Security Now! #451 - 04-15-14
TrueCrypt & Heartbleed Pt.2

This week on Security Now!

- Truecrypt phase one audio complete, with interesting results and a teachable moment.
- Heartbleed follow-up including the statement from the guy who introduced the bug and MUCH more
- Maybe a few questions and answers, time permitting.

Programming Note: Triangulating on Steve, Part #2

Security News:

Truecrypt --phase one audit-- complete:

- Matthew Green, (Reesearch Professor @ Johns Hopkins)
  "[The results] don't panic me. I think the code quality is not as high as it should be, but on the other hand, nothing terrible is in there, so that's reassuring.

- https://opencryptoaudit.org/reports/iSec_Final_Open_Crypto_Audit_Project_TrueCrypt_Security_Assessment.pdf

Project Summary:
- Bootloader and Windows kernel driver.

Findings Summary:

During this engagement, the iSEC team identified eleven (11) issues in the assessed areas. Four issues were of Medium severity, and another four of Low severity, with an additional three issues having "Informational" severity (pertaining to Defense in Depth).

Overall, the source code for both the bootloader and the Windows kernel driver did not meet expected standards for secure code. This includes issues such as lack of comments, use of insecure or deprecated functions, inconsistent variable types, and so forth. In contrast to the TrueCrypt source code, the online documentation available at http://www.truecrypt.org/docs/ does a very good job at both describing TrueCrypt functionality and educating users on how to use True-Crypt correctly. This includes recommendations to enable full disk encryption that protects the system disk, to help guard against swap, paging, and hibernation-based data leaks.

The team found a potential weakness in the Volume Header integrity checks. Currently, integrity is provided using a string ("TRUE") and two CRC32s. The current version of TrueCrypt utilizes XTS as the block cipher mode of operation, which lacks protection against modification; however, it is insufficiently malleable to be reliably attacked. The integrity protection can be bypassed, but XTS prevents a reliable attack, so it does not currently appear to be an issue. Nonetheless, it is not clear why a cryptographic hash or HMAC was not used instead.

Finally, iSEC found no evidence of backdoors or otherwise intentionally malicious code in the assessed areas. The vulnerabilities described later in this document all appear to be unintentional, introduced as the result of bugs rather than malice.

Recommendations Summary:

Outside of the specific short and long term recommendations detailed in Section 3 of this doc-ument, iSEC Partners also makes the following high level recommendations.

Update the Windows build environment: The current required Windows build environment depends on outdated build tools and software packages that are hard to get from trustworthy sources. For example, following the reproducible build instructions requires access to VC++ 1.52 (released in 1993 -- 21 years ago), in addition to various Windows ports of GNU tools downloadable from wherever they can be found. Using antiquated and unsupported build tools introduces multiple risks including: unsigned tools that could be maliciously modified, unknown or un-patched security vulnerabilities in the tools themselves, and weaker or missing implementations of modern protection mechanisms such as DEP and ASLR. Once the build environment has been updated, the team should consider rebuilding all binaries with all security features fully enabled. For the purpose of auditing, TrueCrypt should release instructions for how to create reproducible builds.

Improve code quality. Due to lax quality standards, TrueCrypt source is difficult to review and maintain. This will make future bugs harder to find and correct. It also makes the learning curve steeper for those who wish to join the TrueCrypt project.

The assessment explicitly excluded:

- Volume parsing as it relates to a file container
- Rescue Disks code paths activated when the disk does contain the private key
- Cryptographic Analysis, including
  - RNG analysis
  - Algorithm implementation
  - Security tokens
  - Keyfile derivation
  - Hidden Containers
  - Linux and Mac Components
  - All other components not explicitly included

Examples of typical "Medium" severity with "High" difficulty:

TrueCrypt Windows driver:

- The TrueCrypt Windows driver code makes some effort to prevent sensitive in-formation from being paged out during a low memory situation by allocating memory from the non-paged pool. However, sensitive information, such as key material, may still leak to various places during execution, among those kernel stack pages, which can be paged out under certain conditions. If the stack for the system thread created during volume mounting were to be paged out, there is a risk that key information in ReadVolumeHeader() could end up on disk.

  It should be noted that for this to be a threat, the user must be running a configuration in which the main Windows system disk is not encrypted, something the TrueCrypt documentation explicitly recommends against.

- EXPLOIT SCENARIO: A user has a system with a TrueCrypt-encrypted partition on it, in which they save sensitive information. An attacker creates a low memory situation on the user’s ma-chine, forcing key information to be paged out to the unencrypted system disk. The attacker later gains access to the disk and can extract the key from the page file.

- SHORT TERM SOLUTION: Consolidate all sensitive information to one single location. The data can then be locked into memory with the help of functions such as MmLockPagableDataSection() and or KeSetKernelStackSwapEnable() to prevent it from being paged out to disk.

***Decompressor problems* -** (also Medium severity and High difficulty)

- The code to decompress the main bootloader suffers from several implementation weaknesses. Throughout the source code, signed and unsigned integer types are mixed, arrays are accessed without checking if the index is within bounds, and so forth. In several cases, the lack of array bounds checking results out-of-bound accesses actually being performed. Three (3) examples can be found in Appendix A.

  It should be noted that in order to exploit this, an attacker would need access to the disk on which the TrueCrypt-encrypted system resides. An attacker with this level of access could instead perform a more effective evil maid attack.

- EXPLOIT SCENARIO: An attacker modifies the compressed bootloader on the disk to exploit one of the issues in the decompressor. Successful exploitation allows the attacker to modify the TrueCrypt code to record and save the password while the user enters it.

- SHORT TERM SOLUTION: Fix the issues identified in the decompressor. Alter the input buffer handling to take an input size argument and verify that the code does not attempt to read past the end of the buffer while decompressing.

- LONG TERM SOLUTION: Make integer types more consistent throughout the TrueCrypt source code, favoring well-defined unsigned types wherever possible. Perform extensive fuzzing of the bootloader decompressor as well as a more in-depth review of the functionality.

TrueCrypt Windows kernel driver

- The function burn() is used to clear sensitive data throughout most of the True-Crypt Windows kernel driver. In the Windows version, burn() wraps RtlSecureZeroMemory(), which is guaranteed to securely erase memory and will not be optimized out. However, in a handful of places, memset() is used to clear potentially sensitive data. Calls to memset() run the risk of being optimized out by the compiler.

- EXPLOIT SCENARIO: A user has a system with a TrueCrypt-encrypted partition on it, in which they save sensitive information. An attacker creates a low memory situation on the user's ma-chine, forcing key information that should have been securely wiped to be paged out to the un-encrypted system disk. The attacker later gains access to the disk and extracts the key from the paging file.

- SHORT TERM SOLUTION: Alter the above code to call burn() instead of memset().

- LONG TERM SOLUTION: Audit the code for other instances of memset() calls that should be re-placed with calls to burn() to prevent potential information leakage.

Suppression of compiler warnings:

- The Microsoft compilers used to build TrueCrypt will warn against some of the issues men-tioned above. However, in both the bootloader and Windows kernel driver, some of these warn-ings have been suppressed. Some are suppressed with #pragma in the code, while others are suppressed in the build scripts. This results in the code compiling without warnings, even though it contains issues that should be corrected.

  Platform.hused by the bootloader contains the following line:
  #pragma warning (disable: 4018 4102 4704 4769)

- Recommendation: Consider removing the suppressions from both the code and the build environment. Instead, alter the source code to ensure it compiles without warnings.

What is all means:

- Large and complex security software is SO DIFFICULT to write and verify that it becomes porous.
- "We the People" don't have the resources to deeply verify open source software.
- The NSA has all the money and people it needs to throw at this.

So What's next?

Matthew Green: "the second phase was now to perform a detailed crypto review and make sure that there's no bug in the encryption.

Heartbleed Followup:

XKCD: http://xkcd.com/1354/

Quote from Robin Seggelmann who said he accidentally introduced the bug two years ago while working on a research project at the University of Muenster in Germany: http://www.telekom.com/corporate-responsibility/security/229078

> "I was working on a research project at the University of Münster using the OpenSSL encryption library and releasing bug fixes and new features that were developed as part of my work on the OpenSSL project. The various changes were checked by a member of the OpenSSL development team and then incorporated into the official code. In connection with one extension, the TLS/DTLS Heartbeat extension, I failed to check that one particular variable, a unit of length, contained a realistic value. This is what caused the bug, called Heartbleed after the extension. Unfortunately, the OpenSSL developer who reviewed the code also did not notice that a mistake had been made when carrying out the check. As a result, the faulty code was incorporated into the development version, which was later officially released.

Timeline of the Heartbleed discovery and dissemination:
Ben Grubb reporting for the Sydney Morning Herald (I learned of this via twitter - Thanks!)
http://www.smh.com.au/it-pro/security-it/heartbleed-disclosure-timeline-who-knew-what-and-when-20140415-zqurk.html

- Friday, March 21 or before: Neel Mehta of Google Security discovers Heartbleed vulnerability.
- Friday, March 21 @ 10:23: Bodo Moeller and Adam Langley of Google commit a patch for the flaw (This is according to the timestamp on the patch file Google created and later sent to OpenSSL, which OpenSSL forwarded to Red Hat and others). The patch is then progressively applied to Google services/servers across the globe.
- Monday, March 31 or before: Someone tells content distribution network CloudFlare about Heartbleed and they patch against it. CloudFlare later boasts on its blog about how they were able to protect their clients before many others.
- Tuesday, April 1: Google Security notifies OpenSSL about the flaw it has found in OpenSSL, which later becomes known as "Heartbleed". Mark Cox at OpenSSL says the following on social network Google Plus: "Original plan was to push [a fix] that week, but it was postponed until April 9 to give time for proper processes."
- Wednesday, April 2 ~23:30 - Finnish IT security testing firm Codenomicon SEPARATELY discovers the same bug that Neel Mehta of Google found in OpenSSL.
- Friday, April 4 - Content distribution network Akamai patches its servers. They initially say OpenSSL told them about bug but the OpenSSL core team denies this in an email interview. Akamai updates its blog after the denial and Akamai's blog now says an individual in the OpenSSL community told them.
- Friday, April 4 - Rumors begin to swirl in open source community about a bug existing in OpenSSL, according to one security person at a Linux distribution. But no details were apparent so it was ignored by most.

- Saturday, April 5 15:13 - Codenomicon purchases the Heartbleed.com domain name [and begins working on a cool dripping blood logo].
- ....

Who knew of heatbleed prior to release?

- Google (March 21 or prior),
- CloudFlare (March 31 or prior),
- OpenSSL (April 1),
- Codenomicon (April 2),
- National Cyber Security Centre Finland (April 3),
- Akamai (April 4 or earlier) and Facebook (no date given)

Who knew hours before public release?

- SuSE, Debian, FreeBSD and AltLinux.

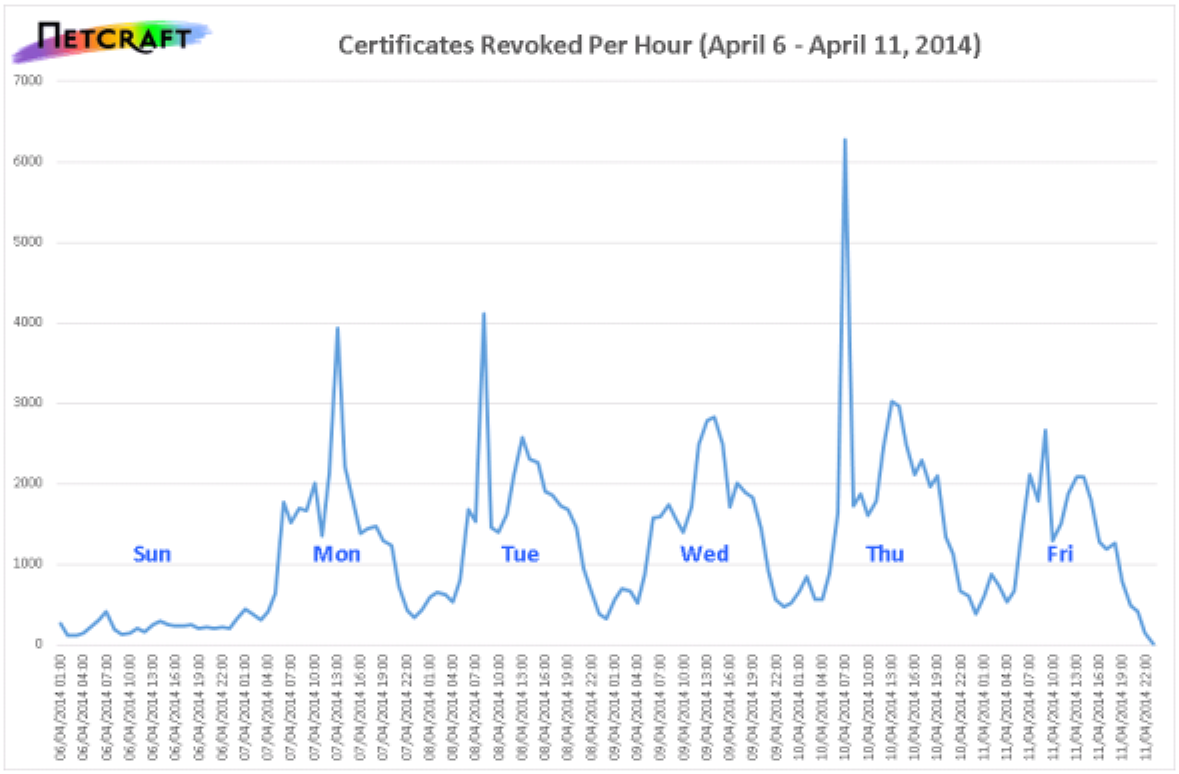
Who did NOT know until public release?

- Many, including Amazon Web Services, Twitter, Yahoo, Ubuntu, Cisco, Juniper, Pinterest, Tumblr, GoDaddy, Flickr, Minecraft, Netflix, Soundcloud, Instagram, Box, Dropbox, GitHub, OKCupid, Wikipedia, Wordpress and Wunderlist.

Netcraft's monitoring of certificate revocation:

- Heartbleed certificate revocation tsunami yet to arrive
- http://news.netcraft.com/archives/2014/04/11/heartbleed-certificate-revocation-tsunami-yet-to-arrive.html
- Activity on certificate revocation lists peaked at a rate of 3,900 revocations per hour on the day the Heartbleed bug was announced (Monday April 7, 2014).
- On a typical Monday, we would expect to see a total of around 22,000-30,000 SSL certificates being revoked over the course of the day.
- On the Monday that the Heartbleed bug was announced to the public, there were 29,000 revocations.
- On the next day (Tuesday), 33,000 certificates were revoked, followed by 32,000 on Wednesday. These were both above average, suggesting that around 5,000 certificates were revoked in direct response to the Heartbleed bug each day.
- Note that fewer revocations usually take place over weekends.

- Certificate authorities MUST revoke certificates within 24 hours if there is evidence of a key compromise.
- A private key is said to be compromised if its value has been disclosed, or if there exists a practical technique by which an unauthorized person may discover its value. Arguably, all certificates on sites vulnerable to the Heartbleed bug should be revoked by now, as such a technique was successfully carried out by the researchers behind heartbleed.com.



How do we reliably detect key-reissue and revocation?

- Unclear whether "Issued Date" is a reliable indicator.
- Perspectives Project:
  - http://bit.ly/sn-451

Several major players doubted that server keys were ever vulnerable.

The Cloudflare Challenge:

- https://blog.cloudflare.com/answering-the-critical-question-can-you-get-private-ssl-keys-using-heartbleed
- http://www.theverge.com/us-world/2014/4/11/5606524/hacker-successfully-uses-heartbleed-to-retrieve-private-security-keys
- https://www.cloudflarechallenge.com/heartbleed
- One of the hackers went so far as to install the Cloudflare private key on his server.
- Modify the HOSTS file and see "www.cloudflarechallenge.com" on the WRONG site!

Bloomberg claims the NSA knew right away:

- "NSA Said to Exploit Heartbleed Bug for Intelligence for Years"
- http://www.bloomberg.com/news/2014-04-11/nsa-said-to-have-used-heartbleed-bug-exposing-consumers.html
- The U.S. National Security Agency knew for at least two years about a flaw in the way that many websites send sensitive information, now dubbed the Heartbleed bug, and regularly used it to gather critical intelligence, two people familiar with the matter said.

There DOES seem to be SOME evidence of previous (November 2013) use of heartbleed:

- https://www.eff.org/deeplinks/2014/04/wild-heart-were-intelligence-agencies-using-heartbleed-november-2013
- [After talking about and dismissing some initial probably false-positive reports]
- The second log seems much more troubling. We have spoken to Ars Technica's second source, Terrence Koeman, who reports finding some inbound packets, immediately following the setup and termination of a normal handshake, containing another Client Hello message followed by the TCP payload bytes 18 03 02 00 03 01 40 00 in ingress packet logs from November 2013. These bytes are a TLS Heartbeat with contradictory length fields, and are the same as those in the widely circulated proof-of-concept exploit.

  Koeman's logs had been stored on magnetic tape in a vault. The source IP addresses for the attack were 193.104.110.12 and 193.104.110.20. Interestingly, those two IP addresses appear to be part of a larger botnet that has been systematically attempting to record most or all of the conversations on Freenode and a number of other IRC networks. This is an activity that makes a little more sense for intelligence agencies than for commercial or lifestyle malware developers.

Bloomberg View

- Leonid Bershidsky / Apr 11th
- Seggelmann has this to say to The Sydney Morning Herald by way of explanation: "It's unfortunate that [open source software] is used by millions of people, but only very few actually contribute to it."
- The open source ideology is attractive to millions of engineers who work on it just for fun, and the challenge of producing beautiful things. As Torvalds wrote, "in a society where survival is more or less assured, money is not the greatest of motivators. It's been well-established that folks do their best work when they are driven by a passion."
- Two conclusions follow.
  - First, where there are people passionate enough about a product to work for free, others will claim their work and use it for profit because they are equally passionate about money.
  - Second, open-source software will never be painstakingly debugged: No one is passionate about that kind of drudgery. This does not mean commercially-produced software is free from bugs, but at least paid developers bear responsibility for the products. In the case of OpenSSL, all the developer will do if something goes wrong is shrug and say he was doing it for free – more or less as Seggelmann did.

- The reason so many big companies – Google, Facebook, Cisco, Juniper Networks, Yahoo!, Amazon and others – were using OpenSSL, was using the free libraries speed up the development process and make it cheaper. Even companies that initially frowned on open source, such as Microsoft, now talk of "a mixed-source" world. "Pragmatism from customers drove commercial vendors of all stripes toward interoperability," the company wrote in a paper titled Microsoft and Open Source Software. "Today, the state of IT in virtually every sector is one of 'mixed source'."

- (Don't we remember that Microsoft took the source code of the Internet stack from one of the BSD's for NT?)

- The fact that for-profit companies have embraced open source does not mean, however, that they are willing to spend serious money to fund it. Why would they want to when it's fueled by passion, as Torvalds explained? As Johns Hopkins University cryptography expert Matthew Green tweeted recently, "Hey companies that use OpenSSL: How many $$ have you spent recovering from Heartbleed? Why not fund OpenSSL so it doesn't happen again?"

Dan Kaminsky waxes poetic about Heartbleed

- http://dankaminsky.com/2014/04/10/heartbleed/
- Dan's own abstract of his much longer note:
- Heartbleed wasn't fun. It represents us moving from "attacks could happen" to "attacks have happened", and that's not necessarily a good thing. The larger takeaway actually isn't "This wouldn't have happened if we didn't add Ping", the takeaway is "We can't even add Ping, how the heck are we going to fix everything else?". The answer is that we need to take Matthew Green's advice, start getting serious about figuring out what software has become Critical Infrastructure to the global economy, and dedicating genuine resources to supporting that code. It took three years to find Heartbleed. We have to move towards a model of No More Accidental Finds.

- (Much) later Dan writes:
  But you know, word from the Wall Street Journal is that there have been all of $841 in donations to the OpenSSL project to address this matter. We are building the most important technologies for the global economy on [a] shockingly underfunded infrastructure. We are truly living through Code in the Age of Cholera.

  Professor Matthew Green of Johns Hopkins University recently commented that he's been running around telling the world for some time that OpenSSL is Critical Infrastructure. He's right. He really is. The conclusion is resisted strongly, because you cannot imagine the regulatory hassles normally involved with traditionally being deemed Critical Infrastructure. A world where SSL stacks have to be audited and operated against such standards is a world that doesn't run SSL stacks at all.

  And so, finally, we end up with what to learn from Heartbleed. First, we need a new model of Critical Infrastructure protection, one that dedicates real financial resources to the safety and stability of the code our global economy depends on – without attempting to regulate that code to death. And second, we need to actually identify that code.

Other heartbleed impacts:

- Android v4.1.1 vulnerable and there are millions of those.
- Cisco and Juniper Networks routers are affected.

"Why security is hard":

- Google patches Android icon permissions attack
- FireEye found malware that could change other icons, sending victims to phishing sites
- http://www.infoworld.com/d/security/google-patches-android-icon-permissions-attack-240495
- Discovered by security research firm, FireEye.
- Google has issued a patch for an attack that could lead an Android user to a phishing site are FireEye spotted a malicious Android application that could modify the icons of other applications so that when they're launched, they send victims to a phishing website.
- The malware is abusing a set of permissions known as "com.android.launcher.permission.READ_SETTINGS" and "com.android.launcher.permission.WRITE_SETTINGS."
- The two permissions have long been classified as "normal," a designation give to application permissions thought to have no malicious possibilities. Android users aren't warned about granting those permissions when they install an application. But using these "normal" permissions, a malicious app can replace legit Android home screen icons with fake ones that point to phishing apps or websites.
- FireEye developed a proof-of-concept attack using Google's Nexus 7 tablet running Android version 4.2.2 to show icons could be modified to send people to another website.
- During their tests, they uploaded their application to Google's Play store but removed it quickly. Google's Play store, which does check applications for security issues, didn't prevent FireEye's application from appearing in the store, they wrote. No one else downloaded the proof-of-concept app, FireEye said.
- FireEye tested a Nexus 7 running CyanogenMod as well as a Samsung Galaxy S4 running Android 4.3 and an HTC One running 4.4.2. All classify the "read_settings" and "write_settings" permissions as normal.
- Google has released a patch to its OEM partners, but many Android vendors are slow to adopt security upgrades. FireEye notified Google of the flaw in October 2013, and Google told FireEye in February it had released the patch.
- Security experts have long noted that patching mobile devices, especially those already in the hands of customers, is inconsistent and slow.
- FireEye wrote: "We urge these vendors to patch vulnerabilities more quickly to protect their users."

Media Watch

Orphan Black returns this coming Saturday! (Season #2)
BBC America / Marathon of entire 1st season.
http://www.sfgate.com/tv/article/Orphan-Black-season-two-delves-into-ethics-5401820.php
'Orphan Black' season two delves into ethics

"As good as it was last year, it's off to an even better start in its sophomore year."

SQRL

SpinRite

Many Corporate Site License purchasers.
One purchase for a lifetime use-forever license.
Corporate Site License:
    Four purchases -- for use on ALL of the machines in a corporate location.