



THE SECURITY

NEWSLETTER

#17

FALL 2010



In this Issue

Editorial	1
Be Our Guest	2
The News	4
Attacks on DLL Preloading	4
iPad Jailbreaking	5
WEP Back To Haunt	5
Cinavia Bug on PS3?	5
XSS Vulnerabilities in SharePoint Server 2007 and JIRA	6
A New Way for TCP Connection	7
Hole196	8
Where Will We Be?	11

Published Quarterly By
Technicolor Security & Content Protection Laboratories
Part of Office of the CTO

Technical Editor: Eric Diehl

Editors: Sharon Ayalde

Contributors: Patrice Auffret
Jean Marc Boucqueau
Raphael Gelloz
Olivier Heen
Mohamed Karroumi
Christoph Neumann
Charles Salmon-Legagneur

CTO: Ben Crosby

Subscribe to the newsletter:
security.newsletter@technicolor.com

EDITORIAL

This summer was hot and full of interesting events. As usual, Defcon and Black Hat, the two main hacking conferences, have brought their load of impressive demonstrations, stoning news, and plethora of disclosed weaknesses. For instance, the video showing the presentation of researcher Jack Barnaby made the headlines. He demonstrated the hacking of an ATM in a live session. And it worked! Attacks that are visual and close to day-to-day users' concerns always attract journalists and bloggers. Many interesting and worthy attacks were disclosed but never made the headlines. They were too complex to be explained in simple terms. They were too esoteric to interest journalists. Thus, we decided to describe some of the most remarkable attacks, at least according to Technicolor's security experts.

XSS is one of the more deployed classes of vulnerability. Christoph explains how Cross Site Scripting (XSS) attacks cracked two iconic Internet entities: Microsoft and Apache. Patrice and Olivier show how a less known flavor of TCP protocol can be used to gain some benefits. Raphael and Olivier accepted my challenge to provide a pedagogical analysis of Hole196. Hole196 is the second presentation of BlackHat Conference that made the headlines. Sohail Ahmad disclosed an intrinsic vulnerability in WPA2. All these attacks are using well-established solutions. And there are still flaws to be discovered.

With all these presented exploits, it was interesting to see what could be done to prevent them. The first mandatory step is to discover the vulnerability. Fuzzing is a new technology used for searching with non-deterministic methods security vulnerabilities or software bugs. Ari Takanen, the CTO of Codenomicon™ will give a good insight in this new breed of tests. His company offers one of the most known fuzzing tool.

With all these network security exploits, never forget the eighth law of Technicolor's security and content protection laboratories': "If you are connected to the Internet, then the Internet is connected to you."

E. DIEHL,
Technical Editor

THE SECURITY

NEWSLETTER

#17

BE OUR GUEST

Ari Takanen

Hi Ari, can you introduce yourself to our readers?

My name is Ari Takanen and I am the CTO and one of the co-founders of Codenomicon.

How did you get involved into security?

The most important change in my career was joining the Oulu University Secure Programming Group (OUSPG) in 1998. Before getting involved in PROTOS research at University of Oulu, I worked as a system administrator where I noticed that timely installation of security updates and fixing broken computer systems was one of the biggest and challenging tasks. Therefore being able to build proactive solutions, protocol fuzzers, that enabled zero-day vulnerability discovery was like a dream come true. Continuing the development at Codenomicon since 2001 was a natural path following the success of PROTOS fuzzing tools.

What is Fuzzing about?

Fuzzing today is about a wide range of technologies and algorithms for creating unexpected anomalous inputs to software in order to crash it. Early techniques mostly involved random white noise tests, whereas today most tools are highly sophisticated model-based test case generators and protocol simulators. The main motivation for fuzzing is to find zero-day vulnerabilities either as part of R&D process or as a penetration testing technique.

What are the main methods for fuzzing?

Most fuzzers can be categorized in two groups: template-based and specification-based fuzzers. The simplest form of a template-based fuzzer is a file fuzzer that takes in a sample file and randomly modifies that to create anomalous mutations. On the other extreme are fuzzers built from hundreds of pages of specifications covering every aspect of a communication interface with thorough fuzz tests. Both approaches have their benefits and drawbacks. Template-based fuzzers are quick to build but do not have much of a test coverage. Specification-based fuzzers give much better confidence in test result, but are complex to build and maintain.

When considering a fuzzer, what are relevant selection criteria and metrics?

You should be really proud of your questions, because you have already listed most of the critical criteria: How is the protocol functionality built (template versus specification)? How is the protocol modeled (flat binary, block-based, tree-like, full executable model)? How are test cases generated from the model (random, anomaly library, scalable)? How are the test case functionalities and models maintained (one-off, by tool user or vendor, or fully editable)? As a result of all these different aspects of fuzzers, the most important metric is really two-fold: How many bugs can the fuzzer find and how quickly.

The popularity of fuzzing is quite recent in the industry: why not before?

Fuzzing is not recent as a technology. Already in 70s, people were using fuzzers to find crash-level issues in software, they just had different names for the same technique: negative testing, white-noise testing, random testing, syntax testing, grammar testing, and robustness testing. The name fuzzing became commonly used only in early 90s especially in the security testing domain where people started using these same techniques to catch buffer overflows. For example, already before year 2005 Codenomicon had more than 50 extremely large software companies and device manufacturers using our commercial fuzzers. Smaller companies, and those who had not really faced security incidents with their products, were much slower to adapt fuzzing tools. The biggest influence in fuzzer adoption has been the competition between fuzzers and static code analyzers. Although complementary solutions, they fight for the same product security budget. Although static analysis is much more expensive and less effective, it is still much easier for many to understand and therefore gets more attention.

How could fuzzing complement code review tools?

A code review can only be as effective as the fingerprint library it contains. Whereas finding simple vulnerable function calls and programming constructs is rather effective process, fixing all those flaws is extremely laborious. And the end result does not guarantee any efficiency from the process. Fuzzing on the other hand is a verification technique. Using fuzzing will only catch those vulnerabilities that can be triggered by external inputs, but it gives rather good confidence that there are very few inputs remaining that can trigger yet another vulnerability. It is not restricted to a set of programming languages or practices. So a complete process for product security will first fuzz the interfaces that the software uses to communicate remotely and fix all found flaws. As a result, you will also receive samples of vulnerable programming constructs and a code trace of all critical remotely accessible code that must be code reviewed for these and other flaws. As a result, you will first catch and eliminate the highest risk flaws, and then teach your code review processes and tools to catch these flaws in also non-critical parts of the code.

What do you think would facilitate the integration of fuzzing as a routine part of product development?

The biggest corner-stone to all product security initiatives is management approval. Every day, we meet more and more enthusiastic developers and test engineers who have found fuzzing as effective addition to their tool-chest. However not until very recently fuzzing and negative testing itself has not been in the spotlights of the software development process. Now with Microsoft SDL and BSIMM maturity model enthusiasts and pioneers have conceptual tools to talk to the quality managers and decision makers. Understanding the Return On Investment (ROI) to product security is critical: each bug left to code after release costs tens or hundreds of thousands of USD. Every bug will be found, it is a matter of who finds it.

THE SECURITY

NEWSLETTER

#17

What do you see as the main trends in the area of fuzzing?

There are two visible trends in Fuzzing today. Firstly, in the tool development the biggest trend is the emergence of model-based fuzzing techniques. The first fuzzers used random mutation and could only generate a few test cases. Now testers can choose from a variety of model-based fuzzers. Template based fuzzers can be used to test all IP-based traffic, but the best testing coverage is achieved by using specification based fuzzers. As creating and maintaining specification based fuzzers requires a vast amount of expertise, this domain has quickly been taken over by commercial fuzzing tool vendors. Many open source fuzzing frameworks have become unmaintained, with only one or a few active projects remaining.

Another trend in the field of fuzzing is new customers. The initial users of fuzzers were software developers and testers, and the fuzzing processes were mostly controlled by centralized product security teams. Now as the tools develop, new customers, like system integrators, security consultants and even end-users, are finding them. Fuzzing use cases are also expanding; the same tools can be used to test internal software projects and to verify the quality of outsourced projects. For example, mobile operators, defense organizations or banks, all organizations with critical infrastructure, can enforce security by demanding that all their vendors perform fuzzing as a part of their development process. Software vendors can use fuzzing to demonstrate the quality of their products making it easier for them to enter the critical infrastructure market. In my opinion, the greatest remaining challenge for the future is convincing consumer product companies about fuzzing.

Thank you

A. TAKANEN
CTO of Codenomicon
Interview by O. HEEN



THE SECURITY

NEWSLETTER

#17

THE NEWS

Attacks on DLL Preloading

On August 23, Microsoft released a security advisory warning² that insecure library loading could allow remote code execution:

« This issue is caused by specific insecure programming practices that allow so-called “binary planting” or “DLL preloading attacks”. These practices could allow an attacker to remotely execute arbitrary code in the context of the user running the vulnerable application when the user opens a file from an untrusted location ». This is seen in applications which pass an insufficiently qualified path when loading an external library.

Windows, like many operating systems, has the ability to load Dynamic Linked Libraries (DLLs at runtime). For instance, a media player application loads a DLL for a specific video/audio codec when playing a file. If the library name is specified without a path, Windows searches it in different locations: the directory holding the application, Windows System directories or the current working directory

Let us go back to our use case. When a player application plays back a file, the directory holding the content becomes the current one. If the file requires a codec library not yet installed by the application, Windows attempts to load it from the current directory. A typical attack “plants” a malicious DLL with the name of the codec DLL, together with the video file.

For decades, hackers exploited this flaw. Microsoft regularly published in service package (XP SP2 and after) patches to reduce risks and restrict the DLL loading behavior with new security policies.

Resurgence of this flaw occurred when HD Moore, creator of the Metasploit penetration testing suite, tweeted about a newly patched iTunes flaw. Any kind of remote directory (WebDAV site, SMB/CIFS e.g., shared repertoires under network favorite folder) could be exploited for this attack, and not only the local directory. Attackers could create remote attacks whenever the user navigates on hostile or infected WEBDAV sites. One week later, many ethical hackers reported³ that the vulnerability affected hundreds of applications: adobepro, skype, notepad++, tortoisefsvn, flashplayer9, etc.

Because the cause of the vulnerability could not be strictly classified as a remote code execution - but more likely relies on unsecure coding on application side - Microsoft put the burden of addressing the problem to application developers. Microsoft released guidance and tools for mitigating the issue for both the end users and developers⁴. Unfortunately, this flaw affects so many applications that it will take some time to fix them all. In the mean time, it is important to follow the

Microsoft's guidance⁵ and to educate developers to be security aware. This will take even more time.

C. SALMON LEGAGNEUR

Android Antipiracy Framework Cracked



In September 2008, the quite large “Operating Systems family” welcomed a new birth from famous parents: Android. Based on Linux kernel, owned by Google, SDK available for free⁶, and distributed in the promising Nexus One, a.k.a. Google Phone, Android gathered most assets to become the reference for mobile phone and PDA markets.

Nexus One did not convince, Google stopped promoting his own hardware brand while meeting an unexpected success with Android. Motorola (with Droid), Samsung, and HTC are gaining market share with this open and unlocked Operating System. In June 2010, Android 2.2 source code was released.

In August 2010, Justin Case demonstrated how to break the License Verification Library (LVL)⁷, enabling circumventing protection measures deployed by developers willing to control the distribution of their applications. Android was cracked... Really?

Android makes a tight balance between security, performance and openness. Most applications are written in Java and compiled into bytecode. Due to the need for cross compatibility, the bytecode is fairly readable. Android integrates the Dalvik JVM with dex format input. This bytecode can easily be disassembled using JesusFreke's tool Smali/Baksmali^{8,9}.

Android SDK includes an LVL that can be included in any application project. It handles all of the licensing-related communication with the Android Market client and the licensing service. An application can determine its licensing status for the current user by simply calling a library checker method and implementing a callback that receives the status. Google strongly recommends combining LVL usage with code obfuscation. Justin Case showed how to ensure that an application will always receive a LICENSED result if code was not obfuscated. It is a 3-step method:

- Disassemble .apk with Baksmali and locate LicenseValidator class
- Open this class containing constants and verification methods, modify mapping between possible for “NOT_LICENSED” and the function

² “Microsoft Security Advisory 2269637: Insecure Library Loading Could Allow Remote Code Execution” (Microsoft, August 31, 2010), <http://www.microsoft.com/technet/security/advisory/2269637.mspx>.

³ “SecurityFocus,” n.d., <http://www.securityfocus.com/>.

⁴ “Secure loading of libraries to prevent DLL preloading attacks” (Microsoft, August 24, 2010), <http://support.microsoft.com/kb/2389418>.

⁵ Jonathan Ness and Maarten Van Horenbeeck, “An update on the DLL-preloading remote attack vector,” TechNet Blogs, Security Research & Defense, August 31, 2010, <http://blogs.technet.com/b/srd/archive/2010/08/31/an-update-on-the-dll-preloading-remote-attack-vector.aspx>.

⁶ “Android SDK,” Android Developers, n.d., <http://developer.android.com/sdk/index.html>.

⁷ Breaking Google's Android Licensing Verification Library (LVL), 2010, http://www.youtube.com/watch?v=eMZ4qYbpN_s&feature=youtube_gdata_player.

⁸ “smali,” Google Code, n.d., <http://code.google.com/p/smali/>.

⁹ “smali/baksmali used for great evil,” JesusFreke's AndBlog, August 27, 2010, <http://jf.andblogs.net/2010/08/27/smalibaksmali-used-for-great-evil/>.

THE SECURITY

NEWSLETTER

#17

- Reassemble with Smali

Nobody has seriously endangered Android yet. This hack highlights the fact that the Android LVL is aligned with global Android objectives: easy-to-use and not constraining. The other side of the coin is that used as such, this solution alone is not offering a serious protection level. Developers have the solution in hand: code obfuscation.

J.M. BOUCQUEAU

iPad Jailbreaking

At the end of July, two vulnerabilities were disclosed on Apple's iPad, iPod touch, and iPhone devices. A buffer overflow in FreeType allows arbitrary code execution from specially crafted pdf files, and an integer overflow in IOsource permits an application to gain system privilege. The combination of both exploits can give full control of the devices if the user downloads a forged pdf file.

The site JailBreakMe.com used these exploits to allow user to jailbreak their iPhones and iPads just with one click! Jailbreaking is the act to unlock a mobile phone to be usable on any operator. It was not illegal!

Indeed, at the end of July, the US Copyright Office and the Library of Congress announced six new exemptions that authorize circumventing protection measures as defined by the Digital Millennium Copyright Act (DMCA)¹⁰:

- extracting small video sequences from a CSS protected DVD, to create a new work, for criticism or education purpose,
- making mobile phone applications interoperable with other handsets,
- jail breaking phones,
- circumventing video games for the purpose of good faith testing for, investigating, or correcting security flaws or vulnerabilities,
- circumventing computer programs protected by dongles if they are bugged or obsolete,
- and eBooks if no edition allow access to speak aloud function or special formats displaying

Apple reacted quickly and two weeks later issued new versions that correct these flaws: iOS 3.2.2 for iPads and iOS 4.0.2 for iPhones. This quick Apple's update is a good thing. Regardless of jailbreaking, malware could exploit the closed vulnerabilities for more malevolent actions.

E. DIEHL

¹⁰ "Statement of the Librarian of Congress Relating to Section 1201 Rulemaking." U.S. Copyright Office, July 23, 2010, <http://www.copyright.gov/1201/2010/Librarian-of-Congress-1201-Statement.html>.

WEP Back To Haunt

At Blackhat conference, Meiner et al¹¹ presented an attack against Cisco's Wireless Protected Access (WPA) Migration Mode. The WPA Migration Mode is a feature implemented on Cisco's access points, which enables both WPA and legacy Wired Equivalent privacy (WEP) clients to connect to the same access point. The authors show that it is quite simple to recover the WEP key. This was already known when a WEP client is connected using classical cracking attacks^{12,13}. The authors however also demonstrate that even in absence of WEP clients it is possible to recover the WEP key. Furthermore, the authors show how to bypass an additional security mechanism, named "broadcast key rotation", which according to Cisco's documentation¹⁴ "significantly improves the security".

The WPA Migration Mode builds upon the fact that IEEE 802.11 networks are switched. This makes it possible for the access point to decrypt the frames sent by one station and encrypt it again (using a different encryption scheme) prior to forwarding it to its destination. The access point can distinguish the type of authentication procedure used during association. Consequently, to allow legacy WEP clients associating to the access point multicast and broadcast traffic is encrypted using WEP. This latter point opens a breach used by the authors to recover the WEP key when no WEP clients are connected. They broadcast spoofed ARP messages to the access point (using a technique called "bitflipping"¹⁵ to fool the WEP encryption). This yields many encrypted ARP replies and ARP requests generated by the access point. Applying classical cracking tools¹⁶ on those packets recovers the WEP key.

The authors propose some techniques to mitigate the above attack. However, the best solution is to disable the WPA Migration Mode and use only WPA2.

C. NEUMANN



Cinavia Bug on PS3?

Early July, some US users had the unpleasant experience of not fully enjoying a downloaded movie, "The Losers", on their PS3 Blu-ray reader. Users got one of the two following messages after about 20 minutes of playback:

"Playback stopped. The content being played is protected by Cinavia and is not authorized for playback on this device."

"Audio outputs temporarily muted. Do not adjust the playback volume."

¹¹ L. Meiners and D. Sor, "WPA Migration Mode: WEP is back to haunt you..." in (presented at the Black Hat USA 2010, Las Vegas, NV, USA, 2010).

¹² A. Stubblefield et al., "Using the Fluhrer, Mantin, and Shamir Attack to Break WEP," in Proceedings of Network and Distributed System Security Symposium (presented at the Network and Distributed System Security Symposium, San Diego, USA: ISOC, 2002).

¹³ "Aircrack-suite," <http://www.aircrack-ng.org/>.

¹⁴ "Cisco IOS Software Configuration Guide for Cisco Aironet Access Points" (CISCO, October 2003), http://www.cisco.com/en/US/docs/wireless/access_point/12.2_13_JA/configuration/guide/s13pdf.pdf.

¹⁵ N. Borisov et al., "Intercepting mobile communications: the insecurity of 802.11," in Proceedings of the 7th annual international conference on Mobile computing and networking (Rome, Italy: ACM, 2001), 180-189.

¹⁶ "Aircrack-suite."

THE SECURITY

NEWSLETTER

#17

The content being played is protected by Cinavia. and is not authorized for playback on this device."

Cinavia™^{17,18} is Verance's audio watermark technology. It is required by the Blu-ray standard AACS content protection system¹⁹. PS3 being AACS compliant, it has to implement it.

Actually, the first message (Message Code 1) should be displayed when playing back a screener or a camcorder version of the movie whereas the second message (Message Code 3) should be displayed with a backup copy of a Blu-ray disc. Indeed, Cinavia™ marked the incriminated movie before theatrical and Blu-ray release. The PS3 blocked playback as expected when encountering illegal copies.

Cinavia™ analyzes the soundtrack of the movie. It compares the source of the audio indicated by the embedded watermark (i.e., theatrical or Blu-ray disc) to the format in which the movie is provided to the playback device. If the player detects any discrepancy, the movie is either muted or not played at all. For downloaded movies,.

Cinavia™ protection is limited to Blu-ray players only. The technology does not affect any playback methods that do not include a Cinavia™ detector. The AACS Licensing Authority required that all Blu-ray players sold after Q4 2009 support Cinavia™ and thus to have the detection module in place²⁰. A list of current device players implementing Cinavia™ can be found here²¹. The Cinavia™ detector module was added to version 3.1 of the PS3 firmware update, released at the end of 2009.

Cinavia™ is likely to have a negligible impact on piracy. Attackers already identified several means to circumvent this protection:

1. **Logistics.** The first idea is to fully bypass the playback control engine by using a soundtrack originating from an alternative source. One can for instance retrieve the AC3 or DTS stream from a DVD (which is not expected to carry Cinavia watermark) and remux it with the BD video stream.
2. **Desynchronization.** The second strategy exploits a known shortcoming of the technology against desynchronization. A crude example is to speed-up playback by x1.5 which has been reported to prevent the PS3 from picking up the audio watermark²². More sound preserving techniques are also available which provide exactly the same effect. This known weakness traces back as far as the SDMI challenge.
3. **Protocol.** Another technique is to suspend playback for at least

10 minutes to reset some internal state of the watermarking module. As a result, the consumer gets an extra 20 minutes of unmuted playback.

These attacks demonstrate that implementing a watermark detector in consumer electronics is risky. The adversary has a watermark oracle that tells her whether she defeated or not the system. This lesson has been repeatedly taught during the last decade, e.g. SDMI challenge, or BOWS contests²³.

M. KARROUMI

XSS VULNERABILITIES IN SHAREPOINT SERVER 2007 AND JIRA

Cross-site scripting (XSS) is a vulnerability found in web applications that enables attackers to inject and execute malicious JavaScript and HTML in the victim's browser. Attackers may use this vulnerability to steal a user's session cookie or change the look of the vulnerable site. XSS holds the second position in the OWASP Top 10²⁴, a classification of the 10 most critical vulnerabilities of web applications. Indeed, many systems are vulnerable to XSS attack as shown by the recent vulnerability of Microsoft's SharePoint Server 2007^{25,26} and the very serious attack on the bug tracking system JIRA of apache.org²⁷.

In the majority of cases, XSS injects JavaScript using a website's HTTP query parameters or HTML forms. If the server uses the submitted data to generate a result page without verifying the submitted data, the injected script is returned as is to the browser and the latter executes the script. In Microsoft's SharePoint Server 2007, the server did not sanitize the parameter "cid0" on the page "/_layouts/help.aspx"²⁸. Adding to the HTTP query cid parameter value a piece of script such as "<script>alert('XSS')</script>" lets the browser execute the script which in this example yields a popup alert dialog with the message 'XSS'. The final malicious URL looks like this

17 "Cinavia Frequently Asked Questions" (Verance, 2010), http://www.verance.com/pdf/Cinavia_FAQs.pdf.

18 "What is Cinavia technology and what does it do?," Cinavia, <http://www.cinavia.com/pages/technology.html>.

19 "AACS for Blu-ray Disc Pre-recorded Book, v0.951" (Intel, IBM, Matsushita, Industrial Co., Ltd., Microsoft, Sony, Toshiba, Walt Disney, and Warner Bros, January 12, 2010), http://www.aacsla.com/specifications/AACS_Spec_BD_Prerecorded_Final_0.951.pdf.

20 "ADVANCED ACCESS CONTENT SYSTEM ("AACS") ADOPTER AGREEMENT" (AACS LA, June 19, 2009), AACS Final Adopter Agreement, http://www.aacsla.com/license/AACS_Adopter_Agrmt_090619.pdf.

21 Hajj 3, "Cinavia Protected Disks and Blu-ray Players (disks and players ONLY)," SlySoft Forum, July 28, 2010, <http://forum.slysoft.com/showthread.php?t=41885>.

22 Cinavia and Ps3 quick fix, 2010, http://www.youtube.com/watch?v=R13bvBQEvrc&feature=youtuve_gdata_player.

23 S. Craver et al., "How we broke the BOWS watermark," in . vol. 6505, 2007, 46, <http://adsabs.harvard.edu/abs/2007SPIE.6505E..46C>.

24 "OWASP Top Ten Project," OWASP, n.d., http://www.owasp.org/index.php/Category:OWASP_Top_Ten_Project.

25 Dan Goodin, "Microsoft SharePoint bug exposes credentials, sensitive data," The Register, April 29, 2010, http://www.theregister.co.uk/2010/04/29/microsoft_sharepoint_security_bug/.

26 "XSS vulnerability in Microsoft SharePoint Server 2007," High-Tech Bridge SA - Advisories, April 2010, http://www.htbridge.ch/advisory/xss_in_microsoft_sharepoint_server_2007.html.

27 pgollucci, "apache.org incident report for 04/09/2010," Apache blog, April 13, 2010, https://blogs.apache.org/infra/entry/apache_org_04_09_2010.

28 Goodin, "Microsoft SharePoint bug exposes credentials, sensitive data."

THE SECURITY

NEWSLETTER

#17

`http://host/_layouts/help.aspx?cid=MS.WSS.manifest.xml%00%3Cscript%3Ealert%28%27XSS%27%29%3C/script%3E&tid=X29`. Note that, URL encoding yields strings difficult to interpret by the human eye as, for instance, the character “<” is replaced by “%3c”. Any other JavaScript may replace the quite harmless alert popup, such as scripts that download the authentication cookie to a server controlled by the attacker or scripts that modify sensitive data of the victim.

In order to carry out a XSS attack based on HTTP queries the victim’s browser must access the prepared URL. Attackers may, for instance, send the URL per email or include it within a link of a website they control. In the case of the attack on apache.org the attackers posted the following issue on Apache’s issue tracker: “ive got this error while browsing some projects in jira `http://tinyurl.com/XXXXXXXX`” (The URL has been intentionally obfuscated). Tinyurl is a URL redirection and shortening service. In this case, tinyurl hides the actual malicious URL from the victim. Indeed if the URL had not been hidden, an expert eye may have spotted the XSS attack by inspecting the malicious URL. Several administrators of apache.org clicked on the link, and therefore executed the scripts on their machines. The attack was crafted to steal the session cookies, which in this case gave the attackers access to session cookies with administrator privileges compromising the apache.org’s infrastructure. Even if it is not clear if the attackers actually used the cookies to break into apache.org’s system²⁹, it is sufficient to have these cookies to log in as administrator on the JIRA bug tracking system.

In the meanwhile, Microsoft and Atlassian (the editor of JIRA) released security fixes that solve above issues.

What can we learn from above incidents? How to protect from XSS attacks?

On the server side, the server should check and sanitize all returned parameters before using them. This means adding escape characters or refusing special characters to the input strings, yielding a harmless non-executable string. Additional mechanisms such as setting the HTTPOnly parameter for cookies should prevent access of scripts to cookies on the victim’s machine. The latter measure would have prevented the attackers from downloading the session cookie in the apache.org case.

On the client side, carefully examining the URL is the best, but requires expertise and in practice may not be feasible. URL shortening tools such as tinyurl should raise suspicion, as it may be an attempt to hide a malicious URL. Browser plugins like NoScript³¹ prevent the browser from executing scripts and are able to detect XSS attacks. Such tools are very effective but may affect the quality of experience on many web sites (especially web 2.0 sites).

C. NEUMANN

A «NEW WAY» FOR TCP CONNECTION

TCP – for Transmission Control Protocol – is the most used protocol on the Internet. It is the basis for almost all activity: surf, e-mail, file transfer, etc. TCP is also a complex and ancient protocol with many implementation variations, caveats, and lurking threats.

Recently, researchers T. Beardsley and J. Qian found a new method to establish a TCP connection³². They called it the TCP split handshake. This new method has interesting consequences for network security.

Standard TCP Connection Methods

The standard method to establish a TCP connection is described in IETF document RFC 793³³. This is the classical three-way handshake. It establishes a channel from a client to a server [figure-1] using three messages. The standard also mentions another highly theoretical method called the simultaneous open. It uses four messages [figure-2], basically simultaneously opening two half-way connections.

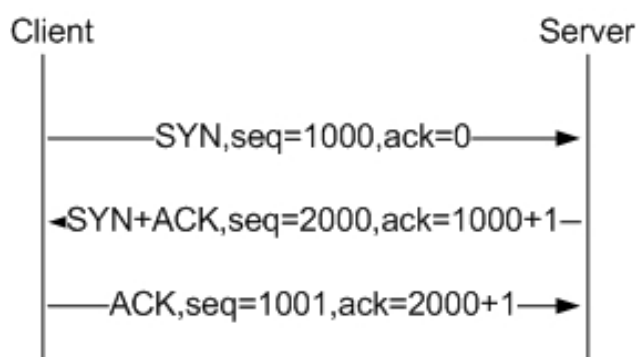


Figure 1 - Three-way Handshake



Figure 2 - Simultaneous Open

²⁹ “XSS vulnerability in Microsoft SharePoint Server 2007.”

³⁰ A brute force attack on the login page of apache.org took place simultaneously and it is not clear which one of the two attacks was successful.

³¹ “JavaScript/Java/Flash blocker for a safer Firefox experience!,” NoScript, n.d., <http://noscript.net/>.

³² T. Beardsley and J. Qian, “The TCP Split Handshake: Practical Effects on Modern Network Equipment,” *Network protocols and Algorithms* 2, no. 1 (May 18, 2010), <http://www.macrothink.org/journal/index.php/npa/article/view/285>.

³³ “RFC793 Transmission Control Protocol,” September 1981, <http://www.ietf.org/rfc/rfc793.txt>.

THE SECURITY

NEWSLETTER

#17

A New Method: The TCP Split Handshake

T. Beardsley and J. Qian mixed both connection methods in order to create a working TCP channel. They called it the TCP split handshake. It can reverse the direction of connection from server to client, instead from client to server. Like the simultaneous open, it uses four messages. Unlike the simultaneous open, it does not require simultaneity of the handshake and the messages slightly differ. First, the client sends one SYN packet to the server. Then, the server starts a classic three-way handshake towards the client by using information gathered from the first SYN [figure-3]. The magic goes here: this should not work, but it does! (at least on the three most popular operating systems: Windows, Linux, and MacOS).

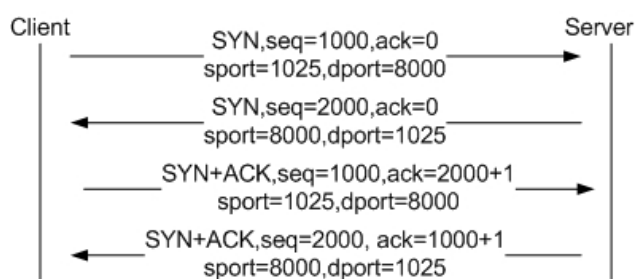


Figure 3 - Split Handshake

The Risk: Evasion from Protection Systems

Because the TCP split handshake significantly differs from the current usage, it has the potential to fool protection systems that do not handle this use case. In the scenario where a malicious server (e.g., a web server) is attacking clients (e.g., browsers) the TCP split handshake can provide a means to evade protection systems (e.g., Intrusion Prevention Systems). Regarding the structural reversion of the connection, it is even possible that some protection systems may be totally confused and may not check some packets, considering they do not go in the sense of an attack).

First Tested Equipment

T. Beardsley and J. Qian tested several devices already. They sometimes obtained worrying results. At least one Intrusion Prevention System (IPS) did not block an attack under the TCP split handshake scenario, although it did routinely block the same attack under the standard scenario.

At first sight, Network Address Translation (NAT) devices seem to better resist. But this is rather by chance. The tested devices did not fully implement the TCP specification, giving less success chance to the TCP split handshake. Other NAT devices may still experience problems.

Conclusion

It is sometimes believed that all TCP weaknesses are known, but this study unveils a totally new concept: reversing the TCP connection establishment! It is a quite deep paradigm change in the client-server model, and it is very difficult to forecast all the consequences, good or bad. Nevertheless, we may see soon some exploits using TCP split handshake appear. Our security law number 8 has never been so true: "when you're connected to the Internet, the Internet is connected to you."

P. AUFFRET, O. HEEN

HOLE196

Sohail Ahmad, a researcher from the wireless security company AirTight Networks, presented the WPA/WPA2 vulnerability termed "Hole196" at Black Hat Arsenal and DEF CON 18³⁴.

There has been a lot of buzz about this Hole196 vulnerability but it does not really provide new attack types. But this gives us the occasion to detail one typical wireless attack.

The attack targets Wi-Fi networks protected by WPA2 Enterprise mode. It lets an authorized user accessing the protected traffic of other authorized users. The attack is irrelevant in personal mode as any authorized user can already sniff all the wireless traffic.

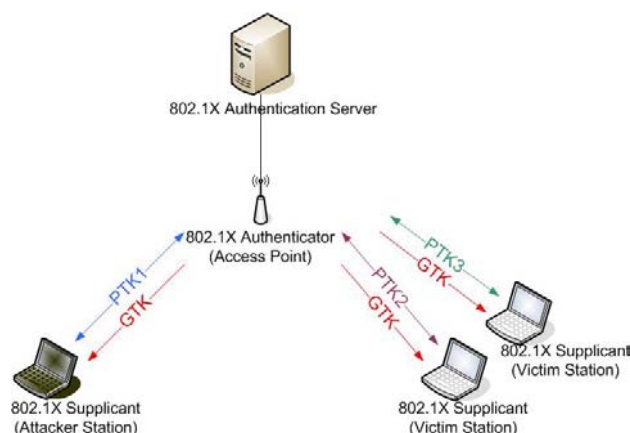


Figure 4 - the Setup of the Attack

The Hole196 attack does not allow an unauthorized user to gain access to a WPA2 network or to crack any WPA2 keys. However, a malicious insider, already authenticated, could leverage this vulnerability to send spoofed broadcast or multicast packets appearing to come from the access point. It would enable him to perform ARP poisoning and man-in-the-middle attacks, inject malware onto other authorized devices, or launch a denial of service without using disassociation frames.

34 "WPA2 Hole196 Vulnerability," Airtight networks, 2010, <http://www.airtightnetworks.com/WPA2-Hole196>.

THE SECURITY

NEWSLETTER

#17

WPA2 Details

WPA2 uses IEEE 802.1x for authentication of the station. The station cannot access the network if the authentication fails. After the authentication phase, the station and the authentication server share a Master Key (MK). They both compute a 256-bit Pairwise Master Key (PMK). The authentication server then delivers the PMK to the access point. The next step is to derive the 384 bits Pairwise Transient Key (PTK) from the PMK, during a 4-way handshake between the station and the access point. The PTK is the concatenation of three 128-bit keys (Figure 5).

PTK = 384-bit Pairwise Transient Key		
KCK = 128-bit EAPoL Key Confirmation Key	KEK = 128-bit EAPoL Key Encryption Key	TEK (=TK) = 128-bit Temporal Encryption Key
Bits 0-127	Bits 128-255	Bits 256-383

Figure 5 - Pairwise Key Hierarchy

The Key Confirmation Key (KCK) used to authenticate the messages during the 4-way handshake and the Group Key Handshake (to renew the Group Temporal Key - GTK).

The Key Encryption Key (KEK) used to encrypt the 128-bit GTK during the 4-way handshake and the Group Key Handshake.

The Temporary key (TK) used to protect the packets with AES.

For its operations, WPA2 uses AES-CCMP (AES Counter mode CBC MAC protocol): the encryption is realized by AES in Counter mode with a 128-bit session key (TK) and a 128-bit block size, the integrity control is performed by AES CBC-MAC, a keyed hashing function using TK. The Message Integrity Code is added to the payload and everything is encrypted with the same key TK.

For the purpose of understanding the Hole196 attack we only have to remember that WPA2 uses two keys to protect data frames:

1. A Pairwise Transient Key (PTK – 384 bits) derived from the PMK, which is unique to each station, and from which we extract the TK for protecting unicast traffic between that station and the access point.
2. A GTK to protect multicast/broadcast data sent to several/all stations on a network. This key is shared by all the authorized WLAN stations and is supposed to be a one-way key: used for encryption by the access point and used for decryption by the stations.

The Hole196 Vulnerability

The attack uses a variant of an ancient technique called ARP poisoning. If correctly executed and undetected by defense mechanism, this technique maps the IP address of the access point to the MAC address of the attacker. Later on, victim stations that connect will take the attacker for the access point.

The name “Hole196” stems from a note on the page 196 of the 2007 IEEE 802.11 Revised Standard³⁵: “Pairwise key support with TKIP or CCMP allows a receiving STA to detect MAC address spoofing and data forgery. The RSNA architecture binds the transmit and receive addresses to the pairwise key. If an attacker creates an MPDU with the spoofed TA, then the decapsulation procedure at the receiver will generate an error. GTKs do not have this property.” On page 38 an extra piece of information is given: “Data origin authenticity is only applicable to unicast data frames. The protocols do not guarantee data origin authenticity for broadcast/multicast data frames, as this cannot be accomplished using symmetric keys and public key methods are too computationally expensive.”

It means that MAC address spoofing and data forgery can be detected in PTK encrypted traffic but not in GTK encrypted traffic. So, an insider can send GTK encrypted broadcast packets to the rest of the WLAN, and the access point will ignore them because it only sends and does not receive GTK broadcast messages. Those packets can have the MAC address of the access point as the source of the frame, and all other stations will believe it comes from the access point.

In a nutshell, an attacker will embed malicious payloads inside GTK encrypted broadcast packets.

For example, to launch a man-in-the-middle attack, the insider will follow these steps:

1. The malicious insider broadcasts fake ARP request packets with the IP address of the actual gateway, but the MAC address of the attacker’s machine. This packet is encrypted using the GTK and sent directly to the stations. All the stations that receive this packet will update their ARP tables – mapping the gateway’s IP address to the attacker’s MAC address.
2. A poisoned station will now send all its traffic to the access point (encrypted with its private key PTK) with the attacker’s MAC address as the destination.
3. The access point uses the station PTK to decrypt the packet and forward it to the attacker, now encrypted with the attacker’s PTK. The attacker, as an authorized station, uses its own PTK to decrypt the packet.
4. The attacker filters the traffic as it sees fit (sniff credentials, inject/alter data, drop packets, ...)
5. The attacker then forwards the traffic to the real gateway to prevent the victims from noticing any abnormal behavior.

A classic ARP poisoning attack over Wi-Fi would have the insider sending its ARP requests to the access point (encrypted with its PTK) and then the access point would broadcast it both to the WLAN and to the LAN (in the clear) where it could be detected. As a result, the Hole196 attack is stealthier.

35 “IEEE Std 802.11™-2007” (IEEE Computer Society, June 12, 2007), <http://standards.ieee.org/getieee802/download/802.11-2007.pdf>.

THE SECURITY

NEWSLETTER

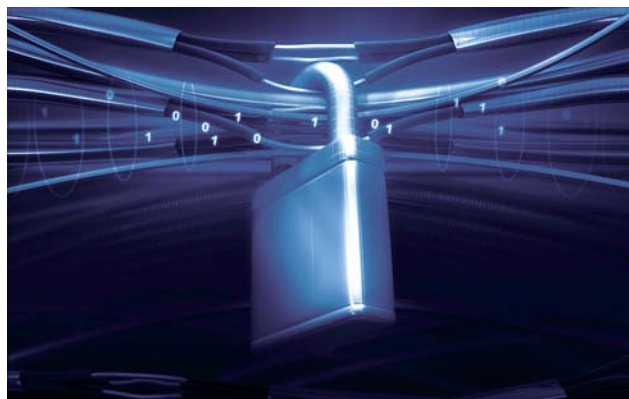
#17

Other attacks are possible by misusing GTK traffic. In particular, a variant of the Hole196 attack provides an efficient denial of service against the whole Wi-Fi network.

This talk did not mention the fact it is possible to perform stealthy ARP cache poisoning using ARP (request or reply) unicast packets: these packets get encrypted with PTK keys and the access point does not forward them on the LAN... The only drawback is that the poisoning must be performed for each targeted machine on the WLAN, whereas an ARP broadcast poisons all machines at once.

What an Insider Can Do

Insider attacks are still the most prominent and costly threat to businesses. According to the January 2010 Cyber Security Watch Survey by CERT, "51% of respondents who experienced a cyber security event were still victims of an insider attack". Insiders can be disgruntled employees, contractors, or corporate spies. The Hole196 vulnerability can be exploited to gather sensitive data such as VoIP over Wi-Fi conversations, intellectual property/trade secrets, identities and credentials, financial information/credit card transactions over Wi-Fi terminals. Insiders are interested in these data to sell them (e.g., to competitors). Personally Identifiable Information³⁶ (PII – e.g., a name and a social security number) can be used for identity theft in various situations: fraud, terrorism, etc. Finally, the leakage of such information can result in fines or loss of image.



Countermeasures

There is nothing in the standard to patch or fix the hole. But there can be proprietary fixes of WPA2: disable broadcast traffic, have the station authenticate the GTK packets as coming from the access point (may require firmware upgrades), assign unique per-station GTKs during 802.1x authentication.

The following elements can also be considered:

- The attack is not so stealthy: broadcast coming from a station, rather than from the AP, can at least raise an alarm. This gives a chance of detection by a Wireless Intrusion Detection System or even by a piece of software on legitimate stations.
- Intrusion Detection Systems could be installed on Stations to detect malicious activities such as ARP poisoning but it would be difficult to deploy on an enterprise environment.
- Station isolation or PSPF (Public Secure Packet Forwarding) is a proprietary feature that can be used to prevent authorized devices from exchanging data. The insider could continue to send spoofed GTK packets but the access point would no longer forward station traffic to the attacker. However 1) the attacker could set up a rogue gateway on the LAN side 2) PSPF does not prevent other types of attacks based on Hole196.

In any case, a VPN tunnel within the encrypted 802.11 session should be deployed to prevent data interception.

As a Conclusion

The Hole196 vulnerability facilitates breaking the separation between legitimates users. This shows well the importance of insider attacks. This also shows the importance of defense in depth: routine use of VPNs for client stations and Intrusion Detection/Prevention Systems at various points in the network.

R. GELLOZ, O. HEEN

³⁶ "Personally identifiable information," in Wikipedia, the free encyclopedia, n.d., http://en.wikipedia.org/wiki/Personally_identifiable_information.

THE SECURITY

NEWSLETTER

#17

WHERE WILL WE BE?

10th ACM Workshop on Digital Rights Management (ACM DRM 2010), Chicago, USA, October 4, 2010

- An introduction to interoperable digital rights locker, by Eric Diehl and Arnaud Robert (Disney)

2010 ACM Multimedia, Industrial Exhibit, Firenze, Italy, October 25-29, 2010

- Multimedia security technologies for movie protection, by Michael Arnold et al.

17th ACM Symposium on Virtual Reality Software and Technology (VRST 2010), Hong Kong, November 22-24, 2010

- Deconstruction of virtual objects for protection, by Marc Eluard, Sylvain Lelievre, Yves Maetz, and Alain Durand

Opening day for "Maths et Entreprise" GDR, Institut Poincaré, Paris, France, December 1, 2010

- Invited talk: Elliptic curve cryptography, by Marc Joye

13th Annual International Conference on Information Security and Cryptology (ICISC 2010), Seoul, Korea, December 1-3, 2010

- Protecting white-box implementation by using dual ciphers, by Mohamed Karroumi

4th International Conference on Pairing-Based Cryptography (Pairing 2010), Yamanaka Hot Spring, Japan, December 13-15, 2010

- Program chairs: Marc Joye and Atsuko Miyaji

Japan Advanced Institute of Science and Technology (JAIST), Nomi, Ishikawa, Japan, December 15, 2010

- Invited talk: The arithmetic of Huff curves and its cryptographic applications, by Marc Joye

Technicolor Sponsored Conferences

10th ACM Workshop on Digital Rights Management (ACM DRM 2010), Chicago, USA, October 4, 2010

2nd IEEE Workshop on Information Forensics and Security (WIFS 2010), Seattle, USA, December 12-15, 2010

EXTENSIVE WORLDWIDE PRESENCE



Vancouver
Hollywood
Indianapolis

Princeton
New York
Guadalajara

Mexico
Manaus
Paris

Rennes
London
Madrid

Piaseczno
Rome
Bangalore

Beijing
Bangkok
Sydney

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

technicolor



© Copyright 2010 Technicolor. All rights reserved. All trade names referenced are service marks, trademarks, or registered trademarks of their respective companies.