technicolor

THE SECURITY NEWSLETTER #22

WINTER 2013

THE SECURITY NEWSLETTER #22

In this Issue

Published Quarterly By
Technicolor Security & Content Protection Laboratories

Technical Editor: Eric Diehl

Contributors: Gwenael Doërr
Alain Durand
Marc Eluard
Raphael Gelloz
Olivier Heen
Marc Joye
Mohamed Karroumi
Benoit Libert
Christoph Neumann

EVP: Gary Donnan
CSO: Rachel Orand

Subscribe to the newsletter:
security.newsletter(at)technicolor.com

Report vulnerability:
security(at)technicolor.com

EDITORIAL

It is often claimed that our digital world is generating a digital divide; some people will have full access to new benefits while others will be left far behind. It seems there is a similar divide with piracy: high-end versus low-cost.

"Low-cost" piracy is the day-to-day piracy that we all encounter. This type of piracy uses common malware, phishing, SPAMs, and forged sites. It is usually not extremely sophisticated but highly automated. The attacker looks for easy, fast sources of revenue, scanning randomly to find the most vulnerable targets. Breaking the user's security must take no effort to maximize the attacker's revenue. It is mass piracy. We are all potential targets. Fighting it should not be too difficult; good practices such as up-to-date antivirus software and, most of all, education are effective shields. This piracy makes headlines and gets a lot of attention.

On the other hand, there is "high-end" piracy, sometimes called Advanced Persistent Threat (APT), which aims at precise targets. The attacker seeks to reach a goal at any price. He uses sophisticated custom tools that exploit the latest zero day exploits. Humans, not robots, drive it. The targets are government institutions and companies. Business intelligence is often the driver. Fighting it requires a high level of expertise. APT is stealthy and does not get a lot of attention from the press. Nevertheless, it is real.

There is a similar divide in content piracy. Early upstream piracy, such as pre-theatrical, requires technical and social engineering skills, as well as some form of organization. The cost of an upstream leak is high for the attacker. Downstream, ant piracy, which distributes bootleg copies or rip discs, is far simpler. The cost of these late leaks is low for the attacker (if not null through sharing sites).

In the middle, researchers and passionate hobbyists help security evolve by creating new solutions, or responsibly disclosing vulnerabilities. In this issue, we are proud to invite one of them: Joan Daemen. A few years ago, he co-authored AES, the new NIST-approved standard encryption algorithm. This year, he is co-author of the recently NIST-approved hash algorithm, SHA3.

E. DIEHL
Technical Editor

THE SECURITY
NEWSLETTER
#22

BE OUR GUEST

Joan Daemen
ST Microelectronics, Brussels

In 2000, Rijndael was adopted by NIST as the standard for symmetric encryption. This year you repeat this success with hash functions: Keccak was selected as the winner of the NIST hash function competition. Can you tell us more about this competition?

The competition started some time ago, in 2007. There were 64 submissions (of which 51 were considered valid by NIST). Then, like the AES competition, the list was shrunk to 14 after a first evaluation phase and then to five finalists after a second evaluation phase.

How did you come up with the design of Keccak?

This is a long story. I had a design called Panama; it is a hash function and stream cipher, similarly to Keccak. It was based on my PhD thesis and became public in 1997. Unfortunately, it was broken in 2002 by researchers from KULeuven. But I still believed that the grounds were solid. So together with my colleagues Gilles Van Assche and Michael Peeters (at that time with ST Microelectronics), we started working on trying to tweak it and then abandoned it. Then some years later, as there were rumors for a SHA-3 competition, we decided to pick up Panama again. Guido Bertoni (also with ST Microelectronics) expressed interest in collaborating. We started working on a successor to Panama, made some progress, and came up with a better design called RadioGatún. The problem with RadioGatún was how to formulate security claims such as collision resistance because of its variable output length. The idea was to define a function behaving like a random oracle except that it could exhibit internal collisions. That led to the sponge construction, which was initially meant only as a reference security model. While trying to improve RadioGatún, we thought why not design a permutation and use it in a sponge construction. Our efforts led to the Keccak-F permutation that is used in Keccak. All in all, it took us four-to-five years to design Keccak.

Why did you choose the name "Keccak"?

We came up with it during an ESC workshop (Early Symmetric Cryptography) that is held every two years in January in Luxemburg. We were present and that is basically where we decided to switch from the RadioGatún structure to the sponge function of Keccak. I had this music on my PC, a ritual dance from Indonesia (which is called the Kecak dance [1]). It makes violent noise and is pretty funny to hear. The idea was that the function should work like this dance: chop everything into pieces, turn it around and chop it again. We decided to keep the name but we wrote it Keccak (with 2 c's) to make it searchable on the web.

1 *Kecak Dance in Bali*, 2007, http://www.youtube.com/watch?v=WSTIcTWTNrY.

THE SECURITY
NEWSLETTER
#22

SHA-1 (but also MD5) and SHA-2 are still widely deployed. When do you think Keccak (SHA-3) will be used?

I don't know. NIST has already announced that SHA-2 has no real security problems, so there is no big urgency to migrate from SHA-2 to SHA-3. But if you ask my opinion, I would of course recommend to migrate to SHA-3 as soon as the standard is available. Keccak is also more than just a hash function. It offers additional features such as lending more naturally to stream encryption or authenticated encryption. Furthermore, all operations provided by a block-cipher can be readily handled in sponge mode with a permutation, except a block encryption such as ECB or CBC. It can be seen as an alternative cryptography, based not on block cipher but on permutation. Keccak presents the advantage of being very fast and compact in hardware. Moreover, the round function of Keccak has an algebraic degree of only 2, which leads to efficient protection against side-channel attacks using masking schemes.

How does Keccak compare to SHA-2 in terms of security?

It is always difficult to say something about security. I think we established a wide safety margin. If we look at the "theoretical attacks" that have been published (or more precisely the distinguishers against the Keccak-F permutation), it turns out that the full 24-round version has a structural property not present in a random permutation, that can be detected with an attack with complexity 2^{1575}, meaning you need that many input/output pairs. So it is really theoretical. Looking at more practically oriented attacks, there are collision attacks on reduced versions up to 4 rounds (that is, 4 out of 24). That is a good safety margin. I think we took double the number of rounds we really need. I don't know what is the safety margin of SHA-2. The problem with SHA-2 is that, along with many people working in cryptanalysis, we believe that SHA-2 provides security partly based on obscurity, because it is very hard to describe and evaluate its structure and characteristics. In that sense, the design of Keccak is cleaner. Of course this is not a neutral statement. But I think that Keccak has a better safety margin.

What is your next challenge?

Now we are defining a number of modes and variants of sponge/Keccak for keyed applications with fewer rounds. We use the fact that attacks against keyed modes are much harder than attacks against unkeyed modes. For example, the hash function Panama was badly broken but its companion stream cipher is not broken at all. We are also working on side-channel resistant implementations of Keccak.

Thank you!

J. DAEMEN (ST Microelectronics, Brussels)
Interview by M. JOYE

THE NEWS

PS3 hacked again

In 2006, the initial version of Sony PlayStation 3 (PS3) allowed the execution of customer-developed software. In August 2009, with the advent of the newer PS3 slim, this was no longer possible. Since this date, hobbyists have attempted to jailbreak the device.

On December 2010, Georges Holtz, aka GeoHot, disclosed the private key used to sign PS3's firmware, the so-called LV1 key.[2] He guessed the key by exploiting an error of implementation in the signing software. Sony recovered from the attack.

Nevertheless, LV1 was not the ultimate key; it "only" protected the firmware. In October 2012, hackers by the nickname of "The Three Musketeers" went a step further.[3] They released the private key used for the boot loader, called LV0. With this key, it is possible to install any arbitrary software on the console. Unfortunately, the Musketeers did not disclose how they discovered the private key.

As the key of the boot loader is usually the ultimate key, it is difficult to forecast how Sony may recover while keeping compatibility with the legacy base.

E. DIEHL

Hugo awards: reality or science fiction?

If you are a science-fiction fan, you know about the Hugo awards, which reward the best science-fiction works (movies, novels, novellas, etc.) of the year.

The 2012 edition was held in early September in Chicago and was at the same time available via video streaming on the Internet. Everything was going smoothly when suddenly Internet viewers got the following message: "Worldcon banned due to copyright infringement." This happened when a reward was given for a *Doctor Who* script. As some clips of the corresponding episode were performed as an illustration, Vobile's fingerprint-based copyright infringement tool detected the copyrighted material and signaled it to Ustream's streaming tool, which automatically stopped the Internet broadcast. But Worldcon had authorization from the BBC to use the clips. Despite numerous complaints from the Internet community, it was not possible to resume the broadcast before the end of the ceremony. The incident shows how stupid, badly implemented robots can rule the world (a favorite sci-fi theme).

A. DURAND

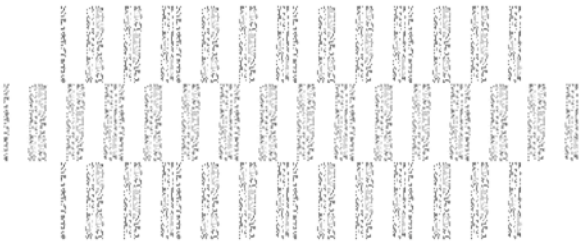
2 Jonathan Fildes, 'Hackers uncover secret PS3 keys', *BBC*, January 6, 2011, sec. Technology, http://www.bbc.co.uk/news/technology-12116051.
3 The Three Musketeers, '#5102182', *Pastie*, October 22, 2012, http://pastie.org/5102182.

THE SECURITY
NEWSLETTER
#22

WoW: a World of Watermarks?

World of Warcraft, aka. *WoW*, is a massively multiplayer online role-playing game (MMORPG) released in November 2004 by Blizzard Entertainment. Players log on a realm and control a character avatar within a game world. They can explore the landscape, fight various monsters, complete quests, polish their non-fighting skills, and interact with non-player characters or other players. With over 10 million subscribers as of the Fall of 2012, *World of Warcraft* is currently the world's most popular MMORPG ever.

In September 2012, a controversy erupted on the *Owned Core* forums.[4] A user reported a rather intriguing repetitive pattern that can be revealed by applying an extreme sharpening filter onto JPEG screenshots taken from the game (cf. below). It quickly appeared that these repetitive patterns were different for each user and that they could be a watermark. With a little bit of collaboration, the forum users found that the embedded watermarks consist of approximately 88 bytes of unencrypted data, encoding (i) the player's account name, (ii) the full information of the played realm including its IP address, and (iii) a timestamp. Apparently, these traitor-tracing watermarks could trace back as far as the patch 2.1.0 in 2007 that introduced JPEG screenshots for the first time.



Most likely, Blizzard Entertainment introduced these invisible watermarks to combat rogue private servers that break *WoW*'s End User License Agreement (EULA) and Terms of Service (ToS) in several ways. Indeed, when using private rogue servers, players do not have to pay the monthly subscription to Blizzard Entertainment. The $88M judgment against Scapegaming in 2010[5] explains very well how seriously Blizzard takes this issue.

The core of the controversy on Owned Core did not focus on yet another tracking method, which is not even mentioned in WoW's ToS. Users expressed real concern that this sensitive information was transmitted over an unsafe communication channel and that malicious hackers could exploit it. After being awarded the 2012 Big Brother Award in the Consumer Protection category[6], Blizzard Entertainment was probably not looking for such extra publicity.

G.DOËRR

The danger of Steam

Security researchers from the vulnerability research and consultancy firm ReVuln[7] found a way to remotely exploit local bugs in Steam and its applications.

The Steam platform, built and operated by Valve Corporation, is very popular in the market for digital distribution of software (mainly games, but also other types of application). Within a user-friendly environment, users manage their software collection and participate in Steam's social network (Steam Community). Gamers can install and play a game on any device that can run it. The Steam Cloud enables backup of game configurations and saves. In order to connect to Steam, users need to install the Steam client, which runs on Windows, Mac OS X, Linux, iOS, Android and PS3.

The installation of the Steam client registers the `steam://` URL protocol handler. Traditionally, a URL handler enables launch of an external application when a link beginning by this handler is clicked on or typed in the address bar of a browser. Some handlers such as `http://` are internal to the browser while `ftp://` can be used to launch a ftp client. Browsers are not the only applications that support such links: mail clients, multimedia readers and office suites do too. The `steam://` URL handler is used to send commands to the Steam client. Such commands include the possibility of installing, launching (with parameters) or updating games. [8, 9]

4 Sendatsu, 'Looking inside your screenshots', *Owned core*, 12sep 12, http://www.ownedcore.com/forums/world-of-warcraft/world-of-warcraft-general/375573-looking-inside-your-screenshots.html.

5 Ben Kuchera, 'The $88 million server: private WoW server op loses big', *Ars Technica*, August 17, 2010, http://arstechnica.com/gaming/news/2010/08/the-88-million-server-private-wow-server-op-loses-big.ars.

6 'Category Consumer Protection', *BigBrotherAwards*, August 9, 2012, https://www.bigbrotherawards.de/2012/.cons.

7 Dennis Fisher, 'ReVuln Emerges as New Player in Vulnerability Sales Market', threatpost, October 12, 2012, https://threatpost.com/en_us/blogs/revuln-emerges-new-player-vulnerability-sales-market-101212.

8 'Steam browser protocol', *Valve Developer Community*, https://developer.valvesoftware.com/wiki/Steam_browser_protocol.

9 'Command Line Options', *Valve Developer Community*, https://developer.valvesoftware.com/wiki/Command_Line_Options.

THE SECURITY
NEWSLETTER
#22

A user can be tricked to click on a malicious `steam://` link or its browser can be forced to redirect silently to such an URL. Not all applications behave in the same way when this occurs: some do not ask the user for confirmation; others do, but truncate the URL. Users may also be tempted to disable warnings in their browser. The researchers provided several examples of vulnerabilities in popular games and in the Steam client itself that can be exploited using this attack vector.[10]

Security-conscious users can disable the `steam://` URL handler or use a browser that prompts them before following `steam://` link. ReVuln also advises Steam to disable the passing of parameters to games.

Two lessons can be learned from this security news. First, we should not assume that external attackers cannot reach local systems or services. Secondly, we should be very careful when interacting with third party applications, especially web browsers.

R. GELLOZ

DRM for 3D printers

DRM (Digital Rights Management) is deployed in numerous areas, including music, movies and video games. Since mid-October 2012, a new application domain emerged for DRM: 3D printers. In fact, a patent, "Manufacturing control system",[11] was granted to the company *Intellectual Ventures*. Run by former Microsoft CTO Nathan Myhrvold, the company is known as a "patent-hoarding" company (according to Shane Robison, CTO of HP).

This patent describes a device able to produce a 3D object but that must authenticate it to a server to obtain the rights to print. Thus, it may not be possible to print a new fancy basket after retrieving 3D drawings (so-called *physibles*) on the Internet. In very simple terms, a fingerprint of the object to be printed is sent to the server. If the object is known by the database, the printer will apply some rules. If the object is unknown, the 3D printer will apply other rules.

Obviously, in domains where DRM has been used previously, there was counterfeiting and a lot of unauthorized copying of copyrighted content. 3D printing will not be different. With the advent of 3D printers for less than $500, copyright litigations will flourish.[12]

3D printing can benefit society. For example, a young girl suffering from arthrogryposis was helped by researchers at a Delaware hospital. They drew a 3D model of an exoskeleton, sent her the files and she printed it in 3D.[13] With that exoskeleton, she can move her members again.

With this very broad patent, Intellectual Ventures covers many cases, regardless of their societal impact or copyright issues. Nevertheless, DRM for 3D printers has a long way before being established and well balanced.

M.ELUARD, E. DIEHL

Credit : Objet

10 ReVuln, *ReVuln - Steam Browser Protocol Insecurity*, 2012, http://vimeo.com/51438866.
11 Edward Jung et al., 'United States Patent: 8286236 - Manufacturing control system', October 9, 2012.
12 Clive Thompson, 'Clive Thompson on 3-D Printing's Legal Morass', *Wired*, May 30, 2012, sec. Wired Design, http://www.wired.com/design/2012/05/3-d-printing-patent-law/.
13 Leslie Katz, '3D-printed 'magic arms' give little girl new reach', *CNET*, August 6, 2012, http://news.cnet.com/8301-17938_105-57487822-1/3d-printed-magic-arms-give-little-girl-new-reach/.

THE SECURITY
NEWSLETTER
#22

SHA-3 IS BORN

In early October of this year, the US National Institute of Standards and Technology (NIST) announced the winner of the Secure Hash Algorithm (SHA) competition: SHA-3 is KECCAK, created by G. Bertoni, J. Daemen and G. Van Assche of STMicroelectronics and M. Peeters of NXP Semiconductors. The NIST offered no financial reward. The only satisfaction to the winners is the prestige. This is Daemen's second victory; he was a co-winner of the NIST AES competition.

The competition started in 2007. NIST set various acceptance criteria, for example, that algorithms should be royalty-free. NIST received 64 proposals. Cryptography experts analyzed and reviewed the algorithms throughout the competition. KECCAK won not only for its enhanced security (resistant to attacks that succeed against current Hash algorithms), but also thanks to its higher performance, especially on hardware.

What is a Hash function?

Hash functions transform a message of arbitrary length into a fixed-length string, called a digest or hash value. For example, a gigabyte movie file is reduced down to 160, 256 or 512 bits. Hash functions have no secret key. They are simpler than ciphers and have a much broader range of applications. Data integrity and identification, random generation, user authentication, and password storage are just a few examples that we use daily without realizing it.

What are the properties of a Hash function?

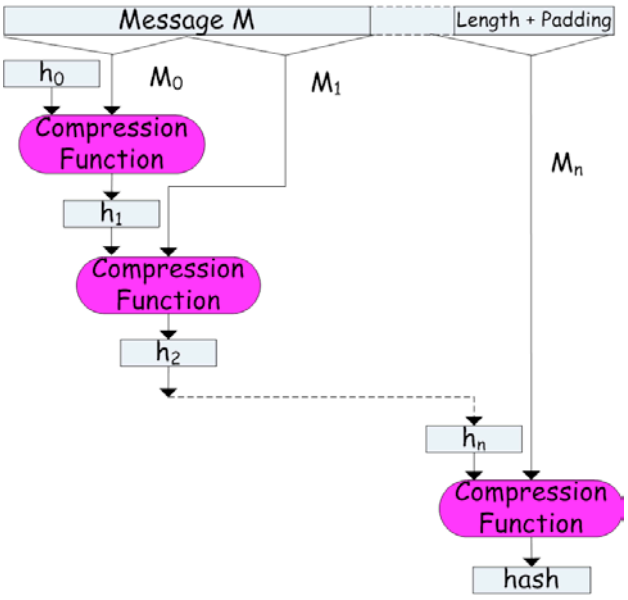
Hash functions are deterministic: knowing the algorithm, one can verify that the hash value matches the hashed file. They need two additional properties: "one-way" and "collision-resistant." "One-way" (aka pre-image resistant) means it is not feasible to guess the original message from the digest. "Collision-resistant" means it is very unlikely that another message (even if they differ only by one bit) hashes to the same value. When that happens, it is called a "collision." Since a Hash function maps an infinite set (of all possible files) to a smaller, finite set of digest values, collisions exist. The key point is that it is not computationally feasible to exhibit one collision.

Birthday Attacks

A birthday attack aims at finding collisions. It consists of computing digests of randomly chosen messages until the result equals any previously computed one. This attack is powerful, since, at each new computation, a new target value is added. Statistically, if the digest length is n, one would compute in average $2^{n/2}$ digests before finding a collision. This mathematical property is known as birthday paradox.[14] For instance, with the former SHA-1 hash function, a birthday attack may find a collision in 2^{80} hashing operations, since SHA-1 outputs 160 bits.

It is possible to create collisions deliberately. If this takes more than $2^{n/2}$ attempts, then the Hash function is considered secure, because such an attack is no better than the birthday attack. However, if a collision can be found in less than $2^{n/2}$ attempts but still more than 2^{80}, then the Hash function is considered theoretically broken. If it is less than 2^{64} trials, then a supercomputer, or distributed calculus on a very large network of computers, might find collisions in few months. On February 15, 2005, Chinese researchers Wang, Yin and Yu announced that they found an attack that would produce a collision in 2^{69} operations against SHA-1. A few months later, the same researchers announced that they improved their attack to a complexity of 2^{63} hashing operations. They exploited inner construction of the Hash function to make the attack effective.

Constructions

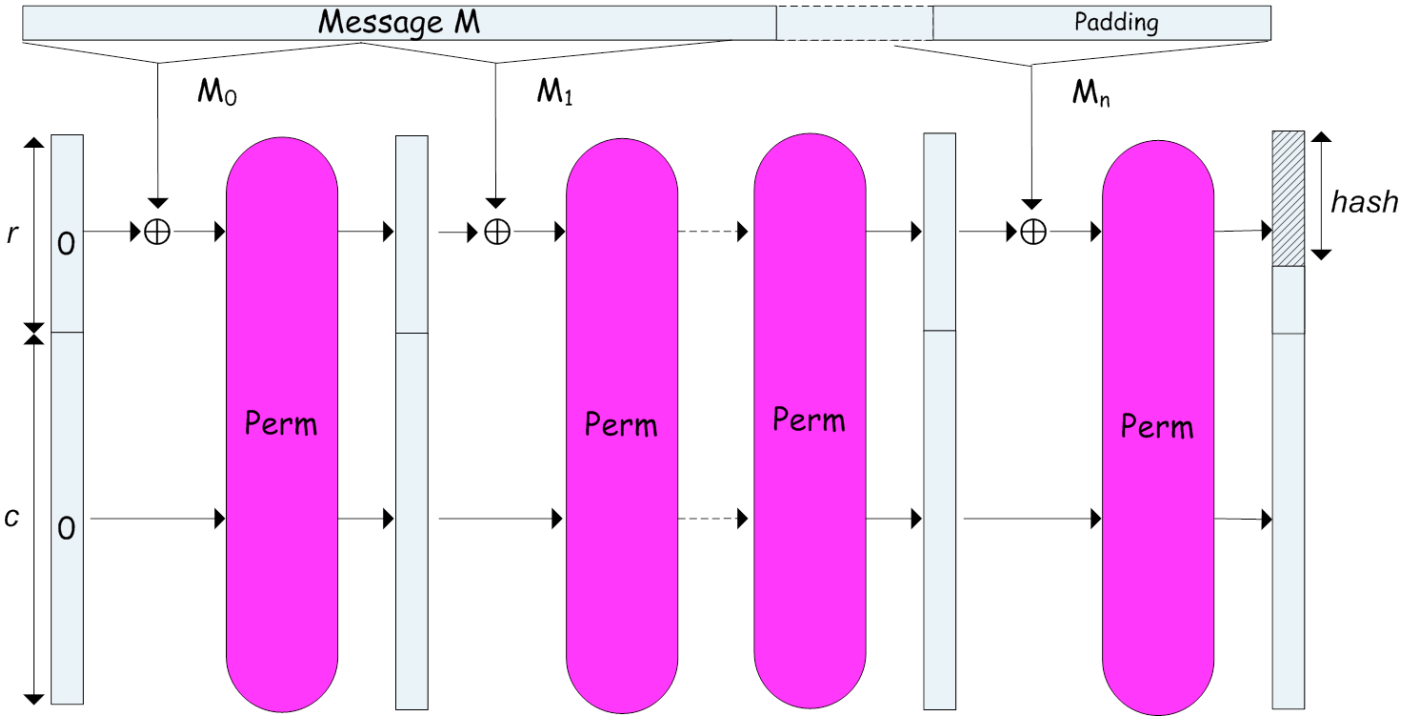


Hash functions such as SHA-1 are generally built upon iterated compression functions. A compression function transforms a fixed-length input into a smaller fixed-length output. To hash, one pads the input message, and breaks it into blocks of predefined size, e.g. 128 bytes for SHA-1. Then, the first block of the message to hash is passed together with a fixed initial value h through a Compression Function to get a first result. This result becomes the input to the Compression Function with the second message block and so on. The result of the last block, which contains an encoding of the length of the message and padding, is the hash value of the entire message.

The most famous hash families are MD, RIPEMD and former SHA families.

14 If you have 23 people in a room, there is more than 50% chance that two of them have the same birthday!

THE SECURITY
NEWSLETTER
#22



SHA-3 is based on a different construction principle. It uses the so-called Sponge Function. A Sponge Function uses an initial b-bits memory state and a permutation function, f, which permutes the memory state. The memory state can be expressed as $b = r + c$ where r is the bit-rate and c is the capacity. To compute a hash, one initializes the memory state to 0, pads the input message, and breaks it into blocks of size r-bits (where r is smaller than b). Then, the first message block is XORed into r-bit of the memory state and then f is applied to get a first result. Next, the second message block is XORed with r-bit of the first result and then f is applied again to get a second result and so forth. The process is repeated until the last block that contains the padding.

SHA-3 uses the sponge construction with a 24-round permutation. It also, supports variable output length 224, 256, 384 and 512 bits with different capacities and bit-rates. For the 256-bit version and 64-bit words. For a 64-bit processor and 256-bit output, SHA-3 sets r to 136 bytes (i.e. 1,088 bits). Because r is always greater than the expected digest length n, after the final block permutation, the leading n bits of the state are the desired hash value.

Also, SHA-1 operations are mostly based on so-called ARX (addition, rotation, XOR) whereas SHA-3 does not make use of arithmetic operations. This makes it easier to protect against Differential Power Analysis attacks. This feature is particularly interesting for smart-card applications, when the hash function is used for generating keyed-HMAC values.

Conclusion and recommendation

Sponge functions open a new direction for building hash functions. The resistance level is shown to be closely related to the capacity length of the memory state. In the SHA-1 case, this corresponds nearly to the size of the chaining value that seems too small and thus leads to weaknesses. Therefore, we recommend the use of SHA-3 for long-term security applications.

For new systems, SHA-256 or SHA-512 should be used instead of SHA-1, since there is no reason to believe that a practical attack on those functions is imminent. However, they are based on the same design principle and attacks will get better with time. They will certainly be affected by improved attacks.

On the other hand, there is no need to recall your SHA-1 based software; no collision has been found yet. The press sometimes wrongly claims that "SHA-1 is broken" to make it sensational enough to attract readers. Cryptographers are excited too – but mainly because of the breakthrough or advance made in this field, not because they believe there is an immediate threat.

M. KARROUMI

THE SECURITY
NEWSLETTER
#22

SHAKING SSL

This has been a tough year for Secure Socket Layer (SSL) and Transport Layer Security (TLS). Numerous vulnerabilities concerning the widely-used SSL/TLS protocol have been published at major security conferences (ACM CCS, Usenix Security, Crypto, NDSS, ESORICS). The vulnerabilities concern different aspects of SSL/TLS: *certificate verification, key generation, the protocol itself and side channels*. This article provides an overview of these latest developments.

Recap on SSL/TLS

SSL and its successor TLS are cryptographic protocols that protect network communication from eavesdropping and tampering. It is the de facto standard for Internet communications. It is the security layer of `https`, and is used for quite diverse applications such as online banking, mail applications and online shopping. SSL/TLS is also used in the professional domain, for instance in virtual private networks, cloud administration, access to digital content and more.

In this article, we describe all the recent attacks and their consequences.

Certificate verification vulnerabilities

At ACM CCS 2012, two papers demonstrated vulnerabilities due to poor implementation of certificate verification in widely deployed applications. The first paper[15] examines a large set of popular Android apps, while the second one [16] focuses on a large set of non-browser software, such as the Amazon EC2 Java library, Paypal's merchant SDK or mobile banking applications.

SSL/TLS relies on certificates that authenticate the end-points of the communication. Usually, certificates are used to authenticate the server only. Generally, clients are not authenticated by SSL/TLS. Proper certificate validation by the client should at least include verification of:

1. the subject of the certificate, called the Common Name (CN), which should match the destination domain name of the communication,
2. the signing Certification Authority (CA), which should be a trusted CA,
3. the validity of the certificate with respect to its expiration date and possible revocation lists , and
4. the correctness of the signature itself.

Incomplete verification of the certificate opens the door to Man-In-The-Middle (MITM) attacks. The MITM attack intercepts messages sent in a SSL/TLS protocol, using a MITM SSL proxy. The SSL proxy replaces the original server certificate with its own certificate, which goes unnoticed unless the client properly verifies the signature. The MITM attacker can eavesdrop and potentially modify the messages of the SSL/TLS communication.

Using static code analysis, Fahl et al. examined 13,500 Android apps. [17] Eight percent (1,074 apps) accepted any SSL certificate or hostname (CN) for a certificate and thus were potentially vulnerable to MITM attacks. Based on numbers provided by the Google Play app market, the apps involved are installed on 39.5-to-185 million devices (Google Play app market only provides ranges). Three of these apps are installed on 10-to-50 million devices. Exploiting weak SSL verification, the authors were able to capture sensitive information, such as credentials from American Express, Diners Club, bank accounts, diverse mail and social network accounts. Furthermore, the authors could inject false virus signatures into an antivirus application using the weak SSL implementation. Virus detection could be deactivated by injecting an empty virus signature base or by injecting the signature of the antivirus app itself (which then deleted itself). As a countermeasure, the authors suggest that the Android app market should include automatic static code analysis that would help detect bad certificate verification.

Georgiev et al.[18] came to very similar conclusions on another set of applications and libraries: SSL certificate validation is widely broken and vulnerable to MITM attacks. Some of the software involved are SDKs or middleware, such as libcloud, Apache ActiveMQ, Apacha Axis, Paypal Payment or Amazon Flexible Payment Service. All applications that rely on these frameworks are thus vulnerable. By attacking these applications, the authors could gather information such as credit card numbers and login credentials for Google, Yahoo! and Windows Live services.

15 Sascha Fahl et al., 'Why eve and mallory love android: an analysis of android SSL (in)security', in *Proceedings of the 2012 ACM conference on Computer and communications security, CCS '12* (New York, NY, USA: ACM, 2012), 50–61, doi:10.1145/2382196.2382205.

16 Martin Georgiev et al., 'The most dangerous code in the world: validating SSL certificates in non-browser software', in *Proceedings of the 2012 ACM conference on Computer and communications security, CCS '12* (New York, NY, USA: ACM, 2012), 38–49, doi:10.1145/2382196.2382204.

17 Fahl et al., 'Why eve and mallory love android'.

18 Georgiev et al., 'The most dangerous code in the world'.

THE SECURITY
NEWSLETTER
#22

The broken applications and libraries usually do not implement SSL themselves but rely on SSL libraries such as OpenSSL, GnuTLS, JSSE, etc. According to the authors, the root cause of broken certificate validation lies in poorly designed APIs of these SSL libraries. The numerous options and parameters lead to developer misunderstandings. For example, to enable hostname verification, Amazon's Flexible Payments Service PHP library sets c`URL's CURLOPT_SSL_VERIFYHOST` parameter to true. Unfortunately, the correct, default value of this parameter is 2; setting it to true silently changes it to 1 and disables certificate validation. Another example is the `SSLSocketFactory` API of JSSE: it silently skips hostname verification if the algorithm field in the SSL client is NULL rather than HTTPS.

These findings highlight the importance of systematic adversarial testing during application development. Most vulnerabilities could have been easily discovered by such tests. Finally, the APIs of most SSL libraries need to be redesigned and clarified.

Key generation vulnerabilities

In August 2012, Heninger et al. pinpointed another vulnerability in TLS [19], mostly caused by poor implementations of the key generation phase. In TLS, the server has an RSA public key. This public key is used to check a server's signature of handshake messages – in order to hedge against MITM adversaries during a Diffie-Hellman key agreement. This public key is also used by the client to encrypt its session key material.

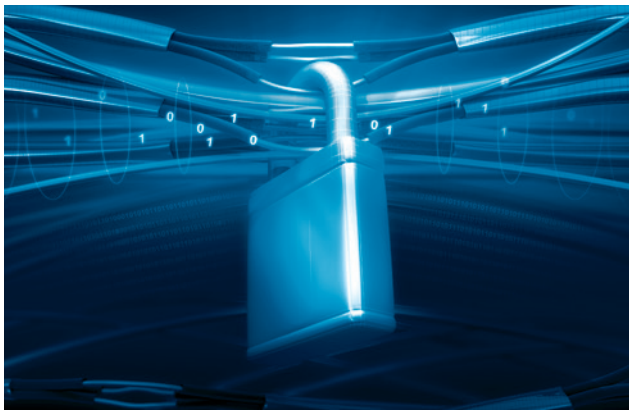
In an Internet-wide survey, a non-negligible fraction (0.75%) of examined public keys was found to collide because of a key generation phase using insufficient entropy (similar observations were made by Lenstra et al.[20]). Heninger et al. were further able to compute the RSA private keys of 64,000 TLS hosts (or 0.50% of the considered candidates) by merely computing greatest common divisors (GCD) among scrutinized moduli. Within an 11-million population of RSA moduli, this was done in two hours using an algorithm for computing GCDs between all pairs in a much faster way than by testing each pair individually.

Such a key-recovery attack either renders the handshake vulnerable to a man-in-the-middle attack or directly exposes the session key. This highlights the fact that secure random number generation remains a delicate problem in security applications.

Protocol vulnerabilities

In October 2012, Mavragiannopoulos et al. presented a new cross-protocol attack against implementations of the TLS handshake protocol that optionally support elliptic curve cryptography on the server's side.[21] This attack is in the spirit of the Wagner-Schneier attack, which takes advantage of the fact that server certificates do not authenticate the server's choice among algorithms supported by the client.[22]

In the Wagner-Schneier attack, a MITM adversary can intercept a server's hello message when the latter contains parameters – consisting of a prime number p and a generator g of the multiplicative group (Z/pZ)* -- for a Diffie-Hellman key agreement protocol. The adversary can then trick the client into interpreting p and g as an RSA modulus and RSA exponent, respectively. The client replies by returning k^g mod p which, in the client's perspective, is an RSA encryption of the pre-master secret k. However, since p is prime, the adversary can easily compute the session key k and impersonate the server. As noted by Mavragiannopoulos et al., the SSLRef 3.0b1 implementation resists this attack because it properly parses packets and reads their length before their content. The risk of misinterpretation at the client is eliminated because packets asking for a Diffie-Hellman key agreement contain one additional field.



However, they came up with a more subtle attack where an appropriate parsing of packets may not suffice. In short, a MITM adversary can obtain Elliptic Curve Diffie-Hellman (ECDH) parameters in some server's certificate and present them as standard Diffie-Hellman parameters (intended for a key agreement in a subgroup of (Z/pZ)*) to the client. The attack succeeds under two conditions: first, certified ECDH parameters fall in a range that coincides with that of potential standard Diffie-Hellman parameters and second, the adversary is able to infer the Diffie-Hellman session key. This occurs with low but noticeable probability when, for example, the ECDH parameters specify curves over small fields: in this case, the y-coordinate of the server's ephemeral Diffie-Hellman

19 N. Heninger et al., 'Mining Your Ps and Qs: Detection of Widespread Weak Keys in Network Devices', in *21st Usenix Security Symposium Proceedings* (presented at the Usenix Sec 12, Bellevue, USA: USENIX Association, 2012), https://www.usenix.org/system/files/conference/usenixsecurity12/sec12-final228.pdf.

20 A. K. Lenstra et al., 'Ron was wrong, Whit is right', *IACR eprint archive 64 (2012)*, https://easterhegg.ch/slides/rwwr-pres.pdf.

21 Nikos Mavrogiannopoulos et al., 'A cross-protocol attack on the TLS protocol', in *Proceedings of the 2012 ACM conference on Computer and communications security*, CCS '12 (New York, NY, USA: ACM, 2012), 62–72, doi:10.1145/2382196.2382206.

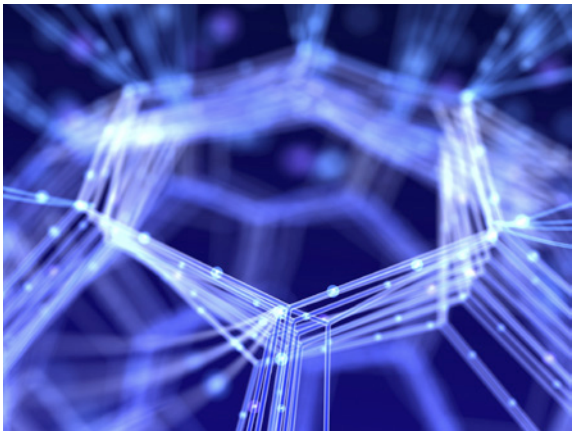
22 David Wagner and Bruce Schneier, *Analysis of the SSL 3.0 protocol* (CiteSeerX, 1996), http://www.counterpane.com/ssl.pdf.

key (which is an elliptic curve point) is likely to take on the values 0, 1, or -1 and be unfortunately interpreted as an ephemeral key for a Diffie-Hellman key exchange over (Z/pZ)*.

On average, the attacker must obtain about $2^{(40)}$ certificates within the time frame of a client session in order to successfully impersonate a server. Still, it is not completely unrealistic if the adversary can simultaneously initiate connections with many servers and only aims at attacking a random client. Therefore it seems advisable for servers' certificates to authenticate the entire negotiated cipher suite in the server's hello message.

Side channels

In 1998, Bleichenbacher published an attack against the PKCS#1 RSA encryption scheme.[23] This led to practical attacks against implementations of SSL (TLS did not exist at the time) that were quickly patched in an ad-hoc manner. "And some things that should not have been forgotten were lost. History became legend. Legend became myth."[24]



During the ESORICS 2012 conference, three German researchers presented an application of Bleichenbacher's attack against the XML encryption standard.[25] This attack is not strictly targeted at TLS/SSL, for which the ad-hoc patches are still efficient. However, it strikes in the very close context of secure web services. This new attack reveals the encryption key of an XML payload within minutes to hours, depending on the scenario.

In Bleichenbacher's attack, the attacker tries to learn if given ciphertext is PKCS#1 conformant. To this end, the attacker captures an encrypted message, uses it to forge many chosen ciphertexts and submits these ciphertexts to a server for decryption. The attacker then observes the server's reaction: accept or reject. Note that the observation channel is the critical element in the attack. Using carefully crafted ciphertexts, the attacker can deduce bits of information from each observation. In the original attack, 106 tries suffice, hence the nickname "Million Questions Attack". It has been improved in the meantime.

The new attack recreates the ideal conditions of Bleichenbacher's attack against PKCS#1, in the context of XML encryption. In a first variant, the observation channel is a timing channel. The XML encryption scheme decrypts the payload only if the tested key is PKCS#1 conformant. Therefore, the attacker may forge a long encrypted payload and submit it to the server: a slow answer reveals that the server decrypted the long payload, which in turn reveals that the key was conformant. Network jitter may obfuscate the observation of tiny time differences, but additional techniques increase the payload up to practically observable decryption times. In the favorable case where the attacker is collocated with a virtual machine of the victim, the attack succeeds in 200 minutes. This case really happens in multitenant clouds.

In the second variant, the observation channel does not depend on a network connection. Instead, it exploits a known weakness of the Cipher-Block Chaining mode used in XML encryption. The attacker is able to modify the last byte of the encrypted plaintext, which contains the number of padding bytes. This allows the forging of packets that are correctly parsed when the tested key is PKCS#1 conformant, and probably not correctly parsed otherwise. The attacker can carry out a Bleichenbacher's attack by submitting such ciphertexts and observing the occurrence of error messages.

The classic countermeasure against Bleichenbacher's attack is returning a random key in case of non-compliance. Then the payload is always decrypted: with the right key if compliant, with the random key otherwise. This countermeasure is only efficient against the timing observation channel. The authors do not detail any countermeasure against the second variant of the attack.

23 Daniel Bleichenbacher, 'Chosen ciphertext attacks against protocols based on the RSA encryption standard PKCS #1', in *Advances in Cryptology — CRYPTO '98*, ed by. Hugo Krawczyk, Lecture Notes in Computer Science 1462 (Springer Berlin Heidelberg, 1998), 1–12.
24 J. R. R. Tolkien, The Fellowship of the Ring, Second Revised Edition. (Houghton Mifflin, 1966).
25 Tibor Jager, Sebastian Schinzel, and Juraj Somorovsky, 'Bleichenbacher's Attack Strikes again: Breaking PKCS#1 v1.5 in XML Encryption', in *Computer Security – ESORICS 2012*, ed by. Sara Foresti, Moti Yung, and Fabio Martinelli, Lecture Notes in Computer Science 7459 (Springer Berlin Heidelberg, 2012), 752–769.

THE SECURITY
NEWSLETTER
#22

Plaintext-Recovery Attacks Against Datagram TSL

The Datagram Transport Security Layer (DTLS) protocol has grown increasingly popular since its introduction in 2004. It provides the security services of TLS to datagram protocols. In February 2012, Al Fardan and Paterson, two researchers at the Royal Holloway University of London, published an attack against implementations of DTLS.[26] The objective of the attack is to recover DTLS-protected plaintext, under reasonable network assumptions.

The attack is a variant of Vaudenay's Padding Oracle Attack.[27] Since DTLS is based on TLSv1.1 and since TLSv1.1 is patched against Vaudenay's attack, DTLS should have been immune. But, in the field of protocol security, there is no such thing as transitive security.

Vaudenay's Padding Oracle Attack exploits an error side channel in the decryption mechanism of TLS. During the decryption, un-patched TLS will send acknowledgment or error messages depending upon whether the padding is valid or not. This channel can in turn be exploited to gather information on the plaintext and, ultimately, retrieve a whole plaintext.

In DTLS, no message is emitted, so the error-side channel does not exist. However, Al Fardan and Paterson noticed a timing difference in the processing of packets. Per se, this difference is so small (a few tens of µs) that it cannot be measured through the network. The authors found several efficient amplification means. For instance, they use very long trains of DTLS messages, use the DTLS heartbeat extension for provoking a message with a predictable response delay, voluntarily choose slow encryption algorithms, etc.

This results in practical attacks against OpenSSL and GnuTLS implementations. Under favorable conditions against OpenSSL, a plaintext block can be recovered with 0.94 probability at a cost of roughly 7,000 bytes of network traffic per byte. The implementation can be fixed to render the processing time independent of decryption failure. Actually, this fix was already mandated in the OpenSSL specification for TLS v1.1, but not for DTLS.

We see two lessons learned, or re-learned, from this attack. First, vulnerable protocols can be constructed from unbroken cryptographic primitives. Second, transitive security does not exist. In both case, this leads to one well known conclusion, already stated by S. Vaudenay in 2002: "It confirms that security analysis must not be limited to the block cipher but must rather be considered within the whole environment: [...] we can really have insecure standards which use unbroken cryptographic primitives."

One final side-channel attack is called CRIME, disclosed at EKOPARTY 2012. In CRIME, the attacker deduces information about a secret cookie just by observing the compression ratio achieved by the server. The attacker submits its own cookie, successively taking the values: a, b, c, d, ... For one of these values, the compression ratio in the server response is the best, thus revealing one character of the secret cookie. Under favorable attack conditions, the attacker may continue and enumerate step-by-step the value of the secret cookie. For the moment, the workaround is just deactivating compression in the ClientHello message.

Conclusions and recommendations

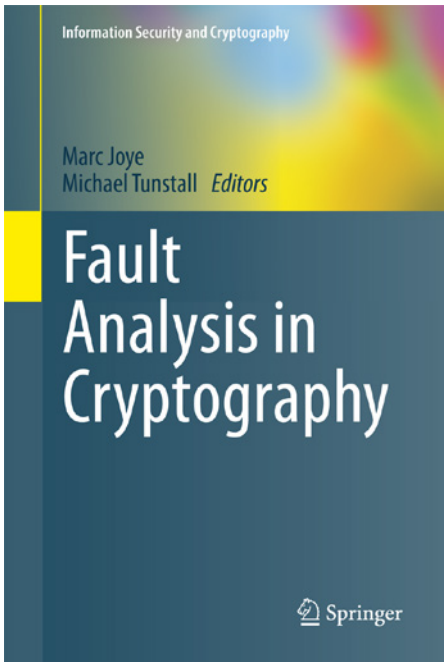
SSL/TLS and their many variants are widely used security protocols. Widely used also means highly exposed. This explains the interest and vigor of the research community in finding practical attacks. Regarding the many recent attacks, one question may arise: is the effort sufficient to secure TLS/SSL? In some cases, static analysis would have avoided vulnerabilities. In others, countermeasures were specified but not implemented. In still other cases, basic verification was just not carried out. These are all sequels of "best effort security." For an exposed protocol such as TLS/SSL, best effort is not enough: verification must be unconditional.

O. HEEN, B. LIBERT, C. NEUMANN

26 N. J. AlFardan and K. G. Paterson, 'Plaintext-recovery attacks against datagram TLS' (presented at the Network and Distributed System Security Symposium, San Diego, California, USA, 2012), http://www.internetsociety.org/sites/default/files/P01_1.pdf.

27 Serge Vaudenay, 'Security Flaws Induced by CBC Padding — Applications to SSL, IPSEC, WTLS...', in *Advances in Cryptology — EUROCRYPT 2002*, ed by. Lars R. Knudsen, Lecture Notes in Computer Science 2332 (Springer Berlin Heidelberg, 2002), 534–545.

BOOKS PUBLISHED BY THE TEAM



Fault Analysis in Cryptography,
Marc Joye and Michael Tunstall, Eds,
Information Security and Cryptography, Springer, 2012

Securing Digital Video:
Techniques for DRM and Content Protection,
Eric Diehl, Springer, 2012

THE SECURITY
NEWSLETTER
#22

WHERE WILL WE BE?

International Conference on Practice and Theory in Public-Key Cryptography (PKC 2013), Nara, Japan, February 26 – March 1, 2013

- **Paper presentation: Robust Encryption, Revisited**
  by Pooya Farshim, Benoît Libert, Kenneth G. Paterson, and Elizabeth A. Quaglia
- **Paper presentation: Efficient Completely Context-Hiding Quotable and Linearly Homomorphic Signatures**
  by Nuttapong Attrapadung, Benoît Libert, and Thomas Peters

SPIE Media Watermarking, Security, and Forensics 2013, San Francisco, CA, USA, February 3-7, 2013

- **Paper presentation: A Sneak Peek into the Camcorder Path,**
  by Chérif Ben Zid, Séverine Baudry, Bertrand Chupeau and Gwenaël Doërr

Workshop on Redefining and Integrating Security Engineering (RISE'12), Washington DC, USA, December 14, 2012

- **Paper presentation: Security Engineering and Modeling of Set-top Boxes,**
  by Jose Francisco Ruiz, Marcos Arjona, Antonio Maña, Antoine Monsifrot, Michel Morvan and Andre Rein

IEEE Workshop on Information Forensics and Security (WIFS'12), Tenerife, Spain, December 2-5, 2012

- **Paper presentation: AC-3 Bit Stream Watermarking,**
  by Xiao-Ming Chen, Michael Arnold, Peter Baum and Gwenaël Doërr
- **Tutorial: A Primer on Content Protection Systems,**
  by Gwenaël Doërr, Alain Durand, and Ton Kalker

TECHNICOLOR SPONSORED CONFERENCES

IEEE Workshop on Information Forensics and Security (WIFS'12), Tenerife, Spain, December 2-5, 2012

NOTES

EXTENSIVE WORLDWIDE PRESENCE

Vancouver	Palo Alto	Mexico	Rennes	Piaseczno	Beijing
Hollywood	New York	Manaus	London	Rome	Bangkok
Indianapolis	Guadalajara	Paris	Madrid	Bangalore	Sydney

technicolor

TECHNICOLOR WORLDWIDE HEADQUARTERS
1, rue Jeanne d'Arc
92443 Issy-les-Moulineaux France
Tel. : 33(0)1 41 86 50 00 - Fax : 33 (0) 1 41 86 58 59

www.technicolor.com