by comments to provide the necessary context and connection with the existing literature. The work is divided into two basic parts. The first part deals with the automated testing of randomness and the consequences of detected weaknesses. The second part describes the cryptographic protocols designed to prevent the loss of randomness and to make its exploitation more difficult for an attacker.

Keywords: key generation, randomness, RSA, secrecy amplification, secure multi-party computation, statistical randomness testing.

Abstrakt

Náhodná data jsou v kryptografii široce používána, ať už pro generování kryptografických klíčů, nepredikovatelného doplnění zpráv nebo vytvoření unikátních autentizačních výzev. Náhodná sekvence bitů se v některých případech nepoužije přímo, ale je pouze dílčím vstupem algoritmu, např. pro generování velkých prvočísel nebo inicializaci počátečního stavu pseudonáhodného generátoru. V implementacích generátorů náhodných čísel se ale může objevit chyba, která následně vyústí ve výhodu pro útočníka, například v použití predikovatelného klíče, kterým útočník může dešifrovat zprávy, které mu nejsou určeny. Proto je nutné generátory důkladně a nejlépe i průběžně testovat. Protože však může náhodné číslo nabývat v principu libovolné hodnoty, testování není zcela přímočaré. Cílem je najít takovou systematickou závislost mezi jednotlivými bity dat, která se v těch opravdu náhodných vyskytne jen velmi zřídka. A nejlépe s co nejmenším množstvím dat tak, aby vlastní testování proběhlo co nejrychleji. Pokud se taková závislost v datech objeví, lze je odmítnout, případně dále analyzovat procesy (v případě pseudonáhodných generátorů), které za přítomností nežádoucí detekovatelné závislosti stojí. Výstupy moderních šifer a hašovacích algoritmů by měly být od opravdu náhodných dat statisticky neodlišitelné.

Cílem práce je seznámit čtenáře s výzkumnými výsledky v oblasti statistického testování náhodnosti a jeho využití pro analýzu existujících implementací kryptografických algoritmů. Součástí jsou i ukázky nalezené zranitelnosti s velkým celosvětovým dopadem nebo naopak umožnění přímého měření fenoménu, který byl dosud měřen jen nepřímo. Zároveň se práce věnuje i problematice tvorby kryptografických klíčů v částečně kompromitovaném prostředí – dnes již běžnou situací, kterou lze vzhledem k čím dál komplikovanějšímu propojení softwarových a hardwarových komponent očekávat i nadále.

Práce je prezentována formou kolekce vybraných vědeckých publikací doplněných komentáři, které poskytují výsledkům potřebný kontext a provázání s existující literaturou. Práce je rozdělena na dvě základní části. První se věnuje automatickému testování náhodnosti a důsledkům detekované nenáhodnosti. Ve druhé části jsou popsány kryptografické protokoly, které umožňují ztrátě obsažené náhodnosti (entropie) zabránit, případně její vynucenou ztrátu učinit pro útočníka obtížnější.

Klíčová slova: bezpečné protokoly pro více účastníků, generování klíčů, náhodnost, RSA, statistické testování náhodnosti.

Acknowledgements

I want to express the greatest appreciation to my wife, Linda. I am still struggling to understand why she is still willing to support me in the work I love, despite being too little with her and our kids.

I also owe a lot to my long-time tutor Vashek Matyas which I will at least partially repay by caring about the next generation of lab colleagues. While Vashek showed me the world of security research, the curiosity itself was already ignited many years before thanks to my parents, who grown me in the spirit of exploring and sharing.

I would also like to thank all my colleagues and students – frequently also friends – for all the inspiration, help, and shared joy of teaching and research. To Martin Ukrop (for shared and realized visions for our lab), to Vlasta Šťávová (for a courage to make her way while staying gentle), to Zdenek Říha and Roman Lacko (for all the teaching support), to Dušan Klinec (for shared vibes), to Marek Sýs (for teaching me a patience), to Matúš Němec (for the sense of humour during the late hours just before the deadline), to Milan Brož (for significant impact despite self-pronounced insignificancy), to Vláďa Sedláček (for his honesty and ideals), to Jan Krhovják, Marek Kumpošt, Jirka Kůr and Andriy Stetsko (for our steps in Ph.D. journey), to Dan Cvrček, Vašek Lorenc and Víťa Bukač (for vital insights from the industry side), to Lukáš Němec (for all DIY perks), to Lydia Kraus and Lenka Knapová (for showing me the power of methodology), to Martin Stehlík (for shared thoughts during the long runs), to Radim Ošťádal (for unbelievable persistency). There are many more of you to whom I am grateful - I would not be able to make much progress without you all. Thank you.

Petr Švenda

Contents

Part I

Commentary

Chapter 1

Introduction

Randomness plays a crucial role in the cryptographic systems (Kerckhoffs' principle [96]) as the random bits are required to generate secret keys for cryptographic algorithms. While generation of secret keys typically requires truly random bits, output of many cryptographic functions is also required to be indistinguishable from a truly random data, despite being generated deterministically. The failure to meet the indistinguishability requirement translates into the leakage of some information about the sensitive input plaintext, used key or both, e.g., recovered via linear, differential and other cryptanalytic techniques.

Therefore, the vital step is to detect unwanted bias in a data, e.g., in the form of a correlation between selected bits. While detecting a completely malfunctioning random generator producing just binary zeroes is trivial, the detection of correlated bits with a complex relation is more difficult – yet when discovered by an attacker, it may lead to compromise of the whole encryption key. The statistical randomness tests aim to detect even small biases given only a decent amount of data for testing. Although already widely used, there is a large room for improvement – both in the testing speed as well as detection sensitivity.

Once a bias in data is detected, a generator can be further analyzed to establish the root cause. Such an analysis then serves as a guidance for a cryptographic function designer during the design phase or can be used to assess the properties of otherwise black-box implementations like keypair generation in smartcards. As many real-world examples [80, 114, 86, 161, 2] demonstrate, flaws in design and implementation are common. In past, a small bias used to be frequently neglected [74, 152, 34, 178], just to be later found to have deeper reasons as e.g., in the

case of the RC4 algorithm [126, 145], sometimes with devastating security implications as for backdoored Dual-EC-DRBG/NIST SP 800-90 [19, 44] or flawed Infineon RSA keypair generation [132].

An opposite view can start with the assumption that an attacker already learned some bits of random keys used in a system, e.g., by a compromise of some of the interacting users. The question is whether we can utilize additional cryptographic protocols to decrease or eliminate an attacker's advantage. Research shows that secure multiparty protocols are a promising candidate to protect the system against both intentional compromises as well as unintentional design or implementation flaws.

1.1 Focus of the thesis

This thesis presents my research contributions to the area of randomness generation, statistical testing and analysis of resulting vulnerabilities if the randomness indistinguishability property is not satisfied. Subsequently, key generation and usage protocols for scenarios assuming partial compromise of either randomness state or protocol participants are devised. The following research questions are addressed:

How to extract and test entropy from a physical source?
Various techniques were designed and practically deployed to extract unbiased bits from a physical phenomenon like decay of radioactive particles, amplification of thermal noise or complex visual patterns produced by lava lamps. Even if believed to contain some fraction of random bits, physical sources are typically producing a significantly biased and correlated output. Therefore, two principal components are required: 1) extractor of entropy that will create an unbiased sequence from (typically) longer, but biased one and 2) test(s) to check if the resulting sequence is free of any detectable biases.

How to detect a weakness in a cryptographic function or its implementation?
Atop of other requirements, the output of modern cryptographic functions is typically required to be indistinguishable from the truly random data, unless the used key (for ciphers) or input data (e.g., for unkeyed hash functions) is known. Even a slight bias is now considered a weakness as its presence translates to a leak of some information about the input plaintext, a used key or additional metadata like origin software library. The very similar procedure of statistical testing used for truly random number generators (TRNGs) can be applied to test output of these functions,

typically combined with specific input data generation strategy (e.g., input with a very low Hamming weight) and configuration of function parameters (e.g., number of internal rounds) to stress the function under the test to exhibit the detectable bias.

What are the practical implications of a detectable bias?
The detection of any bias in the output of a cryptographic function or random number generator is just the first step hinting about a potential vulnerability. Once detected, the bias can be used to reason about the necessary improvements to a mathematical construction of a cryptographic function (e.g., increasing the number of internal rounds to improve confusion and diffusion properties), the source of the data seen (e.g., the software library that generated a given key) or even to a practical exploitation of the vulnerability (e.g., reconstruction of a secret key).

How to generate and use random bits in a partially compromised environment?
Finally, the secure key generation in the presence of an active attacker is now a frequently desired property for the modern cryptographic systems. Such an attacker is capable of compromising not only parts of a random state but also to control the execution of subset of the protocol participants. The aim is to address an even stronger attacker model than the classical Dolev-Yao attacker model [62] as this model does not concern compromised protocol participants. This thesis deals with the topic of the collaborative generation of secret keys suitable for cryptographic smartcards or lightweight ad-hoc networks with significant limitations on memory (restricting the number of pre-distributed keys), computational power (preventing public key cryptography algorithms) and energy (restricting the number of exchanged messages).

My Ph.D. thesis was focused primarily on the lightweight key establishment in partially compromised environments. This habilitation thesis deals with two additional research areas: randomness statistical testing and secure use of cryptographic smartcards. While research during my Ph.D. study was conducted with a very limited number of co-authors from our laboratory, the work presented in this thesis is a result of wider research collaborations with co-authors from several different institutions.

1.2 Thesis structure

The thesis is structured with respect to contributions I made to the area of randomness generation, statistical testing, related vulnerability analysis, and randomness

used in protocols for a collaborative key establishment in partially compromised environments.

The first part of chapter 2 describes an adaptive statistical randomness testing approach we proposed and applied to a large number of cryptographic functions. The second part of that chapter explains the method developed for detection of bias in generated RSA keys and its utilization for a key to library attribution, dissection of particular vulnerability found and options for systematic prevention of flaws using multiparty cryptographic protocols.

Chapter 3 tackles the issue of key establishment in partially compromised environments with a focus on the devices with limited storage and computational power. Two main classes of protocols are discussed: secrecy amplification protocols aiming at the collaborative distribution of fresh non-compromised keys using multiple communication paths and structured probabilistic key pre-distribution to increase the likelyhood that a shared key can be established, yet to decrease the attacker advantage after a physical node capture followed by a compromise of stored keys.

The thesis is concluded in chapter 4 with possible future research directions. The collection of selected research papers is appended at the end of the thesis.

Chapter 2

Randomness analysis

Many documented flaws in design and implementation of random or pseudorandom number generators exist – e.g., in an early version of Netscape SSL [80], OpenSSL FIPS Object Module [114], Java 2 ME [86], or Sun's MIDP Reference Implementation of SSL [161]. They have a critical impact on the security of many common services. For example, a devastating flaw (resulting in the predictability of the randomness generator) in the Debian OpenSSL caused that SSH keys, SSL certificates, DNSSEC keys, OpenVPN keys, and DH/DSA session keys generated in the period of more than two years had to be considered compromised [2]. In some cases, intentional introduction of such a vulnerability into a standardized algorithm is suspected, as in case of the Dual-EC-DRBG/NIST SP 800-90 generator [159, 19, 44].

The physical random number generators usually do not provide uniform output unless post-processed by a suitable entropy extractor [176, 173, 94]. As the given extractor is based on the assumptions about the prior distribution of the unprocessed output, invalid assumptions may result in a predictable sequence, so the extractor output has to be tested as well.

Additionally, both newly designed as well as widely used cryptographic primitives (block cipher, stream cipher, hash function, pseudo-random generators, etc.) are subjected to various analytic techniques like linear, differential and algebraic cryptanalyse that look for flaws or information leakage in the primitive design [85, 15, 127]. The standard techniques try to find any significant correlations between the tested primitive input (plaintext), output (ciphertext) and key bits (if used). An existence of correlated bits indicates a weakness of the function, which might be exploited to

predict the bits of a secret key or next output bits of the pseudorandom generator [153]. Although these techniques can be partially automated, the aid of the skilled cryptanalyst is still needed.[1]

2.1 Adaptive randomness testing

Fully automated statistical test suites like NIST STS [154], Dieharder [36] and TestU01 [108]) are often used as a quick and easy to run tool before the deeper cryptanalysis is performed [160]. Each test suite (often called *battery*) typically consists of tens of empirical tests of randomness, examining the correlation of a function output bits. Each test looks for a predefined pattern of bits (or block of bits) in data, and thus it examines randomness property from its specific point of view. Each test computes a histogram of a specific feature of bits (or block of bits) and compares it with the one expected for the truly random data. The tested data are considered non-random if histograms differ significantly. Although there is an unlimited number of tests in principle, batteries opt for implementation of only several (carefully) selected ones for the practical reasons. The complexity of tests usually determines the amount of data necessary, with up to several GBs usually required. Also, in general, the more data available for analysis, the smaller bias can be spotted.

NIST STS [154], Dieharder [36] (an extended version of the Diehard [116]) and TestU01 [108] are the most commonly used. The NIST STS is the basic battery required by NIST to test RNGs of cryptographic devices by the FIPS 140-2 certification process [136] with four out of total 15 of NIST STS tests required as power-up tests executed on-device. The Dieharder battery is an extension of the original Diehard battery [116] with some (but not all) NIST STS tests also included and generally more sensitive than NIST STS. TestU01 can be viewed as the current state-of-the-art of randomness testing. TestU01 is a library that implements more than 100 different tests of randomness grouped into six sub-batteries called Small Crush, Crush, Big Crush, Rabbit, Alphabit, BlockAlphabit with increasing amount of data required for the analysis.

There are also other, less known and used batteries including Donald Knuth's tests [100], Crypt-X suite [39], PractRand [63], RaBiGeTe [148], CryptoStat [93], YAARX [20], ENT [180], SPRNG [117], gjrand [90] and BSI test suite [155].

[1] The description of related work in this chapter is adapted and extended with input from our publications, mainly [171, 178, 131, 132, 120].

There are two main generic limitations of standard batteries with respect to the analysis of pseudo-random number generators and cryptographic functions: 1) Insufficient sensitivity to detect bias present in unweakened functions with full number of rounds and other standard security parameters and/or with given amount of data. 2) The difficulty of test results interpretation. While the first limitation can be addressed by analysis of functions with a reduced number of rounds, provision of more data and design of more sensitive tests, the second limitation requires a used test to be able to identify concrete dependent bits and provide this crucial information to a cryptanalyst.

In addition to the generic randomness testing batteries, various approaches targeting specifically the pseudo-randomness property of cryptographic functions were proposed. A strong method focused on testing of hash functions and symmetric ciphers based on the representation of each output bit as a Boolean function in the algebraic normal form (ANF) and related statistical tests based on the number of its monomials was proposed in [73]. The strong *d-monomial test* adapted to perform chosen IV statistical attacks on stream ciphers is used in automated cryptanalytic tool [69]. Similarly, a greedy method was proposed to find distinguishers from randomness for stream and block ciphers based on the maximum degree monomial test [164]. The CryptoStat [93] tool is focused on testing block ciphers, and message authentication codes and consists of several tests, each computing the probability that a block of bits of the ciphertext equals to bits taken from a plaintext and a key. The TEA block cipher [181] is a particularly popular target for analysis of function biased outputs [95, 84, 76, 104].

Contributions

We focus on the automated dynamic construction of new statistical tests fine-tuned for the given data stream in contrast to the standard practice of pre-defined fixed static test. This represents a shift in the testing paradigm. Instead of relying on a set of carefully selected but fixed tests (as is the case of standard statistical batteries), every evaluation of the input data itself will produce new test(s) specific for the input data. The process (and its success or failure to find a distinguisher) serves as the actual test itself. If a working distinguisher is found, data are non-random. Additionally, the distinguisher found can be analyzed to determine why it works – which bits are used and in what dependency on others. We explored two principal options for the construction of tests (distinguishers):

1. Arbitrarily complex test expressed as software circuits (EACirc).

2. Simple test in the form of Boolean functions (BoolTest).

The area of statistical testing of outputs of cryptographic functions in our research group was initiated by me, followed by the steadily growing group of involved people. I devised the initial idea of adaptive randomness statistical tests, representation in the form of software circuit processing separate blocks of an input stream and did the initial implementation, utilization of genetic programming optimization heuristic method and analyzed several round-reduced cryptographic functions. Together with colleagues, we later significantly extended the set of analyzed functions, improved the sensitivity of tests found by variations of software circuit output and systematically examined various parameters including the influence of input data length, software circuit depth and compared extensively with NIST STS and Dieharder batteries. The initial publication [169] described the idea and provided results for seven stream ciphers, showing comparable results to STS NIST and was accompanied with a testing tool called *EACirc*. The significantly extended version of this paper [170] included an additional analysis of 18 SHA-3 candidates, more detailed comparison with NIST STS and Dieharder and basic methodology for interpretation of software circuits (distinguishers) found.

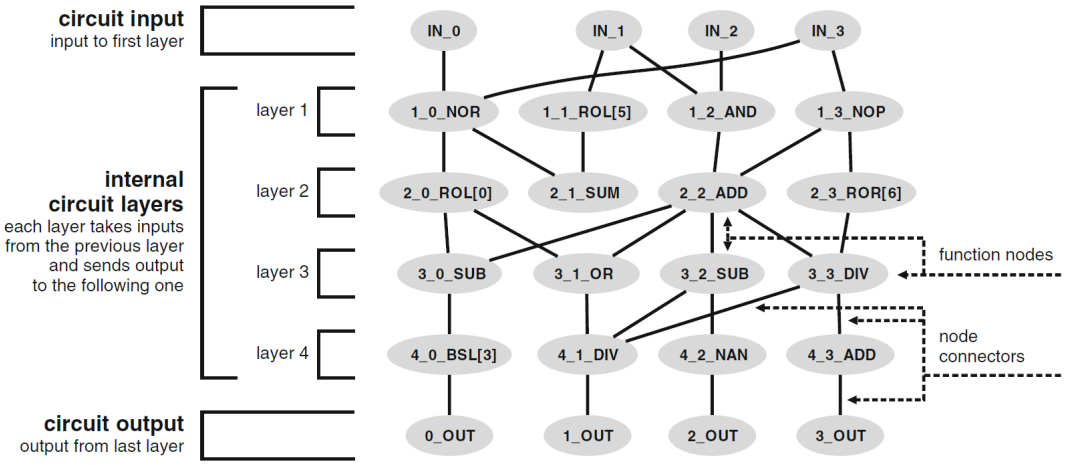

Figure 2.1: The simple example of a software circuit as used by EACirc randomness distinguisher. Both connectors between the nodes as well as specific functions executed inside the nodes are searched for using Cartesian Genetic Programming. Adapted from [170].

The subsequent work [163] significantly improved *EACirc* sensitivity to spot bias in the analyzed data. Instead of directly encoding the distinguisher decision about the input origin (a truly random stream or a tested function output), the circuit outputs only 1 or 0 for every input block from a tested stream. The resulting histogram of obtained ones and zeroes for the tested input data is then compared with reference histogram obtained for truly random data stream (obtained from a physical TRNG). If these two distributions differ enough, then the current circuit serves as distinguisher able to spot the bias in data. With the Dieharder battery being still slightly superior, the new approach was able to construct distinguisher in two round-reduced functions (Hermes, Fubuki) where neither the NIST STS nor Dieharder spotted any bias. Another advantage was in the amount of required data to spot a bias – usually, an order of magnitude less data was required. The method was also used to find more efficient distinguisher attack against 4-round TEA algorithm [104] as well as to investigate the resilience of avalanche effect for candidates to CAESAR competition against the improper initialization [174].

Our quest to design a more sensitive testing methodology requiring fewer input data continued with the further simplification of the distinguisher structure and resulted in a method called *BoolTest* [171]. Instead of a circuit-like structure with multiple layers and connectors in between, the candidate distinguishers are expressed as Boolean functions with the defined degree, represented in ANF and computed over the bits in the currently processed block. The resulting polynomial is evaluated over all blocks from the input data stream with the comparison of obtained distribution to the expected one (for truly random data) same as in the previous approach [163]. Thanks to the limitations placed on the searched Boolean functions, the resulting distinguisher directly identifies the function's biased output bits, providing a better feedback about the tested function.

In both cases (*EACirc* and *BoolTest*), dynamically constructed tests are found by a suitable heuristic optimization as the optimal solution cannot be found in a reasonable time (except for the trivial cases). We used the Cartesian genetic programming (CGP) [124] for EACirc software circuits and a brute-force of simpler sub-components combined with variations of the greedy algorithm for the *BoolTest*.

We also practically utilized the statistical testing batteries to detect problems with truly random number generators for cryptographic smartcards of two hardware vendors [178].

Our research in this area continues with the development of the largest (100+)

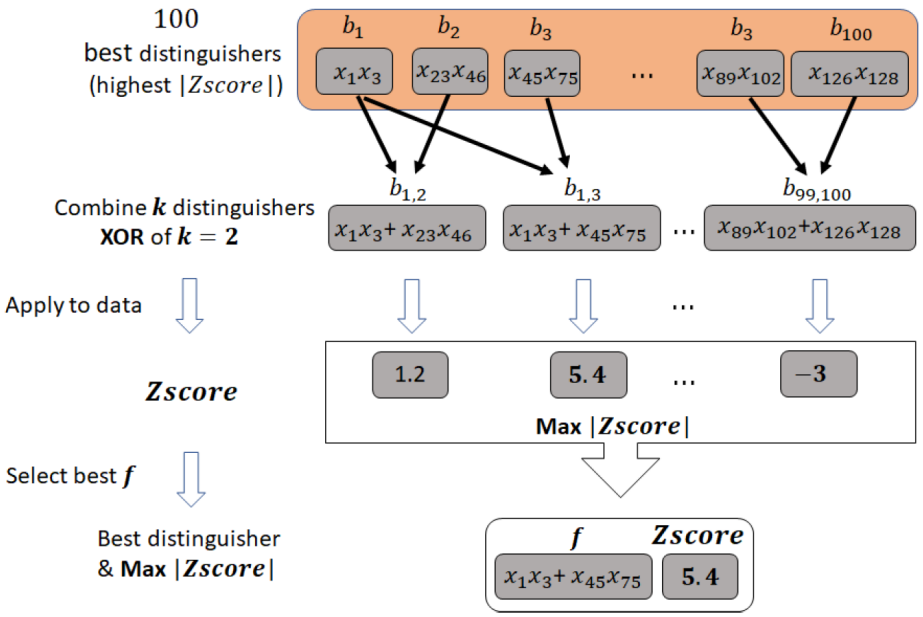

Figure 2.2: The example of randomness distinguisher based on Boolean functions as used by BoolTest. All possible monomials with a degree 2 are evaluated using tested data, and the 100 best performing are then selected for further combinations. Adapted from [171].

public suite of round-reduced cryptographic functions (block and stream ciphers, hash functions, lightweight ciphers, PRNGs) called *CryptoStreams* [5] and multi-batteries randomness testing portal with necessary results interpretation methodology. The project is a basis for the development of new cryptanalysis methods, which can be quickly tested on the exhaustive set of real-world cryptographic functions instead of one or few selected (as is the common practice nowadays).

Our earlier work focused on the design of a new entropy extractor based on Carter-Wegman universal families of hashing functions [30] with the aim to construct a well-performing extractor without the detectable bias. While technically slower than custom extractor described in [103], it allowed us to formally reason about the properties of extracted sequence (as long as the distribution of bits from an unprocessed CMOS sensor output is retained).

Articles in collection

[169] P. Svenda, M. Ukrop, and V. Matyas. Towards cryptographic function dis-

tinguishers with evolutionary circuits. In *10th International Conference on Security and Cryptography (SECRYPT'13)*, pages 135–146. SciTePress, 2013

I proposed the idea of dynamic construction of statistical tests, performed implementation, analyzed results and wrote significant part of the text. Contribution 50%.

[172] M. Sýs, P. Švenda, M. Ukrop, and V. Matyáš. Constructing empirical tests of randomness. In *Proceedings of the 11th International Conference on Security and Cryptography (SECRYPT'14)*, pages 1–9. ICETE, 2014

I cooperated on the design of the improved algorithm, performed part of experiments and contributed to text writing. Contribution 30%.

[171] M. Sys, D. Klinec, and P. Svenda. The efficient randomness testing using Boolean functions. In *Proceedings of the 14th International Conference on Security and Cryptography (SECRYPT'17)*, pages 92–103. SCITEPRESS, 2017

I cooperated on design and analysis of experiments and contributed to text writing. Contribution 20%.

Other relevant publications

[30] J. Bouda, J. Krhovjak, V. Matyas, and P. Svenda. Towards true random number generation in mobile environments. In *Nordic Conference on Secure IT Systems*, pages 179–189. Springer, 2009

I analyzed entropy extractor for camera sensor, minor contribution to text writing. Contribution 15%.

[168] P. Svenda and V. Matyas. On the origin of yet another channel. In *Proceedings of the 21st International Workshop on Security Protocols (IWSP'13)*, pages 223–237. Springer, 2013

I proposed the idea of automatic search for attack strategies, performed experiments for 2 out of 3 scenarios and wrote main part of the text. Contribution 67%.

[170] P. Svenda, M. Ukrop, and V. Matyas. Determining cryptographic distinguishers for eStream and SHA-3 candidate functions with evolutionary circuits. In *E-Business and Telecommunications*, volume 456, pages 290–305. Springer Berlin Heidelberg, 2014

I proposed the idea of dynamic construction of statistical tests, performed implementation, analyzed results and wrote significant part of the text. Contribution 40%.

[174] M. Ukrop and P. Svenda. Avalanche effect in improperly initialized CAESAR candidates. In *Electronic Proceedings in Theoretical Computer Science*, volume 233, pages 72–81. Open Publishing Association, 2016

I cooperated on the design of the experiments and analysis and contributed to text writing. Contribution 30%.

[104] K. Kubíček, J. Novotný, P. Svenda, and M. Ukrop. New results on reduced-round Tiny Encryption Algorithm using genetic programming. *IEEE Infocommunications*, vol. 8, 2016

I cooperated on the design of the experiments and analysis and contributed to text writing. Contribution 30%.

[171] M. Sys, D. Klinec, and P. Svenda. The efficient randomness testing using Boolean functions. In *Proceedings of the 14th International Conference on Security and Cryptography (SECRYPT'17)*, pages 92–103. SCITEPRESS, 2017

I cooperated on design and analysis of experiments and contributed to text writing. Contribution 20%.

2.2 Bias in the RSA keypair generation

An actual realization (software implementation) of a cryptographic algorithm or protocol is a well-known source of subtle differences when processing different input data, (so-called *side-channel leakage*), varying between the different implementations and possibly leading to the detection of a specific type or even version of hardware or software (so-called *fingerprinting*). The fingerprinting technique is almost universally applicable to any device or application including the web servers, email clients, databases, IoT networked devices or, e.g., electronic passports. The typical differences found are within the content of returned messages and error codes and response behavior dictated by the implementation's state model [9, 60]. The same implementation may also behave differently based on processed input data, forming time-based, power consumption-based and other types of side channels.

A large number of practical attacks in recent years [107, 37, 17, 24] testifies how difficult it is to make an implementation secure, robust and without any side-channel leakage. Even major libraries such as OpenSSL, Java JCE or Microsoft CryptoAPI were hit by multiple problems including extraction of RSA private keys [37], AES

secret keys [17] or decryption of sensitive message via padding oracle attacks [21, 24] remotely from a targeted web server and generation of vulnerable keys by a weak or a malfunctioning random generator [2, 83]. It is reasonable to expect that similar problems will occur in future for these and other cryptographic libraries as well.

The prediction of an impact for a future bug depends not only on the nature of the bug (unknown in advance) but also on the overall popularity of the affected cryptographic library within the targeted usage domain. A security bug in OpenSSL will probably cause more harm than a bug in an unknown or sparsely used library.

Yet, the estimation of the popularity of a given library is a complicated affair. As a library produces random keys, it is difficult to attribute a particular key to its originating library based only on the bits of the key. A common approach is to make indirect estimates based on additional information such as specific strings inserted into certificates, default libraries used by a software package which is identified by other means (e.g., the Apache HTTP server typically uses OpenSSL), specific key properties (uncommon key lengths or domain parameters) or a library popularity based on the positive ratings (stars or likes) to establish prior probabilities.

Server fingerprints were used to probabilistically determine the operating system, or even the versions of the deployed software [134, 137], estimating the number of servers running a Microsoft OS [133] or popularity of software packages handling the SSH connection [10]. A direct identification of software packages running on other cores in a cloud environment based on cache side-channels was demonstrated by [89, 88].

However, all these approaches leave a large uncertainty about the total number of keys produced by a specific library or the real origin of a given single key from specific domain – for example TLS keys used for secure HTTP connection.

Measurements and analyses of the TLS ecosystem have a long history with large-scale scans starting in 2010 with the EFF SSL Observatory project [4], followed by analyses of both valid certificates [47, 68, 67, 66, 72] (the majority of papers) as well as invalid ones [46], including also certificates from increasingly popular Certificate Transparency repositories [175] and client SSH authentication keys crawled from GitHub [54, 14]. The used cryptographic algorithms and key lengths were analyzed [65, 87], showing that more than 85% of currently valid TLS certificates use the RSA algorithm (and more than 99% for SMTP certificates [122]) – making the RSA a representative algorithm of the ecosystem.

Despite the RSA algorithm being the most prominent and widely used asymmetric cryptography algorithm, only very few prior publications are concerned with the identification of the library responsible for generating an RSA key directly from the key itself. The task was partially done by [125] using particular biases in private keys generated by OpenSSL and observed in datasets of weak factorable keys [83, 81, 2]. However, the method only worked when private keys and their primes were known and used a feature specific to OpenSSL keys. Hence the technique can be extended neither to study public keys generated by OpenSSL nor to other cryptographic libraries in general.

To the contrary, a significant body of literature is devoted to resiliency of the RSA algorithm against various attacks. Besides attacks on the messages (e.g., padding oracle [21, 29, 35] or related messages [183, 51]), a large portion of attacks aims to deduce the private key from the corresponding public key. The attacks can be divided into two classes based on the assumptions about the key:

1) No additional information – methods such as Pollard p-1 [149], Pollard Rho [150, 33], and a class of several sieving methods (e.g., NFS, GNFS).

2) Partial information – low private or public exponent [50, 182, 26, 23], implementation and side-channel attacks, and attacks based on Coppersmith's method [49].

The usage of generic factorization attacks is limited to small RSA keys due to their exponential time complexity – the current public record for a general 768-bit RSA [99] utilizes NFS. Only attacks from the second class are known to be used to break RSA moduli used in practice. Except for Wiener's attack [182] for a small private exponent, other notable attacks belong to the same class as Coppersmith's attack [50] or do require side-channel leak and active probing from an attacker. Coppersmith's algorithm can be viewed as a universal tool for attacking RSA keys generated with improperly chosen parameters or originating from a faulty implementation. The algorithm was adapted for various scenarios where some bits of a factor, of the private exponent or of the message are known [48, 50, 22] with a nice overview in [121].

In 2012, two independent teams [83, 109] analyzed RSA public keys on the Internet used as SSL certificates, SSH host keys and PGP keys and found that a small, but significant portion (0.5% of TLS, 1% of SSH) of public RSA keys shared prime factors (because of insufficient entropy during the generation process).

Such keys are trivially factorable by a computation of the greatest common divisor (GCD) using Euclid's algorithm. Malfunctioning random number generator leads to a discovery of weak keys in national IDs of Taiwanese citizens factorable by adapted Coppersmith's algorithm [18].

Proper generation of RSA keys is described in several standards (e.g., FIPS 186-4 [97], IEEE 1363-2000 [1], see [113] and [130] for an overview), having different requirements for the form of the primes. One feature is common to all these standards – the primes should be generated randomly using a large amount of entropy. In addition to specialized construction methods (e.g., provable primes), the generation of RSA primes is typically performed in several iterations, repeating two fundamental steps: a random candidate is generated and then tested for primality. Since the primality test is a time-consuming process, several authors have proposed various speedups for the candidate generation process ([32, 119, 92], see [91] for an overview of such methods). The current state of the art focused on constrained devices is described in [91], where the authors decreased the number of primality tests with a negligible loss of entropy (0.5 bits).

A potential flaw in implementation of cryptographic algorithm or compromise of single protocol participant can be alleviated by utilization of secure multiparty computation based on common cryptographic primitives like RSA, Diffie-Hellman or Elliptic curve cryptography [185, 43, 42, 27, 79, 77, 53, 31, 166, 166, 82]. Using specifically constructed cryptographic protocols, one can distribute trust between multiple components, preferably originating from non-crossing supply chains with different internal implementation and operated by an independent owners and thus reducing the likelihood of compromises and system-wide error. This design draws from protective-redundancy and component diversification [45, 13, 52]. As long as one of the components remains honest, the secret cannot be reconstructed or leaked. Such a scheme can be used to support a wide-range of commonly used cryptographic operations (e.g., random number and key generation, decryption, signing etc.) [156, 138, 75, 157, 123, 135, 25, 16, 110, 147, 115]

The scheme can be constructed so that all participants are required to participate in a particular execution (*t-of-t*) or relaxed to require only k participants (*k-of-t*) [61, 146, 78, 165, 28]. Unfortunately, the typical secure multiparty scheme is not backward compatible with commonly used cryptographic procedures (e.g., digital signing using RSA or ECDSA algorithms), requiring involved parties to run new software implementation. Such operational changes due to new and (often)

interactive protocol come with practical obstacles of updating of large number of system clients. The threshold RSA signature scheme [79] requires collaboration during every signature operation. A more efficient generation based on 3-prime RSA [166] is not suitable for devices with fixed API exposing only standard 2-prime RSA operations (e.g., smartcards). The protocols securing against active adversaries [82], are time-consuming even on standard CPUs while having prohibitively long keypair generation phases on performance-limited hardware. The collaborative method splitting the key generation between card manufacturer and cardholder [144] constructs 4-prime 4096-bit RSA key from two 2048-bit parts during an interactive protocol executed before the card's first use. In some cases, backward compatible schemes were designed [144, 58] and practically deployed [38]. Note that the secure multiparty computation can also be applied to any generic program that can be represented as a finite Boolean circuit [184, 139, 59, 102, 158, 11], yet for the price of significant computational, memory and communication overhead. We focus only on common cryptographic operations like signing or decryption.

Contributions

I started the collection of a large number of RSA keys generated by cryptographic smartcards with the idea of finding similarities to keys produced by the open-source libraries. The aim was to better understand the otherwise closed-source implementations used by smartcards and independently audit its security. Our work then examined the biases in both public and private keys from a majority of cryptographic libraries used nowadays [178] and showed significant detectable differences in the distribution of key bits. This surprising discovery (given that the RSA algorithm is in use for more than 40 years) opened a new analytic method how to:

a) Attribute a given key to its origin library,

b) quantify popularity of libraries in large datasets (e.g., IPv4 TLS scans),

c) spot irregularities hinting at potential vulnerabilities.

In [131], we extended our previous work [178] by applying statistical inference to approximate a share of libraries matching an observed distribution of RSA keys in an inspected dataset (e.g., an Internet-wide scan of TLS handshakes). While [178] primarily described the fact that detectable bias exists and can be used for probabilistic attribution of a single given key to its origin library, [131] was specifically

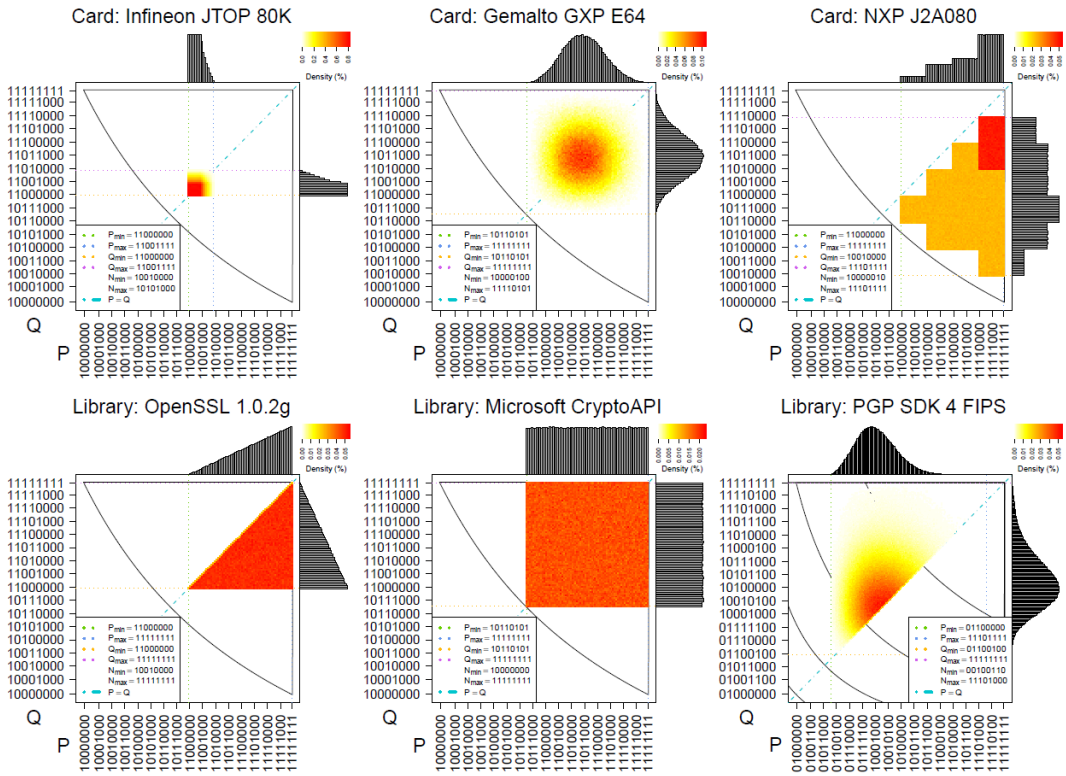

Figure 2.3: The example of differences between private keys as produced by different cryptographic libraries. Each graph depicts the distribution of the most significant byte of prime P and Q respectively and their relation as computed from a dataset of 1 million keypairs generated by a given library. Adapted from [178].

focused on estimation of the share distribution within a large dataset instead of particular keys from a particular library. With this reframing of the research question, probabilistic attribution is significantly more accurate (with less than 2% estimated misclassification error), sensitive enough even to detect transient events such as a periodic insertion of keys from a specific library into Certificate Transparency logs and inconsistencies in archived datasets. To our best knowledge, this is the first accurate measurement of the popularity of cryptographic libraries not based on proxy information like web server fingerprinting, but directly on the number of observed unique keys.

The wide-scale analysis of cryptographic libraries done in [178] also showed notable differences for one particular smartcard vendor (Infineon with the *RSALib* cryptographic library), which was not attributable to the "benign" implementation de-

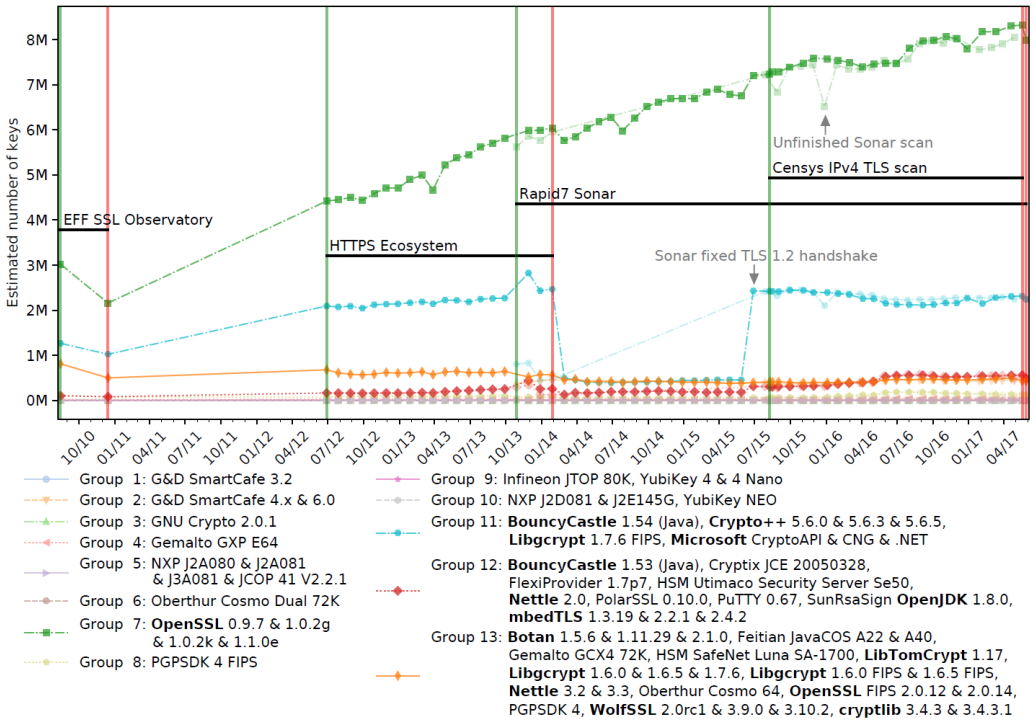

Figure 2.4: The first direct measurement of the popularity of different cryptographic libraries utilizing our method based on RSA bias. The OpenSSL library (green) is the most popular one and still growing. The fraction of keys from the Microsoft cryptographic libraries (blue) remains relatively stable (the big decrease between years 2014 and 2015 is caused by mistake in data collection methodology by Sonar organization providing the archived datasets). Adapted from [131].

cisions performed during the primes selection. The keys produced by *RSALib* were notable in many respects, including non-uniform distribution of remainders of public key modulo small primes. Moreover, such non-uniformity in public key must have been caused by the specific structure of primes used. We inferred the structure of primes (introduced by the manufacturer to speed up the costly prime generation), shown that primes in this form came with a significantly reduced entropy and devised practical factorization technique [132] based on the Coppersmith factorization method [48]. The method was efficient enough to factorize RSA keys (generated by *RSALib*) with a widely used key length of 2048 bits. Combined with the fact that affected chips were used in many domains including the citizen's identity cards in many European countries, Trusted Platform Modules (TPMs) utilized for the Microsoft BitLocker disk encryption software, authentication and software signing

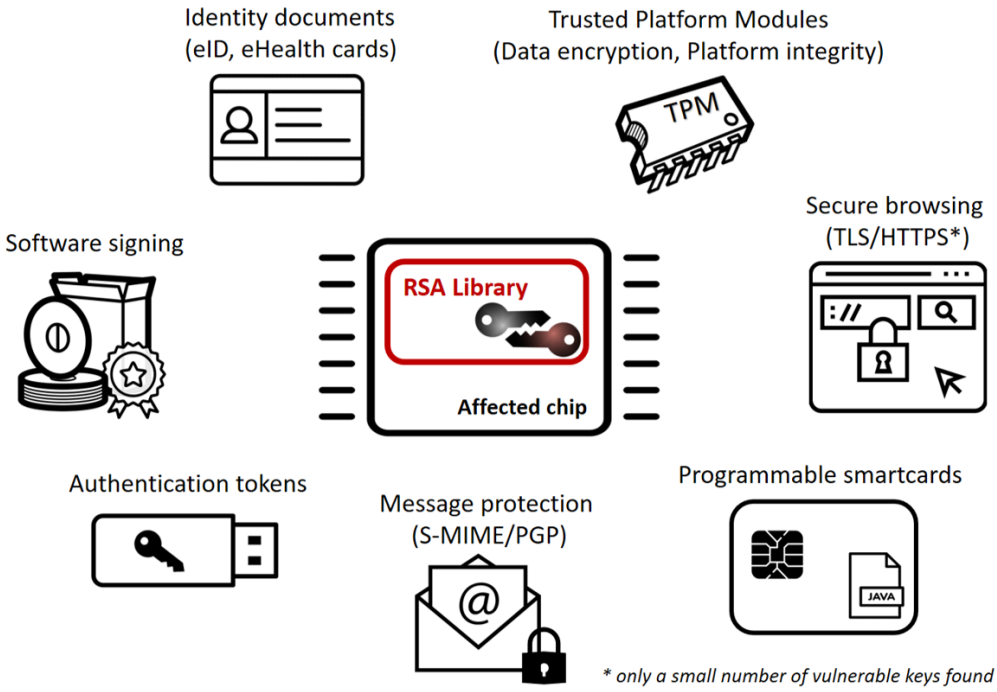

Figure 2.5: The usage domains impacted by ROCA vulnerability [132] with at least 1 billion chips impacted worldwide [71]. The RSA with 2048 bits is typically used with the electricity cost for the practical factorization equal to several thousand euros.

tokens among others. This cryptographic vulnerability was assigned the CVE-2017-15361 code and was addressed with software updates and procedural changes by many major vendors, certification laboratories as well as national countries. The open-source implementation of the vulnerable keys detection [6] released during the responsible disclosure resulted in a highly popular GitHub repository and was incorporated in several security scanning tools, including Let's Encrypt certification authority key eligibility assessment module.

In parallel, we worked on another line of research focused on secure multiparty cryptographic protocols suitable for cryptographic smartcards. Coincidentally, such a protocol would have prevented ROCA-like vulnerabilities, if utilized. In [120], we devised a suite of protocols based on elliptic curves for a distributed generation, and use of private keys for decryption and signature. As both the unintentional (bugs) and intentional errors (backdoors) cannot be ruled out, we intentionally used not a single, but multiple chips from different manufacturers. The combined group remains

cryptographically secure unless a critical error is present in all different chips. We also designed and built a high-performance scalable platform containing hundreds of cards running in parallel, thus achieving hundreds of operations per second, using the design proposed by us earlier on [55]. The work also required non-trivial engineering effort with the publication of first publicly available implementation of low-level ECPoint operations without any proprietary operations [7] and on-card performance profiler [8].

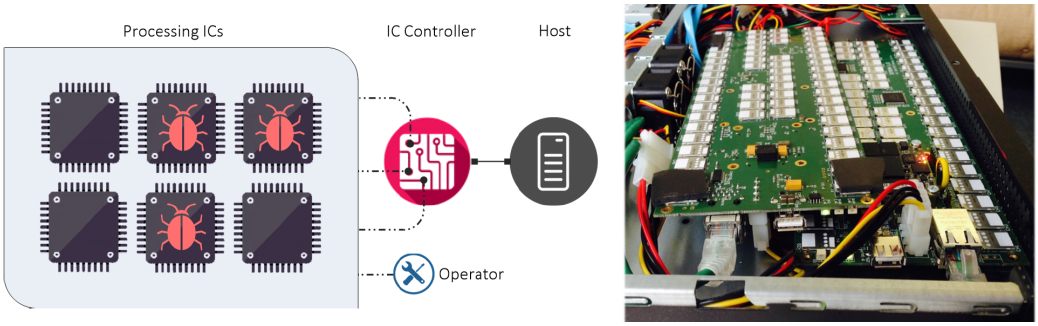

Figure 2.6: **Left:** The basic design of the Myst secure multiparty computation platform using secure cryptographic chips with some chips assumed to be flawed or backdoored (depicted with the bug icon). **Right:** The actual realization with array of SIM-sized cryptographic smartcards. Adapted from [120].

The secure execution environment was also used by us to design data retention mechanism, which protects the operator of a service (e.g., anonymization mix) against unjustified demands to reveal the service logs outside the period required by a law [101]. All logs are encrypted under a key available only inside a smartcard, which will release the key only when the logs are not older than the specified period. The current time is checked on-card against multiple trusted time servers.

Articles in collection

[178] P. Švenda, M. Nemec, P. Sekan, R. Kvašňovský, D. Formánek, D. Komárek, and V. Matyáš. The million-key question – investigating the origins of RSA public keys. In *The 25th USENIX Security Symposium (USENIX Security '16)*, pages 893–910. USENIX, 2016

I led the research, proposed the idea of testing RSA keypairs from smartcards for vulnerabilities, collected most of the keys, supervised a team of students working on analysis and wrote a significant part of the text. The paper received *Best paper award*.

Contribution 25%.

[131] M. Nemec, D. Klinec, P. Svenda, P. Sekan, and V. Matyas. Measuring popularity of cryptographic libraries in Internet-wide scans. In *The 33rd Annual Computer Security Applications Conference (ACSAC'17)*. ACM, 2017

I led the research, cooperated on the design of improved classification algorithm, experiments and analysis of usage domains and wrote a significant part of the text. Contribution 20%.

[132] M. Nemec, M. Sys, P. Svenda, D. Klinec, and V. Matyas. The return of Coppersmith's attack: Practical factorization of widely used RSA moduli. In *24th ACM Conference on Computer and Communications Security (CCS'17)*, pages 1631–1648. ACM, 2017

I led the research, co-spotted irregularities in keys produced by a vulnerable chip, performed analysis of vulnerability impact on multiple usage domains, handled responsible disclosure process and wrote a significant part of the text. The paper received *Real world impact award*. Contribution 25%.

[120] V. Mavroudis, A. Cerulli, P. Svenda, D. Cvrcek, D. Klinec, and G. Danezis. A touch of evil: High-assurance cryptographic hardware from untrusted components. In *24th ACM Conference on Computer and Communications Security (CCS'17)*, pages 1583–1600. ACM, 2017

I co-designed used smartcard infrastructure, performed major part of efficient implementation and participated on text writing. Contribution 25%.

Other relevant publications

[55] D. Cvrcek and P. Svenda. Architecture considerations for massively parallel hardware security platform. In *The 5th International Conference on Security, Privacy, and Applied Cryptography Engineering (SPACE'15), LNCS 9354*, pages 269–288. Springer, 2015

I cooperated on design and implementation of highly parallel architecture, implemented and analysed experiments and wrote major part of the text. Contribution 50%.

[101] S. Kopsell and P. Svenda. Law enforcement and data retention in the light of an anonymisation services. In *Masaryk UJL & Tech.*, volume 5, page 305. HeinOnline, 2011

I co-designed the used architecture and protocols, performed on-card implementation and analysed experiments and participated on text writing. Contribution 50%.

Chapter 3

Key establishment in compromised networks

Ad-hoc networks of nodes with limited capabilities often handle sensitive information and security of such networks is a typical baseline requirement. Such networks consist of a high number of interacting devices, the price of which should be as low as possible – limiting computational and storage resources and (expensive) tamper resistance. Systems secure by design with a strong focus on autonomous self-defense are desired, as an accurate detection and subsequent reaction is usually difficult. Lightweight security solutions are preferable, providing a low computational and communication overhead. When considering key management in nodes of limited capabilities, symmetric cryptography is frequently the preferred approach, yet with a low number of pre-distributed keys due to limited memory. [1]

3.1 Secrecy amplification

Substantial improvements in resilience against node capture (fraction of nodes with their keys compromised by an attacker) or key exchange eavesdropping (fraction of keys compromised) can be achieved when a group of neighboring nodes cooperates in an additional *secrecy amplification* (SA) protocol after the initial key establishment protocol. A strong majority of secure links can be obtained even when the initial network compromise is at 50% [179].

[1] The description of related work in this chapter is adapted and extended with input from our publications, mainly [179, 106, 143].

The SA concept was originally introduced in [12] for the *key infection* plaintext key exchange, but can also be used for a partially compromised network resulting from node capture in probabilistic pre-distribution schemes introduced by Eschenauer, and Gligor [70] and significantly studied later [40, 57, 64, 111]. Due to an attacker action, the communication link between nodes A and B secured by a link key K can be compromised. When the group of neighbors of nodes A and B cooperates in an additional protocol, communication link(s) protected by the previously compromised key K can be secured again, if a new key K' can be securely transported to both nodes A and B using non-compromised path. The exact way the new key value K' is transported is specified by a particular secrecy amplification (SA) protocol. The SA protocol is executed both for compromised links as well as the already secure ones, as a network operator usually has no knowledge which links are compromised. It serves as either prevention of future compromise or a response to a partial compromise which already happened. While SA protocol can try all possible paths to propagate a new key, the price would be a huge communication overhead. Therefore, the proposed SA protocols aim to find a good tradeoff between the number of paths tried and the probability of finding at least one secure path.

Different key distribution schemes and corresponding attacks result in different compromise patterns, in turn, influencing how successful an SA protocol will be. The random compromise pattern arises when a probabilistic key pre-distribution scheme of [70] and many later variants of [40, 57, 111, 112] are used, and an attacker extracts keys from several randomly captured nodes. Another pattern results from key distribution based on idea of "key infection" [12], later extended by [56, 98, 179] and others. Here, link keys are exchanged in plaintext (no keys are pre-distributed), and an attacker can compromise them if the transmission can be eavesdropped by the attacker's eavesdropping nodes.

The earliest published SA protocols are so-called *node-oriented* and were also proposed in a multi-hop (more than a single intermediate node) and multi-path (different set of intermediate nodes) variants [12, 56, 98, 179]. In *group-oriented* protocols [179] proposed later, an identification of the parties in the protocol is no longer "absolute" (e.g., node designation A, B, C), but it is given by the relative distance from other parties. This distance can be approximated from the minimal transmission power needed to communicate with a given neighbor or the signal strength measured during message reception. Based on the actual distribution of the neighbors, the node closest to the indicated distance(s) is chosen for a particular protocol run. As there is no need to re-execute the protocol for all k-tuples (as was

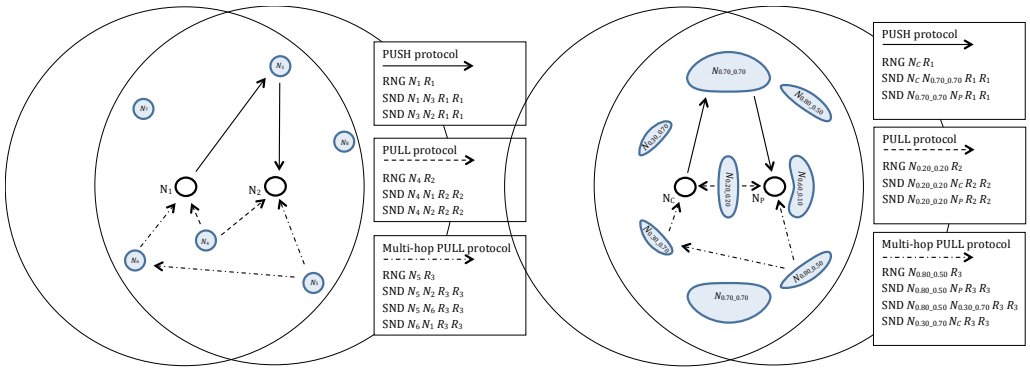

Figure 3.1: **Left:** An example of instructions of several node-oriented SA protocols. The Push, Pull, and multi-hop version of Pull are included. **Right:** An example of instructions of a basic hybrid SA protocol. Selected node-relative identification (distance from N_C and N_P) of involved parties are displayed as the geographic most probable areas, where such nodes will be positioned. Adapted from [143].

the case for *node-oriented* protocols), all the neighbors can be now involved into a single execution, reducing the communication overhead significantly. The main disadvantage is the complicated synchronization of the parallel executions and also complicated security analysis due to the high number of nodes involved. *Hybrid protocols* [142] combine properties of both node- and group-oriented protocols using the relative distance from other parties (as group-oriented) but a small number of steps of each protocol (as node-oriented).

Contributions

The work on lightweight key distribution protocols for devices with limited computational power in a partially compromised network was the main topics of my Ph.D. study with the first result published in 2004 and continued until now. I started the initial investigation of the security protocols for wireless sensor networks (WSN) in our laboratory, later followed with five more postgraduate students with three already graduated. While WSN was the main topic for my Ph.D. study, I gradually became more involved with the areas described in chapter 2 – this fact is reflected by my smaller author contribution to the papers presented, as I now act only as an advisor for the WSN research during the design phase and less involved later.

During the Ph.D. study, I designed, implemented and analyzed several secrecy amplification protocols, initially designed manually [56], later automatically [177,

179] using genetic programming combined with the custom network simulator. A candidate secrecy amplification protocol consists of a sequence of instructions (generate new value, combine existing values, send a value to a selected neighbor) operating over several memory registers. The actual instructions are searched for by genetic programming with the quality of candidate computed using custom network simulator [3].

This stream of work continued after my Ph.D. thesis with a thorough examination of the influence of various system parameters on the quality of resulting protocols [162]. Subsequently, we proposed a new class of *hybrid* secrecy amplification protocols, combining simplicity of early SA protocols (and thus achieving a low number of messages) and dynamic selection of participating nodes based on their relative distance [140]. We performed the detailed comparison of all known SA protocols on unified network settings [143], showing that hybrid design offers a favorable tradeoff between the fraction of secured links and transmission overhead while having simple implementation and small network synchronization requirements. I helped with the design of experiments to verify the resilience of SA protocols against more nuanced attacker models [141]. I proposed the idea of combining the keys with only small entropy (resulting from a key establishment based on RSSI extraction [151, 128]) to form single key with enough entropy to withstand brute-force attack. We designed a suite of efficient protocols for this use-case [129].

The proposed protocols for probabilistic key pre-distribution were also practically implemented into a transparent security middleware called WSNProtectLayer for TinyOS operating system used for WSN nodes and tested on our laboratory testbed as well as in the field on real devices [118].

While the common attacker models used in the area typically assumed too simplistic randomly-behaving attacker behavior, we examined more realistic attacker models derived from our field experiments and shown the practical limits of achieved compromise on the network with secrecy amplification protocols deployed as defense measure [141].

I designed the probabilistic key pre-distribution scheme securely utilizing keys stored in the memory of neighboring nodes with significantly improved node-capture resilience [167]. The idea was further extended with the use of hash chains to derive shared key between two nodes and increasing the resilience further again [106].

Within the area of wireless sensor networks, I also proposed to use genetic programming to automatically search for attacker strategies [168] expresses as a se-

quence of simple instructions with attacks against message routing, optimal placement of eavesdropping nodes used against secrecy amplification and selective node capture. The related work [105] investigated the inherent tradeoff between the ability to detect and react on network attacker on nodes and privacy-preserving mechanism trying to prevent linkability of events. As the intrusion detection nodes can also be compromised, sharing more information helps intrusion detection but also leaks more information towards an attacker – a privacy-preserving mechanism decrease the amount of information available to an attacker but also the accuracy of detection.

Articles in collection

[141] R. Ostadal, P. Svenda, and V. Matyas. Attackers in wireless sensor networks will be neither random nor jumping – secrecy amplification case. In *International Conference on Cryptology and Network Security (CANS'16)*. Springer, 2016

I helped with design of experiments, analysis of results and contributed to text writing. Contribution 15%.

[106] J. Kur, V. Matyas, and P. Svenda. Two improvements of random key predistribution for wireless sensor networks. In *International Conference on Security and Privacy in Communication Systems (SecureComm'12)*, pages 61–75. Springer, 2012

I co-designed the improved key establishment protocol, help with analysis and significantly contributed to text writing. Contribution 30%.

[162] T. Smolka, P. Švenda, L. Sekanina, and V. Matyáš. Evolutionary design of message efficient secrecy amplification protocols. In *European Conference on Genetic Programming (EuroGP'12)*, pages 194–205. Springer, 2012

I led the research, designed the experiments, cooperated on its analysis and wrote main part of the text. Contribution 35%.

Other relevant publications

[129] L. Nemec, R. Ostadal, V. Matyas, and P. Svenda. Entropy crowdsourcing – protocols for link key updates in wireless sensor networks. In *26th International*

Workshop on Security Protocols (SPW'18), LNCS 11286. Springer, 2018

I proposed the idea of combination of key shares with low-entropy, helped to design relevant protocols and contributed to text writing. Contribution 20%.

[128] L. Nemec, R. Ostadal, V. Matyas, and P. Svenda. Adaptive secrecy amplification with radio channel key extraction. In *14th International Conference on Distributed Computing in Sensor Systems (DCOSS'18), LNCS 11286*. Springer, 2018

I helped with the design of experiments and contributed to text writing. Contribution 13%.

[143] R. Ošťádal, P. Švenda, and V. Matyáš. On secrecy amplification protocols. In *9th International Conference on Information Security Theory and Practice (WISTP'15), LNCS 9311*, pages 3–19. Springer, 2015

I helped to design and analyse experiments and contributed to text writing. Contribution 15%.

[142] R. Ošťádal, P. Švenda, and V. Matyáš. A new approach to SA in partially compromised networks. In *4th International Conferenece on Security, Privacy, and Applied Cryptography Engineering (SPACE'14, LNCS 8804*, pages 92–109. Springer, 2014

I cooperated on the design and analyse experiments with hybrid protocols and contributed to text writing. Contribution 20%.

[179] P. Švenda, L. Sekanina, and V. Matyáš. Evolutionary design of secrecy amplification protocols for wireless sensor networks. In *Second ACM Conference on Wireless Network Security (WISEC'09)*, pages 225–236, 2009

I proposed the idea of the algorithm, designed, implemented and analyzed the experiments and wrote main part of the text. Contribution 60%.

Chapter 4

Conclusion and future work

My research contributions through the span of almost ten years are explained and put into the context of the state-of-the-art research. The unifying topic is randomness, covering its generation, statistical analysis to detect any bias present as well as bias use and (frequently) misuse with applications to the analysis of cryptographic primitives as well as utilization in the key establishment protocols. The research spans from the fundamental research on entropy extractors and on cryptographic protocols to the practical exploitation of a real-world vulnerability in hardware chips. The individual addressed research domains are accompanied with the list of the articles co-authored by me with an explanation of my contribution. The selected articles are also attached to this text for interested readers[1].

The research work outlined can be continued in several directions. The statistical testing using BoolTest was so far used only to detect the bias presence and not for an interpretation of the exact distinguisher found. The open question is how to extend the knowledge of distinguisher found to a particular structure inside a cryptographic function responsible for the bias presence – ideally in a semi-automatic way. The simple and straightforward structure of distinguishers found by BoolTest shall be directly extendable to perform the key recovery attack.

Building on the results produced en masse by our CryptoStreams project with 100+ cryptographic functions; we plan to investigate deeper the instances with surprising bias detected and to establish shared patterns among the distinguishers found. The randomness statistical testing used to be perceived only as first and

[1]The full-texts of selected articles are excluded from this document to avoid copyright violation and are only available in the printed version.

rather weak testing methodology for cryptographic function analysis. With the dynamically adapting statistical tests and deeper analysis of the distinguishers found, the goal is not only to find a weakness in some of the functions, but to reason about the shared properties of whole groups of the cryptographic functions.

The systematic and large-scale analysis of cryptographic software libraries for the presence of subtle biases in the keys produced with the application for library attribution can be further extended. The use of machine learning techniques to detect additional correlated bits may increase the classification accuracy for a single key and discover key generation implementations. Additionally, an accurate classification in a scenario with only tens of keys from 2-3 unknown libraries remains is still an open question while having a practical use and implications for the security audit of companies. While we focused so far mainly on the RSA algorithm, other algorithms like ECC can be investigated as well, including black-box implementations in cryptographic smartcards. Are we able to identify a library or a system that produced a given ECC public key or participate in the ECC-based authentication protocol? Can we detect the incorrect implementation of ECC operations despite having no access to its source code? As a broader vision, the suite of techniques proposed shall lead to easier verification that an analyzed implementation is not deviating in any detectable way from the expected case.

Finally, the secure multiparty protocols (MPC) provide the building block for the cryptographically strong computational environment, that removes the reliance on a single trusted entity – both for hardware manufacturing and delivery as well as for correctness of the software implementation. One can envision the secure virtual processor, which is realized by a proper combination of N physical components and diverse software implementations and providing secure computation until all N components fail. Having N as a changeable parameter, one can easily set the proper tradeoff between security, performance, and cost and even adapt dynamically in time.

While progress in the design of universally composable MPC schemes addressing very strong attacker model with a minimum of additional assumptions is continuously made, these schemes are still coming with significant performance and memory overhead, making it impractical on limited devices like smartcards. More efficient and practically usable MPC schemes can be constructed, if additional assumptions (like the hardness of discrete logarithm and EC Diffie-Hellman problem) is made as was also demonstrated by our work [120]. The recent work of Chadran et al. [41]

address the model relevant for smartcards (so-called *corrupted token model*) with universally composable MPC scheme which requires the only assumption of the existence of a one-way function, but being still prohibitively expensive for common smartcards. Would it be possible to close this gap and make some scheme practically usable on smartcards while not requiring additional assumptions like discrete logarithm problem?

To facilitate a faster adoption, a design of MPC schemes that would be backward compatible with the vast legacy security infrastructure is favorable. One example is a multiparty creation of ECDSA signatures on smartcards with an unmodified signature verification procedure – similarly as a scheme already demonstrated for two-party RSA signatures.

Bibliography

[1] IEEE Std 1363-2000: IEEE Standard – specifications for public-key cryptography. IEEE, 2000. cit. [2018-11-11]. Available from `https://books.google.cz/books?id=KKc8nQAACAAJ`.

[2] Dsa-1571-1 OpenSSL – predictable random number generator, 2008. [cit. 2017-09-20]. Available from `https://www.debian.org/security/2008/dsa-1571`.

[3] Petr Svenda and Jiri Kur, Sensor Security Simulator (S3), 2008. cit. [2018-11-11]. Available from `https://www.fi.muni.cz/~xsvenda/s3.html`.

[4] The EFF SSL Observatory, 2010. [cit. 2017-09-20]. Available from `https://www.eff.org/observatory`.

[5] CRoCS-MUNI, CryptoStreams project, 2018. cit. [2018-11-11] Available from `https://github.com/crocs-muni/CryptoStreams`.

[6] CRoCS-MUNI, ROCA: Infineon RSA key vulnerability dection tool, 2018. cit. [2018-11-11]. Available from `https://github.com/crocs-muni/roca`.

[7] OpenCryptoProject, JCMathLib project, 2018. [cit. 2018-11-11]. Available from `https://github.com/OpenCryptoProject/JCMathLib/`.

[8] Petr Svenda, JCProfiler project, 2018. cit. [2018-11-11]. Available from `https://github.com/OpenCryptoProject/JCProfiler/`.

[9] F. Aarts, J. De Ruiter, and E. Poll. Formal models of bank cards for free. In *IEEE Sixth International Conference on Software Testing, Verification and Validation Workshops (ICSTW'13), 2013*, pages 461–468. IEEE, 2013.

[10] M. R. Albrecht, J. P. Degabriele, T. B. Hansen, and K. G. Paterson. A surfeit of SSH cipher suites. In *Proceedings of the 2016 ACM SIGSAC Conference on*

Computer and Communications Security, CCS '16, pages 1480–1491. ACM, 2016.

[11] M. R. Albrecht, C. Rechberger, T. Schneider, T. Tiessen, and M. Zohner. Ciphers for MPC and FHE. In *Advances in Cryptology - EUROCRYPT'15, 2015, Proceedings, Part I*, pages 430–454, 2015.

[12] R. Anderson, H. Chan, and A. Perrig. Key infection: Smart trust for smart dust. In *12th IEEE International Conference on Network Protocols*, pages 206–215. IEEE, 2004.

[13] A. Avizienis. The N-version approach to fault-tolerant software. In *IEEE Transactions of Software Engineering*, volume 11, pages 1491–1501, 1985.

[14] M. Barbulescu, A. Stratulat, V. Traista-Popescu, and E. Simion. RSA weak public keys available on the Internet. In *International Conference for Information Technology and Communications*, pages 92–102. Springer-Verlag, 2016.

[15] G. V. Bard. *Algebraic Cryptanalysis*. Springer Publishing Company, ISBN 978-0-387-88756-2, 2009.

[16] M. Bellare and G. Neven. Multi-signatures in the plain public-key model and a general forking lemma. In *13th ACM conference on Computer and communications security (CCS'06)*, pages 390–399. ACM, 2006.

[17] D. J. Bernstein. Cache-timing attacks on AES, 2005. [cit. 2017-09-20]. Preprint available at `https://cr.yp.to/antiforgery/cachetiming-20050414.pdf`.

[18] D. J. Bernstein, Y.-A. Chang, C.-M. Cheng, L.-P. Chou, N. Heninger, T. Lange, and N. van Someren. Factoring RSA keys from certified smart cards: Coppersmith in the wild. In *Advances in Cryptology - ASIACRYPT'13*, pages 341–360. Springer-Verlag, 2013.

[19] D. J. Bernstein, T. Lange, and R. Niederhagen. Dual EC: a standardized back door. In *The New Codebreakers*, pages 256–281. Springer, 2016.

[20] A. Biryukov and V. Velichkov. Automatic search for differential trails in ARX ciphers. In *Cryptographers' Track at the RSA Conference*, pages 227–250. Springer, 2014.

[21] D. Bleichenbacher. Chosen ciphertext attacks against protocols based on the RSA encryption standard PKCS# 1. In *Annual International Cryptology Conference (CRYPTO'98)*, pages 1–12. Springer, 1998.

[22] D. Bleichenbacher and A. May. New attacks on RSA with small secret CRT-exponents. In *Public Key Cryptography - PKC'06*, pages 1–13. Springer-Verlag, 2006.

[23] J. Blömer and A. May. New partial key exposure attacks on RSA. In *Advances in Cryptology – CRYPTO'03*, pages 27–43. Springer-Verlag, 2003.

[24] H. Böck, J. Somorovsky, and C. Young. Return Of Bleichenbacher's Oracle Threat (ROBOT). In *27th USENIX Security Symposium (USENIX Security 18)*, pages 817–849, Baltimore, MD, 2018. USENIX Association.

[25] A. Boldyreva. Threshold signatures, multisignatures and blind signatures based on the Gap-Diffie-Hellman-Group signature scheme. In *Public Key Cryptography - PKC 2003*, pages 31–46, 2003.

[26] D. Boneh and G. Durfee. Cryptanalysis of RSA with private key d less than $N^{0.292}$. In *Advances in Cryptology — EUROCRYPT '99*, pages 1–11. Springer-Verlag, 1999.

[27] D. Boneh and M. Franklin. Efficient generation of shared RSA keys. In *Advances in Cryptology (CRYPTO'97)*, page 425. Springer, 1997.

[28] A. Bonnecaze and P. Trebuchet. Threshold signature for distributed time stamping scheme. *Annales des Télécommunications*, 62(11-12):1353–1364, 2007.

[29] M. Bortolozzo, G. Marchetto, R. Focardi, and G. Steel. Secure your PKCS#11 token against API attacks! In *3rd International Workshop on Analysis of Security APIs (ASA-3)*, July 2009.

[30] J. Bouda, J. Krhovjak, V. Matyas, and P. Svenda. Towards true random number generation in mobile environments. In *Nordic Conference on Secure IT Systems*, pages 179–189. Springer, 2009.

[31] F. Brandt. Efficient Cryptographic Protocol Design Based on Distributed El Gamal Encryption. In *8th International Conference Information Security and Cryptology (ICISC'05), Revised Selected Papers*, pages 32–47, 2005.

[32] J. Brandt, I. Damgård, and P. Landrock. Speeding up prime number generation. In *Advances in Cryptology (ASIACRYPT '91)*, pages 440–449. Springer-Verlag, 1993.

[33] R. P. Brent. An improved Monte Carlo factorization algorithm. *BIT Numerical Mathematics*, 20(2):176–184, 1980.

[34] D. R. Brown and K. Gjøsteen. A security analysis of the NIST SP 800-90 elliptic curve random number generator. In *Annual International Cryptology Conference (CRYPTO'07)*, pages 466–481. Springer, 2007.

[35] D. R. L. Brown. A Weak-Randomizer Attack on RSA-OAEP with e = 3, 2005. cit. [2018-11-11], Available at `http://eprint.iacr.org/2005/189`.

[36] R. G. Brown, D. Eddelbuettel, and D. Bauer. Dieharder: A random number test suite 3.31.1. *http://www.phy.duke.edu/~rgb/General/dieharder.php*, 2013.

[37] D. Brumley and D. Boneh. Remote timing attacks are practical. In *Computer Networks*, volume 48, pages 701–716. Elsevier, 2005.

[38] A. Buldas, A. Kalu, P. Laud, and M. Oruaas. Server-supported RSA signatures for mobile devices. In *European Symposium on Research in Computer Security (ESORICS'17)*, pages 315–333. Springer, 2017.

[39] W. Caelli et al. Crypt-X: Randomness testing suite, 1998. [cit.] Available at `https://web.archive.org/web/19990224063612/http://www.isrc.qut.edu.au/cryptx/index.html`.

[40] H. Chan, A. Perrig, and D. Song. Random key predistribution schemes for sensor networks. In *IEEE Symposium on Security and Privacy (S&P'03)*, pages 197–213, 2003.

[41] N. Chandran, W. Chongchitmate, R. Ostrovsky, and I. Visconti. Universally composable secure two and multi-party computation in the corruptible tamper-proof hardware token model. In *IACR Cryptology ePrint Archive, 2017: 1092*, 2017.

[42] D. Chaum, C. Crépeau, and I. Damgard. Multiparty unconditionally secure protocols. In *20th annual ACM Symposium on Theory of Computing*, pages 11–19. ACM, 1988.

[43] D. Chaum, I. B. Damgård, and J. Van de Graaf. Multiparty computations ensuring privacy of each party's input and correctness of the result. In *Conference on the Theory and Application of Cryptographic Techniques*, pages 87–119. Springer, 1987.

[44] S. Checkoway, J. Maskiewicz, C. Garman, J. Fried, S. Cohney, M. Green, N. Heninger, R.-P. Weinmann, E. Rescorla, and H. Shacham. A systematic analysis of the Juniper Dual EC incident. In *ACM SIGSAC Conference on Computer and Communications Security (CCS'16)*, pages 468–479. ACM, 2016.

[45] L. Chen and A. Avizienis. N-version programming: A fault-tolerance approach to reliability of software operation. In *Digest of Papers FTCS-8: Eighth Annual International Conference on Fault Tolerant Computing*, pages 3–9, 1978.

[46] T. Chung, Y. Liu, D. Choffnes, D. Levin, B. M. Maggs, A. Mislove, and C. Wilson. Measuring and applying invalid SSL certificates: The silent majority. In *2016 ACM on Internet Measurement Conference (IMC'16)*, pages 527–541. ACM, 2016.

[47] J. Clark and P. C. van Oorschot. SoK: SSL and HTTPS: Revisiting past challenges and evaluating certificate trust model enhancements. In *IEEE Symposium on Security and Privacy*, pages 511–525. IEEE, 2013.

[48] D. Coppersmith. Finding a small root of a bivariate integer equation; factoring with high bits known. In *Proceedings of EUROCRYPT'96*, pages 178–189. Springer-Verlag, 1996.

[49] D. Coppersmith. Finding a small root of a univariate modular equation. In *Advances in Cryptology — EUROCRYPT '96*, pages 155–165. Springer-Verlag, 1996.

[50] D. Coppersmith. Small solutions to polynomial equations, and low exponent RSA vulnerabilities. *Journal of Cryptology*, 10(4):233–260, 1997.

[51] D. Coppersmith, M. Franklin, J. Patarin, and M. Reiter. Low-exponent RSA with related messages. pages 1–9. Springer-Verlag, 1996.

[52] B. Cox and D. Evans. N-variant systems: A secretless framework for security through diversity. In *Proceedings of the 15th USENIX Security Symposium, Vancouver, BC, Canada, July 31 - August 4, 2006*, 2006.

[53] R. Cramer, I. Damgård, and J. B. Nielsen. Multiparty computation, an introduction. In *Contemporary cryptology*, pages 41–87. Springer, 2005.

[54] Batch-GCDing Github SSH Keys, 2015. [cit. 2017-09-20]. Available from `https://cryptosense.com/batch-gcding-github-ssh-keys/`.

[55] D. Cvrcek and P. Svenda. Architecture considerations for massively parallel hardware security platform. In *The 5th International Conference on Security, Privacy, and Applied Cryptography Engineering (SPACE'15), LNCS 9354*, pages 269–288. Springer, 2015.

[56] D. Cvrček and P. Švenda. Smart dust security-key infection revisited. In *Electronic Notes in Theoretical Computer Science*, volume 157, pages 11–25. Elsevier, 2006.

[57] R. D. Pietro, L. V. Mancini, and A. Mei. Random key-assignment for secure wireless sensor networks. In *1st ACM Workshop on Security of Ad Hoc and Sensor Networks*, pages 62–71, 2003.

[58] I. Damgård and M. Koprowski. Practical threshold RSA signatures without a trusted dealer. In *International Conference on the Theory and Applications of Cryptographic Techniques*, pages 152–165. Springer, 2001.

[59] I. Damgård, V. Pastro, N. P. Smart, and S. Zakarias. Multiparty computation from somewhat homomorphic encryption. In *Advances in Cryptology (CRYPTO'12)*, pages 643–662, 2012.

[60] J. De Ruiter and E. Poll. Protocol state fuzzing of TLS implementations. In *USENIX Security Symposium*, pages 193–206, 2015.

[61] Y. Desmedt and Y. Frankel. Threshold cryptosystems. In *Advances in Cryptology (CRYPTO '89)*, pages 307–315, 1989.

[62] D. Dolev and A. Yao. On the security of public key protocols. In *IEEE Transactions on information theory*, volume 29, pages 198–208. IEEE, 1983.

[63] C. Doty-Humphrey. Practically Random: Specific tests in PractRand, 2014. [cit. 2018-11-11]. Available at `http://pracrand.sourceforge.net/`.

[64] W. Du, J. Deng, Y. S. Han, and P. K. Varshney. A pairwise key pre-distribution for wireless sensor networks. *ACM Conference on Computer and Communications Security (CCS'03)*, pages 42–51, 2003.

[65] Z. Durumeric, D. Adrian, A. Mirian, M. Bailey, and J. A. Halderman. A search engine backed by internet-wide scanning. In *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security*, pages 542–553. ACM, 2015.

[66] Z. Durumeric, M. Bailey, and J. A. Halderman. An internet-wide view of internet-wide scanning. In *Proceeding of USENIX Security Symposium*, pages 65–78, 2014.

[67] Z. Durumeric, J. Kasten, D. Adrian, J. A. Halderman, M. Bailey, F. Li, N. Weaver, J. Amann, J. Beekman, M. Payer, et al. The matter of Heartbleed. In *Proceedings of the 2014 Conference on Internet Measurement Conference*, pages 475–488. ACM, 2014.

[68] Z. Durumeric, J. Kasten, M. Bailey, and J. A. Halderman. Analysis of the HTTPS certificate ecosystem. In *Proceedings of the 2013 ACM Internet Measurement Conference*, pages 291–304. ACM, 2013.

[69] H. Englund, T. Johansson, and M. Sönmez Turan. A framework for chosen IV statistical analysis of stream ciphers. In *INDOCRYPT 2007*, pages 268–281. Springer Berlin Heidelberg, 2007.

[70] L. Eschenauer and V. D. Gligor. A key-management scheme for distributed sensor networks. In *9th ACM Conference on Computer and Communications Security, Washington, DC, USA*, pages 41–47. ACM, 2002.

[71] Estonia Information System Authority. Roca vulnerability and eid: Lessons learned.

[72] A. P. Felt, R. Barnes, A. King, C. Palmer, C. Bentzel, and P. Tabriz. Measuring HTTPS adoption on the web. In *26th USENIX Security Symposium*, pages 1323–1338. USENIX Association, 2017.

[73] E. Filiol. A new statistical testing for symmetric ciphers and hash functions. In *ICICS 2002*, pages 342–353. Springer Berlin Heidelberg, 2002.

[74] S. Fluhrer, I. Mantin, and A. Shamir. Weaknesses in the key scheduling algorithm of RC4. In *International Workshop on Selected Areas in Cryptography*, pages 1–24. Springer, 2001.

[75] Z. Galil, S. Haber, and M. Yung. Cryptographic computation: Secure fault-tolerant protocols and the public-key model. In *Conference on the Theory and Application of Cryptographic Techniques*, pages 135–155. Springer, 1987.

[76] A. Garrett, J. Hamilton, and G. Dozier. A comparison of genetic algorithm techniques for the cryptanalysis of TEA. In *International Journal of Intelligent Control and Systems*, volume 12, pages 325–330. Springer, 2007.

[77] R. Gennaro, S. Jarecki, H. Krawczyk, and T. Rabin. Secure distributed key generation for discrete-log based cryptosystems. In *Eurocrypt'99*, pages 295–310. Springer, 1999.

[78] R. Gennaro, S. Jarecki, H. Krawczyk, and T. Rabin. Secure distributed key generation for discrete-log based cryptosystems. *J. Cryptology*, 20(1):51–83, 2007.

[79] N. Gilboa. Two party RSA key generation. In *Annual International Cryptology Conference (CRYPTO'99)*, pages 116–129. Springer, 1999.

[80] I. Goldberg and D. Wagner. Randomness and the Netscape browser. *Dr Dobb's Journal-Software Tools for the Professional Programmer*, 21(1):66–71, 1996.

[81] M. Hastings, J. Fried, and N. Heninger. Weak keys remain widespread in network devices. In *Proceedings of the 2016 ACM on Internet Measurement Conference (IMC'16)*, pages 49–63. ACM, 2016.

[82] C. Hazay, G. L. Mikkelsen, T. Rabin, and T. Toft. Efficient RSA key generation and threshold Paillier in the two-party setting. In *CT-RSA*, pages 313–331. Springer, 2012.

[83] N. Heninger, Z. Durumeric, E. Wustrow, and J. A. Halderman. Mining your Ps and Qs: Detection of widespread weak keys in network devices. In *The 21st USENIX Security Symposium (USENIX Security'12)*, pages 205–220. USENIX, 2012.

[84] J. Hernández and P. Isasi. Finding efficient distinguishers for cryptographic mappings, with an application to the block cipher TEA. In *Computational Intelligence*, volume 20, pages 517–525. Blackwell, 2004.

[85] H. M. Heys. A tutorial on linear and differential cryptanalysis. In *Cryptologia*, volume 26, pages 189–221, Bristol, PA, USA, July 2002. Taylor & Francis, Inc.

[86] K. Hole, V. Moen, and A. Klingsheim. Challenges in securing networked J2ME applications. *Computer*, (2):24–30, 2007.

[87] The ICSI Certificate Notary, 2017. [cit. 2017-09-20]. Available from `https://notary.icsi.berkeley.edu/`.

[88] M. S. Inci, B. Gulmezoglu, T. Eisenbarth, and B. Sunar. Co-location detection on the cloud. In *International Workshop on Constructive Side-Channel Analysis and Secure Design*, pages 19–34. Springer-Verlag, 2016.

[89] G. Irazoqui, M. S. IncI, T. Eisenbarth, and B. Sunar. Know thy neighbor: crypto library detection in cloud. *Proceedings on Privacy Enhancing Technologies*, 2015(1):25–40, 2015.

[90] G. Jones. Gjrand random numbers, 2007.

[91] M. Joye and P. Paillier. Fast generation of prime numbers on portable devices: An update. In *Cryptographic Hardware and Embedded Systems - CHES 2006*, pages 160–173. Springer-Verlag, 2006.

[92] M. Joye, P. Paillier, and S. Vaudenay. Efficient generation of prime numbers. In *Cryptographic Hardware and Embedded Systems — CHES 2000*, pages 340–354. Springer-Verlag, 2000.

[93] A. Kaminsky and J. Sorrell. Cryptostat: a Bayesian statistical testing framework for block ciphers and MACs. *Rochester Institute of Technology, Rochester, NY*, 2013.

[94] J. Kamp and D. Zuckerman. Deterministic extractors for bit-fixing sources and exposure-resilient cryptography. *SIAM Journal on Computing*, 36(5):1231–1247, 2006.

[95] J. Kelsey, B. Schneier, and D. Wagner. Related-key cryptanalysis of 3-way, biham-des, cast, des-x, newdes, rc2, and tea. In *International Conference on Information and Communications Security*, pages 233–246. Springer, 1997.

[96] A. Kerckhoffs. La cryptographie militaire. *Journal des sciences militaires, vol. IX, pp. 5–83*, 1883.

[97] C. F. Kerry and R. Charles. FIPS PUB 186. Federal Information Processing Standards Publication, Digital Signature Standard (DSS). 2013.

[98] Y. H. Kim, M. H. Kim, D. H. Lee, and C. Kim. A key management scheme for commodity sensor networks. In *4th International Conference on Ad Hoc and Wireless networks, LNCS 3738*, pages 113–126. Springer, 2005.

[99] T. Kleinjung, K. Aoki, J. Franke, A. K. Lenstra, E. Thomé, J. W. Bos, P. Gaudry, A. Kruppa, P. L. Montgomery, D. A. Osvik, H. Te Riele, A. Timofeev, and P. Zimmermann. Factorization of a 768-bit RSA modulus. In *Proceedings of the 30th Annual Conference on Advances in Cryptology*, CRYPTO'10, pages 333–350. Springer-Verlag, 2010.

[100] D. E. Knuth. *The Art of Computer Programming*, volume 2. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, first edition, 1969.

[101] S. Kopsell and P. Svenda. Law enforcement and data retention in the light of an anonymisation services. In *Masaryk UJL & Tech.*, volume 5, page 305. HeinOnline, 2011.

[102] B. Kreuter, A. Shelat, and C.-H. Shen. Billion-gate secure computation with malicious adversaries. In *USENIX Security Symposium*, volume 12, pages 285–300, 2012.

[103] J. Krhovják, V. Matyas, and P. Svenda. The sources of randomness in mobile devices. *In Proceeding of NORDSEC'07*, pages 73–84, 2007.

[104] K. Kubíček, J. Novotný, P. Svenda, and M. Ukrop. New results on reduced-round Tiny Encryption Algorithm using genetic programming. *IEEE Infocommunications*, vol. 8, 2016.

[105] J. Kur, V. Matyas, A. Stetsko, and P. Svenda. Attack detection vs. privacy–how to find the link or how to hide it? In *International Workshop on Security Protocols*, pages 189–199. Springer, 2011.

[106] J. Kur, V. Matyas, and P. Svenda. Two improvements of random key predistribution for wireless sensor networks. In *International Conference on Security and Privacy in Communication Systems (SecureComm'12)*, pages 61–75. Springer, 2012.

[107] D. Lazar, H. Chen, X. Wang, and N. Zeldovich. Why does cryptographic software fail?: A case study and open problems. In *Proceedings of 5th Asia-Pacific Workshop on Systems*, pages 1–7. ACM, 2014.

[108] P. L'Ecuyer and R. Simard. TestU01: A C library for empirical testing of random number generators. In *ACM Trans. Math. Softw.*, volume 33, New York, NY, USA, Aug. 2007. ACM.

[109] A. K. Lenstra, J. P. Hughes, M. Augier, J. W. Bos, T. Kleinjung, and C. Wachter. Public keys. In *Advances in Cryptology (Crypto'12)*, volume 7417 of *Lecture Notes in Computer Science*, pages 626–642. Springer-Verlag, 2012.

[110] Y. Lindell and B. Pinkas. An efficient protocol for secure two-party computation in the presence of malicious adversaries. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 52–78. Springer, 2007.

[111] D. Liu and P. Ning. Establishing pairwise keys in distributed sensor networks. In *10th ACM Conference on Computer and communications security*, pages 52–61. ACM Press, 2003.

[112] D. Liu, P. Ning, and R. Li. Establishing pairwise keys in distributed sensor networks. *ACM Trans. Inf. Syst. Secur.*, 8(1):41–77, Feb. 2005.

[113] D. Loebenberger and M. Nüsken. Analyzing standards for RSA integers. *AFRICACRYPT'11*, abs/1104.4356, 2011.

[114] G. Lowe. OpenSSL Security Advisory – OpenSSL FIPS Object Module vulnerabilities. In *http://www.openssl.org/news/secadv_20071129.txt*, 2007.

[115] S. Lu, R. Ostrovsky, A. Sahai, H. Shacham, and B. Waters. Sequential aggregate signatures, multisignatures, and verifiably encrypted signatures without random oracles. *Journal of Cryptology*, 26(2):340–373, 2013.

[116] G. Marsaglia. The Marsaglia random number CDROM including the diehard battery of tests of randomness. *http://www.stat.fsu.edu/pub/diehard/*, 1995.

[117] M. Mascagni and A. Srinivasan. Algorithm 806: SPRNG: A scalable library for pseudorandom number generation. *ACM Transactions on Mathematical Software (TOMS)*, 26(3):436–461, 2000.

[118] V. Matyas, P. Svenda, A. Stetsko, D. Klinec, F. Jurnecka, and M. Stehlik. WSNProtectLayer: Security Middleware for Wireless Sensor Networks. In *Securing Cyber-Physical Systems*, pages 119–162. CRC Press, sep 2015.

[119] U. M. Maurer. Fast generation of prime numbers and secure public-key cryptographic parameters. *Journal of Cryptology*, 8(3):123–155, 1995.

[120] V. Mavroudis, A. Cerulli, P. Svenda, D. Cvrcek, D. Klinec, and G. Danezis. A touch of evil: High-assurance cryptographic hardware from untrusted components. In *24th ACM Conference on Computer and Communications Security (CCS'17)*, pages 1583–1600. ACM, 2017.

[121] A. May. Using LLL-Reduction for Solving RSA and Factorization Problems. In *The LLL Algorithm: Survey and Applications*, pages 315–348. Springer-Verlag, 2010.

[122] W. Mayer, A. Zauner, M. Schmiedecker, and M. Huber. No need for black chambers: testing TLS in the e-mail ecosystem at large. In *Availability, Reliability and Security (ARES), 2016 11th International Conference on*, pages 10–20. IEEE, 2016.

[123] S. Micali, K. Ohta, and L. Reyzin. Accountable-subgroup multisignatures: extended abstract. In *8th ACM Conference on Computer and Communications Security (CCS'01)*, pages 245–254, 2001.

[124] J. F. Miller and P. Thomson. Cartesian Genetic Programming. In *3rd European Conference on Genetic Programming EuroGP'00*, volume 1802 of *Lecture Notes in Computer Science*, pages 121–132. Springer-Verlag, 2000.

[125] I. Mironov. Factoring RSA Moduli II. [cit. 2018-11-11]. Available from `https://windowsontheory.org/2012/05/17/factoring-rsa-moduli-part-ii/`.

[126] I. Mironov. (Not so) random shuffles of RC4. In *Annual International Cryptology Conference (CRYPTO'02)*, pages 304–319. Springer, 2002.

[127] N. Mouha. ECRYPT II: Tools for cryptography. *[cit. 2018-11-11]. Available at `http://www.ecrypt.eu.org/tools/overview`*, 2010.

[128] L. Nemec, R. Ostadal, V. Matyas, and P. Svenda. Adaptive secrecy amplification with radio channel key extraction. In *14th International Conference on Distributed Computing in Sensor Systems (DCOSS'18), LNCS 11286*. Springer, 2018.

[129] L. Nemec, R. Ostadal, V. Matyas, and P. Svenda. Entropy crowdsourcing – protocols for link key updates in wireless sensor networks. In *26th International Workshop on Security Protocols (SPW'18), LNCS 11286*. Springer, 2018.

[130] M. Nemec. The properties of RSA key generation process in software libraries. *Diploma thesis, Masaryk University*, 2016.

[131] M. Nemec, D. Klinec, P. Svenda, P. Sekan, and V. Matyas. Measuring popularity of cryptographic libraries in Internet-wide scans. In *The 33rd Annual Computer Security Applications Conference (ACSAC'17)*. ACM, 2017.

[132] M. Nemec, M. Sys, P. Svenda, D. Klinec, and V. Matyas. The return of Coppersmith's attack: Practical factorization of widely used RSA moduli. In *24th ACM Conference on Computer and Communications Security (CCS'17)*, pages 1631–1648. ACM, 2017.

[133] NetCraft April 2017 Web Server Survey, 2017. [cit. 2017-09-20]. Available from `https://news.netcraft.com/archives/2017/04/21/april-2017-web-server-survey.html`.

[134] NetCraft operating system detection, 2017. [cit. 2017-09-20]. Available from `http://uptime.netcraft.com/accuracy.html#os`.

[135] A. Nicolosi, M. N. Krohn, Y. Dodis, and D. Mazières. Proactive two-party signatures for user authentication. In *Proceedings of the Network and Distributed System Security Symposium, NDSS'03, San Diego, California, USA*, 2003.

[136] NIST. FIPS 140-2 security requirements for cryptographic modules. NIST, May 2001.

[137] Nmap Remote OS Detection, 2017. [cit. 2017-09-20]. Available from `https://nmap.org/book/osdetect.html`.

[138] K. Ohta and T. Okamoto. A digital multisignature scheme based on the fiat-shamir scheme. In *Advances in Cryptology - ASIACRYPT '91*, pages 139–148, 1991.

[139] C. Orlandi. Is multiparty computation any good in practice? In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP'11), 2011*, pages 5848–5851. IEEE, 2011.

[140] R. Ostadal, P. Svenda, and V. Matyas. A new approach to sa in partially compromised networks. In *Security, Privacy, and Applied Cryptography Engineering – 4th Int. Conf., SPACE 2014, LNCS 8804*. Springer, 2014.

[141] R. Ostadal, P. Svenda, and V. Matyas. Attackers in wireless sensor networks will be neither random nor jumping – secrecy amplification case. In *International Conference on Cryptology and Network Security (CANS'16)*. Springer, 2016.

[142] R. Ošťádal, P. Švenda, and V. Matyáš. A new approach to SA in partially compromised networks. In *4th International Conferenece on Security, Privacy,*

and Applied Cryptography Engineering (SPACE'14, LNCS 8804, pages 92–109. Springer, 2014.

[143] R. Ošťádal, P. Švenda, and V. Matyáš. On secrecy amplification protocols. In *9th International Conference on Information Security Theory and Practice (WISTP'15), LNCS 9311*, pages 3–19. Springer, 2015.

[144] A. Parsovs. Identity card key generation in the malicious card issuer model. In *MTAT Research seminar report*, 2014. cit. [2018-11-11].

[145] G. Paul, S. Rathi, and S. Maitra. On non-negligible bias of the first output byte of RC4 towards the first three bytes of the secret key. *Designs, Codes and Cryptography*, 49(1-3):123–134, 2008.

[146] T. P. Pedersen. Non-interactive and information-theoretic secure verifiable secret sharing. In *Advances in Cryptology - CRYPTO '91*, pages 129–140, 1991.

[147] B. Pinkas, T. Schneider, N. P. Smart, and S. C. Williams. Secure two-party computation is practical. In *International Conference on the Theory and Application of Cryptology and Information Security*, pages 250–267. Springer, 2009.

[148] C. Piras. RaBiGeTe: random bit generators tester, 2004. [cit. 2018-11-11]. Available at `http://cristianopi.altervista.org/RaBiGeTe_MT/`.

[149] J. M. Pollard. Theorems on factorization and primality testing. volume 76, pages 521–528. Cambridge University Press, 1974.

[150] J. M. Pollard. A Monte Carlo method for factorization. *BIT Numerical Mathematics*, 15(3):331–334, 1975.

[151] S. N. Premnath, S. Jana, J. Croft, P. L. Gowda, M. Clark, S. K. Kasera, N. Patwari, and S. V. Krishnamurthy. Secret key extraction from wireless signal strength in real environments. *IEEE Transactions on mobile Computing*, 12(5):917–930, 2013.

[152] R. Rivest. RSA security response to weaknesses in key scheduling algorithm of RC4. *Technical note, RSA Data Security, Inc*, 2001.

[153] S. Ruhault. SoK: security models for pseudo-random number generators. *IACR Transactions on Symmetric Cryptology*, 2017(1):506–544, 2017.

[154] A. Rukhin. A statistical test suite for the validation of random number generators and pseudo random number generators for cryptographic applications, version STS-2.1. NIST, 2010.

[155] W. Schindler and W. Killmann. Evaluation criteria for true (physical) random number generators used in cryptographic applications. In *International Workshop on Cryptographic Hardware and Embedded Systems*, pages 431–449. Springer, 2002.

[156] C.-P. Schnorr. Efficient signature generation by smart cards. *Journal of cryptology*, 4(3):161–174, 1991.

[157] C. P. Schnorr and M. Jakobsson. Security of discrete log cryptosystems in the random oracle and generic model. *The Mathematics of Public-Key Cryptography. The Fields Institute*, 1999.

[158] C.-h. Shen et al. Fast two-party secure computation with minimal assumptions. In *Proceedings of the 2013 ACM SIGSAC conference on Computer & communications security*, pages 523–534. ACM, 2013.

[159] D. Shumow and N. Ferguson. On the possibility of a back door in the NIST SP800-90 Dual EC PRNG. In *Crypto 2007 Rump session*, 2007.

[160] E. Simion. The relevance of statistical tests in cryptography. *IEEE Security & Privacy*, 13(1):66–70, 2015.

[161] K. I. F. Simonsen, V. Moen, and K. J. Hole. Attack on Sun's MIDP reference implementation of SSL. In *Proceedings of NORDSEC'05*, 2005.

[162] T. Smolka, P. Švenda, L. Sekanina, and V. Matyáš. Evolutionary design of message efficient secrecy amplification protocols. In *European Conference on Genetic Programming (EuroGP'12)*, pages 194–205. Springer, 2012.

[163] M. Sýs, P. Svenda, M. Ukrop, and V. Matyas. Constructing empirical tests of randomness. In *Proceedings of the 11th International Conference on Security and Cryptography*, 2014.

[164] P. Stankovski. Greedy distinguishers and nonrandomness detectors. In *INDOCRYPT 2010, LNCS 6498*. Springer, 2010.

[165] D. R. Stinson and R. Strobl. Provably secure distributed schnorr signatures and a (t, n) threshold scheme for implicit certificates. In *Information Security*

and Privacy, 6th Australasian Conference, ACISP 2001, Sydney, Australia, July 11-13, 2001, Proceedings, pages 417–434, 2001.

[166] T. Straub. Efficient two party multi-prime RSA key generation. In *International Conference on Communication, Network, and Information Security*, pages 100–105. ACTA Press, 2003.

[167] P. Svenda and V. Matyas. Authenticated key exchange with group support for wireless sensor networks. In *IEEE International Conference on Mobile Adhoc and Sensor Systems Conference (MASS'07)*. IEEE, 2007.

[168] P. Svenda and V. Matyas. On the origin of yet another channel. In *Proceedings of the 21st International Workshop on Security Protocols (IWSP'13)*, pages 223–237. Springer, 2013.

[169] P. Svenda, M. Ukrop, and V. Matyas. Towards cryptographic function distinguishers with evolutionary circuits. In *10th International Conference on Security and Cryptography (SECRYPT'13)*, pages 135–146. SciTePress, 2013.

[170] P. Svenda, M. Ukrop, and V. Matyas. Determining cryptographic distinguishers for eStream and SHA-3 candidate functions with evolutionary circuits. In *E-Business and Telecommunications*, volume 456, pages 290–305. Springer Berlin Heidelberg, 2014.

[171] M. Sys, D. Klinec, and P. Svenda. The efficient randomness testing using Boolean functions. In *Proceedings of the 14th International Conference on Security and Cryptography (SECRYPT'17)*, pages 92–103. SCITEPRESS, 2017.

[172] M. Sýs, P. Švenda, M. Ukrop, and V. Matyáš. Constructing empirical tests of randomness. In *Proceedings of the 11th International Conference on Security and Cryptography (SECRYPT'14)*, pages 1–9. ICETE, 2014.

[173] L. Trevisan and S. Vadhan. Extracting randomness from samplable distributions. In *41st Annual Symposium on Foundations of Computer Science (FOCS'00)*, page 32. IEEE, 2000.

[174] M. Ukrop and P. Svenda. Avalanche effect in improperly initialized CAESAR candidates. In *Electronic Proceedings in Theoretical Computer Science*, volume 233, pages 72–81. Open Publishing Association, 2016.

[175] B. VanderSloot, J. Amann, M. Bernhard, Z. Durumeric, M. Bailey, and J. A. Halderman. Towards a complete view of the certificate ecosystem. In *Proceedings of the 2016 ACM on Internet Measurement Conference*, pages 543–549. ACM, 2016.

[176] J. Von Neumann et al. Various techniques used in connection with random digits. *Applied Math Series*, 12(36-38):1, 1951.

[177] P. Švenda and V. Matyáš. *From Problem to Solution: Wireless Sensor Networks Security (chapter in book)*. Nova Science Publishers, New York, USA, 2008. ISBN 978-1-60456-458-0.

[178] P. Švenda, M. Nemec, P. Sekan, R. Kvašňovský, D. Formánek, D. Komárek, and V. Matyáš. The million-key question – investigating the origins of RSA public keys. In *The 25th USENIX Security Symposium (USENIX Security '16)*, pages 893–910. USENIX, 2016.

[179] P. Švenda, L. Sekanina, and V. Matyáš. Evolutionary design of secrecy amplification protocols for wireless sensor networks. In *Second ACM Conference on Wireless Network Security (WISEC'09)*, pages 225–236, 2009.

[180] J. Walker. Ent: A pseudorandom number sequence test program, 2008. [cit. 2018-11-11]. Available at `https://www.fourmilab.ch/random/`, urldate = 2017-05-20,.

[181] D. J. Wheeler and R. M. Needham. TEA, a tiny encryption algorithm. In *International Workshop on Fast Software Encryption*, pages 363–366. Springer, 1994.

[182] M. J. Wiener. Cryptanalysis of short RSA secret exponents. *IEEE Transactions on Information Theory*, 36:553–558, 1990.

[183] O. Yacobi and Y. Yacobi. A new related message attack on RSA. In *Public Key Cryptography (PKC'05)*, volume 3386 of *Lecture Notes in Computer Science*, pages 1–8. Springer-Verlag, 2005.

[184] A. C. Yao. Protocols for secure computations. In *Foundations of Computer Science, 1982. SFCS'08. 23rd Annual Symposium on*, pages 160–164. IEEE, 1982.

[185] A. C.-C. Yao. How to generate and exchange secrets. In *Foundations of Computer Science, 1986., 27th Annual Symposium on*, pages 162–167. IEEE, 1986.

Part II

Collection of articles

Appendix A

Selected research articles

The appendix contains the 10 research articles (out of more than 30 published total) selected as the representatives of my research contributions into investigated areas. The complete articles are inserted into the corresponding appendix of the printed version of this thesis[1].

Article A: P. Švenda, M. Nemec, P. Sekan, R. Kvašňovský, D. Formánek, D. Komárek, and V. Matyáš. The million-key question – investigating the origins of RSA public keys. In *The 25th USENIX Security Symposium (USENIX Security '16)*, pages 893–910. USENIX, 2016

Article B: M. Nemec, M. Sys, P. Svenda, D. Klinec, and V. Matyas. The return of Coppersmith's attack: Practical factorization of widely used RSA moduli. In *24th ACM Conference on Computer and Communications Security (CCS'17)*, pages 1631–1648. ACM, 2017

Article C: V. Mavroudis, A. Cerulli, P. Svenda, D. Cvrcek, D. Klinec, and G. Danezis. A touch of evil: High-assurance cryptographic hardware from untrusted components. In *24th ACM Conference on Computer and Communications Security (CCS'17)*, pages 1583–1600. ACM, 2017

Article D: M. Nemec, D. Klinec, P. Svenda, P. Sekan, and V. Matyas. Measuring popularity of cryptographic libraries in Internet-wide scans. In *The 33rd Annual Computer Security Applications Conference (ACSAC'17)*. ACM, 2017

[1]The fulltexts of the articles are excluded from the publicly available electronic version of this text to avoid copyright violation.

Article E: P. Svenda, M. Ukrop, and V. Matyas. Towards cryptographic function distinguishers with evolutionary circuits. In *10th International Conference on Security and Cryptography (SECRYPT'13)*, pages 135–146. SciTePress, 2013

Article F: M. Sys, D. Klinec, and P. Svenda. The efficient randomness testing using Boolean functions. In *Proceedings of the 14th International Conference on Security and Cryptography (SECRYPT'17)*, pages 92–103. SCITEPRESS, 2017

Article G: J. Bouda, J. Krhovjak, V. Matyas, and P. Svenda. Towards true random number generation in mobile environments. In *Nordic Conference on Secure IT Systems*, pages 179–189. Springer, 2009

Article H: R. Ostadal, P. Svenda, and V. Matyas. Attackers in wireless sensor networks will be neither random nor jumping – secrecy amplification case. In *International Conference on Cryptology and Network Security (CANS'16)*. Springer, 2016

Article I: J. Kur, V. Matyas, and P. Svenda. Two improvements of random key predistribution for wireless sensor networks. In *International Conference on Security and Privacy in Communication Systems (SecureComm'12)*, pages 61–75. Springer, 2012

Article J: P. Švenda, L. Sekanina, and V. Matyáš. Evolutionary design of secrecy amplification protocols for wireless sensor networks. In *Second ACM Conference on Wireless Network Security (WISEC'09)*, pages 225–236, 2009