HAKIN9

PRACTICAL PROTECTION

IT SECURITY MAGAZINE

HACK APPLE

HACKING TOOLS ON IOS

APPLE MEMORY TRICKS

APPLE OS X AND IOS HACKING

INTERVIEW WITH DAVID HARLEY, MACVIRUS.COM

Vol.6 No.10
Issue 10/2011(46) ISSN: 1733-7186

PLUS

DOUBLE TOOL TIME COLUMN:
INTERCEPTION WITH PAROS PROXY & PREY: FROM PRAYING TO PREYING

eLearnSecurity
Forging security professionals

It's here!
Penetration testing for Students

Click here
To enter the
early bird list

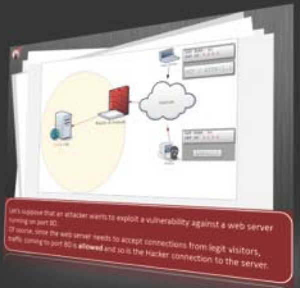
[illegible]

IP addresses: 192.168.5.100
Subnet Mask: 255.255.0.0

This is to say that 192.168.5.3 is in the same network with 192.168.76.135

80% of beginners remain beginners or give up completely
We know the pain of being a beginner.
You either don't have the foundational skills or you don't have a clear path to follow. Don't give up. There is a better way.
Our course will teach you basics of networks and web apps.

It's not just about 1337 instructors
Expert teachers hardly remember what took them to the expert status. It's a fact. There is no way to effectively teach beginners other than help them building strong foundations and showing them the correct path.


Let's suppose that an attacker wants to exploit a vulnerability against a web server running on port 80.
Of course, since the web server needs to accept connections from legit visitors, traffic coming to port 80 is **allowed** and so is the Hacker connection to the server.

Anatomy of a XSS attack
Let's see which actors are involved in a XSS attack and which role each of them play (click on any of them to begin).

You can do it
If you keep studying without a clear learning path you are probably wasting time. Secret is path and perseverance.
Better a single step in the correct direction than 10 random steps.
Our course will save you months of struggling and frustrations.

You gotta see this.

www.elearnsecurity.com

eLearnSecurity
Forging security professionals

Still hacking virtual machines?

Coliseum
Web application security lab

Coliseum Lab is here!

The most epic web app hacking lab
you have ever seen

CLICK HERE

14 educational challenges
in a multi-platform
environment.

Epic!

www.coliseumlab.com

Hakin9 team

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

Managing Editor: *Patrycja Przybylowicz*
patrycja.przybylowicz@hakin9.org

Editorial Advisory Board: Rebecca Wynn, Matt Jonkman, Donald Iverson, Michael Munt, Gary S. Milefsky, Julian Evans, Aby Rao

DTP: Ireneusz Pogroszewski
Art Director: Ireneusz Pogroszewski
ireneusz.pogroszewski@software.com.pl

Marketing Manager: Małgorzata Bocian
m.bocian@*hakin9.org*

Proofreaders: Donald Iverson, Michael Munt, Elliott Bujan, Bob Folden, Steve Hodge, Jonathan Edwards, Steven Atcheson

Top Betatesters: Ivan Burke, John Webb, Nick Baronian, Felipe Martins, Alexandre Lacan, Rodrigo Rubira Branco

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

Senior Consultant/Publisher: Paweł Marciniak

CEO: Ewa Dudzic
ewa.dudzic@software.com.pl

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

Publisher: Software Press Sp. z o.o. SK
02-682 Warszawa, ul. Bokserska 1
Phone: 1 917 338 3631
www.hakin9.org/en

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 smartdraw.com program by SmartDraw

Mathematical formulas created by Design Science MathType™

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.

Dear Readers,

We decided to dedicate this issue to Apple security. For a very long time Apple was considered to be more safe than any other computer and iOS more virus proof than the rest of popular systems. This opinion drastically changed in the last few years. Many films about hacking apple devices were published on YouTube and a lot of holes have beed discovered by a wide range of specialists. We can't also forget that the jailbreak law status is still unclear in some parts of the world. With the greater popularity and extending the offer apple became more attractive target to hackers and now has to deal with it, as well as its users and clients.

The introducing article: *Hacking Tools on iOS* by Alexandre Lacan doesn't refer to security but rather on the contrary. In this text you will find out which of popular hacker tools can be installed on iOS or can play with iPhone or iPad. It's a great piece for those who never considered their iPod Touch as a discreet pentest tool. In the next article: *Apple Memory Tricks,* Israel Torres will show you how by using osxmem you can safely experiment on your system to see what is floating around outside of the expected boundaries. This is definitelly something for those who enjoy run into things.

Our long time contributors: Gary S. Milefsky and Julian Evans will up-date our knowledge on apple (in)security. In their articles they point out the most popular holes and vulnerabilities in apple system. Fortunately, in the end, they both give us some advices how to tune up the Apple Software.

Just behind the back cover you will find the interview with David Harley. David is a Director of the Anti-Malware Testing Standards Organization, a Fellow of the BCS Institute, and runs the Mac Virus website. Lately he agreed to tell us about some interesting security issues he deales with in his professional life. The interview wasn't censored by any company David works for. So, it truly deserves your full attention and I hope you will find it educating and enjoyable.

There is also a nice surprise for those who love playing with tools. It this issue Tool Time column is doubled!

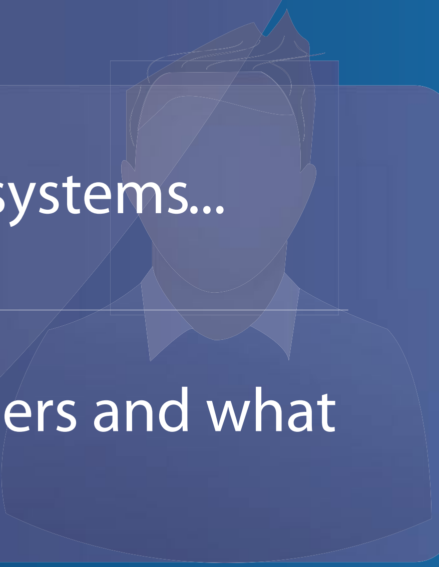
Enjoy the reading!
Patrycja Przybyłowicz
& Hakin9 team

idtheft protect

Be reactive...

- Your systems are being attacked 24 hours a day...
- You understand the threats and are protected against them...

Be proactive...



- My users' behaviour threatens our systems...
- I understand what motivates my users and what threats are coming my way...

ID Theft Protect provides information on threats from a user perspective.

Visit: **http://id-theftprotect.com**

Learn
Web Application Security
with...

Coliseum
Virtual labs
100% practical hands on training
by eLearnSecurity

FIND OUT
14 educational challenges

✓ Real world scenarios
✓ No set-up time
✓ Play on MS SQL Server
✓ Got stuck? We support!

www.coliseumlab.com

GOOGLE PATCHES 32 CHROME V.14 SECURITY FLAWS

Google has released 32 patches for its web browser search engine, Google Chrome. The bugs were identified in bug bounty and Google has paid $ 14,000 for the identification these bugs in Chrome. The company has also updated the Chrome browser edition to version 14. The 15 out of the 32 vulnerabilities are considered to be high by the company. None of Chrome's security flaws were flagged as *critical*. The Updated version 14 of chrome is available for download for Mac OS X, Windows, and Linux operating systems.

Source: ID Theft Protect

HACKERS BREAK INTO NBC TWITTER ACCOUNT

Hackers have broken into the Twitter account of NBC News and posted messages claiming that there has been a terrorist attack at Ground Zero in New York. Coming two days before the tenth anniversary of the 9/11 attacks, the prank by a group calling themselves the *script kiddies* was greeted with widespread opprobrium from other twitter users. *Breaking News! Ground Zero has just been attacked. Flight 5736 has crashed into the site, suspected hijacking. More as the story develops*, was the first tweet this afternoon. It was followed by two others, including one that started *This is not a joke*. The fourth tweet said *NBCNEWS hacked by The Script Kiddies*.

Source: ID Theft Protect

MICROSOFT DELIVERS SECURITY BULLETINS AND DIGINOTAR PATCH

Microsoft delivered 5 security bulletins (all rated „important" *http://www.id-theftprotect.com/news.php?news_id=1427*) that address 15 vulnerabilities affecting Windows, Microsoft Office and Microsoft Server Software on Patch Tuesday. In addition to that, Microsoft has also released updated security advisory and has added six more DigiNotar root certificates to its Windows Untrusted Certificate Store.

Source: ID Theft Protect

ORACLE FIX HTTP APACHE 2.0 2.2 KILLER FLAW

Oracle has patched the Apache Killer bug, which could have HTTP servers running on the Apache 2.0 and 2.2 versions. The vulnerabilities identified in the Apache Server were affecting the Oracle's offering and causing other security threats, which could allow hackers to crash their systems. The vulnerabilities are considered as critical by the company's security experts and can be easily exploited. Affected products include Oracle Fusion Middleware 11g Release 1, versions 11.1.1.3.0, 11.1.1.4.0, 11.1.1.5.0; Oracle Application Server 10g Release 3, version 10.1.3.5.0; and Oracle Application Server 10g Release 2, version 10.1.2.3.0. The database giant decided to fix the flaw ahead of its next patch update, scheduled for 18 October

Source: ID Theft Protect

ROYAL BANK OF CANADA TARGETED IN DIGINOTAR EMAIL SCAM

Cyber Criminals are targeting online banking customers by using the latest breach in Certificate Authentication in DigiNotar. The hackers are attacking the RBC Royal Bank of Canada, customers by sending them mails stating that their *Your Digital Certificate has expired* and also providing a link for upgrading it, containing the BlackHole Exploiting kit. Once the link is opened the malware steals all the login credentials and sends them to remote computers.

Source: ID Theft Protect

THE FACEBOOK FRIEND REQUEST DATA MINING PLUGIN

A new cross platform Java Facebook proof of concept (PoC) hacking plug-in data mining tool could turn your closest friends into your worst enemies – social engineering anyone? The plug-in tool (created by a team at RISST), called *Facebook Pwn http://code.google.com/p/fbpwn/*, lets Facebook criminals (and normal users like us) – steal personal profile information from any target of their choice on a social network – in this instance Facebook. A hacker would create a new blank Facebook account and then download Facebook Pwn (which is FREE). Read more... *http://www.julianevansblog.com/2011/09/the-facebook-friend-request-data-mining-plugin.html.*

Source: ID Theft Protect

US MAN FOUND GUILTY IN $3M CARD FRAUD

A 21 year old US (*http://www.id-theftprotect.com/news.php?news_id=1424*) man received a 14 year prison sentenced on Friday for running an online business that sold counterfeit credit cards encoded with stolen account information with losses estimated at more than $3 million. Tony Perez III, of Hammond, Indiana, US, pleaded guilty to the charges on April 4. In his plea, Perez said he sold counterfeit credit cards encoded with stolen account information. Perez found customers through criminal *carding forums*, Internet discussion groups set up to aid in the buying and selling of stolen financial account information and related services. When the US Secret Service raided his apartment in June 2010, they found data for 21,000 stolen credit cards and equipment needed to encode them onto blank cards. Credit card companies said losses from the card numbers in Perez's possession topped more than $3 million. In addition to the prison term, Judge Liam O'Grady of U.S. District Court for the Eastern District of Virginia (US) ordered Perez to pay $2.8 million in restitution and a $250,000 fine.

Source: ID Theft Protect

DIGINOTAR

Another certificate authority compromise has come to light; DigiNotar was broken into and several fraudulent certificates were issued as a result. DigiNotar hired a third party security auditing company and they removed what they believed to be all of the fraudulent certificates at that time but missed one. The Iranian hacker also issued himself a certificate for *google.com* and all subdomains; Gmail is believed to be the main target. It was issued on July 10th and remained active until DigiNotar finally caught it in September. DigiNotar has since revoked the certificate; Google and Mozilla also blacklisted the entire certificate authority for their internet browsers. Because of the attacks, the company petitioned for bankruptcy and was declared bankrupt by Dutch courts. This being another addition to many recent certificate authority break-ins, it is starting to raise questions in the security community. An absolute trust is currently given to certificate authorities and these attacks prove their susceptibility to abuse this trust.

By Schuyler Dorsey, eLearnSecurity

COMODO HACKER

A hacker dubbed *Comodo hacker* has claimed responsibility for many of the certificate authority compromises. He described himself as a 21 year old Iranian student in press interviews. He announced his intent of the DigiNotar attack was to punish the Dutch government for its presence and actions in Srebrenica in 1995. Along with Comodo and DigiNotar, he also claims he has access to GlobalSign and four other certificate authorities but this has yet to be confirmed. He attempted to hack StartCom in June but they were able to detect and block the attack before any fraudulent certificates had been assigned. Comodo has been using his Pastebin to announce his attacks and even post the details of the hacks. He included details of the attacks such as the private key used to create the certificate in order to prove his identity as the hacker. Many people question the independence of the Comodo hacker and question if it was actually an Iranian-state sponsored attack.

By Schuyler Dorsey, eLearnSecurity

CELEBRITY HACKS

Adding to the list of celebrity hacks, Scarlett Johansson's phone was hacked and private pictures were then leaked onto the internet. He lawyers reacted quickly and have requested all sites hosting the pictures to remove them. This recent attack brings a new question to light; how are these phones being hacked? Many independent security researchers claim they are unsure of how the compromises occur. Many of the same attacks that affect PC's also affect smart phones such as Trojans and phishing. It could have been something as simple as her clicking a malicious URL on her phone that took her to an attacker's site. Some experts suggest that the stronger possibility in this case is that she synced her phone's pictures with an online cloud service. The cloud service could've been compromised in some way whether it was an exploit of their site, a weak password or someone successfully resetting her password. With this possibility, an even greater emphasis should be placed on web application auditing with the growing use of cloud services.

By Schuyler Dorsey, eLearnSecurity

JAPAN'S MITSUBISHI HEAVY INDUSTRIES UNDER ATTACK

Japan's largest defense contractor, Mitsubishi Heavy Industries Ltd has been compromised in a recent cyber attack. Several systems were accessed and the company believes their submarine, missile and nuclear power plant components were the targets. They believe no crucial information has been leaked yet, only IP addresses. They first discovered the attack on August 11th. A MHI spokesperson stated that 45 servers and 38 workstations had been infected with malware at the time of discovery. The culprits have not been identified yet and MHI claims they do not yet have any clues. Many cyber-warfare analysts speculate it is just the first of many attacks as American defense companies have already fallen victim to many. Mitsubishi Heavy Industries Ltd manufactures many weapons including missiles, fighter jets and submarine systems.

By Schuyler Dorsey, eLearnSecurity

ANDROID MALWARE

Researchers at North Carolina State University have discovered a new strain of malware on Android 2.3 based devices called GingerMaster. It uses a recently found root exploit and it is repackaged into several legitimate apps. The researchers also noted that all current mobile anti-virus software cannot detect the attack. NetQin states, the malware elevates itself to root privilege uses the exploit. This affects all Android 2.2 and 2.3.3 devices. GingerMaster then installs a root shell into the system partition. The malware silently launches a service in the background and collects private information from the phone such as device ID, phone number and hardware info; it then uploads this information to a remote server and waits for further instructions. NetQin also noted that the malware has the ability to be installed completely without any user awareness. With the reoccurring zero day exploits being taken advantage of in mobile devices, information security professionals may need to rethink their mobile deployment strategies and place a higher emphasis on mitigating the risks involved.

By Schuyler Dorsey, eLearnSecurity

Hacking Tools on iOS

When Defcon17, in 2009, Thomas Wilhelm proposed to transform an iPod Touch into a pentest tool. Even if some tools are not available in iOS, many utilities can play with an iPhone or iPad. In this article you will learn how to install the most useful tools for hacker.

For starters, it requires a iDevice (iPod, iPhone and iPad) jailbroken. An abundant literature is available on the Internet, this article will not dwell on the subject.

A first use Cydia, we can apply a filter to display *User*, Hack *or* development. The filter *Developer* is used to filter ... nothing. Let's be crazy, and activate the filter *Developer*.

iOS Command Line

To access the command line, you must first install the OpenSSH server through Cydia, available in the *Network* (Figure 1 & Figure 2). Then, install an SSH client. Currently, the best client available is iSSH ($ 7.99 on the App Store). iSSH also has VNC, RDP, and Telnet, they can be tunneled through a SSH tunnel. There are also Pterm ($ 3.99 on the App Store). Then just connect to address 127.0.0.1 with user IDs *mobile* or *root*, the default password is *alpine*.

It is possible to install applications via the command line. To update the package list:

```
# apt-get update
```

To install the new package versions:

```
# apt-get upgrade
```

To install a package and its dependencies

```
# apt-get install <packagename>
```

To remove a package:

```
# apt-get remove <packagename>
```

To list installed packages:

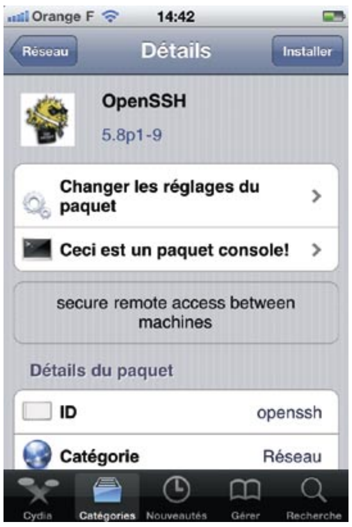
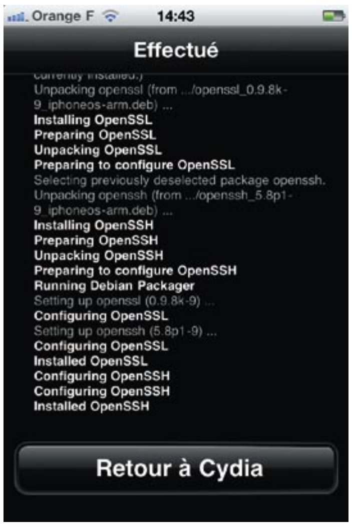


Figure 1. *Openssh_installing*



```
Unpacking openssl (from .../openssl_0.9.8k-
9_iphoneos-arm.deb) ...
Installing OpenSSL
Preparing OpenSSL
Unpacking OpenSSL
Preparing to configure OpenSSL
Selecting previously deselected package openssh.
Unpacking openssh (from .../openssh_5.8p1-
9_iphoneos-arm.deb) ...
Installing OpenSSH
Preparing OpenSSH
Unpacking OpenSSH
Preparing to configure OpenSSH
Running Debian Packager
Setting up openssl (0.9.8k-9) ...
Configuring OpenSSL
Setting up openssh (5.8p1-9) ...
Configuring OpenSSL
Installed OpenSSL
Configuring OpenSSH
Configuring OpenSSH
Installed OpenSSH
```

Figure 2. *Openssh_installing 2*

```
# dpkg-l
```

To the respring iDevice:

```
# respring
```

Secure Your SSH Access to iDevice

The first step is obviously to change the password of the two default accounts (*mobile* and *root*), because it is well known. In 2009, an Australian named Ashley Towns 21 years was one of the first to create a virus named *IKEE*, which operated the SSH connection for jailbroken iPhone and password by default, to change the wallpaper by putting a picture of a 80's singer Rick Astley. To change the password, log in as root and type:

```
# passwd mobile
# passwd
```

Be careful, It is less known that the SSH password iOS is limited to 8 characters. So if your password is: *passwords-are-very-secure-when-they-are-long-and-complex*, you can connect by typing only *password*. This significantly reduces the time to attack against a forcebrute iDevice. We are in 2011, thank you Apple.

Then it is good to change the listening port of OpenSSH by editing `/etc/sshd_config`. We will modify the line *# Port 22* in *Port 53194*. Then we can ban the authentication password by uncommenting the line *# AuthorizedKeysFile .ssh/authorized_keys* and crossing the line *# PasswordAuthentication yes* to *# PasswordAuthentication no*.

Then we'll create a pair of public/private key for authentication. On the computer, type the command `ssh-keygen-t rsa` to create two keys that are stored without the `~ /.ssh/`. He must then place the contents of the public key (extension *.pub*) in `/var/root/.ssh/authorized_keys`.

Finally, we must change the key `ProgramArguments` file `/Library/LaunchDaemons/com.openssh.sshd.plist` so that our argument takes IOS configuration file on every boot: remove the `-i` and replace `-f /etc/ssh /sshd_config`. The changes will take effect after restarting the iDevice.

Patching Holes

There are several types of jailbreak. The jailbreak type *userland* act in the software environment. If you used the tool *jailbreakme.com* of Comex, you exploited a flaw in the PDF reader of the iOS. This flaw exists even after the jailbreak, so you have the patch it. To correct the vulnerability, install via Cydia *PDF Patcher 2*. Otherwise, an attacker can take control of the iPhone simply by sending you a pdf by email.

Monitor Active Connections

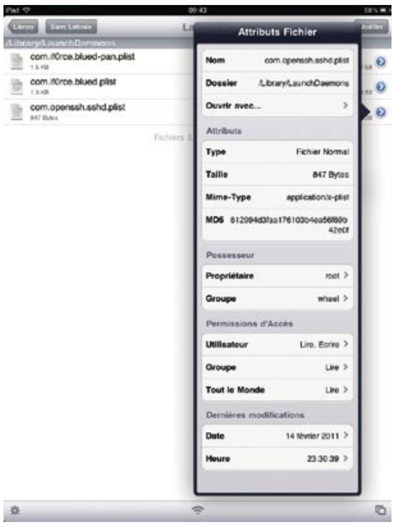
Firewall IP is a tool fee ($ 4.49) available on Cydia. Whenever an application tries to connect to a server, an alert appears on the screen to select the desired action. This software does not enforce rules on the form of IP packets, but only to control network access of installed applications.

The Essential Command Line

In Cydia, many components can be useful: network-cmds, adc-cmds, sudo, top, Make, Patch, Vi, Aptitude, Lynx, Python, Python Setup Tools, Ruby, cURL, inetutils, wget, subversion, Background (to pass any application in the background), Insomnia (For the wifi is always active), interpreters ... The installation of Ruby, Python or Perl (1) is essential for the installation of other tools written in these languages.

ifile – The Indispensable

This is the indispensable tool for digging into the iDevice. Available in Cydia for $ 4, this software support many file types: txt, plist, pdf, sound, text, zip. It allows



Nom com.openssh.sshd.plist
Dossier /Library/LaunchDaemons

Attributs
Type Fichier Normal
Taille 847 Bytes
Mime-Type application/x-plist
MD5 612094d3faa1761030b4ea56f89o42e0f

Possesseur
Propriétaire root
Groupe wheel

Permissions d'Accès
Utilisateur Lire, Ecrire
Groupe Lire
Tout le Monde Lire

Dernières modifications
Date 14 février 2011
Heure 23:30:39

Figure 3. *ifile_iPad*

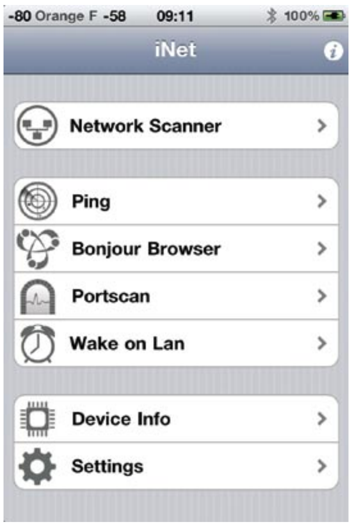


Figure 4. *iNet_iPhone*

you to navigate the file system, via a graphical interface. It is very convenient to edit configuration files or access data in your applications. Only drawback, it still lacks a sqlite viewer to be perfect (Figure 3).

Available on The Appstore

It is well known that Apple is closely monitoring the *morality* of applications posted on the Appstore. However, one can find some interesting tools:

inet inet and pro ($ 4.99)
iNet scan the network, and the port (only port 1 to 1000) and show the devices connected to your local network, With their name, Ip address and Vendor (Figure 4).

Nessus
Only the client Nessus is available on the appstore. It is therefore not possible to scan from an iPhone, but it can view the audit trail and create new ones.

zScan ($ 2.39)
zScan Is A fast network scanner with multithreading support. It has built-in application and database software banner grabbing version detection. Also it has vulnerabilities detection plugins. It works on wifi, 3G, Edge and GPRS. Unfortunately, there are few updates and only 4 plugins are available.

Other software is available, practical but always respect the moral as *apple* (Figure 5).

Available on Cydia

Pirni – Sniffer and ARP Spoofing
Freely available in text mode, or paid ($1.99) in graphical mode. Pirni is a native network sniffer for iPhone. Pirni comes with an ARP spoofer that routes all the network traffic through your iDevice and then uses packet forwarding to send it to it's normal recipent (ie. the router) After a successful network sniffing, you

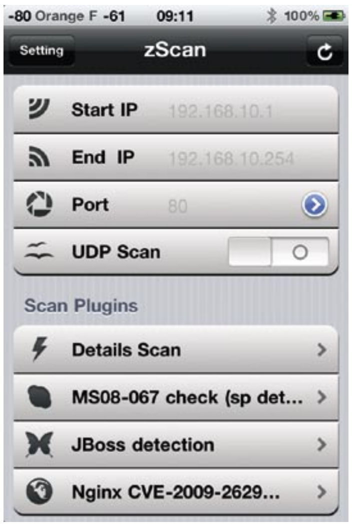
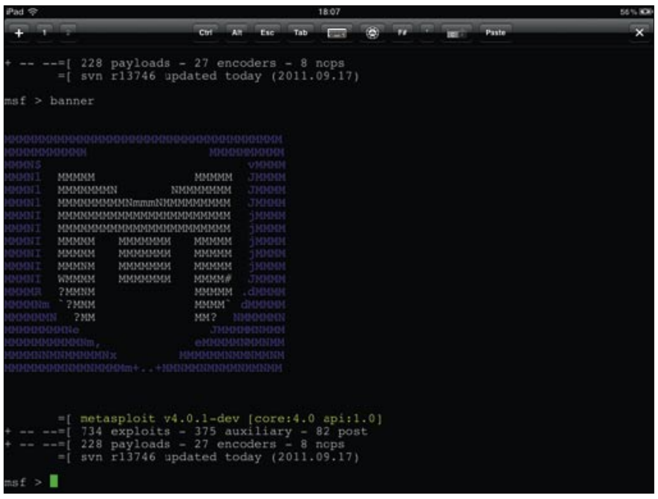


Figure 5. *zSQcan_iPhone*



```
+ -- --=[ 228 payloads - 27 encoders - 8 nops
       =[ svn r13746 updated today (2011.09.17)

msf > banner

MMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMM
MMMMMMMMMMMM                MMMMMMMMMMMM
MMMN$                              vMMMM
MMMNl  MMMMM              MMMMM   JMMMM
MMMNl  MMMMMMMN       NMMMMMMM   JMMMM
MMMNl  MMMMMMMMMNmmmNMMMMMMMMM   JMMMM
MMMNI  MMMMMMMMMMMMMMMMMMMMMMM   jMMMM
MMMNI  MMMMMMMMMMMMMMMMMMMMMMM   jMMMM
MMMNI  MMMNM    MMMMMMM    MMMMM   jMMMM
MMMNI  MMMNM    MMMMMMM    MMMMM   jMMMM
MMMNI  MMMNM    MMMMMMM    MMMMM   jMMMM
MMMNI  WMMMM    MMMMMMM    MMMM#   JMMMM
MMMMR  ?MMNM               MMMMM .dMMMM
MMMMNm `?MMM               MMMM` dMMMMM
MMMMMMN  ?MM               MM?  NMMMMMN
MMMMMMMMNe                    JMMMMMNMMM
MMMMMMMMMMNm,              eMMMMMNMMNMM
MMMMNNMNMMMMMNx        MMMMMMNMMNMMNM
MMMMMMMMNMMNMMMMm+..+MMNMMNMNMMNMMNMM

       =[ metasploit v4.0.1-dev [core:4.0 api:1.0]
+ -- --=[ 734 exploits - 375 auxiliary - 82 post
+ -- --=[ 228 payloads - 27 encoders - 8 nops
       =[ svn r13746 updated today (2011.09.17)

msf >
```

Figure 6. *Metasploit_iPad*

can transfer the dumpfile to your computer and open it up with Wireshark (or any other traffic analyzer that supports pcap) to analyze the traffic.

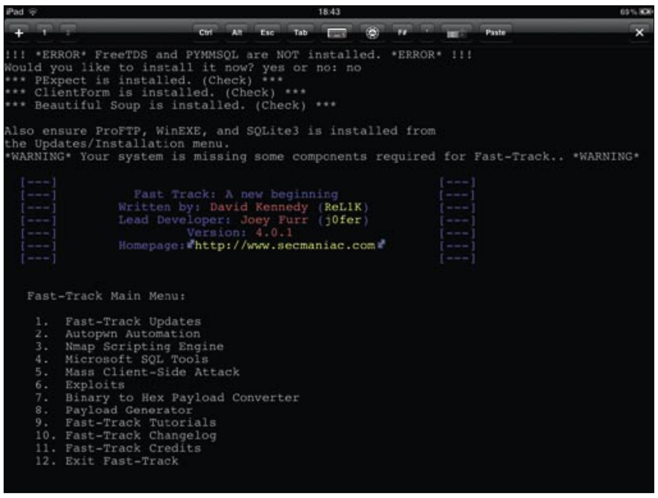
Although there is a version of aircrack ported IOS (by installing the repositories *http://cydia.xsellize.com*), the iOS's wifi has some major drawbacks in it's hardware design, thus we can not properly set the device in promiscious mode. Do not imagine to break a wireless network with just an iPhone.

Nmap

Nmap has been ported to iOS. There is only version 5.00, rarely updated. But it is very useful for running a discovery of hosts, a computer scanner. One day I was able to access the internal network of a fast food restaurant with a simple nmap scan on the iPhone on the wireless network open to the public.

Metasploit

The most popular pentest tools is also available in IOS. Attention, Metasploit does not work with the version of Ruby 1.9 available on Cydia. If you installed it, it must be uninstalled and then install version 1.8. After



```
!!! *ERROR* FreeTDS and PYMMSQL are NOT installed. *ERROR* !!!
Would you like to install it now? yes or no: no
*** PExpect is installed. (Check) ***
*** ClientForm is installed. (Check) ***
*** Beautiful Soup is installed. (Check) ***

Also ensure ProFTP, WinEXE, and SQLite3 is installed from
the Updates/Installation menu.
*WARNING* Your system is missing some components required for Fast-Track.. *WARNING*

  [---]                                                 [---]
  [---]         Fast Track: A new beginning             [---]
  [---]       Written by: David Kennedy (ReL1K)         [---]
  [---]       Lead Developer: Joey Furr (j0fer)         [---]
  [---]                Version: 4.0.1                   [---]
  [---]       Homepage: http://www.secmaniac.com        [---]
  [---]                                                 [---]


   Fast-Track Main Menu:

    1.  Fast-Track Updates
    2.  Autopwn Automation
    3.  Nmap Scripting Engine
    4.  Microsoft SQL Tools
    5.  Mass Client-Side Attack
    6.  Exploits
    7.  Binary to Hex Payload Converter
    8.  Payload Generator
    9.  Fast-Track Tutorials
    10. Fast-Track Changelog
    11. Fast-Track Credits
    12. Exit Fast-Track
```

Figure 7. *Fasttrack_iPad*

installation, be careful and do not update for ruby and rubygems Cydia (in the settings of the package, it is possible to ignore the updates).

```
# apt-get remove ruby rubygems
# cp /private/var/
# wget http://apt.saurik.com/dists/tangelo-0.9/main/
binary-iphoneos-arm/debs/ruby_1.8.6-p111-5_iphoneos-arm.deb
# dpkg -i ruby_1.8.6-p111-5_iphoneos-arm.deb
# wget http://apt.saurik.com/dists/tangelo-0.9/main/
binary-iphoneos-arm/debs/rubygems_1.2.0-3_iphoneos-arm.deb
# dpkg -i rubygems_1.2.0-3_iphoneos-arm.deb
# rm -f ruby*
```

Then download and install Metasploit:

```
# wget http://updates.metasploit.com/data/releases/
                    framework-4.0.0.tar.bz2
# tar jxpf framework-4.0.0.tar.bz2
# cd msf3
# svn update
# ./msfconsole
```

enjoy and have fun! (Figure 6)

Social_Engineering_Toolkit and Fast-Track

It is possible to integrate SET (*Social Engineering Toolkit*) to Metasploit iOS.

```
# cd /private/var
# svn co http://svn.secmaniac.com/social_engineering_
                    toolkit/ SET/
```

Edit the `/private/var/SET/config/set _ config` giving the msf path variable the path `/private/var/msf3` (Figure 7).

```
# svn co http://svn.secmaniac.com/fasttrack fasttrack
# cd fasttrack
```

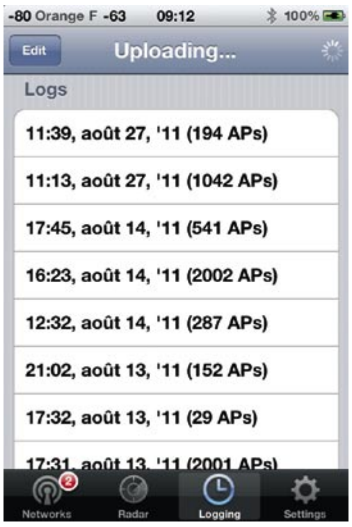

-80 Orange F -63 09:12 100%
Uploading...
Logs
11:39, août 27, '11 (194 APs)
11:13, août 27, '11 (1042 APs)
17:45, août 14, '11 (541 APs)
16:23, août 14, '11 (2002 APs)
12:32, août 14, '11 (287 APs)
21:02, août 13, '11 (152 APs)
17:32, août 13, '11 (29 APs)
17:31, août 13, '11 (2001 APs)

Figure 8. *Wififofum_iPhone*

References:
Debugging on iPhone – *http://blogs.csoonline.com/a_new_iphone_hacking_tool*
Hacking with the iPad Touch – *https://www.defcon.org/images/defcon-17/dc-17-presentations/defcon-17-thomas_wilhelm-hacking_ipod_touch.pdf*
Penetration Testing for iPhone/iPad Applications – *http://www.mcafee.com/us/resources/white-papers/foundstone/wp-pen-testing-iphone-ipad-apps.pdf*
(1) How to install Perl on iOS *http://ihackerpro.com/forum/index.php?topic=1287.0*

```
# python setup.py install
# python ./fast-track -i
```

TetherMe ($ 4.99 in Cydia)

If you need to enable sharing on your iPhone connexion, TetherMe is able to use the mechanisms of Apple and to enable tethering. Then you can share your Internet access USB, WiFi or Bluetooth.

Wardriving

WiFiFoFum (coupled with insomnia + backgrounder) is one of my favorite tools. WiFiFoFum detects the wifi signal and store their GPS position. You can put your iPhone in listening, a drive in the car fun to map access points. The software allows you to export your card as a gateway to the Google Earth KML format (Figure 8).

Conclusion

Many other tools are available to suit your needs:

- Netcat
- Dns2tcp that provides encapsulation of TCP session in DNS packets
- StealthMac which assigns a random MAC address each time you start
- Stunnel for creating a secure SSL tunnel
- TCPDump
- Nikto, a Web vulnerable scanner
- Medusa
- John the Ripper
- PenTBox

One day I was asked if the iPhone is a good phone. I said, *it's not a phone, but a real computer with a telephony function.* Its portability and discretion make a very good tool pentest in certain situations where discretion is important.

ALEXANDRE LACAN
lades@canett.com | http://twitter.com/lades51 | http://www.canett.com

Apple Memory Tricks

As a researcher it is always fun to run into things while playing with unexpected behavior. Recently while implementing a crypto challenge in C on my Mac running Lion I noticed that some of the resultant output was not matching the expected test input. Figuring it was a fluke I didn't think about it too much more until curiosity got the best of me a few days later...

What you will learn...

- Using C to passively play with live Apple Memory in Terminal Shell

What you should know...

- Basic programming using gcc and an understanding of OS memory

Going back to the code I noticed that sometimes I would get Segmentation Faults (aka segfaults). A lot of the stuff I play with just covers the 'happy path' and doesn't delve into error handling. Sure it's a horrible coding practice but it's not all bad in the sandbox (where it just needs to exist and die for the moment). I ended up narrowing down and writing out the stub of interest in under 15 lines for added functionality. Because it shows the user what is running/had run in memory it aptly named it *osxmem.c* (Listing 1). A really simple and passive way to poke around memory and get interesting results.

Testing osxmem was tedious at first due to not knowing where limitations existed. It just seemed that the more I looked around the more I found. Enter *osxmem-chunks.sh* (Listing 2) I created an artificial limit of 70k sequential entries (0-69,999) and cut each chunk into 10k chunks; allowing me to keep an eye as chunks were created and not having to wait for the entire output log to complete. Naturally the limits can be modified in the bash script via `$FSTART`, `$FFINIS` and `$FCHUNK`. Note for it to make sense on larger numbers you'll need to update the data type in osxmem accordingly.

After the first run of osxmem-chunks there appeared to be consistency with the first 26 entries before diving into the ocean of segfaults and nulls (looking at Console as Crash Report spews all the segfaults is interesting to say the least) (Figure 1). Pages-deep there would be chunks within the chunks that logged more data running in memory; stagnate, in use, etc. The resultant data appears to be what was going on within the Terminal Shell at the time it was running. (In other news don't strictly rely on the data being there the next time you go looking for it as it appears to be in flux and things load and unload – don't be surprised if you log something tangible during the batch process only to get a null value in return when attempting to access it independently.)

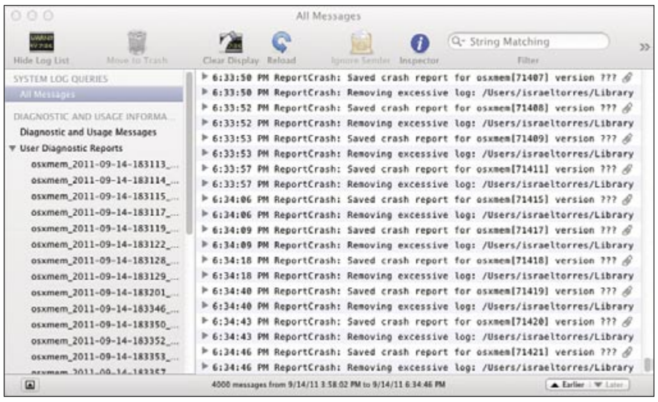
When osxmem-chunks completed I went through the logs and was able to find sequential segments that were still available via osxmem. Thinking about creating a filter based off osxmem (where it just ran through the default 26 integers and the arbitrary input) I created an extended version named *osxmem-react.c* (Listing 3) which scans portions of desired areas and acts upon either the filter itself. In the included source you can enter `./osxmem-react VARIABLE= (e.g. ./osxmem-react LOGNAME= will` return the account name; in this case 'israeltorres' – by not including the = sign you'll get it in the returned data) and get the resultant back of what the variable is assigned to. It can be further modified to trigger additional functions based on other segments and the data they contain for their entries. For example interested in see if `__dyld_link_edit_error` is floating around – you may run into it around sequence 27608 (Figure 2). Need to verify something? Throw it into memory and see if you can find it.

Listing 1. *Source osxmem*

```
/*
 osxmem.c
 Israel Torres hakin9@israeltorres.org
 Wed Sep 14 14:22:07 PDT 2011
 "Apple Memory Tricks"
 demo to:
 play with argv[] results (list desired variable based on integer input)
 compile with:
 gcc osxmem.c -o osxmem && ./osxmem (use with osxmem-chunks.sh for fun results)
 created and tested on gcc version 4.2.1 / Mac OS X 10.7.1 11B26
 */
#include <stdio.h>
int main (int argc, char * argv[]){
    int x = 0;
    if (argc == 2){ // enter integer value to target
    x = atoi(argv[1]);
    printf("%d\t%s\n",x,argv[x]);
    }
    else{  // list known 26 variables in memory
    for (;x < 27; x++)
        printf("%d\t%s\n",x,argv[x]);
    }
return 0;
}
// EOF
```

Listing 2. *Source osxmem-chunks*

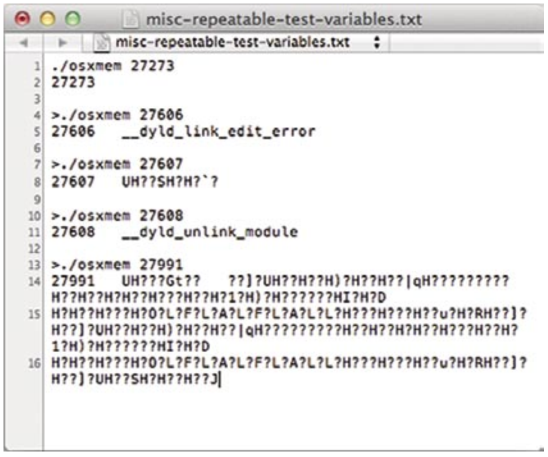
```
#!/bin/bash
# ./osxmem-chunks
# Israel Torres hakin9@israeltorres.org
# Wed Sep 14 14:22:07 PDT 2011
# "Apple Memory Tricks"
# this runs ./osxmem [integer] and saves results in 10k chunks
# use in bash prompt: ./osxmem-chunks.sh (results in ~2 hours)
#
FSTART=0; FFINIS=9; FCHUNK=10000
for (( i = 0; i < 70000; i++ ))
do
   if [ $(( $i % $FCHUNK )) -eq 0 ]; then
      FSTART=$i
      FFINIS=$(($FSTART+$FCHUNK-1))
   fi
   ./osxmem $i >> osxmem-$FSTART-$FFINIS-output.txt
done
#EOF
```


All Messages
Hide Log List
Move to Trash
Clear Display
Reload
Ignore Sender
Inspector
String Matching
Filter
SYSTEM LOG QUERIES
All Messages
DIAGNOSTIC AND USAGE INFORMA...
Diagnostic and Usage Messages
User Diagnostic Reports
osxmem_2011-09-14-183113_...
osxmem_2011-09-14-183114_...
osxmem_2011-09-14-183115_...
osxmem_2011-09-14-183117_...
osxmem_2011-09-14-183119_...
osxmem_2011-09-14-183122_...
osxmem_2011-09-14-183128_...
osxmem_2011-09-14-183129_...
osxmem_2011-09-14-183201_...
osxmem_2011-09-14-183346_...
osxmem_2011-09-14-183350_...
osxmem_2011-09-14-183352_...
osxmem_2011-09-14-183353_...
Earlier
Later

```
6:33:50 PM ReportCrash: Saved crash report for osxmem[71407] version ???
6:33:50 PM ReportCrash: Removing excessive log: /Users/israeltorres/Library
6:33:52 PM ReportCrash: Saved crash report for osxmem[71408] version ???
6:33:52 PM ReportCrash: Removing excessive log: /Users/israeltorres/Library
6:33:53 PM ReportCrash: Saved crash report for osxmem[71409] version ???
6:33:53 PM ReportCrash: Removing excessive log: /Users/israeltorres/Library
6:33:57 PM ReportCrash: Saved crash report for osxmem[71411] version ???
6:33:57 PM ReportCrash: Removing excessive log: /Users/israeltorres/Library
6:34:06 PM ReportCrash: Saved crash report for osxmem[71415] version ???
6:34:06 PM ReportCrash: Removing excessive log: /Users/israeltorres/Library
6:34:09 PM ReportCrash: Saved crash report for osxmem[71417] version ???
6:34:09 PM ReportCrash: Removing excessive log: /Users/israeltorres/Library
6:34:18 PM ReportCrash: Saved crash report for osxmem[71418] version ???
6:34:18 PM ReportCrash: Removing excessive log: /Users/israeltorres/Library
6:34:40 PM ReportCrash: Saved crash report for osxmem[71419] version ???
6:34:40 PM ReportCrash: Removing excessive log: /Users/israeltorres/Library
6:34:43 PM ReportCrash: Saved crash report for osxmem[71420] version ???
6:34:43 PM ReportCrash: Removing excessive log: /Users/israeltorres/Library
6:34:46 PM ReportCrash: Saved crash report for osxmem[71421] version ???
6:34:46 PM ReportCrash: Removing excessive log: /Users/israeltorres/Library
```

4000 messages from 9/14/11 3:58:02 PM to 9/14/11 6:34:46 PM

Figure 1. *Console-Segfaults*


misc-repeatable-test-variables.txt
misc-repeatable-test-variables.txt

```
./osxmem 27273
27273

>./osxmem 27606
27606   __dyld_link_edit_error

>./osxmem 27607
27607   UH??SH?H?`?

>./osxmem 27608
27608   __dyld_unlink_module

>./osxmem 27991
27991   UH???Gt??   ??]?UH??H??H)?H??H??|qH?????????
H??H??H?H??H???H??H?1?H)?H??????HI?H?D
H?H??H???H?O?L?F?L?A?L?F?L?A?L?L?H???H???H??u?H?RH??]?
H??]?UH??H??H)?H??H??|qH?????????H??H??H?H??H???H??H?
1?H)?H??????HI?H?D
H?H??H???H?O?L?F?L?A?L?F?L?A?L?L?H???H???H??u?H?RH??]?
H??]?UH??SH?H??H??J
```

Figure 2. *misc repeatable test variables*

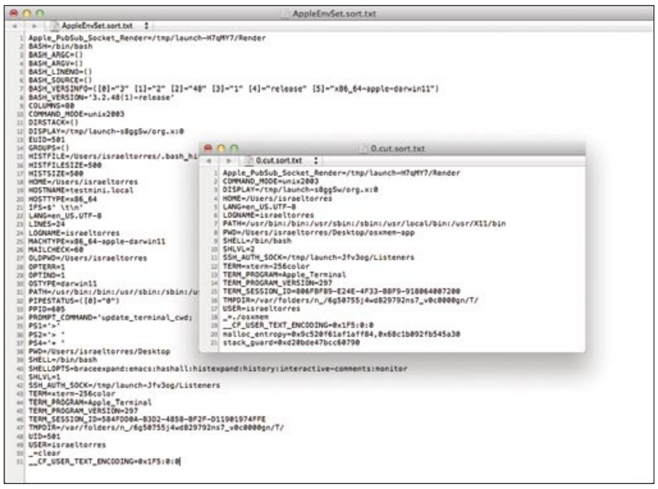
Demonstration

```
#build and run osxmem.c
gcc osxmem.c -o osxmem.c && ./osxmem

#build and run osxmem-react.c
gcc osxmem-react.c -o osxmem-react.c && ./osxmem-react
```

```
#run osxmem in 10k chunks using osxmem-chunks.sh script
(and getting also the length of time involved)
time (./osxmem-chunks.sh)
```

Listing 3. *Source osxmem-react*

```
/*

 Israel Torres hakin9@israeltorres.org
 Wed Sep 14 14:22:07 PDT 2011
 "Apple Memory Tricks"
 demo to:
 play with argv[] results (and reacting upon found data)
 compile with:
 gcc osxmem-react.c -o osxmem-react && ./osxmem-react
 created and tested on gcc version 4.2.1 / Mac OS X 10.7.1 11B26
 */
#include <string.h>
#include <stdio.h>
int main (int argc, char * argv[]){
    int x = 0;
    if (argc == 2){
        char * you, test[256]="\0"; // adjust accordingly
        strcpy(test,argv[1]); // works in a pinch (not recommended)
        int testlen=strlen(test);
        for (;x < 27; x++) // search through known 26 list
            if ((argv[x]!=NULL) && (you = strnstr(argv[x],test,testlen)))
            printf("%s",you+testlen); // list content of desired variable
    }
    else{ // list known 26 variables in memory
        for (;x < 27; x++)
            printf("%d\t%s\n",x,argv[x]);
    }
return 0;
}
// EOF
```



```
AppleEnvSet.sort.txt
Apple_PubSub_Socket_Render=/tmp/launch-H7qMY7/Render
BASH=/bin/bash
BASH_ARGC=()
BASH_ARGV=()
BASH_LINENO=()
BASH_SOURCE=()
BASH_VERSINFO=([0]="3" [1]="2" [2]="48" [3]="1" [4]="release" [5]="x86_64-apple-darwin11")
BASH_VERSION='3.2.48(1)-release'
COLUMNS=80
COMMAND_MODE=unix2003
DIRSTACK=()
DISPLAY=/tmp/launch-s8gg5w/org.x:0
EUID=501
GROUPS=()
HISTFILE=/Users/israeltorres/.bash_hi
HISTFILESIZE=500
HISTSIZE=500
HOME=/Users/israeltorres
HOSTNAME=testmini.local
HOSTTYPE=x86_64
IFS=$' \t\n'
LANG=en_US.UTF-8
LINES=24
LOGNAME=israeltorres
MACHTYPE=x86_64-apple-darwin11
MAILCHECK=60
OLDPWD=/Users/israeltorres
OPTERR=1
OPTIND=1
OSTYPE=darwin11
PATH=/usr/bin:/bin:/usr/sbin:/sbin:/u
PIPESTATUS=([0]="0")
PPID=605
PROMPT_COMMAND='update_terminal_cwd;
PS1='>'
PS2='> '
PS4='+ '
PWD=/Users/israeltorres/Desktop
SHELL=/bin/bash
SHELLOPTS=braceexpand:emacs:hashall:histexpand:history:interactive-comments:monitor
SHLVL=1
SSH_AUTH_SOCK=/tmp/launch-Jfv3og/Listeners
TERM=xterm-256color
TERM_PROGRAM=Apple_Terminal
TERM_PROGRAM_VERSION=297
TERM_SESSION_ID=584FD00A-83D2-4858-8F2F-D11901974FFE
TMPDIR=/var/folders/n_/6g50755j4wd829792ns7_v0c0000gn/T/
UID=501
USER=israeltorres
_=clear
__CF_USER_TEXT_ENCODING=0x1F5:0:0
```

```
0.cut.sort.txt
Apple_PubSub_Socket_Render=/tmp/launch-H7qMY7/Render
COMMAND_MODE=unix2003
DISPLAY=/tmp/launch-s8gg5w/org.x:0
HOME=/Users/israeltorres
LANG=en_US.UTF-8
LOGNAME=israeltorres
PATH=/usr/bin:/bin:/usr/sbin:/sbin:/usr/local/bin:/usr/X11/bin
PWD=/Users/israeltorres/Desktop/osxmem-app
SHELL=/bin/bash
SHLVL=2
SSH_AUTH_SOCK=/tmp/launch-Jfv3og/Listeners
TERM=xterm-256color
TERM_PROGRAM=Apple_Terminal
TERM_PROGRAM_VERSION=297
TERM_SESSION_ID=806FBF89-E24E-4F33-88F9-918064007200
TMPDIR=/var/folders/n_/6g50755j4wd829792ns7_v0c0000gn/T/
USER=israeltorres
_=./osxmem
__CF_USER_TEXT_ENCODING=0x1F5:0:0
malloc_entropy=0x9c520f61af1aff84,0x68c1b092fb545a30
stack_guard=0xd20bde47bcc60790
```

Figure 3. *Apple Environment Variables v osxmem default results*

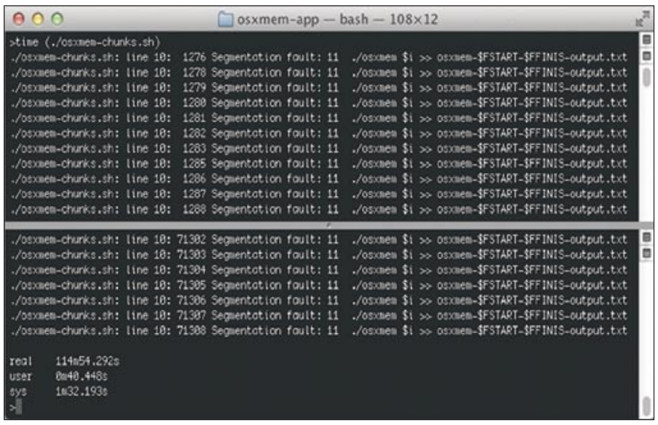
Comparing the set environmental variable assignments in Terminal Shell (Figure 3).

Ignoring `update_terminal_cwd ()` (containing 3 local assignments) we have a total of 51 environmental assignments:

```
# non-artifact method
set | grep -c =
51

# artifact method
set > AppleEnvSet.txt; cat AppleEnvSet.txt | grep -c =
51
```

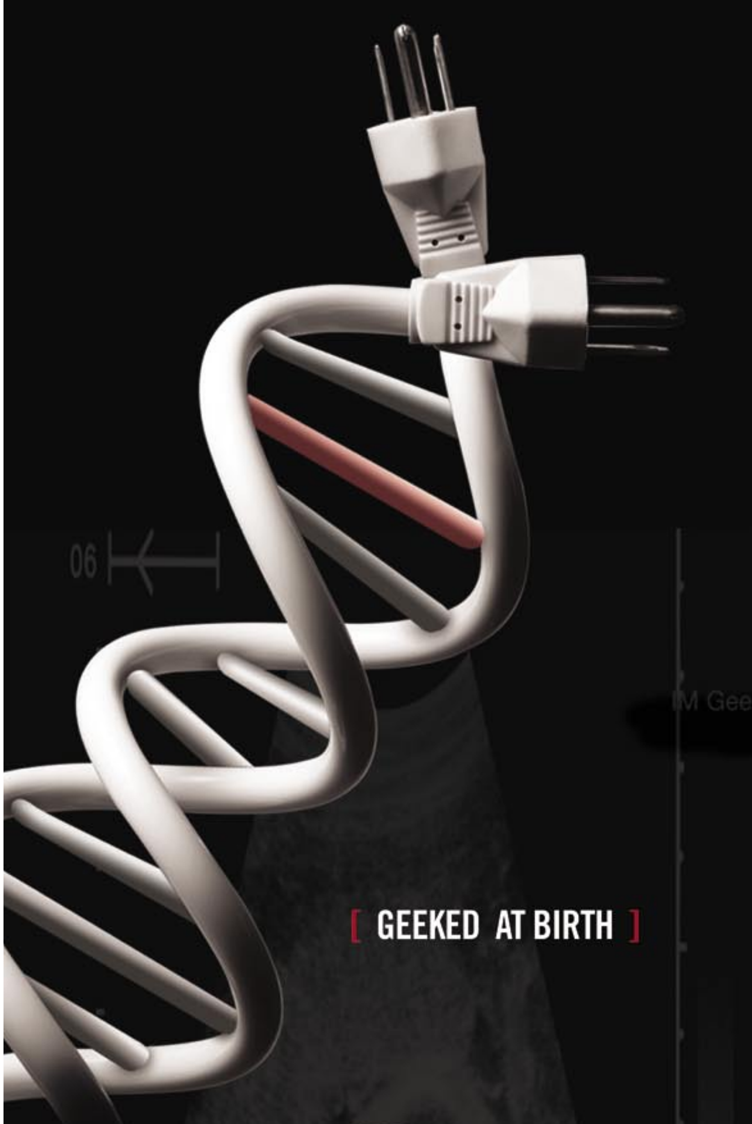
Using osxmem out of the list of 26 resultant data there are 21 assignments found:

```
# non-artifact method
./osxmem | grep -c =
21

# artifact method
osxmem > 0.txt
cat 0.txt | cut -f 2 | grep -c =
21
```



```
osxmem-app — bash — 108×12
>time (./osxmem-chunks.sh)
./osxmem-chunks.sh: line 10:  1276 Segmentation fault: 11  ./osxmem $i >> osxmem-$FSTART-$FFINIS-output.txt
./osxmem-chunks.sh: line 10:  1278 Segmentation fault: 11  ./osxmem $i >> osxmem-$FSTART-$FFINIS-output.txt
./osxmem-chunks.sh: line 10:  1279 Segmentation fault: 11  ./osxmem $i >> osxmem-$FSTART-$FFINIS-output.txt
./osxmem-chunks.sh: line 10:  1280 Segmentation fault: 11  ./osxmem $i >> osxmem-$FSTART-$FFINIS-output.txt
./osxmem-chunks.sh: line 10:  1281 Segmentation fault: 11  ./osxmem $i >> osxmem-$FSTART-$FFINIS-output.txt
./osxmem-chunks.sh: line 10:  1282 Segmentation fault: 11  ./osxmem $i >> osxmem-$FSTART-$FFINIS-output.txt
./osxmem-chunks.sh: line 10:  1283 Segmentation fault: 11  ./osxmem $i >> osxmem-$FSTART-$FFINIS-output.txt
./osxmem-chunks.sh: line 10:  1285 Segmentation fault: 11  ./osxmem $i >> osxmem-$FSTART-$FFINIS-output.txt
./osxmem-chunks.sh: line 10:  1286 Segmentation fault: 11  ./osxmem $i >> osxmem-$FSTART-$FFINIS-output.txt
./osxmem-chunks.sh: line 10:  1287 Segmentation fault: 11  ./osxmem $i >> osxmem-$FSTART-$FFINIS-output.txt
./osxmem-chunks.sh: line 10:  1288 Segmentation fault: 11  ./osxmem $i >> osxmem-$FSTART-$FFINIS-output.txt

./osxmem-chunks.sh: line 10: 71302 Segmentation fault: 11  ./osxmem $i >> osxmem-$FSTART-$FFINIS-output.txt
./osxmem-chunks.sh: line 10: 71303 Segmentation fault: 11  ./osxmem $i >> osxmem-$FSTART-$FFINIS-output.txt
./osxmem-chunks.sh: line 10: 71304 Segmentation fault: 11  ./osxmem $i >> osxmem-$FSTART-$FFINIS-output.txt
./osxmem-chunks.sh: line 10: 71305 Segmentation fault: 11  ./osxmem $i >> osxmem-$FSTART-$FFINIS-output.txt
./osxmem-chunks.sh: line 10: 71306 Segmentation fault: 11  ./osxmem $i >> osxmem-$FSTART-$FFINIS-output.txt
./osxmem-chunks.sh: line 10: 71307 Segmentation fault: 11  ./osxmem $i >> osxmem-$FSTART-$FFINIS-output.txt
./osxmem-chunks.sh: line 10: 71308 Segmentation fault: 11  ./osxmem $i >> osxmem-$FSTART-$FFINIS-output.txt

real    114m54.292s
user    0m40.448s
sys     1m32.193s
>
```

Figure 4. *Using osxmem-chunks.sh to segfault*



[GEEKED AT BIRTH]

University of Advancing Technology
UAT
Learn. Experience. Innovate.

You can talk the talk.
Can you walk the walk?

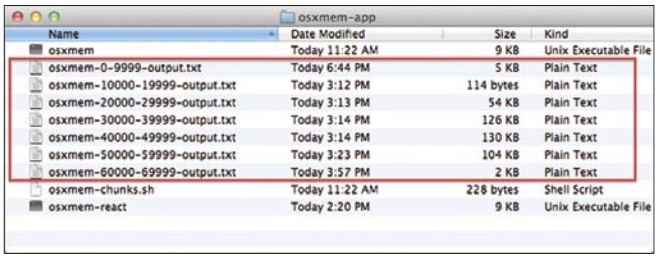
[IT'S IN YOUR DNA]

LEARN:

Advancing Computer Science	Network Security
Artificial Life Programming	Open Source Technologies
Digital Media	Robotics and Embedded Systems
Digital Video	Serious Games and Simulation
Enterprise Software Development	Strategic Technology Development
Game Art and Animation	Technology Forensics
Game Design	Technology Product Design
Game Programming	Technology Studies
Human-Computer Interaction	Virtual Modeling and Design
Network Engineering	Web and Social Media Technologies

www.uat.edu > 877.UAT.GEEK

Please see www.uat.edu/fastfacts for the latest information about degree program performance, placement and costs.



osxmem-app

Name	Date Modified	Size	Kind
osxmem	Today 11:22 AM	9 KB	Unix Executable File
osxmem-0-9999-output.txt	Today 6:44 PM	5 KB	Plain Text
osxmem-10000-19999-output.txt	Today 3:12 PM	114 bytes	Plain Text
osxmem-20000-29999-output.txt	Today 3:13 PM	54 KB	Plain Text
osxmem-30000-39999-output.txt	Today 3:14 PM	126 KB	Plain Text
osxmem-40000-49999-output.txt	Today 3:14 PM	130 KB	Plain Text
osxmem-50000-59999-output.txt	Today 3:23 PM	104 KB	Plain Text
osxmem-60000-69999-output.txt	Today 3:57 PM	2 KB	Plain Text
osxmem-chunks.sh	Today 11:22 AM	228 bytes	Shell Script
osxmem-react	Today 2:20 PM	9 KB	Unix Executable File

Figure 5. *Osxmem-chunks.sh output*

Seeing that the return data looked similar enough to the environment variables a line by line comparison was needed to see which variables were resident and available via osxmem. Here we get both *AppleEnvSet.txt* and *0.txt* into similar formats for line by line comparison:

```
sort AppleEnvSet.txt > AppleEnvSet.sort.txt
cat 0.txt | cut -f 2 | grep = > 0.cut.txt
sort 0.cut.txt > 0.cut.sort.txt
```

We then compare and display only common matching lines:

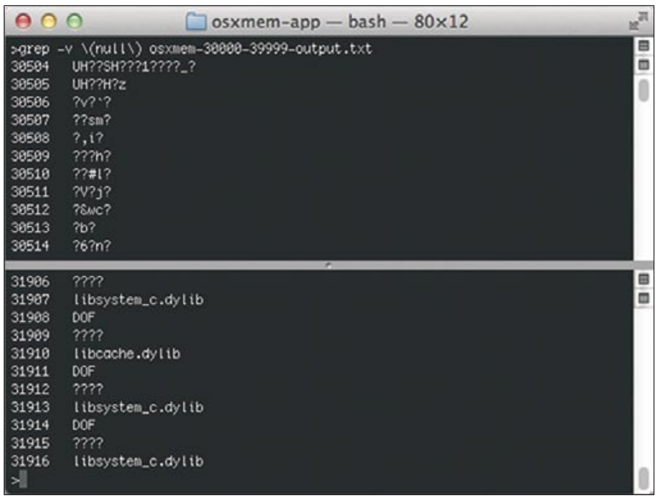
```
comm -12 AppleEnvSet.sort.txt 0.cut.sort.txt | grep -c =
15
```

Then to check line by line and display only the variable names and not the assigned values:

```
comm -12 AppleEnvSet.sort.txt 0.cut.sort.txt | cut -d =
                    -f 1
```

So out of the 51 environmental assignments osxmem finds 15 of them which are the following (in alphabetical order):

```
Apple_PubSub_Socket_Render
COMMAND_MODE
DISPLAY
```



```
>grep -v \(null\) osxmem-30000-39999-output.txt
30504   UH??SH???1????_?
30505   UH??H?z
30506   ?v?`?
30507   ??sm?
30508   ?,i?
30509   ???h?
30510   ??#l?
30511   ?V?j?
30512   ?&wc?
30513   ?b?
30514   ?6?n?

31906   ????
31907   libsystem_c.dylib
31908   DOF
31909   ????
31910   libcache.dylib
31911   DOF
31912   ????
31913   libsystem_c.dylib
31914   DOF
31915   ????
31916   libsystem_c.dylib
>
```

Figure 6. *Using grep to filter out null returns*

Notes

All source code created and tested on:
Mac OS X 10.7.1 11B26
Intel Core 2 Duo 2.93 GHz 4 GB
Darwin Kernel Version 11.1.0
GNU bash, version 3.2.48(1)-release
gcc version 4.2.1

Got More Time Than Money?

Try this month's crypto challenge:
http://hakin9.israeltorres.org

```
HOME
LANG
LOGNAME
PATH
SHELL
SSH_AUTH_SOCK
TERM
TERM_PROGRAM
TERM_PROGRAM_VERSION
TMPDIR
USER
__CF_USER_TEXT_ENCODING
```

Now using osxmem-react `VARIABLE=` we can now see the variable assignments easily

```
./osxmem-react LOGNAME=
which returns: 'israeltorres'
```

Using *osxmem-chunks.sh* we can then delve further into the *unknown* (Figure 4). The complete chunk cycle from 0 – 69999 (into 7 chunks) takes just under two hours (114m54.292s) (based on processor/processes) and also threw up 38,923 segfaults! (Figure 5)

To quickly filter resultant logs filter null returns using grep:

```
grep -v \(null\) osxmem-30000-39999-output.txt (Figure 6)
```

Conclusion

While not for everyone it is always interesting to see what is happening in the background and delve into the unknown. Using osxmem you can safely experiment on your system and see what is floating around outside of the expected boundaries.

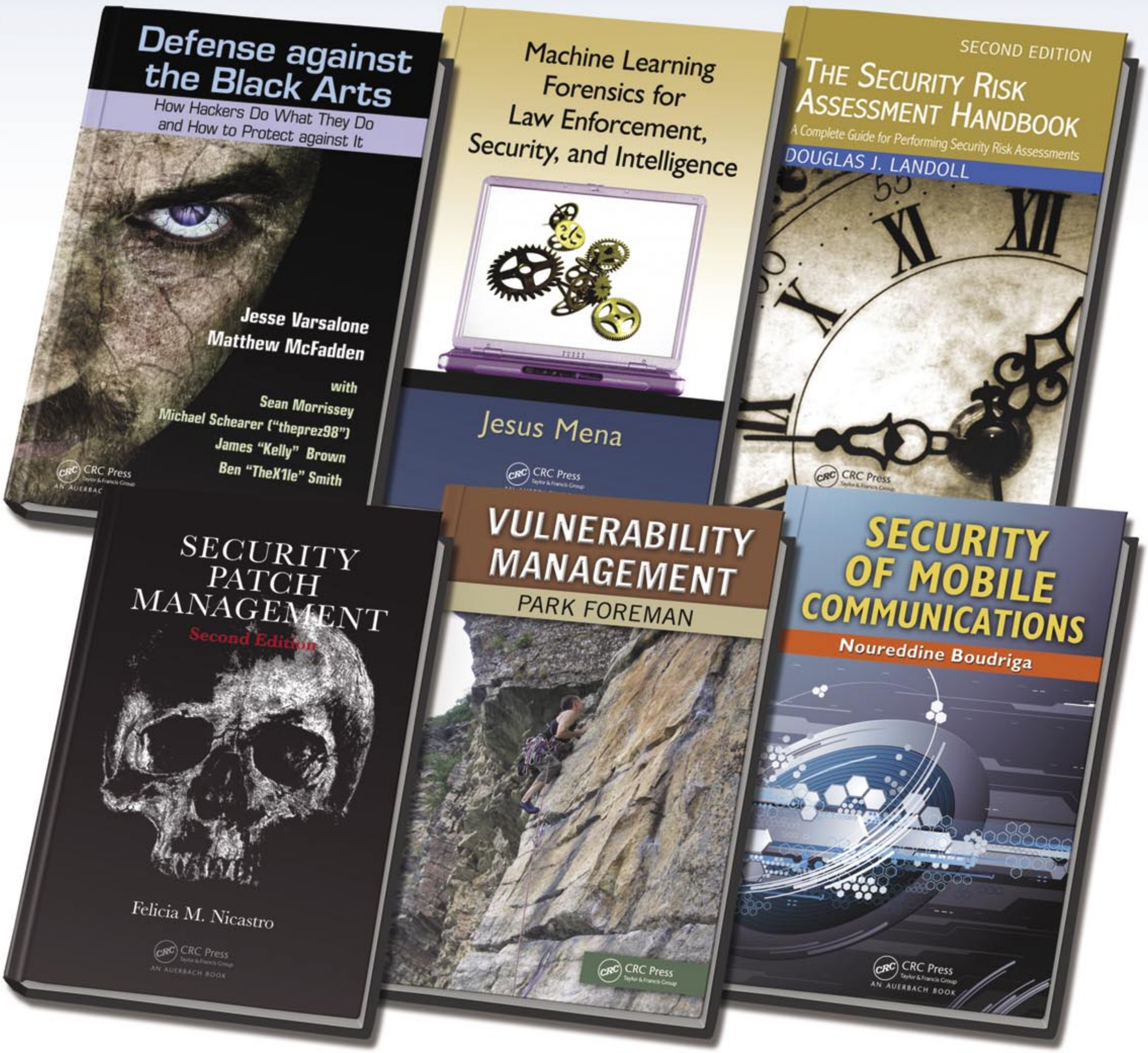
ISRAEL TORRES

Israel Torres is a hacker at large with interests in the hacking realm.
hakin9@israeltorres.org
http://twitter.com/israel_torres
https://plus.google.com/102921309581624765133/posts

SPECIAL SAVE 50% OFFER

Limited Time Offer

Secure Your System with these Critical Volumes


Defense against the Black Arts
How Hackers Do What They Do and How to Protect against It
Jesse Varsalone
Matthew McFadden
with
Sean Morrissey
Michael Schearer ("theprez98")
James "Kelly" Brown
Ben "TheX1le" Smith
Machine Learning Forensics for Law Enforcement, Security, and Intelligence
Jesus Mena
SECOND EDITION
The Security Risk Assessment Handbook
A Complete Guide for Performing Security Risk Assessments
DOUGLAS J. LANDOLL
SECURITY PATCH MANAGEMENT
Second Edition
Felicia M. Nicastro
VULNERABILITY MANAGEMENT
PARK FOREMAN
SECURITY OF MOBILE COMMUNICATIONS
Noureddine Boudriga
CRC Press

Enter promo code **510HA** at checkout to **SAVE 50%**

www.crcpress.com

CRC Press
Taylor & Francis Group

Offer expires 12/31/2011

As Apple Devices Gain Popularity

Do They Become More Vulnerable to Exploitation?

The answer is yes. A few years back before the iPod, iPad and iTouch, there were very few CVEs listed in the National Vulnerability Database at http://nvd.nist.gov but today you can find over 2,700.

What you will learn…

- Apple CVEs (holes)
- Your Apple Hardware
- Tuning up your Apple Software

What you should know…

- The CVE Standard by MITRE
- How to install Firmware and Patches
- Deploying Firewall and HIPS software

Most recently, if you search for CVEs discovered in Apple hardware and software over the last three months alone, you'll find at least 80 CVEs (holes) of which most are of a serious, easily exploitable nature.

Apple iOS Specific CVEs

These are the only well-known holes on the Apple iOS operating system. The good news is that there are very few Apple iOS holes, only three so far, however, they are of a serious nature.

CVE-2011-0228
Summary: The Data Security component in Apple iOS before 4.2.10 and 4.3.x before 4.3.5 does not check the basicConstraints parameter during validation of X.509 certificate chains, which allows man-in-the-middle attackers to spoof an SSL server by using a non-CA certificate to sign a certificate for an arbitrary domain.
Published: 08/29/2011
CVSS Severity: 7.5 (HIGH)

CVE-2011-0227
Summary: The queueing primitives in `IOMobile FrameBuffer` in Apple iOS before 4.2.9 and 4.3.x before 4.3.4 do not properly perform type conversion, which allows local users to gain privileges via a crafted application.
Published: 07/19/2011
CVSS Severity: 7.2 (HIGH)

CVE-2011-0226
Summary: Integer signedness error in psaux/t1decode.c in FreeType before 2.4.6, as used in CoreGraphics in Apple iOS before 4.2.9 and 4.3.x before 4.3.4 and other products, allows remote attackers to execute arbitrary code or cause a denial of service (memory corruption and application crash) via

a crafted Type 1 font in a PDF document, as exploited in the wild in July 2011.
Published: 07/19/2011
CVSS Severity: 9.3 (HIGH)

Twenty Most Recent CVEs in Apple

According to the NVD, many of these newer holes are in Apple software, in particular QuickTime and the Safari web browser, developed for both the Apple operating system and Microsoft Windows.

CVE-2011-0258
Summary: Apple QuickTime before 7.7 on Windows allows remote attackers to execute arbitrary code or cause a denial of service (memory corruption and application crash) via a crafted image description associated with an mp4v tag in a movie file.
Published: 09/06/2011
CVSS Severity: 9.3 (HIGH)

CVE-2011-0228
Summary: The Data Security component in Apple iOS before 4.2.10 and 4.3.x before 4.3.5 does not check the basicConstraints parameter during validation of X.509 certificate chains, which allows *man-in-the-middle* attackers to spoof an SSL server by using a non-CA certificate to sign a certificate for an arbitrary domain.
Published: 08/29/2011
CVSS Severity: 7.5 (HIGH)

CVE-2011-0257
Summary: Integer signedness error in Apple QuickTime before 7.7 allows remote attackers to execute arbitrary code or cause a denial of service (application crash) via a crafted PnSize opcode in a PICT file that triggers a stack-based buffer overflow.
Published: 08/15/2011
CVSS Severity: 9.3 (HIGH)

CVE-2011-0256
Summary: Integer overflow in Apple QuickTime before 7.7 allows remote attackers to execute arbitrary code or cause a denial of service (application crash) via crafted track run atoms in a QuickTime movie file.
Published: 08/15/2011
CVSS Severity: 9.3 (HIGH)

CVE-2008-7296
Summary: Apple Safari cannot properly restrict modifications to cookies established in HTTPS sessions, which allows man-in-the-middle attackers to overwrite or delete arbitrary cookies via a Set-Cookie header in an HTTP response, related to lack of the *HTTP Strict Transport Security* (HSTS) `includeSubDomains` feature, aka a *cookie forcing* issue.
Published: 08/09/2011
CVSS Severity: 5.8 (MEDIUM)

CVE-2011-0252
Summary: Heap-based buffer overflow in Apple QuickTime before 7.7 allows remote attackers to execute arbitrary code or cause a denial of service (application crash) via crafted STTS atoms in a QuickTime movie file.
Published: 08/04/2011
CVSS Severity: 9.3 (HIGH)

CVE-2011-0251
Summary: Heap-based buffer overflow in Apple QuickTime before 7.7 allows remote attackers to execute arbitrary code or cause a denial of service (application crash) via crafted STSZ atoms in a QuickTime movie file.
Published: 08/04/2011
CVSS Severity: 9.3 (HIGH)

CVE-2011-0250
Summary: Heap-based buffer overflow in Apple QuickTime before 7.7 allows remote attackers to execute arbitrary code or cause a denial of service (application crash) via crafted STSS atoms in a QuickTime movie file.
Published: 08/04/2011
CVSS Severity: 9.3 (HIGH)

CVE-2011-0249
Summary: Heap-based buffer overflow in Apple QuickTime before 7.7 allows remote attackers to execute arbitrary code or cause a denial of service (application crash) via crafted STSC atoms in a QuickTime movie file.
Published: 08/04/2011
CVSS Severity: 9.3 (HIGH)

CVE-2011-0248
Summary: Stack-based buffer overflow in the QuickTime ActiveX control in Apple QuickTime before 7.7 on Windows, when Internet Explorer is used, allows remote attackers to execute arbitrary code or cause a denial of service (application crash) via a crafted QTL file.
Published: 08/04/2011
CVSS Severity: 9.3 (HIGH)

CVE-2011-0247
Summary: Multiple stack-based buffer overflows in Apple QuickTime before 7.7 on Windows allow remote attackers to execute arbitrary code or cause a denial of service (application crash) via a crafted H.264 movie.

Published: 08/04/2011
CVSS Severity: 9.3 (HIGH)

CVE-2011-0246
Summary: Heap-based buffer overflow in Apple QuickTime before 7.7 on Windows allows remote attackers to execute arbitrary code or cause a denial of service (application crash) via a crafted GIF file.
Published: 08/04/2011
CVSS Severity: 9.3 (HIGH)

CVE-2011-0245
Summary: Buffer overflow in Apple QuickTime before 7.7 allows remote attackers to execute arbitrary code or cause a denial of service (application crash) via a crafted pict file.
Published: 08/04/2011
CVSS Severity: 9.3 (HIGH)

CVE-2011-1797
Summary: WebKit, as used in Apple Safari before 5.0.6, allows remote attackers to execute arbitrary code or cause a denial of service (memory corruption and application crash) via a crafted web site, a different vulnerability than other WebKit CVEs listed in APPLE-SA-2011-07-20-1.
Published: 07/21/2011
CVSS Severity: 9.3 (HIGH)

CVE-2011-1774
Summary: WebKit in Apple Safari before 5.0.6 has improper libxslt security settings, which allows remote attackers to create arbitrary files, and consequently execute arbitrary code, via a crafted web site.
NOTE: this may overlap CVE-2011-1425.
Published: 07/21/2011
CVSS Severity: 8.8 (HIGH)

CVE-2011-1462
Summary: WebKit, as used in Apple Safari before 5.0.6, allows remote attackers to execute arbitrary code or cause a denial of service (memory corruption and application crash) via a crafted web site, a different vulnerability than other WebKit CVEs listed in APPLE-SA-2011-07-20-1.
Published: 07/21/2011
CVSS Severity: 9.3 (HIGH)

CVE-2011-1457
Summary: WebKit, as used in Apple Safari before 5.0.6, allows remote attackers to execute arbitrary code or cause a denial of service (memory corruption and application crash) via a crafted web site, a different vulnerability than other WebKit CVEs listed in APPLE-SA-2011-07-20-1.
Published: 07/21/2011
CVSS Severity: 9.3 (HIGH)

CVE-2011-1453
Summary: WebKit, as used in Apple Safari before 5.0.6, allows remote attackers to execute arbitrary code or cause a denial of service (memory corruption and application crash) via a crafted web site, a different vulnerability than other WebKit CVEs listed in APPLE-SA-2011-07-20-1.
Published: 07/21/2011
CVSS Severity: 9.3 (HIGH)

CVE-2011-1288
Summary: WebKit, as used in Apple Safari before 5.0.6, allows remote attackers to execute arbitrary code or cause a denial of service (memory corruption and application crash) via a crafted web site, a different vulnerability than other WebKit CVEs listed in APPLE-SA-2011-07-20-1.
Published: 07/21/2011
CVSS Severity: 9.3 (HIGH)

CVE-2011-0255
Summary: WebKit, as used in Apple Safari before 5.0.6, allows remote attackers to execute arbitrary code or cause a denial of service (memory corruption and application crash) via a crafted web site, a different vulnerability than other WebKit CVEs listed in APPLE-SA-2011-07-20-1.
Published: 07/21/2011
CVSS Severity: 9.3 (HIGH)

JailBreaking the Apple iOS

The Apple iOS operating system can be *unlocked* through jailbreaking. This is the process of removing the limitations imposed by Apple on devices running the iOS operating system through use of custom kernels. These devices include the iPhone, iPod Touch, iPad, and the second generation of the Apple TV.

Jailbreaking gives you rootaccess to the operating system. With this level of control, you can download additional applications, extensions, and themes that are unavailable through the official Apple App Store, by using special installers such as Cydia.

You can still use the Apple Store and iTunes. Without jailbreaking your Apple device, you can only run Apple authorized software. If you do jailbreak, you may be at risk of voiding your devices hardware warrantee, so be careful and do it at your own risk. If you reversed the process and restore the original operating system through iTunes, you've undone your jailbreak and therefore your warrantee may be in full force if Apple does not know you've done this process.

Tethered and Untethered Jailbreaks

A tethered jailbreak requires that the device be connected to a computer each time it needs to be booted. Taking place on a computer, it effectively re-jailbreaks an iDevice without losing data or restoring via iTunes. An untethered jailbreak allows the device to be powered off, powered up, and rebooted without the assistance of a computer.

SIM Unlocking the Apple iPhone

This process of unlocking the SIM will allow your iPhone to accept any SIM card without restrictions. This gives you the opportunity to use your phone with an alternative phone company. You can't unlock the SIM, however, without performing a jailbreak first, giving you full access to the device.

For newer devices, you could try downloading Ultrasn0w from Cydia. However, not every iPhone device supports Ultrasn0w. Another alternative to software unlocking is to buy a factory-unlocked iPhone, which can also be found on eBay and throughout the internet. The factory unlocked iPhones come unlocked directly from Apple and don't need to be jailbroken using third-party software. Finally, you could ask your local carrier to unlock the iPhone for you and most will do this for free and some for a small charge.

Is Jailbreaking and SIM Unlocking Ilegal?

While Apple continues to argue that it is illegal, it only breaks their monopoly and control of a device that you, the consumer, or business executive, purchase from them. There was a strong argument by Tim Wu, a professor at Columbia Law School, that jailbreaking is legal and ethical.

He cited an exemption that the Library of Congress issued which notes that locks are used by wireless carriers to limit the ability of the consumer, which has nothing to do with copyright law protections. In July, 2010, the U.S. Copyright Office agreed with Professor Wu and approved exemptions that allow users to jailbreak their devices and unlock their SIMs.

Attacks Against the iPhone

The first iPhone worm appeared in early November 2009, created by a 21-year-old Australian college student, Ashley Towns, of Wollongong. This worm is called iKee and like most malicious software it could cause serious damage to any jailbroken Apple device. After this worm, one of the top anti-virus vendors, F-secure, discovered a similar worm that was designed to take advantage of an SSH default password vulnerability and install software that compromised bank transactions that take place over the jailbroken iPhones.

If you do jailbreak, you put your phone at additional risk. Eight months later, the German government discovered another exploit against the iPhone's weakness, after being jailbroken.

Apple Applications are Vulnerable

As I've discussed earlier, there are numerous *Common Vulnerabilities and Exposures* (CVEs) in the Apple operating system and in various applications. When we think about weaknesses in mobile devices, we need to explore the issues of reverse-engineering, disassembly, debugging and license cracking. Most attacks exploit weaknesses in software, even encryption software, such as SSL or SSH, which were designed to enable secure transactions.

Most attacks look for these software weaknesses for the purpose of cybercrime. The criminals want your identity including your credit card information or your bank account. If you choose to leverage mobile access to your confidential data, you are at risk of an exploited piece of software that you trust, helping cybercriminals siphon off your personal identity.

Hardening Your Apple Device

I'm a strong proponent of *host-based intrusion prevention* (HIPS) systems, however, I have only found one for the iOS that is mature enough to consider discussing in my Hakin9 article.

With that said there are anti-virus programs and firewalls now available for the iOS that you should consider looking at.

In addition, there are ways to harden your iOS by removing known vulnerabilities (CVEs) so this is where I will focus on first. Here's how to harden your Apple device:

- Make sure you update the firmware to the latest version (visit *http://www.apple.com/ios/* to learn more)
- Require a passcode (tap *Settings->General->Passcode Lock*)
- Set auto-lock timeout (tap *Settings->General->Autolock-> 1 Minute or lower*)
- Disable grace period for lock (tap *Settings->General->Passcode Lock->Require Password->Immediately*)
- Erase data upon excessive passcode failures (tap *Settings->General->Passcode Lock->Turn* on Erase Data)
- Enable Fraud Warning in in the Apple Safari web browser (tap *Settings->Safari->Turn* on Fraud Warning)
- Enable Data Protection (tap *Settings->General->Passcode->Data protection is enabled* should be displayed at the bottom of the screen)

- Turn off Ask to Join Networks (tap *Settings->Wi-Fi->Turn off Ask to Join Networks*)
- Turn off Bluetooth when not needed (tap *Setting->General->Bluetooth->Turn off Bluetooth*)
- Forget Wi-Fi networks to prevent automatic rejoin (tap *Settings->Wi-Fi->choose a Wi-Fi network->Forget* this network)
- Erase all data before return, repair, or recycle (tap *Settings->General->Reset->Erase All Contents and Settings*)
- Enable remote wipe functionality. Apple's Mobile Me service provides you the ability to track GPS enabled devices, display messages on the screen, lock a device, and wipe all data remotely. These features are provided free of charge to owners of iPhone 4, iPod Touch (fourth generation or later), and iPad devices, but you have to set it up in advance. If it's lost, before you do this, chances are it's gone for good.
- Encrypt device backups through iTunes (Go into the iTunes software, with the Apple device connected, check *Encrypt [devicetype] backup* under Options and choose a strong password).

Host-based Intrusion Prevention for the Apple iOS

It's early in this platform's lifecycle but it appears that Arxan's EnsureIT for Apple iOS delivers automated embedded software protection that is easy to deploy, durable and resilient. The EnsureIT technology for the iOS defends, detects and reacts to attempted attacks by deploying various security techniques (Arxan calls these *Guards*) and they include obfuscation, checksum and anti-debug directly into the software code of each application for defense-in-depth. This layered protection of diverse Guard types provides control, trust and tamper-resistance for each application, which are the cyber crime targets. The result is an early stage HIPS engine for your iOS. Give it a try and let me know what you think about it.

Firewall for the Apple iOS

The Firewall iP application controls all inbound and outbound connections for your Apple iOS device. It includes traditional Macintosh and Windows firewall software functionality including allow and deny of application connections and port specific blocking. This security tool will also block advertisements and help reduce data usage. It even claims support for IPv6. If you want to try it out and let us know at Hakin9 what you think about it, you can get it at the Cydia Store through the BigBoss repository with a pricetag under $5 USD. Also, if you find any other firewall apps you think are solid for the iOS, let us know.

Anti-virus for the Apple iOS

Intego's VirusBarrier iOS is the first malware-scanning app for Apple mobile devices. Intego's VirusBarrier iOS allows iPhone, iPad and iPod Touch users to manually scan their devices and *detect and eradicate all known malware affecting Windows or Mac OS X*. Please note that for a long time, Apple has famously refused to authorize any anti-virus software for iOS devices until the release of VirusBarrier iOS. This utility can also be deployed to scan ZIP files, poisoned PDFs, email attachments and files downloaded to IOS devices from remote locations such as Dropbox. It automatically updates malware definitions and keeps logs of scans, and repairs infected files. However, VirusBarrier iOS is not a state of the art anti-virus solution as it doesn't yet run automatic or scheduled scans and it cannot scan applications that were already installed before it was installed so you are at risk of having pre-existing malware that you won't notice until your firewall notices surprise port traffic that you didn't expect. You can also get this program from Apple's online store for under $5 USD. Note that Kaspersky makes a Mobile Security Suite that runs on all major mobile device operating systems EXCEPT the iOS and Kaspersky feels Apple needs to open the device and software development at the kernel level to do it right but apple seems to continue to refuse. There are many who question what is hiding in the Kernel and what secrets Apple is trying to protect, including backdoors and eavesdropping but that's a story for another NSA day.

Sources and Credits

Thanks to MITRE's CVE program, the NVD at *NIST.gov*, Apple.com, Google, and Wikipedia for some of the information and content used in my research for this article.

GARY S. MILIEFSKY, FMDHS, CISSP®

Gary S. Miliefsky is a regular contributor to Hakin9 Magazine and has written information security and regulatory compliance articles for other major publications. He is also a frequent speaker at network security events and trade shows throughout the globe including RSA Conference, CSI, White House Conference on eSecurity, Hack in Paris, HackerHalted and many others. He is the founder and Chief Technology Officer (CTO) of NetClarity, Inc, where he can be found at http://www.netclarity.net. He is a 20+ year information security veteran and computer scientist and serves on the Advisory Board of the Center for the Studyof Counter-Terrorism and Cyber Crime at Norwich University. He is a member of ISC2.org and a CISSP®. Miliefsky is a Founding Member of the US Department of Homeland Security (http://www.DHS.gov), serves on the advisory board of MITRE on the CVE Program (http://CVE.mitre.org) and is a founding Board member of the National Information Security Group (http://www.NAISG.org).

National Information Security Group

NAISG™ is a non-profit organization that promotes awareness and education of information security. Members include IT/security admins, managers, law enforcement personnel, the media, educators and students… anyone interested in staying on the cutting edge of information security.

Chapters in
- Atlanta, GA
- Boston, MA
- Dallas, TX
- Houston, TX
- Midland, MI
- Seattle, WA
- and more to come

Membership is free to Hakin9 subscribers!

Meetings count toward CISSP renewal credits

Join a chapter or our global Security TechTips mailing list at www.naisg.org.

Find us on Linked in
- "National Information Security Group."

Register now for December's Securanoia™ Boston 2011 conference at www.securanoia.com.

NAISG™

Import Hooks For Encrypted Python Modules

Every now and again, somebody comes up and ask this question: How can I hide/encrypt/obfuscate my Python code? And the answers may be different, ranging from things like: Python is not the Tool; rewrite it in Perl; distribute only your .pyc or .pyo files; and other creative solutions.

What you will learn…

- Python's import mechanism and hooks
- How to import encrypted Python modules

What you should know…

- Basic knowledge of Python
- Knowledge of Python's C API
- Some knowledge of the XOR cipher.

Well, the answer is Yes! There are ways, and one such way is through the use of Import Hooks.

Import Hooks[1] are objects that can be injected into Python's import mechanism and used to customize how modules are found and loaded, allowing it to import modules *stored in a non-standard way*. As we want to import encrypted modules, we fall into this *non-standard* category, ergo, we need to write an import hook.

The *The Importer Protocol* is presented in the *Python-Enhancement Proposal* (PEP) 302. This protocol describes Import Hooks and explains how the import process works for loading Python modules. According to this PEP, when the Python interpreter finds an import statement, it calls the `__import__` function from the built-in name space with the name of the module and a reference to the global name space. If the name is a sub-module of a package, `__import__` will try to resolve that name relative to that package first. If that fails, `__import__` will try an absolute import.

When the interpreter finds a dotted import, it first splits the name into components and then tries to import those components in order, looking for a component inside the previous one. Thus, import *foo.bar*, becomes first an import of the *foo* module, and when that succeeds, the interpreter imports bar as a sub-module of *foo*, which implies that by the time bar is being imported, *foo* was already loaded successfully. Every time one of these individual imports is made, a hook is invoked to handle the import. If no hooks exist or it can't handle the import, then the built-in method is applied.

According to the PEP, The Importer Protocol involves two objects, the *finder* and the *loader*. The *finder* has the task to let the import process know if it knows of a loader for a given module. The *finder* must implement a function `find_module` of this form:

```
finder.find_module(fullname, path=None)
```

Table 1. Things loader.load_module is responsible for

- The `__file__` attribute of the new module must be set. It could be any string at all, but it must be set.
- The `__name__` attribute must be set.
- If the module is a package, then the `__path__` attribute must be set with a list, although it could be an empty list if not needed.
- The `__loader__` attribute must be set to the Loader object.
- The loader must execute the code of the module inside the new module's global namespace (or `new_module.__dict__`).
- The loader should first look up the module in sys.modules and if found, use that module. On the same note, the Loader should append the new loaded module in `sys.modules`.

where *fullname* is the fully qualified name of the module to be found and path is the package. `__path__` of the top-level module if *fullname* is a sub-module or a sub-package.

One of three things could happen when find_module is called. (a) `find_module` returns None, which implies the module denoted by fullname could not be found, or at least with this *finder* object; (b) `find_module` raises an exception, which then interrupts the import process, propagating an `ImportError`; or (c) `find_module` returns a Loader object which implies the module was found and the loader being returned will be used to load it.

On the other hand, the *loader* must implement a `load_module` function of the following form:

```
loader.load_module(fullname)
```

where fullname is the same argument to the `Finder.find_module` function, the fully qualified name of the module to be loaded. The *Loader* then returns the new loaded module, or could raise an `ImportError` exception if there was a problem loading the module.

The `load_module` function is responsible for several things as seen in Table 1.

The next part of the process is simpler. We just need to tell Python to use our custom *finder* and *loader* objects. There are two ways in which we can accomplish this either with path hooks or meta hooks.

Path hooks are used as part of the `sys.path` processing and are called sequentially to find out if they can handle a given path. Path hooks are registered appending a callable object to `sys.path_hooks` list. Meta hooks, on the other hand, are objects called at the beginning of the import process and are registered by appending them to the `sys.meta_hooks` list.

As meta hooks are general, as in not concerned with an specific path, we'll use them for our proof-of-concept.

Preparing the Tools

After reviewing the import process in Python, let us dive into the practical side of the process. First of all, we should decide on what cipher to use to encrypt the code. For this experiment, along with the strong advisory that this code is just a proof-of-concept, we are going to use a simple XOR cipher.

The XOR cipher[2] is a simple algorithm that is used to encrypt every character of a *plaintext* with a given key, applying the Exclusive-OR operation on the two. Applying the *key* to the resulting *ciphertext* would restore the original *plaintext*. Listing 2 presents a

Listing 1. *(CryptImpHook.h) Header file with XOR key*

```
#ifndef CRYPTIMPHOOK_H_
#define CRYPTIMPHOOK_H_

char qta[] = {
        0x1A, 0x1B, 0xb9, 0x8f, 0xcf, 0x6a,
        0x1A, 0x1B, 0xb9, 0x8f, 0xcf, 0x6a
}; //This will be our key


#endif /* CRYPTIMPHOOK_H_ */
```

Listing 2. *(Cipher.c) Function for encrypting/decrypting using XOR*

```
int XOR(char btoenc, char keybit)
{
    return btoenc ^ keybit;
}
```

Listing 3. *(CryptImpHook_Conv.c) Opening a file and encrypting its contents*

```
…
#include <CryptImpHook.h>
…


void encrypt_data(FILE* in, FILE* out, char* key)
{
    int idx = 0;
    int btoenc;

    while( (btoenc = fgetc(in)) != EOF)
    {
        fputc(XOR(btoenc, key[idx % strlen(key)]),
                    out);
        idx++;
    }
}
```

Listing 4. *(test_script.py): Python test module*

```
import os

def some_function(arg1, arg2, arg3):
    return arg1 + arg2 + arg3

class C(object):
    def __init__(self, arg1, arg2, arg3):
        print some_function(arg1, arg2, arg3)
```

Listing 5. *(CryptImpHook.c) The find_module function*

```
…
PyObject *
CryptImpHook_find_module(self, args)
PyObject *self, *args;
{
 int err;
 char *filename, *fullname, *path;
 CryptImpHook *hook = (CryptImpHook *)self;

 err = PyArg_ParseTuple(args, "s|z", &fullname,
                  &path);

 if(err == 0)
 {
    PyObject_Print(PyErr_Occurred(), stdout,
                  Py_PRINT_RAW);
    PySys_WriteStdout("\n");
    PyErr_Print();
    PySys_WriteStdout("\n");
 }

 filename = cih_get_path(fullname);
 if(filename != NULL)
 {
    hook->mod_file = PyString_FromString(filename);
    free(filename);
    return self;
 }else{
    free(filename);
    return Py_None;
 }

}

char *
cih_get_path(const char *fullname)
{
 int res;
 ssize_t ssize;
 struct stat buf;
 char *filename;

 filename = (char *)malloc(sizeof(char) * 512);
 filename = (char *)memset(filename, 0, sizeof(char)
                  * 512);

 ssize = strlen(fullname);
 strncat(filename, fullname, ssize);
 strncat(filename, IMPHOOK_SUFFIX, strlen(IMPHOOK_
                  SUFFIX));

 //We should have our filename as fullname.pye.
 //If we find it, then we should return self.
 res = stat(filename, &buf);


     return filename;
 }

 return NULL;
 }
```

Listing 6. *(Test.py) CryptImpHook test script*

```
import sys

print "Starting Test. Importing CryptImpHook"

from CryptImpHook import CryptImpHook

print "Adding CryptImpHook to meta_path"

sys.meta_path.append(CryptImpHook())

print "Import class C from EncModule"

#Now lets import our Module

import EncModule

EncModule.C(1, 2, 3)
```

Listing 7a. *(CryptImpHook.c) The load_module function*

```
…
#include <CryptImpHook.h>
…
PyObject *
CryptImpHook_load_module(self, args)
PyObject *self, *args;
{
  int err;
  char *module_code, *fullname;
  PyObject *new_mod, *sys_module_dict, *new_module_dict, *res;

  CryptImpHook *hook = (CryptImpHook *)self;

  PyArg_ParseTuple(args, "s", &fullname);
  new_mod = PyModule_New(fullname);
  Py_INCREF(new_mod);

  err = PyModule_AddObject(new_mod, FILE_ITEM, hook->mod_file);
  if(err != 0)
  {
     PyObject_Print(PyErr_Occurred(), stdout, Py_PRINT_RAW);
     PySys_WriteStdout("\n");
     PyErr_Print();

  }
  err = PyModule_AddObject(new_mod, LOADER_ITEM, self);

  if(err != 0)
  {
     PyObject_Print(PyErr_Occurred(), stdout, Py_PRINT_RAW);
     PySys_WriteStdout("\n");
     PyErr_Print();
     PySys_WriteStdout("\n");
  }

  sys_module_dict = PyImport_GetModuleDict();

  if(sys_module_dict != NULL)
  {
     PyDict_SetItemString(sys_module_dict, fullname, new_mod);
                       PyModule_AddObject(new_mod, "__builtins__", PyDict_GetItemString(sys_module_dict, "__builtin__"));
  }

  module_code = cih_read_module_code(hook->mod_file);
  //Next, we grab the reference to the new module's __dict__
  new_module_dict = PyModule_GetDict(new_mod);

  /*
   * This is really important. the second arg should be Py_file_input because we need
   * the interpreter to believe is a file, and accept multiple statements in multiple lines.
   * Else, the plug-in should throw a SegFault.
   */

  /* Now eval in context with new_mod.__dict__ in both globals and locals;
   * The following (I believe) would be the translation in C of the
   * exec CODE in mod.__dict__
   */
  res = PyRun_String(module_code, Py_file_input, new_module_dict, new_module_dict);
  return new_mod;
}

char *
cih_read_module_code(PyObject *filename)
{
    FILE *fp;
    ssize_t total;
    struct stat buf;
    int err, dec_len;
    int idx = 0;
    int btodec;
    char temp[1];
    char *module_code, *dec_module_code;

     char *chr_filename = PyString_AsString(filename);


    stat(chr_filename, &buf);
    dec_module_code = (char *)malloc(sizeof(char) * buf.st_size); //We grab the file size in bytes
    memset(dec_module_code, 0, sizeof(char) * buf.st_size);

    if(module_code == NULL){
     return NULL;
    }

    fp = fopen(chr_filename, "r");
```

function that takes two characters, and returns the XOR operation between them. To specify the *key* we will be using, lets write a small header file (see Listing 1) to put it there and use it from any place we need to encrypt/decrypt contents.

In addition, it would be really handy to have a program (see Listing 3) that could take care of encrypting the contents of a file, and produce an encrypted file as a result so we can pass our python modules and have an encrypted one back, to be loaded using our import hook.

Now that we have taken care of the encryption, we need a Python module and a way to encrypt it. Take a look at the code in Listing 4. This code imports the *os* module, defines a function called `some_function` that received three integers, and a class called C, that in its `__init__` method, receives three arguments (plus the *self* reference) and prints the result of a call to `some_function`. Let's grab this code and put it in a file called `test_script.py`. We should now use our program `CryptImpHook_Conv` to process `test_script.py` and store an encrypted version of it. We are going to call this version, `EncModule.pye`.

At this point, we are in need of a way to tell python to *understand* our encrypted module. So, we are going to write a python object that will behave like an import hook. That means to load the encrypted file, decrypt it and load it in a python module so we are able to use the code inside it.

Listing 7b. *(CryptImpHook.c) The load_module function*

```

    while( (btodec = fgetc(fp)) != EOF )
    {
        sprintf(temp, "%c", XOR(btodec, qta[idx %
                 ktal]));
        strncat(dec_module_code, temp,
                 sizeof(char));
        idx++;
    }
    err = ferror(fp);

    if((err))
    {
  return NULL;
    }
    fclose(fp);

    return dec_module_code;
}
```

Meta Hooks to Import Encrypted Modules

According to the PEP 302, we need to provide two functions to follow the implementation of the Importer Protocol. Via the Python C API and the object CryptImpHook, these functions are made available. The first function is `find_module`. The `find_module` (see Listing 5) function receives two arguments: *self* and *args*. *Self* is a reference to the `CryptImpHook` object, as the function is a method of that object; *args* is a python tuple with two elements, *fullname*, which is the name of the module to be imported, dots included and path (which is *None* for a top-level module, or a Python list compliant `__path__` for submodules).

After checking that no error occurred in the parsing of the tuple, we need to check if a file exists that matches the *fullname* argument with a given extension, or `IMPHOOK_SUFFIX` which in our case is *.pye* to differentiate it from a normal *.py* or *.pyc* file. If we are able to find a file called *fullname.pye*, then we put the filename in the `mod_file` property of the `CryptImpHook` object, and we return the reference to self (see Lines 22, 23 and 24 in Listing 5). This will tell the import process that `CryptImpHook` is capable of loading this module and that it should look no further to load this code. On the other hand, if we aren't able to find the file, then the `find_module` function returns `Py_None`, which would indicate that CryptImpHook is not capable of loading this module and another *Loader* should be tried instead (this is what would happen with *.py* and *.pyc* files being loaded by this object).

As the python module we encrypted has a *.pye* suffix (`EncModule.pye`), we should be able to load it using our `load_module`, which is the other function needed by the Import Protocol. The `load_module` function (see Listing 7) will then read, decrypt, and load the contents of the file and turn them into a python module. Just like `find_module`, this function receives two arguments, *self* and a python tuple with one element, *fullname*.

After parsing the tuple to grab the name of the module (Line 11), we create a new module with `PyModule_New(fullname)`. This call will create a new empty python module, with the attribute `__name__` set to fullname. We then set `__file__` to `mod_file` and `__loader__` to self (see lines 15 and 23) to fulfill the responsibilities of this function (see Table 1). At this point, we now have an empty python module with all the required attributes set.

Then we must register our new module with *sys.modules* (line 37). We set a reference to `__builtin__` in our new module, so our decrypted code can find python's built-in objects.

Having done all this, we just need to load the encrypted code with the `cih_read_module_code` (*hook->mod_file*) instruction. This function just opens the encrypted file, reads and decrypts the contents with

References

- New Import Hooks (PEP 302) *http://www.python.org/dev/peps/pep-0302* [1]
- XOR Cipher *http://en.wikipedia.org/wiki/XOR_cipher* [2]
- Python C API *http://docs.python.org/c-api/index.html* [3]

the given file (see Listing 1), and returns the decrypted contents to the callee.

With the decrypted code in hand, we proceed to execute it. But first, we need a reference to our new module's `__dict__` attribute, as the code has to be executed with a reference to this dictionary. The second thing is that we need to provide `Py_file_input` as the second argument to the `PyRun_String` function that we are going to use to execute our code into the new python module. If we use something other than `Py_file_input`, we will get *Segfaults* all over.

The line (line 55) `res = PyRun_String(module_code, Py_file_input, new_module_dict, new_module_dict)` is analogous to the exec code in mod.`__dict__` python code you will find in PEP 302.

After all this, we are ready to run the test.

Do you remember we created `EncModule.pye` a while ago? It is time to use it. Put `EncModule.pye` and the compiled `CryptImpHook` in the same directory and just execute the Python test script in Listing 6, and Voila!, we just imported an encrypted module in Python.

You'll find the whole code in *http://dev.gentoo.org/~neurogeek/CryptImpHook.tar.gz*.

JESUS RIVERO

Jesus Rivero, a.k.a Neurogeek, is a Computer Scientist programming for the past 10 years from embedded systems to web applications. Currently, he develops software for the financial world and is a Gentoo GNU/Linux developer.

jesus.riveroa@gmail.com

neurogeek@gentoo.org

Website/blog: http://dev.gentoo.org/~neurogeek

Apple OS X and iOS Hacking News

This month's article focuses on Apple technology hacking that has been identified thus far in 2011. Here you will find a compilation of some high profile media reports and research from the Web on the hacking of Apple technology.

There are two sections – Mac OS X and iOS (iPad; iPod & iPhone). There are many contributors, so we'd like to thank all of them for the use of their research material in this article.

Apple / Mac OS X Lion Security / Hacking News

News: Mac OS X MacDefender Scareware Threat – May 2011

Antivirus firm Intego discovered a new malware known as *MacDefender* targeting Mac OS X users via Safari. According to the report, the malware appeared to be being deployed via JavaScript as a compressed ZIP file reached through Google searches. When a user clicked on a link after performing a search on a search engine such as Google, this took them to a web site whose page contained JavaScript that automatically downloaded a file. In this case, the file downloaded was a compressed ZIP archive, which, if a specific option in a web browser was checked (Open *safe* files after downloading in Safari, for example), would open.

Users running administrator accounts and with the Safari option to open *safe* files automatically checked, appeared to be most at risk, with some claiming that no notification of installation was seen or password required. Only when a screen popped up asking for a credit card number to sign up for virus protection did they realize that malware had been installed on their systems. This was the first major *scareware* threat to the Mac OS X platform. Apple fixed the exploit by simply blocking the MacDefender threat from running or being installed.

Source: Intego

News: Hacking Apple Laptop Batteries – July 2011

Security researcher Charlie Miller, widely known for his work on Mac OS X and Apple's iOS, has discovered an interesting method that enables him to completely disable the batteries on Apple laptops, making them permanently unusable, and perform a number of other unintended actions. The method, which involves accessing and sending instructions to the chip housed on smart batteries, could also be used for more malicious purposes down the road.

What he found is that the batteries are shipped from the factory in a state called „sealed mode" and that there's a four-byte password that's required to change that. By analyzing a couple of updates that Apple had sent to fix problems in the batteries in the past, Miller found that password and was able to put the battery into *unsealed mode*.

From there, he could make a few small changes to the firmware, but not what he really wanted. So he poked around a bit more and found that a second password was required to move the battery into full access mode, which gave him the ability to make any changes he wished. That password is a default set at the factory and it's not changed on laptops before they're shipped. Once he had that, Miller found he could do a lot of interesting things with the battery.

That lets you access it at the same level as the factory can, he said. *You can read all the firmware, make changes to the code, do whatever you want. And those code changes will survive a reinstall of the OS, so you could imagine writing malware that could hide on the chip on the battery. You'd need a vulnerability in the OS or something that the battery could then attack, though*.

Source: Charlie Miller, Apple security expert

News: Apple Safari 5.0.4 Security Vulnerabilities Identified – July 2011

Apple released Safari 5.0.4 – the latest version of Apple's browser software for Windows and Mac users – patching an eye-watering 62 security vulnerabilities in the process. The vulnerabilities, described in an Apple knowledgebase article, were disclosed at the same time as a host of security holes in the iOS software used by the iPhone, iPad and iPod touch were also revealed by the company. What this means is, just like their iPhone/iPod touch/iPad-owning cousins, people who run Safari on their Mac or Windows computers would be wise to check out the latest available security updates as soon as possible. Apple doesn't like to assign severity levels to the security vulnerabilities found in its products, but the bugs in Safari look pretty critical to me. 57 of the 62 bugs can be exploited just by a user visiting a maliciously-crafted website. If that's not a reason to install a security update to your Safari browser, I'm not sure what is.

Source: ID Theft Protect

News: AntiSec Group Posts Passwords From Apple Survey Server – July 2011

A group of computer hackers posted a document it claimed contains usernames and passwords for an Apple Inc. server, the latest in a string of brazen attacks that have compromised government and corporate websites around the world.

AntiSec, a hacking campaign that includes hackers from both the online vigilante group Anonymous and hackers from the now-defunct Lulz Security, posted a document containing a link to a supposed Apple server along with a list of 26 administrative usernames and passwords. AntiSec is Internet shorthand for *anti-security*. The hackers said in a statement posted to Twitter that they had accessed Apple's systems due to a security flaw used in software used by the Cupertino, Calif.-based gadget maker and other companies. *But don't worry*, the hackers said, *we are busy elsewhere*. A spokesman for Apple didn't immediately respond to a request for comment.

Source: ID Theft Protect

News: Apple Mac OS X Server APT and DHX Vulnerability – August 2011

Apple Macs according to iSec are more vulnerable to APTs short for advanced persistent threats. APTs are usually the work of state-sponsored hackers who go to great lengths to infiltrate government and corporate networks with malware that steals classified information and proprietary data. The problem with Macs stems from the OS X server that administrators use to push updates to large numbers of machines. The server's authentication routine is *inherently insecure*.

At the heart of the Mac server's insecurity is a proprietary authentication scheme known as DHX that's trivial to override. While Mac servers can use the much more secure Kerberos algorithm for authenticating Macs on local networks, iSec researchers found it was trivial to force OS X server to resort back to Apple's insecure protocol.

Source: ID Theft Protect

News: Apple Website Hacked by Hacker Called HodLuM – August 2011

One of the Apple Sub-Domains has it is claimed been defaced by known hacker HodLuM. The deface link is just an image uploaded to the Apple domain. The hacker uses the *N00BZ* characters for all Hackers including the cracking group Anonymous, Lulzsec, Turkish hackers, Inj3t0rs and Exploit-DB's. The AOL Postmaster Website was also hacked by HODLUM some months ago.

Source: The Hacker News

News: OS X Lion Password Security Issue Identified – September 2011

Apple has dropped a couple of monumental password security clangers this year with the release on OS X Lion, according to security blogger Patrick Dunstan. Dunstan, who posted an important piece on cracking Mac OS X passwords a couple of years ago, decided to revisit the subject with the release of OS X Lion (version 10.7). Dunstan discovered Apple's developers had made user security worse in two important ways: firstly, it's possible to change the password of the current user without needing to know the original password, as Dunstan explains.

It appears Directory Services in Lion no longer requires authentication when requesting a password change for the current user, he writes. *So, in order to change the password of the currently logged in user, simply use: $ dscl localhost -passwd /Search/Users/bob*.

And that isn't the only backward step. Previously only a user with root (admin) privileges to a machine was able to get at the password hashes for other users, which are held in so-called *shadow files*. With OS X Lion this restriction is easily circumvented.

It appears in the redesign of OS X Lion's authentication scheme a critical step has been overlooked, Dunstan explains. *Whilst non-root users are unable to access the shadow files directly, Lion actually provides non-root users the ability to still view password hash data. This is accomplished by extracting the data straight from Directory Services*.

All users on the system, regardless of privilege, have the ability to access the ShadowHashData attribute from any other user's profile, he adds.

None of the major brute force crackers support OS X Lion hashes because the OS was only released in late July. Dunstan has created a python script to do the job, which is intended for password auditing.

The password security foibles discovered by Dunstan raise further questions about the overall security of Mac OS X Lion, already highlighted by earlier LDAP password security weaknesses. Another Lion flaw discovered by Dunstan enables a skilled hacker to view the computer's password hash data by extracting it from the Directory Services file. (Password hashes are the results of running passwords through encryption algorithms. Those algorithms are supposedly unbreakable, but in truth, automated password-cracking software can run through the millions of possible results from a fixed algorithm to *brute force* the passwords into plain text).

It's important to note that these password attacks don't yet allow a victim's computer to be exploited by far-off strangers; the hacker must have either physical access to the target system, or have been granted remote limited-user access to it. But in the seconds it would take to perform these password takeovers, Defence in Depth says a physical attacker could also visit a Web page rigged with malicious code that would then connect a remote attacker to the compromised machine. From there, the possibilities for exploitation are endless.

Source: Defense in Depth security blog

iOS Security Issues

Dino Dai Zovi conducted research earlier this year (Blackhat) into security concerns surrounding iOS 4.0. He identified it is possible to remotely exploit built-in or third-party apps by injecting a malicious web page in Safari or third-party app with embedded browser (i.e. Facebook or Twitter); malicious e-mail message or attachment in Mail; and MITM and corrupt network communication of third-party app. Dino identified applications that use a UIWebView were at most risk (as they didn't use PIE support) i.e. embedded browser in Twitter and Facebook.

iOS is a very robust OS however jailbroken devices will open iOS to kernel exploits, Apple's review and OTA downloads will bypass Apple's review. iOS is one of the most robust mobile operating systems (aside of QNX – see last month's article in Hakin9 *A brief overview of mobile and tablet app coding security*) which is only vulnerable to exploits if the device is jailbroken.

News: Jailbreaking has just been made a little bit more difficult – July 2011

Apple has released a software update for its iOS devices, including the iPhone 4, iPhone 3GS, iPad 2, iPad and generation 3 and 4 of iPod Touch. Available now, it'll bring your device to version 4.3.4. The free update fixes a vulnerability in the way PDF files handle fonts, where a malicious PDF file could sneak malware into your iOS device, giving hackers access to your hardware. However, that same vulnerability allows easy jailbreaking of iOS 4.3.3 with the web-based JailBreakMe 3.0, which frees users from the tight restrictions Apple places on its iOS software. Apple certainly wanted to stop that as quickly as possible.

Source: Apple Inc

News: iOS update fixes flaw in SSL – July 2011

Apple released a software update to fix a flaw in the latest version of iOS back in July for its mobile operating system for the iPhone, iPad and iPod Touch. The new software, version 4.3.5, addressed a flaw in the way Apple's software authenticates Secure Sockets Layer data; the mishandling of this commonly used communication protocol could allow an attacker with elevated network privileges to *capture or modify* data, Apple wrote on its support blog.

Apple's software update comes just five days after the company released the previous iOS, version 4.3.4, which was primarily a security fix for a PDF bug exploited by Jailbreakme 3.0, a service that allows iPhone and iPad users to *jailbreak* their devices, a process which allows users to install apps unauthorized by Apple. Jailbreakme encountered its own problems; on July 11, Jailbreakme's website was flagged – incorrectly, it turned out – as containing malware. Note: Jailbreakme's founder Nicholas Allegra (nickname *comex*) has been hired (August) as an intern by Apple, no doubt to help identify and close the holes in iOS.

Source: Security News Daily

News: Apple's MDM could be vulnerable to MITM attack vector – August 2011

Security researcher David Schuetz identified a possible *Mobile Device Management* (MDM) attack vector. After having worked out how MDM worked he identified a potential issue with HTTPS which could expose MDM to a *man in the middle* (MITM) attack. David developed a research MDM tool which allowed people to experiment with. The way it works is you enroll a mobile device and then you send commands to the iOS device from the MDM tool and the researcher gets to see what the response is Schuetz noted that without a publicly available MDM server for iOS, it has been difficult – if not impossible to see how users could be exploited via social engineering or other means via MDM. This particular attack POC provides evidence that MDM could very well be a attack vector.

Source: ID Theft Protect

News: iOS 5.0 untethered jailbreak opens the exploit door – September 2011

Great news for all jailbreakers – the Dev-Team has just announced at the world's first iOS jailbreak convention `MyGreatFest`, that they've nearly completed a userland jailbreak for iOS 5 and it will be `untethered.p0sixninja`, one of Chronic Dev-team member, announced that the jailbreak will apparently use five different exploits that have been discovered for the Apple A5 chip which is currently used on the iPad 2 and possibly will also be used for the next generation iPhone 5. This is really good news for people who want their iPhone5 to be jailbroken.

According to a tweet from @veeence the team is waiting for the actual hardware (iPhone 5) to get released and all Apple devices supporting iOS 5 will be eligible for untethered jailbreak. The five exploits, discovered by the team member P0d2g, will allow the next jailbreak to be userland and p0sixninja also added that it will be the most amazing jailbreak yet. To make things more exciting, MuscleNerd announced that he's been working on a 06.15 iPad baseband downgrade and unlock for iPhone 3G and 3GS and it will also be released after iOS 5 will reach users. Seems jailbreaking and the expected exploits, will not be going away anytime soon.

Final Thoughts The Mac OS X Antivirus Protection Question

Most readers will know Mac OS X runs on the UNIX platform, which means the underlying code base is more secure than say Windows. Cracking the UNIX system is very difficult – in fact probably the most difficult of any operating system (QNX might have something to say about that, which is based on the Linux OS). OS X also includes a useful technology called sandboxing which is where applications run and processes are separated. Mac OS restricts both file access and application execution. The sandboxing model is both more secure and less likely to be exploited by malware in the wild. If you visit the Apple website, they will tell you that you still require antivirus for your Mac. What it really comes down to is knowledge – knowledge of what users need to do; be aware of and remembering not to click on links that might be suspicious. Most security experts will tell you, it's always better to be safe rather than sorry, so most if not all will suggest Mac OS X antivirus protection.

The iOS Antivirus Protection Question

iOS which has its roots in FreeBSD focuses on traditional access control techniques such as passwords and idle screen locking to protect the device itself; application provenance which involves testing, verifying and tamper proofing individual apps and secure signing and hashing using a digital signature; encryption – which involves protecting device data in the event of loss or theft; isolation – this limits system and access to sensitive data; and lastly permission-based access control – granting apps a specific set of permissions to limit access to specified data and systems.

iOS architecture restricts what legitimate apps can do within iOS (i.e. there are for example restrictions on scanning another app's memory), which unlike Windows makes it virtually impossible for malware to work. This puts the 'trust 'with Apple (rather than say a mobile security product) who set about banishing antivirus software because iOS apps are sandboxed and cannot communicate with another app (exceptions do apply to some of Apples' own apps). You don't need antivirus running in the background, when apps are sandboxed – it's that simple. That said as highlighted above, jailbreaking a device i.e. jailbreakme (see previous section) opens the door to buffer overrun attacks and load jailbreak code into the iOS startup processes. Safari's PDF viewer was the culprit this time. Jailbreakme is jailbreak code that has one distinctive advantage – it is agnostic. This means it can work on every iOS platform/hardware – iPad, iPhone and iPod Touch.

Conclusion

It is useful to conclude this article by highlighting some useful security hints and tips for Mac OS X users. Most expert users will already know about them, but there is nothing like a useful refresh.

Apple Mac OS X 10.6 provides a useful firewall called Application Firewall and ipfw (an open source FreeBSD packet filtering and traffic accounting firewall). Encryption is provided by an application called 'FileVault'. This will encrypt everything inside a system home directory (not any files or folders outside of it). Some users might also wish to encrypt the virtual memory of their Mac OS – this is turned on by default in 10.6 Snow Leopard. For older OS X versions, users would need to turn on 'secure virtual memory'. Users should also consider using Password Assistant and Keychain (SSO) – the latter stores internet passwords, SSL certificates, servers, applications, websites and more for Single Sign On (SSO).

Security experts agree users should never run as an administrative account as any breach would only allow access to an account rather than the entire OS. In conclusion, there is no doubting that Mac OS X and iOS is more secure than the popular Windows/Windows Phone 7 platform for example. That said, as more and more Apple OS devices reach the market and end users look to synchronize these devices in the iCloud, the security risks could well increase. Then and only then will the malware writers stand up and take notice

JULIAN EVANS

Julian Evans is an internet security entrepreneur and Managing Director of education and awareness company ID Theft Protect. IDTP leads the way in providing identity protection solutions to consumers and also works with large corporate companies on business strategy within the sector on a worldwide basis. Julian is a leading global information security and identity fraud expert who is referenced by many leading industry publications.

Interception With Paros Proxy

Abraham Lincoln once said "Give me six hours to chop down a tree and I will spend the first four sharpening the axe". So, selecting a perfect tool and possessing the knowledge of how to use is most important before starting a task. Now, I am going to talk about a tool which acts as a proxy that modifies the conversation between the client and server.

When we talk about Man-in-the-Middle attacks, we come across two modes in it; one is passive and the other is active. Passive is the one in which the attacker regularly monitors the conversation between two persons without modifying the data or contents affecting confidentiality. Whereas the active is more dangerous compared to passive in which the attacker modifies the whole data between the two parties affecting integrity of data.

Now we will look after a powerful tool known as Paros which acts a proxy between client and server intercepting the data between them.

What is HTTP Proxy?

Proxy is a process in which it receives the requests from clients and forwards them to the server, based upon some predefined filtering rules. HTTP proxy is concerned with internet access where it resides between client and web server. Proxies may able to modify the requests and responses without permissions from both clients and servers. We will see how to modify the http requests arriving from client and pass on to the server using Paros tool.

Paros Proxy

Paros Proxy is a GUI based freeware tool mainly developed with an intention of evaluating the security of web applications. It was totally written in Java language. Its main goal is to assess the vulnerabilities present in the web applications by acting as a HTTP/HTTPS proxy. By using this we can intercept all HTTP and HTTPS data between client and server and can modify the data related to cookies and form fields. It can also be used for performing active MITM attacks by modifying the contents of the conversation between client and server. This tool can be downloaded for free from *http://www.parosproxy.org/*.

Step 1

Download Paros from above said link and install it on your system. As it is totally developed in Java, it requires JRE pre installed in your system to work properly. Start Paros by clicking on the icon. Next thing you should be cautious about is configuring it as a proxy in your system. To do that navigate to *Tools>Options>Local Proxy* and enter the address as *localhost* and port *8080*, also make sure that no other application is using

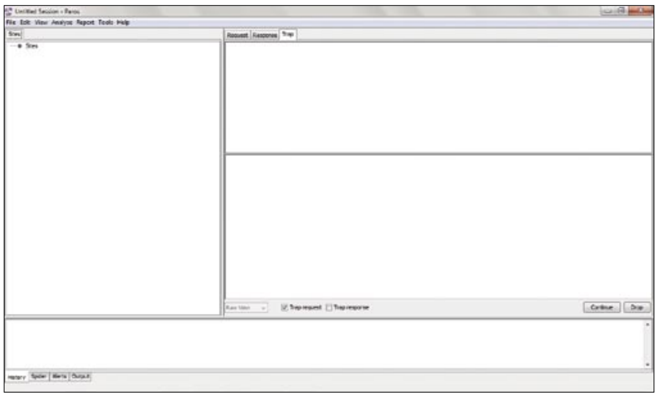
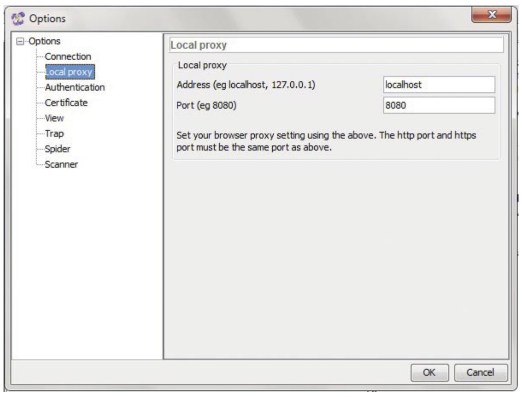


Figure 1. *Paros proxy*



Options

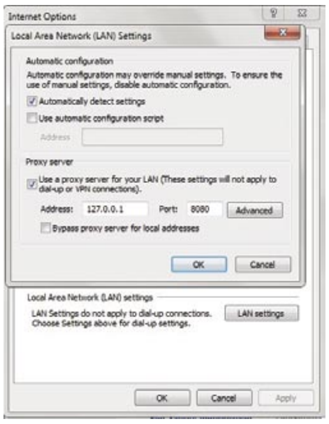
Local proxy

Address (eg localhost, 127.0.0.1): localhost

Port (eg 8080): 8080

Set your browser proxy setting using the above. The http port and https port must be the same port as above.

Figure 2. *Proxy settings in Paros*



Internet Options

Local Area Network (LAN) Settings

Automatic configuration

Automatic configuration may override manual settings. To ensure the use of manual settings, disable automatic configuration.

[✓] Automatically detect settings

[ ] Use automatic configuration script

Address:

Proxy server

[✓] Use a proxy server for your LAN (These settings will not apply to dial-up or VPN connections).

Address: 127.0.0.1 Port: 8080

[ ] Bypass proxy server for local addresses

Local Area Network (LAN) settings

LAN Settings do not apply to dial-up connections. Choose Settings above for dial-up settings.

Figure 3. *Configuring browser proxy*

the port provided above. You can also provide your own proxy server settings if you use one at your premises.

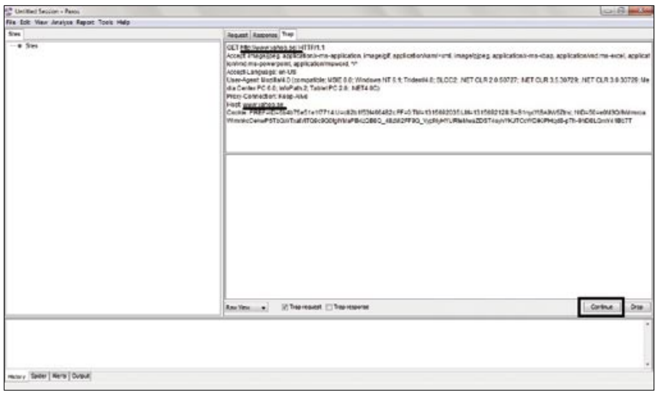
Step 2

You need to change proxy settings in your browser as well to enable the communication flow via Paros proxy. In Internet explorer, configure the proxy by following the path *Tools>Internet Options>Connections>LAN Settings* and enable Proxy server and enter the settings which was given to Paros proxy. If your computer is already using a proxy server, input the same into Paros Local Proxy settings, so that you can capture all the data flowing through the proxy.

Step 3

Now switch to *Trap* tab in Paros and enable the Trap request checkbox. Open the browser and start surfing the web. In my example, I demonstrated how to change search strings which are passed to Google search engine. Now my system is acting as both client and proxy. Initially I will try to access Google web page, but the server returns yahoo website. This is possible by editing the URL and forwarding them to the server. Let's see how to do it

Type in *www.google.com* in the address bar of the browser and press enter. Immediately the request goes to Paros and it waits there. In Paros window you can see that the browser is requesting for google webpage.



```
GET http://www.yahoo.se/ HTTP/1.1
Accept: [illegible]
Accept-Language: en-US
User-Agent: Mozilla/4.0 (compatible; MSIE 8.0; Windows NT 6.1; Trident/4.0; SLCC2; .NET CLR 2.0.50727; .NET CLR 3.5.30729; .NET CLR 3.0.30729; Media Center PC 6.0; InfoPath.2; Tablet PC 2.0; .NET4.0C)
Proxy-Connection: Keep-Alive
Host: www.yahoo.se
Cookie: [illegible]
```

Figure 4. *Modifying URL*

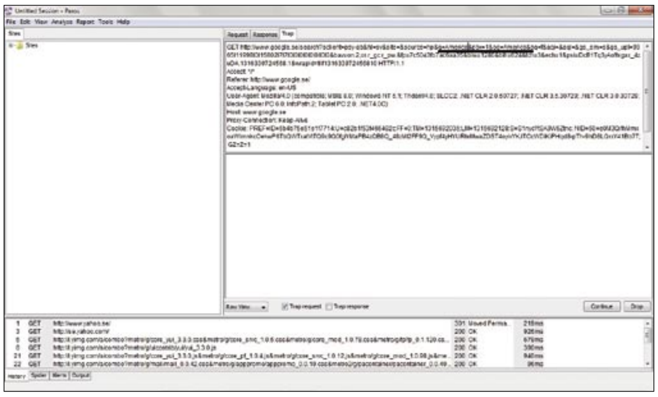


Figure 5. *Editing search string*

Now manually replace the string *Google* with *Yahoo* and press continue button. Keep on pressing continue button till you get no requests from the browser. The moment you press continue, the request is now passed to the web server and it returns yahoo web page. Strange the user requested for google and server responded with yahoo, this is how happens when we modify the HTTP requests.

Step 4

Now, let's have a look how to change search strings. Access the google page and search for let's say *America*. In Paros, search for something which says *…?q=America* and change the search string to *England*. Now server redirects the search to *England* rather than *America*.

By now you are aware of how to change HTTP requests using Paros Proxy. If you can set the Paros proxy settings to your gateway address, i.e Router (if any), then all the traffic in your network will pass through the proxy. But beware that this kind of act will result in serious loss of performance as you need to keep of accepting the requests and sending them to the server will consume lot of time.

Conclusion

Paros is a freeware and easy to use tool, there are so many other HTTP proxies available which you can use with the knowledge possessed by using Paros. Besides HTTP proxy, Paros can also be used as spider and scanner. You can also edit cookies which are sent to the browser. By examining the browser requests for GET and POST methods, we can obtain sensitive information like passwords and usernames in the URL. Also it can trap HTTPS requests send from your browser. This article is for learning and should not be used for any non-ethical purposes.

BHARATH SIVA KUMAR

Occupation: Master's student in Information and communication systems security at KTH, Sweden. Interests: Cloud security, Network and Application security. Hobbies: Bibliophile and pianist. Contact: bharath17@aol.com

Prey: From Praying to Preying

Since the issue 7/2010 article Prey: A new hope, there have been developments in the device tracking tool. It has been enhanced to now be able to monitor lost Android smartphones and tablets when activated. There was a reported case in May 2011 where a Californian harnessed evidence collected from a similar tool, Hidden, to recover his stolen Macbook.

The trend in mobile computing is the increasing popularity and adoption of smartphones as well as tablets which are compact compared to laptops and netbooks.

This is an ideal segment for Prey to aid whilst permitting you to have peace of mind in tracking your laptops (*Windows*, *Ubuntu*, *Mac OS*, *Linux*) too.

Setup

Prey is a simple Android app to install and configure.

Note

This article will focus on setting up Prey 0.5.3 on a Dell Streak.

- Register for an account with the Prey website (ie. *http://control.preyproject.com/signup*).
- Install the Prey app from *Android Market* (Figure 1).
- Launch Prey and a prompt will appear reminding you to register with the *Prey website. Click on Yes* (Figure 2).
- Enter your account information to add your device to your *Control Panel* (Figure 3).
- The *Phone Added*! message will appear upon successful registration. Click *OK* to proceed (Figure 4).
- Click Activate to protect Prey (Figure 5).
- The *Prey app* menu is simple with several options available (Figure 6).
- Accessing the *Device* section of the *Control Panel*, you will see the registered *Android device* (Figure 7).

Key Features

The *Prey* app starts collecting information pertaining to its current state when the *SMS activation message* is sent to the device.

The app starts to acquire *Geo* and *Network* information about its location silently in the background. The app will harness the gadget's internal GPS or Wifi hotspots to reveal its geographical location and IP address (Figure 8).

Click on *SMS activation* message from the *Prey* app menu to enter your predefined *SMS* text message that

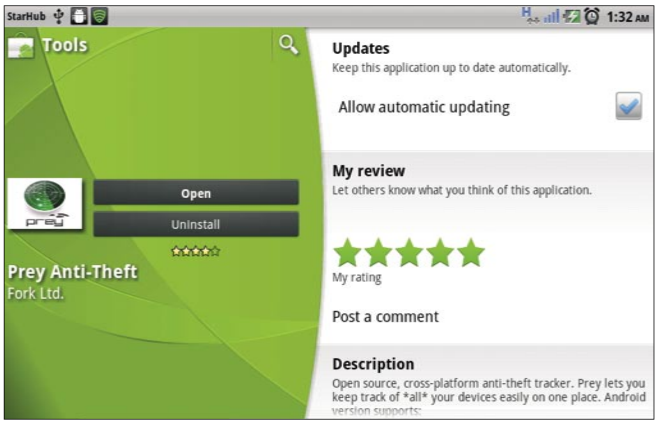
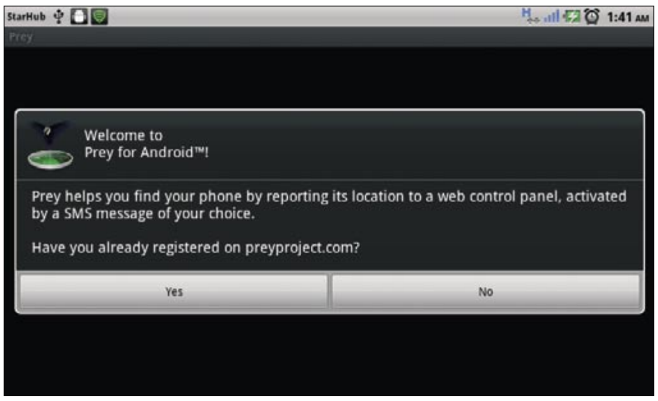

StarHub
1:32 AM
Tools
Open
Uninstall
Prey Anti-Theft
Fork Ltd.
Updates
Keep this application up to date automatically.
Allow automatic updating
My review
Let others know what you think of this application.
My rating
Post a comment
Description
Open source, cross-platform anti-theft tracker. Prey lets you keep track of *all* your devices easily on one place. Android version supports:

Figure 1. *Prey app*



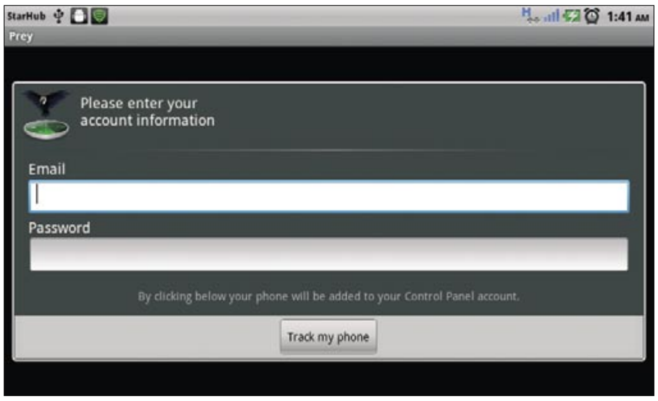
Welcome to Prey for Android™!

Prey helps you find your phone by reporting its location to a web control panel, activated by a SMS message of your choice.

Have you already registered on preyproject.com?

Yes No

Figure 2. *Prey registration prompt*


StarHub 1:41 AM
Prey

Please enter your account information

Email

Password

By clicking below your phone will be added to your Control Panel account.

Track my phone

Figure 3. *Device registration*

you would send from another mobile phone to trigger *Prey* to report its whereabouts. You can deactivate *Prey* by configuring an *SMS deactivation message*.

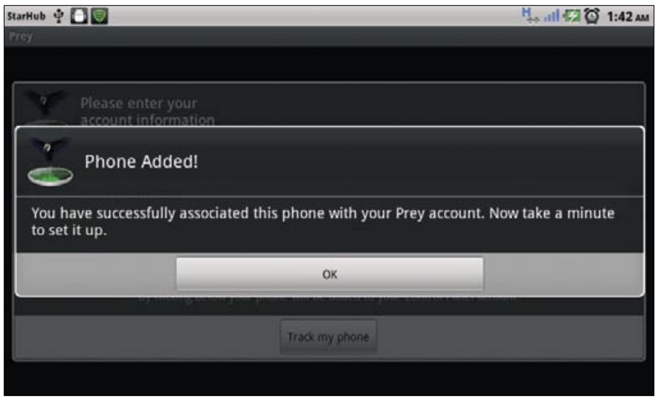
Prey App in Action

The *Dell Streak* is accessible to the mobile network with *Wifi* turned off. It was marked as missing by sending an *SMS* to it with the text message *GO PREY*.

A new report soon appears in the *Control Panel* (Figure 9). The report accurately displays a map with the exact road that the device is currently residing in when reviewed.

Note

The map and IP are removed for privacy reasons (Figure 10).

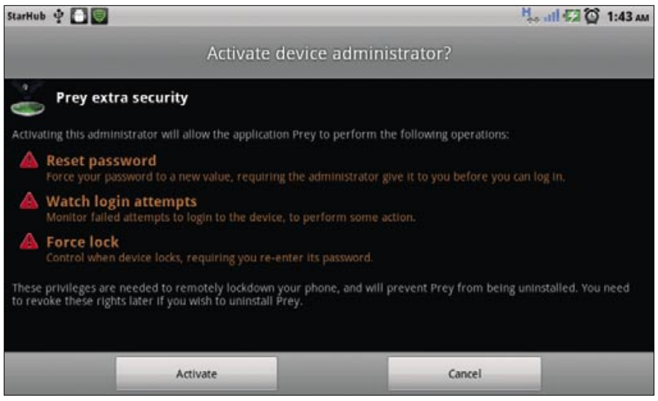

StarHub 1:42 AM

Phone Added!

You have successfully associated this phone with your Prey account. Now take a minute to set it up.

OK

Figure 4. *Registration success*


StarHub 1:43 AM

Activate device administrator?

Prey extra security

Activating this administrator will allow the application Prey to perform the following operations:

Reset password
Force your password to a new value, requiring the administrator give it to you before you can log in.

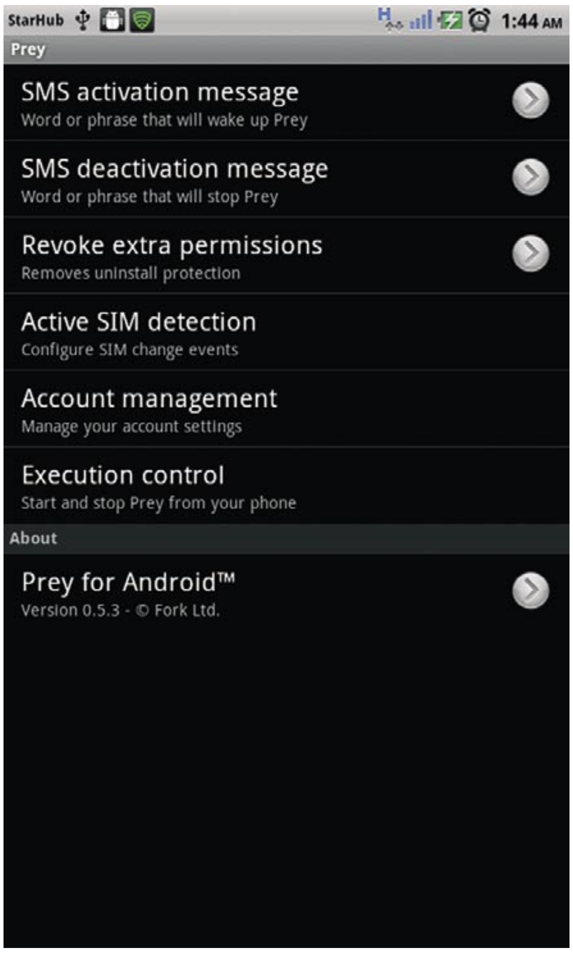
Watch login attempts
Monitor failed attempts to login to the device, to perform some action.

Force lock
Control when device locks, requiring you re-enter its password.

These privileges are needed to remotely lockdown your phone, and will prevent Prey from being uninstalled. You need to revoke these rights later if you wish to uninstall Prey.

Activate | Cancel

Figure 5. *Prey app protection*


StarHub 1:44 AM
Prey

SMS activation message
Word or phrase that will wake up Prey

SMS deactivation message
Word or phrase that will stop Prey

Revoke extra permissions
Removes uninstall protection

Active SIM detection
Configure SIM change events

Account management
Manage your account settings

Execution control
Start and stop Prey from your phone

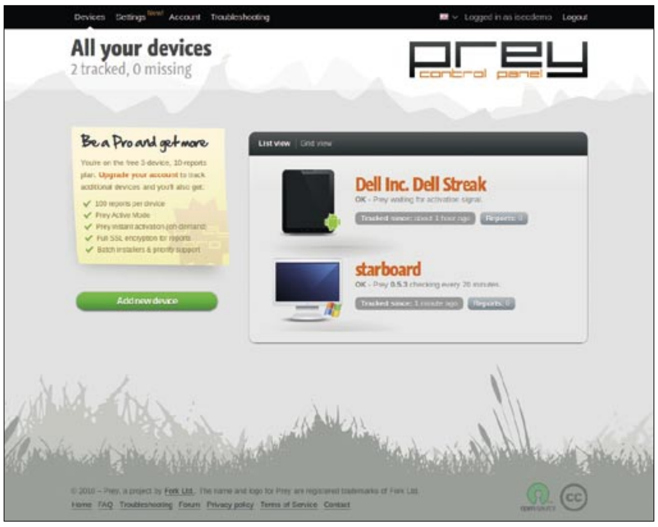
About

Prey for Android™
Version 0.5.3 - © Fork Ltd.

Figure 6. *Prey app menu*

Conclusion

Prey is highly effective on mobile phones and tablets as they are constantly available to the Internet via mobile networks. They do not need to have *Wifi* enabled to report its location back to the possessor. Internet access is also not required to trigger *Prey* on the lost device. An *SMS* message is all that is needed to mark the device as missing. The report with precise location information is instantaneously generated for prompt


Devices Settings Account Troubleshooting Logged in as [illegible] Logout

All your devices
2 tracked, 0 missing

Be a Pro and get more

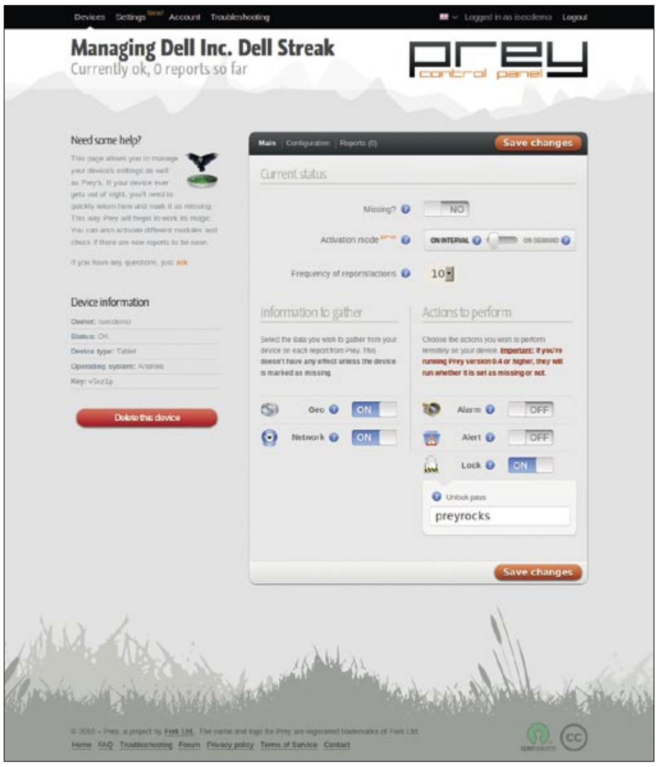
Add new device

List view | Grid view

Dell Inc. Dell Streak
OK - Prey waiting for activation signal.

starboard
OK - Prey 0.5.3 checking every 20 minutes.

Figure 7. *Device inventory*


Devices Settings New! Account Troubleshooting · Logged in as iseedemo Logout

Managing Dell Inc. Dell Streak
Currently ok, 0 reports so far

Need some help?
This page allows you to manage your device's settings as well as Prey's. If your device ever gets out of sight, you'll need to quickly return here and mark it as missing. This way Prey will begin to work its magic. You can also activate different modules and check if there are new reports to be seen.
If you have any questions, just ask

Device information
Owner: iseedemo
Status: OK
Device type: Tablet
Operating system: Android
Key: v5zz1p

Delete this device

Main | Configuration | Reports (0) — Save changes

Current status
Missing? NO
Activation mode: ON INTERVAL / ON DEMAND
Frequency of reports/actions: 10

Information to gather
Select the data you wish to gather from your device on each report from Prey. This doesn't have any effect unless the device is marked as missing.
Geo: ON
Network: ON

Actions to perform
Choose the actions you wish to perform remotely on your device. Important: If you're running Prey version 0.4 or higher, they will run whether it is set as missing or not.
Alarm: OFF
Alert: OFF
Lock: ON
Unlock pass: preyrocks

Save changes

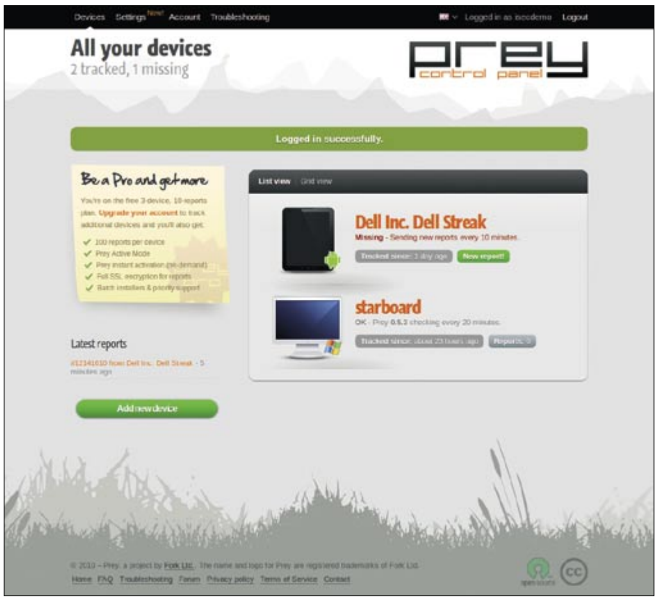
© 2010 – Prey, a project by Fork Ltd. The name and logo for Prey are registered trademarks of Fork Ltd.
Home FAQ Troubleshooting Forum Privacy policy Terms of Service Contact

Figure 8. *Device configuration*

action to be taken to recover the equipment. This is in sharp contrast to its laptop variant.

The purpose of this tool is to provide location and status information with the hope of recovering the device. This is the main rational not to include remote wiping capabilities into the software.

This tool is a double-edged sword. It not only facilitates owners to identiy their device's location but if not managed properly, the owners themselves can be traced. In the case of the *Prey* app, a tell-tale sign that the *Prey* account is compromised is the receipt of the *SMS activation message*. Reclaiming the account could


Devices Settings New! Account Troubleshooting · Logged in as iseedemo Logout

All your devices
2 tracked, 1 missing

Logged in successfully.

Be a Pro and get more
You're on the free 3-device, 10-reports plan. Upgrade your account to track additional devices and you'll also get:
✓ 100 reports per device
✓ Prey Active Mode
✓ Prey instant activation (on-demand)
✓ Full SSL encryption for reports
✓ Batch installers & priority support

List view | Grid view

Dell Inc. Dell Streak
Missing - Sending new reports every 10 minutes.
Tracked since: 1 day ago · New report!

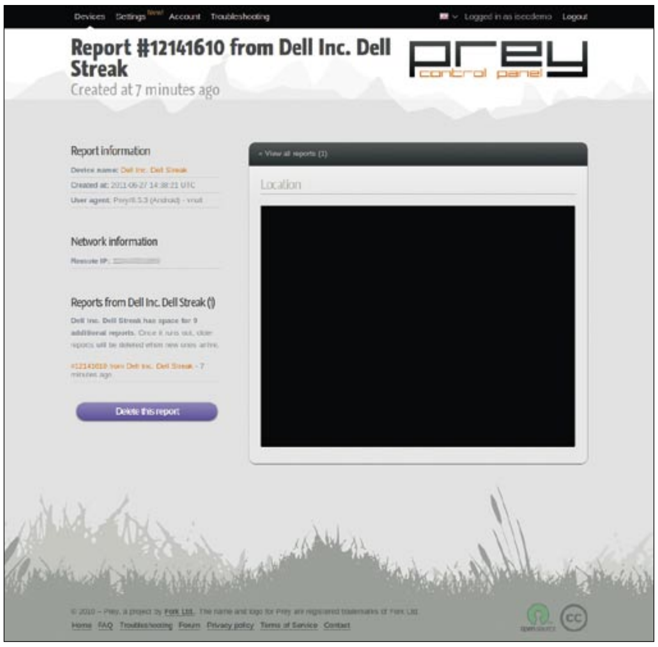
starboard
OK - Prey 0.5.3 checking every 20 minutes.
Tracked since: about 23 hours ago · Reports: 0

Latest reports
#12141610 from Dell Inc. Dell Streak - 5 minutes ago

Add new device

© 2010 – Prey, a project by Fork Ltd. The name and logo for Prey are registered trademarks of Fork Ltd.
Home FAQ Troubleshooting Forum Privacy policy Terms of Service Contact

Figure 9. *New report*


Devices Settings New! Account Troubleshooting · Logged in as iseedemo Logout

Report #12141610 from Dell Inc. Dell Streak
Created at 7 minutes ago

Report information
Device name: Dell Inc. Dell Streak
Created at: 2011-06-27 14:38:21 UTC
User agent: Prey/0.5.3 (Android) - vnull

Network information
Remote IP: [redacted]

Reports from Dell Inc. Dell Streak (1)
Dell Inc. Dell Streak has space for 9 additional reports. Once it runs out, older reports will be deleted when new ones arrive.
#12141610 from Dell Inc. Dell Streak - 7 minutes ago

Delete this report

« View all reports (1)
Location

© 2010 – Prey, a project by Fork Ltd. The name and logo for Prey are registered trademarks of Fork Ltd.
Home FAQ Troubleshooting Forum Privacy policy Terms of Service Contact

Figure 10. *Device location*

be challenging so a strong password is recommended to secure the account. The same password safeguards access to the Prey app menu on the device.

The caveat of the free account is that *HTTPS* is not enabled to furnish integrity of device information to their servers. This is remediated by paying for a *Pro account*. It is also not bulletproof if the thief restores the mobile device to factory default or replaces the firmware.

Prey is not yet available on iPhones or iPads but could be added to the stable in the near future. There has been criticism and scepticism with regards to this service but they can be easily be overcome by opting for the commercial license. Install *Prey* on your portable devices now to have peace of mind and hope in recovering them.

MERVYN HENG

Mervyn Heng, CISSP, loves Information Security and Open Source. These interests are translated into his life in Singapore where he practises the 2 philosophies and attempts to transfer these passions to his friends through awareness. If you have any comments or queries, please contact him at commandrine@gmail.com.

Attend the second

Android Developer Conference

AnDevCon II

The Android Developer Conference

November 6-9, 2011 • San Francisco

100% Developer Focused!

30+ Expert Speakers

70+ Technical Classes and Workshops

"A lot of useful, cutting-edge information."
—Alfred Mirzagitov, Sr. Software Engineer, Webroot

"AnDevCon had lots of great information, excellent speakers and a coherent program."
—Paul Verger, Software Developer, Pico Software

"There were great presentations with very professional lecturers. Go for it!!!"
—Eyal Zmora, Software Engineer, NDS Technologies

Google KEYNOTE!

Chet Haase and Romain Guy present:

"Android Awesomeness"

Register Early and SAVE!

Download the complete course listing at

www.AnDevCon.com

A BZ Media Event

AnDevCon™ is a trademark of BZ Media LLC. Android™ is a trademark of Google Inc. Google's Android Robot is used under terms of the Creative Commons 3.0 Attribution License.

Facebook and the Fuzz

Smartphones, Social Media, and Policing Civil Disturbances

Mobile telecoms is a very, very hot topic in Britain this year. Much of the year saw the investigation playing out around mobile phone "hacking" by journalist – this apparently touched everyone from the Queen to various minor celebrities. In reality, the hacking in question was nothing more than some journalist being aware of how to access voicemail for which default PIN codes were in use. Nonetheless, the scandal involved politicians on all sides, and led to calls for the resignation of the Prime Minister.

Perhaps more important, however, is the role of the Blackberry in the wave of riots and looting that burned across the UK in August. On Thursday 4th August, a 29 year old man was shot by police in the north London suburb of Tottenham, in an incident allegedly involving an illegal firearm. The next day, peaceful protests began against police. Police shootings are rare in the UK; in the last three years there have been a total of eight. By the evening of Saturday the 6th of August, these had turned into violent attacks against businesses and the police. By Sunday evening, more attacks were occurring in other London districts, such as Brixton, Battersea (15 km

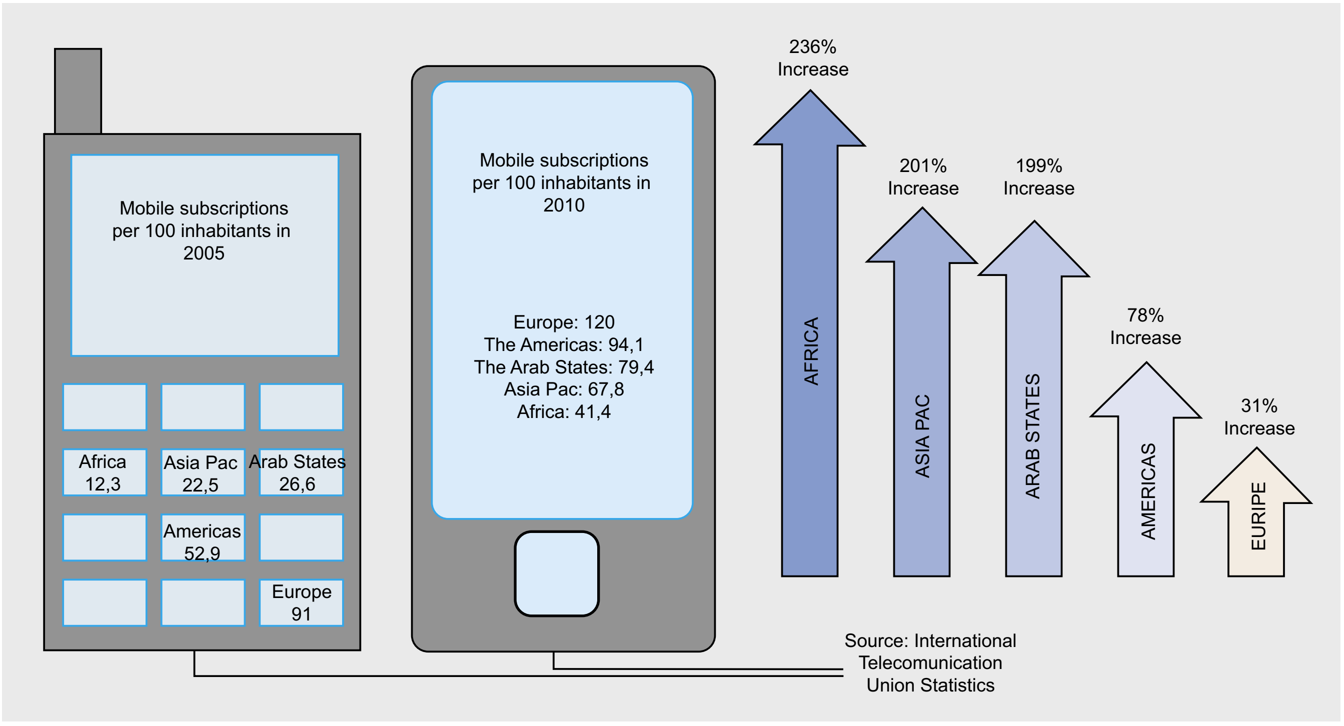


Figure 1. *Mobile subscriptions per 100 inhabitants*

from Tottenham), and Croydon (25km). By the following Tuesday, violence had spread to other cities, such as Manchester, Nottingham and Bristol. All of these incidents had marked similarities – gangs of mainly young people, attacking shops and looting shops selling *luxury* goods such as electrical items and trainers. More importantly, these groups were not related, and the disorder had nothing to do with the original protest against police.

Blackberries are extremely popular among the young in London. It transpired that many of the specific incidents had been co-ordinated using *Blackberry Messaging* (BBM). BBM is a device-to-device service, often touted as *untraceable*. There was much discussion in Parliament and in the British media around the possibility of suspending the service, or using *national technical resources* (i.e. Britain's electronic spying group, GCHQ) to provide intelligence to the police. The problem is, Blackberries are not just indispensible for young looters with a taste for the latest Nikes; they are also a vital communications tool for many of Britain's businesses. The role of mobile telecoms in maintaining (and threatening) security is not a problem that is confined to the UK.

The Mighty Device

The ubiquitous smartphone – you can shop, chat, manage your e-mails (work and personal), announce your location (and check those of friends). The list goes on. More than that, in recent times we have seen real social change enabled by technology including such devices – for example in Moldova, Iran, and more recently in Tunisia and Egypt – as smartphones have facilitated the rapid, first hand, communication of events. Moreover, the popularity and increasing affordability of smartphones has led many organisations to consider their use to provide ready access to corporate resources. This is perceived to lead to significant gains, in terms of cost and efficiency. These activities can be performed via the handset's web browser, customised applications or proprietary systems such as Microsoft Exchange. Whatever the technology used, clearly the smartphone has become the custodian of a large amount of corporate data, sometimes including highly sensitive information.

The Whirlwind Tour Of Some Issues

All this potential does not come without risk. Most smartphones available at present display inherent security flaws. In many cases, the security features in handsets are not an intrinsic part of the device design, and amount to merely superficial additions to what is otherwise a product driven mainly by the need for a fundamentally open architecture. Such security as exists in Android and iOS can easily be overridden after user provisioning. As the complexity and functionality of smartphones increases, so too does their susceptibility

Join hakin9 team!

If you would like to help our team in creating hakin9 magazine you can join our authors or betatesters today!

All you need to do, is to send an email to:

editors@hakin9.org

and give us a brief description of your field of interest.

We look forward to hearing from you!

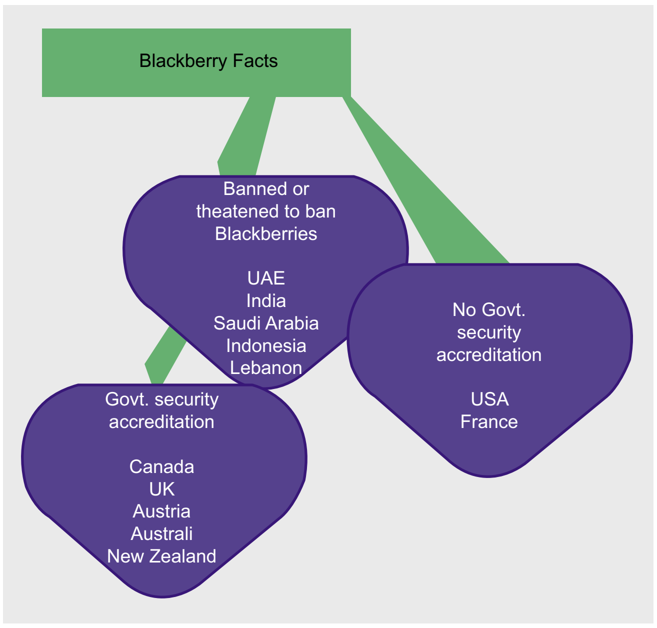

Blackberry Facts
Banned or theatened to ban Blackberries
UAE
India
Saudi Arabia
Indonesia
Lebanon
No Govt. security accreditation
USA
France
Govt. security accreditation
Canada
UK
Austria
Australi
New Zealand

Figure 2. *Blackberries and Government Security Accreditation*

to security vulnerabilities, and the potential impact of a security breach. The means to secure smartphones designed with a consumer, rather than a business user in mind, do exist in the form of third party add-ons, but uptake has been patchy at best.

Some figures about smartphone uptake give a feel for the size of the issue:

- 200000 Android devices activated every day in August 2010
- August 2010 – first malware reported for Android OS
- 4 paid for Android Apps with greater than 250,000 copies distributed by end January 2011
- 1627 free Android Apps with greater than 250,000 copies distributed by end January 2011

In a corporate context, smartphones represent an extension of the corporate perimeter, even more so if the same device is used for both corporate and personal functions. Since they are not equipped with security measures of equivalent strength to those usually found in other mobile IT equipment like laptops, it is hardly surprising that they are now often targeted by criminal organisations and other threat agents. Indeed, given the fact that they are highly likely to be used outside the physical premises, this increases their attractiveness as a target. Rapid uptake of the technology simply ensures that the *bad hats* will be more familiar with the possibilities – for example rogue apps or smartphone-focussed malware.

The BlackBerry platform may be offered as a counter-example to this trend. Since it was designed from the outset for business users, we can broadly say it is based on more secure design principles than other platforms on the market. Nonetheless, the security of BlackBerry devices still depends on effective deployment of a robust set of security policies. The Blackberry has been accepted for use with Restricted material by some governments, and banned by some others, which can be interpreted as an indication that it is secure enough to prevent eavesdropping by a fairly sophisticated organisation. That said, not all Western governments have been as accepting – France, for instance, has a specific ban on the use of Blackberries in a government context.

What Does All This Mean?

In the UK, there is a suggestion that the government will attempt to bring in new legislation in September 2011 that enables the *temporary* disablement of particular messaging and social media services. This may well be another piece of not-very-well thought out legislation. It is probably unworkable, given the international and amorphous nature of social media technologies. More importantly, social media is a confusing mixture of private free speech and public announcement. As this is is being written, a number of individuals have been arrested in the UK for alleged offenses relating to the use of social media to incite public disturbances. In one case, a person arrested for starting a Facebook page dedicated to rioting in her home town. However, this rioting never occurred.

What it does however do is signpost the way legislation is likely to move, as governments realise the real significance of the intersection of mobile telephony, social media, legitimate economic activity and criminality. Let's think about what all this means to the individual. Firstly, there will now be more agencies than ever that are interested in intercepting mobile communications than ever before. Secondly, with that increase in surveillance, individuals need to be aware that Facebook, for example, isn't just for communicating with your friends – you could be inadvertently communicating to the police and *other agencies*. Saying something stupid or just plain ill-considered could come back to haunt you. We can expect to see police forces themselves slowly move more into the social media space – various parts of London are the subject of, what are, in effect, *crowd sourcing* identification of looters, with CCTV footage being displayed on large screens. The net result? The forces of law and order have moved into cyberspace, and they will not be leaving in the foreseeable future.

DRAKE

Drake has worked on information security and strategy with government agencies, the military, financial institutions and other blue chip organisations in Europe, the Middle East, and Africa since Boris Yeltsin was President.

WE @ iViZ

HATE

FALSE POSITIVES

AS MUCH AS YOU DO

1st Cloud based Application Penetration Testing service

No Tools | No Consultants | Zero False Positives | Cost effective

www.ivizsecurity.com

Interview With David Harley

David Harley is an IT security researcher, author and consultant to the security industry living in the United Kingdom, known for his books on and research into malware, Mac security, anti-malware product testing, and management of email abuse. He is a director of the Anti-Malware Testing Standards Organization, a Fellow of the BCS Institute, and runs the Mac Virus website, and we persuaded him to take some time out from talking to the press and writing presentations to talk to us instead.

How did you get started in the field of Anti-malware research?

I think I actually told part of that story in Viruses Revealed, but I'm happy to revisit it. Perhaps it will gain a bit of glamour since I'm now ten years further away from the event. In 1989 I was working (primarily as an administrator) for a research department in a hospital in London, but I was taken on by what was then the Imperial Cancer Research Fund (now Cancer Research UK) to work with a small team set up to run a couple of major meetings/conferences for people involved in the Human Genome Mapping Project. Just after I started there, the hospital rang me to ask about a problem they had with the AIDS trojan (*http://en.wikipedia.org/wiki/AIDS_(trojan_horse)*). As it happened, while I had no direct involvement in security at the time, I knew enough about that trojan to refer them to Jim Bates, who'd cracked the encryption and had developed tools to remove the malware and reverse the encryption. I always say that I can remember the beginning of my involvement in security because it was the same day that my daughter was born, and in fact I got the phone call just as I was on my way out to the hospital.

Having I set up antivirus on workstations for the HGM11 conference: Dr. Solomon's provided us with free scanning software and I wrote some utilities to make it harder to escape the eagle eye of what was essentially an on-demand scanner: not quite a complete memory-resident shell, but a suite of MS-DOS utilities that I used to enforce scanning at login and logout, scheduled scanning, in fact several of the features we now take for granted in modern security software in a multi-tasking operating system. Subsequently, I was taken on by the IT unit, and one of my first jobs was to clean up and document the AV installation process, I was commissioned to do an article on viruses for an external newsletter, and things kind of snowballed from there.

In one of your bios, it was mentioned that you are interested in the 'psychosocial aspects of Security'. What exactly do you mean by that?

Probably this reflects my academic background, which is a slightly strange mixture: I started off in the 1960s reading social sciences and psychology, and at the time I started to drift into security, I was finishing what was primarily a computer science degree. But I actually think that makes for quite a good mixture in this field. While I'd like to think I have a better than fair technical grasp of the topics I'm alleged to be expert in, I haven't been a hands-on tech guy for a long time, and there are many people who can do malware analysis and development better than I do, but I think that it's important to have people in the industry who can place malware and anti-malware technology in a broader social context. Technology is critical, but you can't, as they say, fix social problems with purely technical solutions.

Tell us a little bit about the Anti-Malware Testing Standards Organization (AMTSO) and what role do you play as Director?

AMTSO was founded as a result of the recognition by mainstream testers and the AV industry that AV comparative testing was a free-for-all. Our mission statement says we focus *on addressing the global need for improvement in the objectivity, quality and relevance of anti-malware testing methodologies*. You can criticise us for the ways in which we've approached that aim and our failure to fix the problem (we haven't fixed global warming, either!), but I still believe in the aim. The Board of Directors is a small group of elected and unpaid volunteers who try to translate the will of the AMTSO members into somewhat effective action through cooperation with other groups, workshops and provision of resources, public relations and so on. Because I have a fairly public profile, I get caught up in PR issues a lot, but I'm actually trying to focus on documentation, which is the area in which I'm most comfortable in my own skin.

What kind of advice would you offer to those who want to pursue a career in Anti-Malware development?

Don't do it. Not, at any rate, if you have the urge to be loved. I'm not altogether the best person to ask about this, as while I've never been involved at first hand in product development – well, not in a commercial environment. My background is more in incorporating commercial products and services into a multi-layered defensive strategy in a customer environment.

What are some of the characteristics an Anti-Malware researcher should possess?

Research is a very broad church, and I sometimes think I'm on a different planet to some of the people who also describe themselves as researchers. Even in the anti-virus industry, attitudes in some areas have become less hardcore and isolationist in recent years, though there's still general adherence to ethical standards that aren't necessarily understood by the rest of the world, or even other sectors of the security industry. Doggedness. An acute sense of responsibility towards your customers, whoever they may be. Always remaining mindful of the fact that security is one (admittedly important) part of a much bigger picture: for instance, a totally secure business would probably not be profitable... An ability to recognize the limitations of your own knowledge and expertise: the worst speech defect in the world is the inability to say *I don't know...*.

What does Small Blue-Green World do?

Small Blue-Green World (*http://www.smallblue-greenworld.co.uk*) started when I left the UK's National Health Service in 2006. I was intending to develop along more independent lines as a writer and consultant, and I figured it was going to be more of a publishing venture than it's actually turned out to be. However, we have a lot of *stub* processes that are likely to get more attention sooner or later, and we have done a few things such as travel books (albeit with very limited circulation).

For a long time SBGW was basically an administrative shell for my own consultancy and authoring services, but it's showing signs of taking on more of a life of its own.

Recently you launched the 'Geek Peninsula' blog, what is the objective of this blog?

Originally, it was just a matter of the pun *Geek Peninsula* popping into my head – I guess I just have one of those minds – and wondering if anyone had used it as a blog name. They hadn't, so I had to think of a use for it, but that didn't take much time. Actually, I guess it's kind of recognition of a slightly odd development in my career, and an example of the way in which SBGW is becoming more than a shell that generates invoices.

It's really about the way I look at my work. It's not just about understanding the technologies of security and malicious logic, but about being an educationalist of sorts. Some of that is a matter of communicating information through blogs, white papers, conference presentations and so on. But lately I find myself spending more and more time talking to the media, and trying to develop a sense of what it is they want (or need) to know about, as well as trying to explain difficult concepts in straightforward language that will be understood by a wider audience than a specialist conference paper. I guess I've gradually realized that while I'll never be as good as someone like David Perry or Graham Cluley at some of that stuff, the amount of time I spend talking to the media has rocketed in the last year or two, and I'm seeing more formal recognition of my notoriety.

Not to blow my own trumpet (much), there was something of a wakeup call when I found myself in Sys-Con Media's Top 25 *Most Powerful Voices in Security*. I can think of many more than 21 people I'd consider more influential than myself, but it was still a bit of a shock to see that metric and realize that people aren't just reading my blogs for ESET or AMTSO or AVIEN. (On the other hand, it was a shock to realize how many people do read them.)

Small Blue-Green World hasn't spent time on PR activities in its own right up to now – it hasn't been necessary, since all that was handled by the companies I've been working with – but as more of my writing/talking is done (like this interview) in contexts where I'm not representing the interests of a customer, it seemed sensible to have a centralized information resource where people can see SBGW as both a brand and a diverse resource in its own right. I'm not quite like the writer Jack Trevor Story (*http://en.wikipedia.org/wiki/Jack_Trevor_Story*), who claimed that he kept everything he'd ever written including laundry lists, but there are evidently people who are as (or more) interested in independent activities such as AVIEN and Mac Virus as the stuff I do for ESET, which is my main customer. So it lists (when I remember to update it) everything of mine that's published in a given week, and some of the press coverage. Also, much of that press coverage is a few words or sentences abstracted from a lengthy email. In PR terms, that's fine, but sometimes it's nice to place those quotes or paraphrases into a more complete context, and that's an aspect I'll probably explore further.

MAC SECURITY

Can you list some of the companies that are leaders in Mac Anti-Malware?

Well, as I work very closely with one particular company that has a Mac AV product, it would be a bit flaky for me to attempt to rank products, but I can certainly enumerate some of the better products. Sophos, Symantec and McAfee have worked in that space for a long time, and it's reasonable to expect their Mac products to integrate well with their products on other platforms. Kaspersky, ESET, Avast!, Panda and F-Secure are newer kids on the Mac block, but they're all consistently high performers in malware tracking generally, and have products effective across several platforms. Intego don't have the same spread of products across platforms, but they've done consistently excellent research in the field of Mac malware, and are often the first off the block with new Mac threats.

What are some of the computer architectural differences between PCs and Macs that makes one more secure than the other?

There is no architectural difference at a hardware level that gives Windows or OS X a clear edge in terms of security, and both operating systems can be configured to be pretty secure. Both operating systems (I'm comparing current NT-derived Windows and OS X rather than, for instance, Windows 98 and OS 9) have outlived many of the older malicious programs that affected them in the '80s and '90s. Clearly, in terms of sheer volume of threats targeting users of each OS, there is no comparison: while the ways in which AV companies measure threat volumes differ very widely, tens and even hundreds of thousands of binaries are processed by AV labs on a daily basis. The volume of Mac-specific threats (or even of cross-platform threats that can affect Mac users) is ridiculously small by comparison. I believe that some of that is simply down to the much higher Mac-using population, of course, and it might even be that there are blackhats who subscribe to the rumours that Macs are secure and Mac users are too bright to succumb to social engineering. I wouldn't go that far myself, though..

Why are commercial companies not as much involved in MAC security compared to PC Security?

Basically, there isn't a vast untapped population of potential Mac customers eager to hand over cash to the AV industry. Lots of Mac users firmly believe that there is no significant risk to them from malware, and those who do actually have a couple of commercial-grade products available to them that are free to home users. Obviously, those products exist because even those companies don't expect to cover the costs of maintaining a commercial consumer product: the main market is in the corporate space where customers have Apple machines and see a need to maximize their protection across all platforms within the enterprise. I do think that it's reasonable to expect a company with an enterprise-oriented product to include Mac protection: it's not just a matter of protecting Apple machines, but also of protecting the enterprise from other threats such as spearphishing using Mac users as a vector.

You have criticized Apple's 'stealth approach to dealing with occasional malware' in one of the InfoSecurity (UK) article interviews. What approach would you recommend?

Well, the attempt at stealthiness isn't such a bad thing in itself, even if it stems from denial rather than a real defensive strategy: if people weren't aware that the XProtect existed, they wouldn't have the unrealistic expectation that it would magic away all past, present and future malware that might affect a Mac user. The problem is that Apple has tried to take a 1980s/90s approach to AV, trying to detect specific malware variants. The MacDefender fake AV highlighted one of the problems with that approach: if the home team learns to block an attack, you change the attack. Of course, this is a weakness with conventional AV, which is why security firms have evolved so many proactive strategies. They can't detect *every* unknown threat (or anything like it), but they *can* detect a significant proportion of unknown threats. Apple doesn't have those decades of technological evolution to draw on: if I were working in Cupertino, I'd be advocating partnering with one or more AV companies to develop a less reactive approach, rather than reinventing a very old wheel. But that would require a cultural shift within a company that rarely acknowledges security issues publicly.

As it stands, XProtect is inadequate because it doesn't detect the whole range of existing threats, it doesn't have full-blown on-demand/on-access scanning, and it won't necessarily detect malware it is supposed to recognize because of limitations in the detection mechanism. For example, assuming that malware attacks will be vectored via applications that set *com.apple.quarantine*. This assumption actually nullified the detection for the iWorks trojan. You can argue that it's better than nothing, but in my experience, a strategy that offers inadequate defence can expose the user to *more* risk than no protection at all if they assume they're better protected than is actually the case. However, since no single solution offers 100% security, you could say the same about strategies that are far more effective. If you were running the best AV scanner in the world and you trusted it so much that you assumed you could click on *anything* because your scanner would protect you, sooner or later you'd pay for that because you'd click on something it *didn't* detect.

Is an updated version of 'Viruses Revealed' book on its way?

Sadly *Viruses Revealed* didn't sell particularly well – at any rate, not after the first few weeks. In fact, books about malware rarely do. So Osborne declined to take up its option to commission a second edition. The rights to the book have reverted to the authors, and we might yet re-use some of the material, but a lot of the material is very much of its time, an era where a much higher proportion of all malware was replicative. The *AVIEN Malware Defense Guide from 2007* (*http://smallbluegreenblog.wordpress.com/publications/*) is a closer reflection of the current landscape, and if I go on to another generalist book on malware, it'll probably be more like that. However, I'm not sure that another book like that is what the world needs most. My next book project is a small contribution to a book on SCADA security, but after that I'm thinking of putting together something on comparative evaluation and testing. As you might not be too surprised to hear.

In one of your blog posts you mentioned that 'Apple's Security Model is pretty good', can you explain to us why?

At one time, Apple's approach to user privilege and automatic updating led the field. They made it quite difficult for an OS X user to run routinely as root, and led him by the hand into updating when updates were available. They also did (and continue to do) sterling work on defensive technologies within the operating system and peripheral technologies. But Microsoft and Linux developers are (whatever some Apple fans try to tell you) also working on these issues. There is no *best* OS in these respects: each OS version does some things well, some not so well, and it's not easy to compare them meaningfully. The flaw in the *Macs Good, Windoze Bad* argument is that these countermeasures are in no sense a complete answer to malicious activity. In particular, there are precious few countermeasures that can't be circumvented by social engineering. Commentary based on the assumption that social engineering somehow *doesn't count* is, in my not particularly humble opinion, misguided and even potentially dangerous. There's no room for wishful thinking in real security.

Who do you follow (on twitter/blog etc) when it comes to Mac AV technology?

I tend not to follow pure product news too closely. I know some people are discontented that I don't do stuff like comparative reviews on Mac Virus, and I wish I could cite a dependable resource for that, but it would be ethically unsound for me to get too far into that while I'm working so closely with an AV company: even I didn't hold that view, it would compromise my ability to be considered impartial as far as some of my readers are concerned. Naturally, I *do* follow individual and company blogs and Twitter accounts where the company has a professional interest in Apple security, but I track all sorts of security-related resources, and good information doesn't always come from commercial sources. Bloggers and tweeters of interest include (just off the top of my head): Paul Ducklin, Mikko Hypponen, Graham Cluley, Luis Corrons, Chester Wisniewski, Charlie Miller, Ryan Naraine, Brian Krebs, and some less obvious people such as John Gruber and Ed Bott.

Can you describe us some of your daily tasks in running Mac Virus website?

Mac Virus is basically a blog site at the moment, though I answer queries by email from time to time. But I don't have time to give it the attention I'd like, and right now my first priority for blogging on Apple-related issues is Infosecurity Magazine (*http://www.infosecurity-us.com/blog/user/david—harley*) Of course, Mac Virus is a bit of a misnomer these days (the name goes back to when Susan Lesch first built the site in the 1990s). I look at mobile devices and iGadgets as well as Macs, and not only Apple products. And there are virtually no Mac viruses nowadays: the threats cover a much wider range. (Of course, the same is true in general of Windows, though we continue to see high profile *true* viruses from time to time: Win32/Induc is an interesting current example.)

What was the hardest problem you had to deal with?

As far as Mac Virus is concerned? Time... I have a ridiculously busy schedule, and I may have to go days or weeks without updating the web site, and there are projects such as the malware descriptions page that I find it very hard to find time for. Mac Virus brings in no income, so paid work has to take precedence. Over recent years, there has been the occasional offer to sponsor or help out with content, but it hasn't often worked out. Fortunately, my friend Old Mac Bloggit (yes, of course that's a pseudonym, but I'm not telling you whose!) occasionally contributes some acerbic and thought-provoking commentary which has raised our game somewhat. Of course, since we deal with topics that Apple fanboiz are sometimes rather sensitive about, we draw a lot of fire as *Mac haters*. Having owned and loved many Macs, and having spent many years supporting Mac users in a professional context, I find that both irritating and amusing. After all, several of my books have been primarily written on Mac laptops... And I also get somewhat irritated by people who expect me to ring them at my own expense to give them free technical support at 4 o'clock in the morning because I'm several time zones away. Not to mention the students who want me to write their assignments for them...

The most challenging problems I've had over the whole of my career in IT were probably during my time running the Threat Assessment Centre for the UK's NHS and my involvement with the security side of its messaging services. I could fill many pages with those war stories, but perhaps not right now.

What was the strangest one?

Probably the correspondent who was convinced that problems with his Mac were due to an international or possibly intergalactic conspiracy. In fact, I've had to politely disengage from many such dialogues in the past in very different contexts. However sympathetic one might feel, it's all too easy to drown in someone else's brainstorm.

What part of your work you consider the most satisfying and enjoyable?

The work I do with the ESET researchers in Russia, Slovakia and thereabouts. It's the nearest I get to hands-on malware analysis and research these days, and I'm grateful to them for letting me share some of the credit for the great work they do. And I wouldn't be human if I didn't get a bit of a buzz from doing an interview like this: everyone likes to feel that their work and opinions are considered to be of interest and value.

ANTI-VIRUS TESTING

What are the fundamental principles of Anti-Malware testing?

They're actually listed at *http://www.amtso.org/amtso---download---amtso-fundamental-principles-of-testing.html*, but the basically codify our belief that everybody involved in such testing must behave ethically, test properly and communicate in a fair and accurate way. But they're not the ten commandments: they are, or should be, a living set of principles that can be modified as we learn more about ourselves and our industries.

What is Dynamic testing and its best practices?

Dynamic testing is, in principle, the opposite of static testing, where a product is expected to detect malware without allowing the malware to execute. Essentially, dynamic testing exercises a product's ability to detect malware using some form of dynamic analysis, where the suspected malware is allowed to execute (albeit in a restricted *safe* virtual environment). It tests technologies and approaches such as emulation, sandboxing active heuristics, behavioural analysis, and so on. (NB, these are closely-related, overlapping technologies, not totally discrete approaches.) Static testing involves the passive scanning of samples that aren't allowed to execute. Since it's quite easy to disguise malicious code so that it isn't detected in this way. Take a Trojan that in its *pure* form is easily detectable, and obfuscate it. Passive scanning may easily miss it, where an on-access scanner with the same signatures will detect it as soon as it is executed.

In the case of Dynamic testing how do you go about Product and Sample selection?

Whether testing is dynamic or static, the essential aim is to select products that have somewhat similar functionality and are configured to the same degree of paranoia. If you have one product that flags every executable as *suspicious* by default and another that doesn't, then leaving them in their default configuration isn't testing detection, it's testing design philosophy, and the first one will probably *detect* far more samples but with many more false positives. It may be that if you configure them to a roughly equivalent level, their performance will also be roughly equivalent. Sample selection is really too complex an issue to discuss in a short(ish) interview. At AMTSO we are hoping to have a *sample selection for testing* guidelines paper ready for voting at the next workshop, in October.

Can you explain Ultracrepidarianism and its relation to AV industry?

My, you have been doing your research on me. It was actually Rob Rosenberger who introduced me to ultracrepidarianism, in a very sane and influential article on *False Authority Syndrome* (*http://vmyths.com/fas/*). It means something like *acting or speaking outside one's own knowledge or ability or experience*. It comes from the Latin ultra crepidem (*beyond the sole of the* *shore*), and the story is that the Greek painter Apelles was criticised by a cobbler for the way he'd depicted a human leg: Apelles accepted the criticism as far as the subject's slipper was concerned, but not of the leg itself, as that was outside the cobbler's expertise (*http://en.wikipedia.org/wiki/Apelles*).

The concept of ill-informed commentary applies to all fields of human endeavour, but it particularly irritates me in the context of testing, where it's widely assumed that anyone can test AV performance. Most testing is amateur. Even some tests that charge the consumer or the vendor are run by people outside the field with questionable experience and expertise. But their conclusions are rarely challenged by the public or the media.

MISCELLANEOUS

One of the most complex computer worm in recently times was Stuxnet. What makes Stuxnet so special and what lessons has the security world learned from that attack?

In some senses, Stuxnet is what we used to call a *media virus*: one of the *special* things about it is self-referential, derivative of its own celebrity. It's a gamechanger of sorts, but not the SCADA-mangling superbug you might think from some of the publicity it's received. It's interesting in many respects, though, not least in some of the innovative 0-days and implementational techniques we found in the binary (*http://go.eset.com/us/resources/white-papers/Stuxnet_Under_the_Microscope.pdf*). The real lesson is that the assumption that SCADA and ICS is somehow uniquely airgapped from all that Internet stuff doesn't hold: many utilities have found that they're actually made more vulnerable by the fact that you can't just take down a power station for a few hours while you apply patches. And now we have lots of researcher attention from both sides of the blackhat/whitehat divide into entirely different types of installation and hardware. That's a little scary, but at least there's an awareness that security isn't something that's only a problem on office desktops.

How would one go about Anti-malware testing in a cloud computing environment?

Actually, there are two main challenges here. One is testing where the products are installed in an environment that could be described as *in the cloud*: because there isn't even universal agreement on what is meant by that term, I'm not sure it's feasible to generate a *one size fits all* testing environment. The other challenge is to build a methodology that accurately

AMTSO has published a document (*http://www.amtso.org/amtso---download---amtso-best-practices-for-testing-in-the-cloud-security.html*) outlining some of the issues, and some of the 3rd-party papers at *http://www.amtso.org/related-resources.html* also address them.

Have you done any AV testing in the cloud? If so, can you describe some of the challenges you faced?

I haven't done direct testing myself for a long time. Small Blue-Green World doesn't have the resources (people or machines) that competent testing demands. I'm kept far too busy writing to devote the kind of intensive setup and maintenance time that most testing requires. Testing in the cloud is particularly demanding. On one hand you have to have a network context that is *natural* enough to allow both the malware and the security software to behave as they would in a real-life scenario: otherwise, the accuracy of the test is compromised because you may effectively prevent part of the AV from doing its job. On the other hand, you can't let the malware have unrestricted access back to the Internet, because you put systems outside your own lab at risk. The first part of the problem is to do with what we sometimes call *whole product testing*: it might be justifiable in some contexts to restrict functionality by taking one aspect of protective technology in isolation, but it's also likely to be seriously misleading in terms of performance evaluation.

Whats the latest and the most interesting project you were involved in?

Oddly enough, the research I've been most interested in recently had very little (direct) connection to malware. There's quite a lot of research on how people use and choose passwords, some of it derived from known collections of exposed passwords such as those exposed in the Gawker and Rockyou attacks (*http://www.scmagazineus.com/good-passwords-are-no-joke/article/204675/*), but there are virtually no equivalent data for PINs. In fact, the only such data I've seen was as a result of research carried out by Daniel Amitay resulting in a reasonably large data set of anonymized passcodes used in conjunction with an iPhone app. Though you can't assume that people will use the same passcodes and passcode selection strategies in other contexts, even on the same device, the data do allow some testing of some interesting initial hypotheses that were previously little more than *received wisdom*. Virus Bulletin (*http://www.virusbtn.com*) published an article in its September 2011 issue that outlines some of my preliminary conclusions, but I plan to cover that area in much more detail elsewhere in 2012.

Interview was prepared by Aby Rao and Hakin9 Team.

In the next issue of Hakin9 Extra magazine:

Botnets

Available to download on October 15th

Soon in Hakin9!

Online Anonymity, Social Network Security, Exploiting Software, Rootkits, Hacking Data, Security SQL Injection, Stuxnet, Port scanner, IP scanners, ISMS, Security Policy, Data Recovery, Data Protection Act, Single Sign On, Standards and Certificates, Biometrics, E-discovery, Identity Management, SSL Certificate, Data Loss Prevention, Sharepoint Security, Wordpress Security

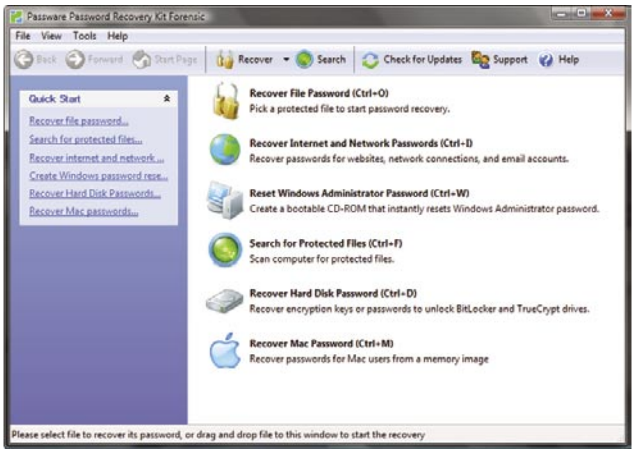
If you would like to contact Hakin9 team, just send an email to en@hakin9.org. We will reply a.s.a.p.

Passware Password Recovery Kit Forensic 11.0

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

Now with Mac User Password Recovery!

Passware Kit Forensic includes over 30 password recovery tools, Encryption Analyzer Professional, Search Index Examiner, FireWire Memory Imager, and a Portable Version to provide immediate password recovery for any protected file detected on a PC or over the network while scanning. It recovers or resets passwords for more than 200 different types of files, as well as decrypts hard drives, PGP archives, and unlocks Windows 7 and Mac OS Lion Administrator accounts. Many types of passwords are recovered or reset instantly.


Passware Password Recovery Kit Forensic
File View Tools Help
Back Forward Start Page Recover Search Check for Updates Support Help
Quick Start
Recover file password...
Search for protected files...
Recover internet and network...
Create Windows password rese...
Recover Hard Disk Passwords...
Recover Mac passwords...

Recover File Password (Ctrl+O)
Pick a protected file to start password recovery.

Recover Internet and Network Passwords (Ctrl+I)
Recover passwords for websites, network connections, and email accounts.

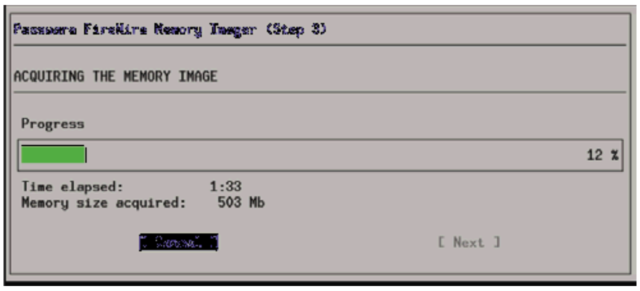
Reset Windows Administrator Password (Ctrl+W)
Create a bootable CD-ROM that instantly resets Windows Administrator password.

Search for Protected Files (Ctrl+F)
Scan computer for protected files.

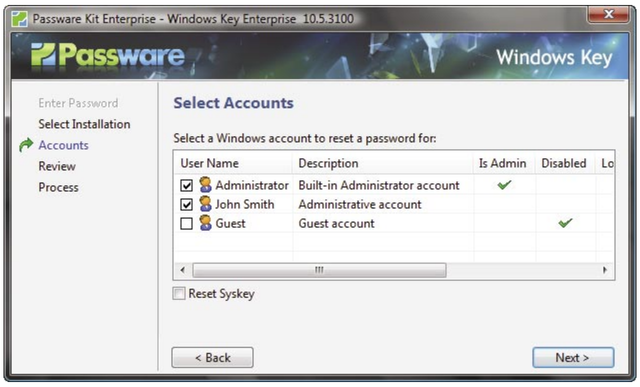
Recover Hard Disk Password (Ctrl+D)
Recover encryption keys or passwords to unlock BitLocker and TrueCrypt drives.

Recover Mac Password (Ctrl+M)
Recover passwords for Mac users from a memory image

Please select file to recover its password, or drag and drop file to this window to start the recovery



```
Passware FireWire Memory Imager (Step 3)

ACQUIRING THE MEMORY IMAGE

Progress
                                                12 %
Time elapsed:           1:33
Memory size acquired:    503 Mb

          Cancel                        [ Next ]
```



Select Accounts

Select a Windows account to reset a password for:

	User Name	Description	Is Admin	Disabled
☑	Administrator	Built-in Administrator account	✓	
☑	John Smith	Administrative account		
☐	Guest	Guest account		✓

Key Features

- Scans computers and network for password-protected files
- Recovers passwords for **200+ file types** Updated
- Unlocks hard drives protected with **BitLocker** and **TrueCrypt**
- Retrieves electronic evidence in a matter of minutes from a Windows Desktop Search Database
- Includes a **Portable Version** that runs from a USB thumb drive and recovers passwords without installation on a target PC
- Acquires memory images over FireWire Updated
- Recovers Mac user login passwords from computer memory New!

Advanced Features

- Instant recovery for many password types
- Acceleration with distributed computing **(Distributed Password Recovery)**
- Multiple-core CPUs and nVidia GPUs acceleration
- **Tableau TACC** hardware acceleration
- 8 different password recovery attacks (and any combination of them) with an easy-to-use setup wizard
- Detailed reports with MD5 hash values

“ *After losing my password to important encrypted documents, I thought it was the end of the world. Thanks for saving my work, Passware.*
Conor LaHiff, LaHiff & Company.

5 editions for consumers, small business, professional, corporate, and forensic users.
Starting from **$49!**

For additional information, please visit:
www.lostpassword.com/kit-forensic.htm

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

Contacts
Nataly Koukoushkina
media@lostpassword.com
Phone: +1 (650) 472-3716 x 101

30 DAYS MONEY BACK GUARANTEE

Copyright © 2011 Passware, Inc. All rights reserved. “Passware” and Passware logo are trademarks of Passware, Inc. All other names mentioned are trademarks, registered trademarks or service marks of their respective owners.

Hacker Halted USA 2011

Oct 21-27, 2011

Intercontinental Hotel. Miami, Florida

www.hackerhalted.com