THE PUBLISHING HOUSE
OF THE ROMANIAN ACADEMY

PROCEEDINGS OF THE ROMANIAN ACADEMY, Series A,
Volume 14, Special Issue 2013, pp. 286–294

THE THREAT LANDSCAPE OF PKI: SYSTEM AND CRYPTOGRAPHIC SECURITY OF X.509, ALGORITHMS, AND THEIR IMPLEMENTATIONS

Blaine HEIN

NATO Communications and Information Agency
E-mail: blaine.hein@ncia.nato.int

With the unavoidable reliance on public key cryptography in modern communications and information systems (CIS) it is critical to maintain visibility into the threat landscape which can adversely impact the trust in public key infrastructure (PKI) implementations. This knowledge is useful as an input to a risk analysis process to determine whether current PKI practices are sufficient, and to determine when to migrate to new algorithms, key lengths, or procedures. This paper provides a discussion of the main attacks against PKI systems, both system and cryptographic in origin. This paper suggests appropriate methods to strengthen PKI systems against these attacks and provides references for additional reading on these attacks.

Key words: Asymmetric Cryptography, Diginotar, Flame, Malware, MITM, Public Key Infrastructur, Social Engineering, Stuxnet.

1. INTRODUCTION

In the last year we have seen a substantial increase in discussions touching on the malicious use of public key cryptography and on some high profile failures of PKI systems. This paper provides analysis and discussion on many of these events, and groups the threats against PKI in two general categories: system security and cryptographic security.

The topics covered under system security might just as easily be seen in an analysis of the threats against any CIS whether the system relied on cryptography or not. Many of these types of attacks have been occurring since before PKI was available as a commercial product. Social engineering, Trojan horses, and malware are terms which have been fully entrenched into our vocabulary.

The information covered under the topic of cryptographic security has a mathematical or algorithmic basis. Over the years the debates on cryptographic security have concentrated on when an algorithm would be broken, and whether it would be broken due to an increase in processing power, or by a new advancement in the field of mathematics.

This paper is not intended to provide a full breakdown of virus implementation, nor is it designed to turn the reader into a crypto mathematician. For those with aspirations in this direction, I have provided a detailed list of references for further reading. Nor will it go into details on the types of mischief an attacker can get into once he has breached a system. This is left to the imagination (or nightmares) of the reader.

This paper will provide recommendations of security mechanisms which will make the exploitation more difficult.

2. SYSTEM SECURITY

Included under the topic of system security, are a variety of topics ranging from the social engineering of a single certificate, all the way to the loss of control of multiple PKI infrastructures. It is important to remember that trust is a critical concept with PKI, and if that trust is convincingly undermined, the loss of trust can exceed the scope of the actual incident.

2.1. Diginotar

The Diginotar attack was a major news event in 2011 made public by the use of a rogue google.com certificate in a large man in the middle (MITM) attack targeting Iranian Internet users. For a complete report on the breach of the Certificate Authorities (CA) the final report on the Black Tulip incident [1] can be downloaded from Fox-IT in The Netherlands. This is an excellent example of System security failures. The following paragraphs provide a high level summary of the attack, and some recommended countermeasures to consider.

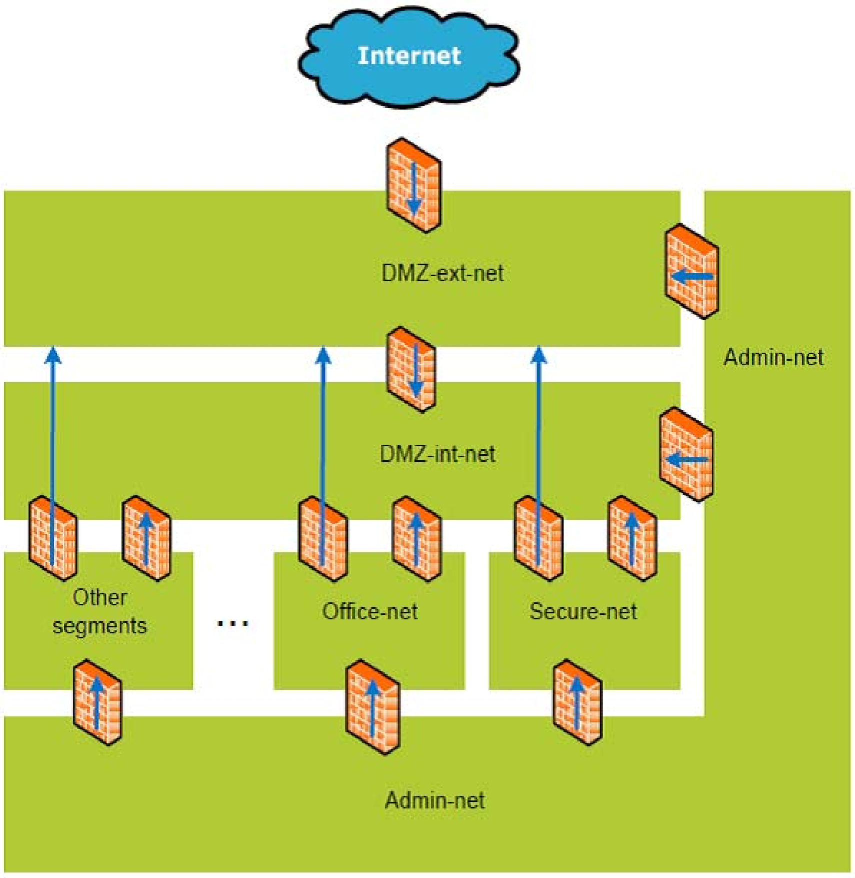

Fig. 1 – Diginotar Network Architecture [1, p. 18].

2.1a. Attack Progression

Reconnaissance and Scanning. Details are very limited within the Fox-IT report; however, it appears that the reconnaissance period for this attack was completed by 17 June 2011. From this point in time, the attack progressed steadily towards the end goal over the course of 3 weeks. Evidence in the report indicates that the attackers likely exploited a known vulnerability on an outdated software application and then used a user name and password discovered on the external web server to attempt connections into a database server.

Further evidence to support an undocumented reconnaissance period was found in the malicious code left on the systems by the attackers. These applications included hard coded IP addresses for internal and external systems, indicating that they were purpose modified for the Diginotar network [1, p. 48].

Exploiting Systems. First points of entry into the network were web servers located in the DMS [1, p. 23]. Next, database and other systems within the office network were compromised over the following 12 days. Tools to allow for the tunnelling of remote desktop protocol (RDP) sessions across non-standard ports (to allow for traversal of the firewalls) were discovered on systems in both the de-militarized zone (DMZ) and the office network.

The third step was the compromise of the secure internal network which included the Certificate Authority servers. Once access was gained to the CA servers, the attackers commenced with attempts to generate rogue SSL certificates. The first successful rogue certificate is thought to have been generated 8 days later.

Keeping Access and Covering the Tracks. Scripts were installed onto the web servers in the DMS to perform the function of a file system for uploading and downloading files.

Auditing subsystems were stopped in several cases to prevent the collection of forensic evidence while the attacker worked on the following steps of the attacks. Over 500 rogue certificates were issued. Some of the rogue certificates were issued with identical distinguished names (but possibly unique key pairs). Tools capable of exporting software certificates and private keys from the servers were found on multiple systems.

CA Database files were manipulated to hide the issuance of some certificates. As these files have integrity protection added by the CA, the easiest approach might have been to restore the database from a previous backup after issuing the certificates. This has the effect to return the CA to the state it was in prior to the issuance of those certificates, and as the integrity mechanisms on the backup file did not need to be tampered with, the files would appear to be intact. The biggest indications of attack are the lack of PKI audit logs during a period which shows attacks against the system from other audit mechanisms.

Certificate Authority implementation flaws exploited. Physical security at the Diginotar site was quite significant with the requirement to pass through multiple security zones and present visual, cryptographic, and biometric authentication data in order to gain physical access to the certificate authorities.

In the end, none of these security mechanisms came to bear to prevent the network based attack. The report [1] does not go into specific details regarding the activities performed on the Certificate Authorities to generate rogue certificates.

The flaws discussed below do not necessarily represent deficiencies within the PKI products. Instead, they reflect choices for the implementation of security mechanisms or configuration options.

Based on the level of details provided within the report, the following conclusions have been drawn.

2.1b. Countermeasures

Transitive Trust. Appendix IX of the report [1, p. 97] shows that five of the eight CAs included a common trust relationship allowing common administrative access between these systems. As soon as administrative privilege escalation was achieved on any system within this domain, all systems with this transitive trust relationship were effectively compromised. Removing transitive trust relationships from between the independent CAs would have added additional work load to gain access to the separate CAs.

Ubiquitous Strong Authentication. Authentication is composed of up to three distinct factors: What you (and only you) know, what you (and only you) have, and what you (and only you) are. Within this context, strong authentication implements at least two of the three factors. The end to end enforcement of strong authentication is critical for the infrastructure components of PKI due to the wide reaching impact of system compromise or suspected system compromise.

The [Black Tulip] report indicates that a smart card is required to log in directly to the CA software. However, there is no reference to strong authentication for the Diginotar Subscription Registration Production Interface (DARPI) application which performs part of the registration authority (RA) functionality.

Additionally, the mandatory use of strong authentication for all system login would decrease the attack surface. Finally, to mitigate against keystroke logging, implementing a smart card reader with an integrated keypad would prevent the smart card PIN from entering into the general system memory.

Separation of roles. PKI implementations normally include the concept of least privilege. Industry best practice such as the Certificate Issuing and Management Components (CIMC) protection profile (PP) requires that there are at least 3 orthogonal roles required to operate the certificate authority. In addition to the CA software, the Networked Hardware Security modules (nCipher netHSM) also implement separate strong authentication.

The automated issuance of CRL's requires that the netHSM be activated during the functioning period of the Certificate Authority. There should, however, be a separate strong authentication required between the Registration Authority (RA) and the CA to mitigate the risk that the active netHSM could be used to generate rogue certificates.

Use of In-house applications. Three in house applications were used to perform the registration authority and card personalization functions. No information is provided in this report as to the independent verification and validation processes that these applications may have been subjected to, or the security mechanisms that they implemented.

Protection of Audit Data. Storage of audit data on the same server as the PKI provides an easy path to covering the trail of the attacker. Deletion of the audit data, combined with restoring a previous backup of the CA database is a fast way to hide the existence of rogue certificates.

If the attacker had more completely erased his actions within the CA systems, (and not generated over 300 certificates which remained in the database) suspicion might have been limited to the compromise of a single CA. The attack could possibly have been repeated by substituting a new google certificate issued by another CA.

Pushing audit data to a separate system with further separation of roles for access control would add an additional layer of security in addition to providing log centralisation and normalisation functionality.

White listing Revocation Data. Certificate Revocation Lists (CRL)s implement a mechanism for blacklisting certificates when they are known to be compromised. OCSP provides an option to implement either blacklisting or white listing. To implement white listing, the CA must provide an explicit status response for all issued certificates. Any certificate which does not match these existing responses is deemed to be compromised (or fraudulent).

2.2. Social Engineering of Certificates

It has already been more than a decade since VeriSign fell victim to a social engineering attack in 2001, and issued two fraudulent code signing certificates with distinguished names linked to Microsoft Corporation. At the time, there was substantial discussion regarding how to remove these certificates and on the ability of Microsoft software to check revocation lists [2]. The Microsoft knowledge base article [3] outlines the mechanisms that were put in place to counter this threat. These certificates still exist in the Microsoft untrusted certificates store. Today, the largest issue with SSL implementations is the difficulty in determining whether the issuer of the SSL certificate is authoritative to issue that certificate. This conundrum exists because there is no correlation between DNS names and PKI Certificate Authorities, and there will likely never be one. To minimize this threat, minimize the number of root certificates trusted by your critical infrastructure components. Ensure this process is well tested before roll out to avoid a self-denial of service.

2.3. Theft of Certificates and Private Keys

Software implementations of cryptographic modules are vulnerable to extraction of private keys from their hosts. This is even more of a risk if these keys are stored on general purpose computing environments running commercial operating systems.

Readily available tools such as Mimikatz [4] have the ability to export keys from Microsoft security stores, even if the keys are marked as non-exportable. With these tools, the compromise of a server must be equated with the compromise of all private keys stored in that system.

Even when the cryptographic module implements a separate password to encrypt the private key, it is still vulnerable to brute force offline password guessing.

Hardware cryptographic modules, separation of roles, and two factor authentications must all be provided to protect critical certificates and keys. Code signing certificates are often overlooked in this scenario.

2.4. Compromise of Registration Authority

The attack against Comodo in the spring of 2011 appears to be the precursor to the Diginotar attack. In this case, the breach was to a registration authority (RA) which allowed the attacker to request and receive nine fraudulent certificates. Microsoft has published instructions to mitigate against these certificates at [5] and an online article [6] provides more background.

The same set of security mechanisms proposed to mitigate the Diginotar attack would likely have hampered the success in this attack as well.

2.5. Supply Chain Security

Security is only as strong as its weakest link. Sometimes even that statement is optimistic. The breach of servers at RSA through an advanced persistent threat (APT) [7] attack resulted in the compromise of a large number of SecurID tokens. However, RSA was not the end target in this attack. The compromised information was used to attack 3 US defence contractors [8]. While SecurID is not a PKI based security mechanism, what would the result be if you outsourced your PKI to a vendor who is subsequently breached? Ensure that your information security management system includes linkages to all aspects of your supply chain. What security mechanisms have your consultants and contractors implemented for IT equipment they connect to your network? Will the compromise of one of your suppliers or business partners result in a breach to your network because of trust relationships?

2.6. Unintentional Consequences of Security Implementations

2.6a. SSL Proxy Implementations

SSL proxies are appliances configured to decrypt SSL connections initiating from within a corporate environment outbound to commercial web services. They are implemented generally to support the implementation of malicious code scanning and data loss protection at the boundary of the corporate network.

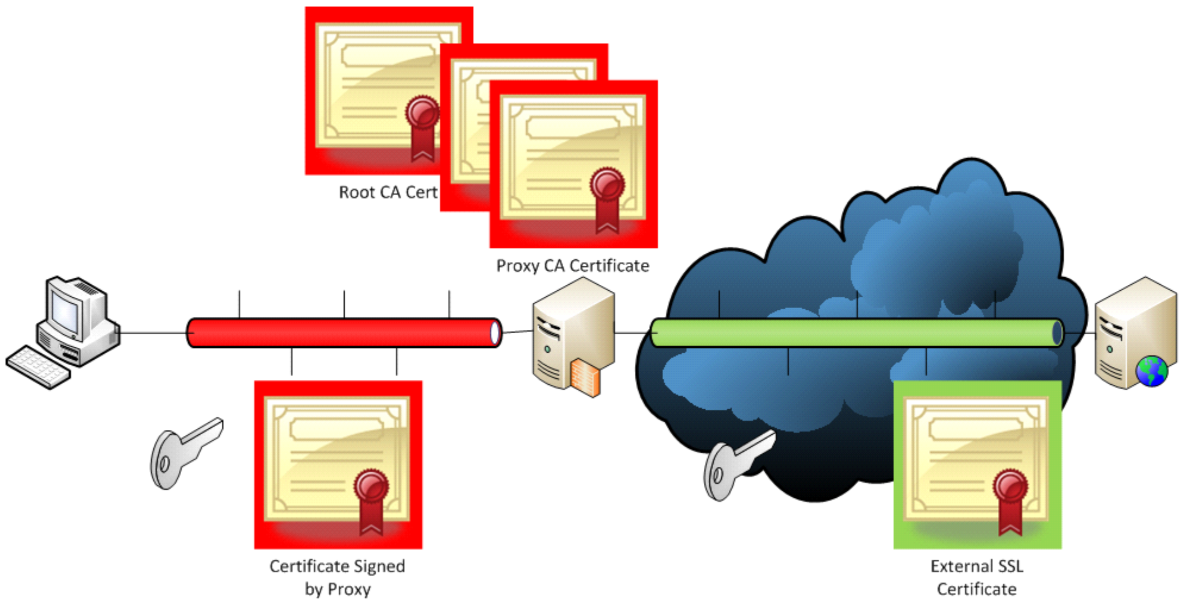

Fig. 2 – SSL Proxy Configuration.

Effectively they implement a man in the middle (MITM) attack. SSL Proxy certificates are functionally the same as intermediate CA certificates. They allow the SSL proxy to intercept the server certificate, replace the key with one known by the proxy, and then resign the certificate. This resigned certificate is presented to, and trusted by the browser. The proxy generates a second SSL connection outbound to the original web server.

Their use has been relatively common for several years, but they have been slowly gathering negative publicity since early 2012 when Trustwave admitted issuing an SSL proxy certificate [9]. Mozilla published a series of letters to participants of their trusted root certificate program offering a period of amnesty instead of revoking the Trustwave Root CA Certificates [10] and an opportunity for them to revoke the “SSL Spying Certificates” [11].

2.6b. TurkTrust

More recently, in what can be better described as a series of unfortunate events instead of malicious activity, the certificate services provider TurkTrust was discovered to have issued intermediate CA certificates to two subscribers. Normal SSL certificates had been requested. One of the subscribers immediately requested the delivered certificate be revoked. The second certificate was used to establish an SSL proxy in late 2012 [12]. Due to this incident the CA certificates have been removed from the browser and TurkTrust has been suspended from the trusted CA certificate program by Mozilla [13].

The chrome browser implements security mechanisms beyond those specified in the X.509 series of standards. This browser implements white listing of SSL public keys for the google.com domain. As TurkTrust was not one of the providers of valid google.com certificates, the chrome browser flagged the TurkTrust certificate as invalid [14].

Ultimately the public exposure of a proxy google.com certificate would put any CA which issued the proxy certificate at risk of blacklisting in several browsers.

3. CRYPTO SECURITY

3.1. Public Key Enabled Malware

The following is not intended to be a detailed analysis of the listed malware. The intent is to identify the manner in which the malware obtained a valid digital signature. The lessons from these malware samples are to limit your trust where practical on critical infrastructure, and to not become the source for signatures on the next generations of malware.

3.1a. Stuxnet

The Stuxnet malware became public in July 2010. Lost in the media coverage speculating on the purpose and authors of this malware was the digital signature which is one of the stealth enabling features. The malware includes two files which are installed as signed drivers by the operating system. The signature allows them to be installed without generating a prompt to authorize the installation. The initial drivers are signed with certificates and keys stolen from Realtek and JMicron [15]. More information regarding the pervasiveness of digitally signed malware is provided at [16]. Finally the Symantec analysis of Stuxnet is provided at [17] which interestingly state the risk level as low.

3.1b. Duqu

The Duqu virus, while carrying a substantially different payload to the Stuxnet virus shares several similarities including the digitally signed driver with a certificate from C-Media. The Laboratory of Cryptography and system Security (CrySyS) at the Budapest University of Technology and Economics has published a detailed analysis of Duqu [18]. A further article quoting the McAfee assessment is published at [19] and again an analysis by Symantec at [20].

3.1c. Flame

The Flame malware took an alternative approach to signing malware. Instead of stealing Certificates and keys, Flame is reported to have implemented a MD5 chosen prefix collision attack against a Microsoft certificate which supported code signing [21]. In response to this event Microsoft has moved the two relevant certificates to the untrusted certificates container [22].

3.1d. 5000 More signed by Adobe

Again in 2012, a compromised server holding a code signing certificate was substantially abused to sign more than 5000 pieces of malware. When initially disclosed, [23]. Adobe downplayed the attack as only enabling the signing of two pieces of malware. Further analysis by other security experts confirmed the scope and highlighted the technical knowledge required to implement the attack [24].

3.2. Attacks against Algorithms

3.2a. Timing Attacks

Timing attacks were already well documented in the mid 1990's at a time when PKI was still in its infancy. General attacks were documented on several discrete logarithmic algorithms including RSA, DSA, and Diffie-Hellman. By 2003 research had matured to the point where controlled implementations were being tested on academic networks against Open-SSL implementations [25]. One implementation of this attack works by measuring the time taken to complete known mathematical operations. The attacker measures the timing differences caused when different information is submitted to the SSL Server. The exact mechanism here is extra reductions during Montgomery reduction for a manipulated input by the attacker. At the time, the technique was being successfully implemented (albeit in a very controlled environment) for key lengths up to 1024 bits.

Blinding, an acceptably efficient mitigation for this attack was already envisioned in much earlier papers [26], but for implementations such as Open-SSL, this functionality was still not enabled by default in 2003. Blinding is a process where an additional calculation with random data is added to obfuscate the timing differences.

3.2b. Short key lengths

In the fall of 2011 compromised certificates based on RSA-512 were detected from a Malaysian Certificate Authority [27]. The commonly agreed attack vector was the factoring of the keys. Microsoft made a tool available to remove trust in RSA-512 certificates in 2012 [28]. The solution here is obvious: follow the attack trends to ensure that you are able to upgrade key lengths before you are forced into an emergency re-key by an attacker factoring keys.

3.2c. Hash collisions

MD5 based hash collisions have been shown to be an implemented reality by malware such as Flame, but the current discussions still speculate on when the theoretical collisions against SHA-1 will arrive. Schneier [29] calculates that collisions will become affordable by 2018. As we have seen, a single chosen plain text hash collision can result in a substantial amount of damage. Solution here is the same as for short key lengths.

3.3. Attacks against Cryptographic Implementations

3.3a. Random Number Generator (RNG)

Over the years there have been many flawed implementations of random number generators, including implementations by Microsoft and Netscape to name just a few. More recently, researchers have uncovered a significant number of active RSA keys which are either default keys duplicate keys, or contain a shared prime factor [30]. Some researchers immediately came to the conclusion that RSA was broken [31], but there are several other causes for weak keys.

Default keys exist due to poor configuration practices where keys included in the device from the factory are not replaced prior to in-servicing the equipment.

Keys that are duplicates or that share a prime factor result from random number generators with low entropy [32]. Duplicate key pairs likely are the result of both primes being generated with poor entropy. The last case (shared prime factor) likely results from the first prime number resulting from poor entropy in the RNG, but the second prime is generated after the RNG is re-seeded to improve entropy. For completeness, the bugs discovered in the Microsoft and Netscape RNG implementations are [33] and [34]. Formal security evaluations are the best protection against flawed RNG implementations.

4. CONCLUSIONS

This paper has only scratched the surface of the majority of these threats, but it provides plenty of links for further analysis. The threats and the suggested countermeasures provide a broad threat landscape to use as an input to a risk assessment process. The risk assessment process is critical in determining the security services and security mechanisms required to secure PKI implementations.

The proliferation of signed malware is here to stay. This paper provides one final lesson; learn the lessons of others before you learn them from scratch by yourself.

6. DISCLAIMER

Any opinions expressed herein do not necessarily reflect the views of the NCI Agency, NATO and the NATO nations, but remains solely those of the author.

REFERENCES

1. Fox-IT BV, *Black Tulip Report of the investigation into the DigiNotar Certificate Authority breach*, 13 August 2012, Online, Available: http://www.rijksoverheid.nl/documenten-en-publicaties/rapporten/2012/08/13/black-tulip-update.html.
2. G. L. Guerin, *Microsoft, Verisign, and Certificate Revocation*, 13 May 2001, Online. Available: http://www.amug.org/~glguerin/opinion/revocation.html.
3. Microsoft, *MS01-017: Erroneous VeriSign-Issued Digital Certificates Pose Spoofing Hazard*, 21 February 2007, Online. Available: http://support.microsoft.com/kb/293818.
4. B. Delpy, *mimikatz*, Online. Available:4. http://blog.gentilkiwi.com/mimikatz. [Accessed 2013].
5. Microsoft, *Microsoft Security Advisory (2524375)*, 6 July 2011, Online, Available: http://technet.microsoft.com/en-us/security/advisory/2524375.
6. G. Keizer, *Solo Iranian hacker takes credit for Comodo certificate attack*, 27 March 2011, Online, Available: http://www.computerworld.com/s/article/9215245/Solo_Iranian_hacker_takes_credit_for_Comodo_certificate_attack.
7. RSA FraudAction Research Labs, *Anatomy of an Attack*, 1 April 2011, Online, Available: http://blogs.rsa.com/anatomy-of-an-attack/.
8. K. Zetter, *RSA Agrees to Replace Security Tokens After Admitting Compromise*, Wired, 6 July 2011, Online, Available: http://www.wired.com/threatlevel/2011/06/rsa-replaces-securid-tokens/.
9. L. Constantin, *Trustwave Admits Issuing Man-in-the-Middle digital Certificate, Mozilla Debates Punishment*, CIO, 8 Febrary 2012, Online, Available: http://www.cio.com/article/699762/Trustwave_Admits_Issuing_Man_in_the_Middle_Digital_ Certificate_Mozilla_Debates_Punishment.
10. L. Constantin, *Mozilla Will Ask All Certificate Authorities to Revoke SSL-Spying Certificates*, CIO, 14 February 2012, [Online]. Available: http://www.cio.com/article/700490/Mozilla_Gives_CAs_a_Chance_to_Come_Clean_About_Certificate_Policy_ Violations.
11. L. Constantin, *Mozilla Gives CAs a Chance to Come Clean About Certificate Policy Violations*, CIO, 20 February 2012, Online, Available: http://www.cio.com/article/700161/Mozilla_Will_CAs_a_Chance_to_Come_Clean_About_Certificate_Policy_ Violations.
12. P. Ducklin, *The TURKTRUST SSL certificate fiasco - what really happened, and what happens next?*, Sophos, 8 January 3012, Online, Available: http://nakedsecurity.sophos.com/2013/01/08/the-turktrust-ssl-certificate-fiasco-what-happened-and-what-happens-next/.
13. M. Coates, *Revoking Trust in Two TurkTrust Certificates*, Mozilla, 3 January 2013, Online, Available: http://blog.mozilla.org/security/2013/01/03/revoking-trust-in-two-turktrust-certficates/.
14. A. Langley, *Imperial Violet: Public key pinning*, 4 May 2011, Online, Available: http://www.imperialviolet.org/2011/05/04/pinning.html.
15. M. Russinovich, *Analyzing a Stuxnet Infection with the Sysinternals Tools, Part 1*, Microsoft, 30 March 2011, Online, Available: http://blogs.technet.com/b/markrussinovich/archive/2011/03/30/3416253.aspx.
16. L. Constantin, *Digitallly Signed Malware Is Increasingly Prevalent, Researchers Say*, IDG News Service, 15 March 2012, Online, Available: http://www.pcworld.com/article/251925/digitally_signed_malware_is_increasingly_prevalent_ researchers_say.html.
17. Symantec, *W32.Stuxnet*, 26 February 2013, Online, Available: http://www.symantec.com/security_response/writeup.jsp?docid=2010-071400-3123-99.
18. Laboratory of Cryptography and System Security (CrySyS), "*Duqu: A Stuxnet-like malwere found in the wild*," 11 October 2011, Online, Available: http://www.crysys.hu/publications/files/bencsathPBF11duqu.pdf.
19. T. Espiner, *McAfee: Why Duqu is a big deal*, ZDNet, 26 October 2011, Online, Available: http://www.zdnet.com/mcafee-why-duqu-is-a-big-deal-3040094263/.
20. Symantec, *W32.Duqu: The precursos to the next Stuxnet*, 23 November 2011, Online, Available: http://www.symantec.com/ content/en/us/enterprise/media/security_response/whitepapers/w32_duqu_the_precursor_to_the_next_stuxnet.pdf.
21. M. Stevens, *Technical Background of the Flame Collision Attack*, Centrum Wiskunde & Informatica, 7 June 2012, Online, Available: http://www.cwi.nl/news/2012/cwi-cryptanalist-discovers-new-cryptographic-attack-variant-in-flame-spy-malware.
22. Microsoft, *Microsoft Security Advisory (2718704) Unauthorized digital Certificates could Allow Spoofing*, 3 June 2012, Online, Available: http://technet.microsoft.com/en-us/security/advisory/2718704.

23. D. Goodin, *Adobe to revoke crypto key abused to sign malware apps (corrected)*, Ars Technica, 28 September 2012, Online, Available: http://arstechnica.com/security/2012/09/adobe-to-revoke-crypto-key-abused-to-sign-5000-malware-apps/.
24. R. NarinE, *Adobe code signing infrastructure hacked by 'sophisticated threat actors'*, ZDNet, 27 September 2012, Online, Available: http://www.zdnet.com/adobe-code-signing-infrastructure-hacked-by-sophisticated-threat-actors- 7000004925/ ?s_cid=e539.
25. D. Brumley and D. Boneh, *Remote Timing Attacks are Practical*, in USENIX, Washington, 2003.
26. P. C. Kocher, *Timing Attacks on Implementations of Diffie-Hellman, RSA, DSS, and Other Systems*, in Advances in Cryptology - CRYPTO '96, 16th Annual International Cryptology Conference, Proceedings, Santa Barbara, 1996.
27. M. Sandee, *RSA 512 Certificates abused in the wild*, 21 11 2011, Online, Available: https://www.fox-it.com/en/blog/rsa-512-certificates-abused-in-the-wild/.
28. Microsoft, *Microsoft Security Advisory (2661254) Update For Minimum Certificate Key Length*, 14 August 2012, Online, Available: http://technet.microsoft.com/en-us/security/advisory/2661254.
29. B. Schneier, *When Will We See Collisions for SHA-1*, 5 October 2012, Online, Available: http://www.schneier.com/blog/archives/2012/10/when_will_we_se.html.
30. D. Auerbach and P. Eckersley, *Researchers Use EFF's SSL Observatory To Discover Widespread Cryptographic Vulnerabilities*, Electronic Frontier Foundation, 14 February 2012, Online, Available: https://www.eff.org/rng-bug.
31. A. K. Lenstra, J. P. Hughes, M. Augier, J. W. Bos, T. Kleinjung and C. Wachter, *Ron was wrong, Whit is right*, 14 February 2012, Online, Available: http://eprint.iacr.org/2012/064.pdf.
32. N. Heninger, *New research: There's no need to panic over factorable keys - Just mind your Ps and Qs*, Online, Available: https://freedom-to-tinker.com/blog/nadiah/new-research-theres-no-need-panic-over-factorable-keys-just-mind-your-ps-and-qs/.
33. L. Dorrendorf, Z. Gutterman and B. Pinkas, *Cryptanalysis of the Random Number Generator of the Windows Operating System*, 4 November 2007, Online, Available: http://eprint.iacr.org/2007/419.pdf.
34. I. Goldberg and D. Wagner, *Randomness and the Netscape Browser*, January 1996, Online, Available: http://www.cs.berkeley.edu/~daw/papers/ddj-netscape.html.

Received May 20, 2013