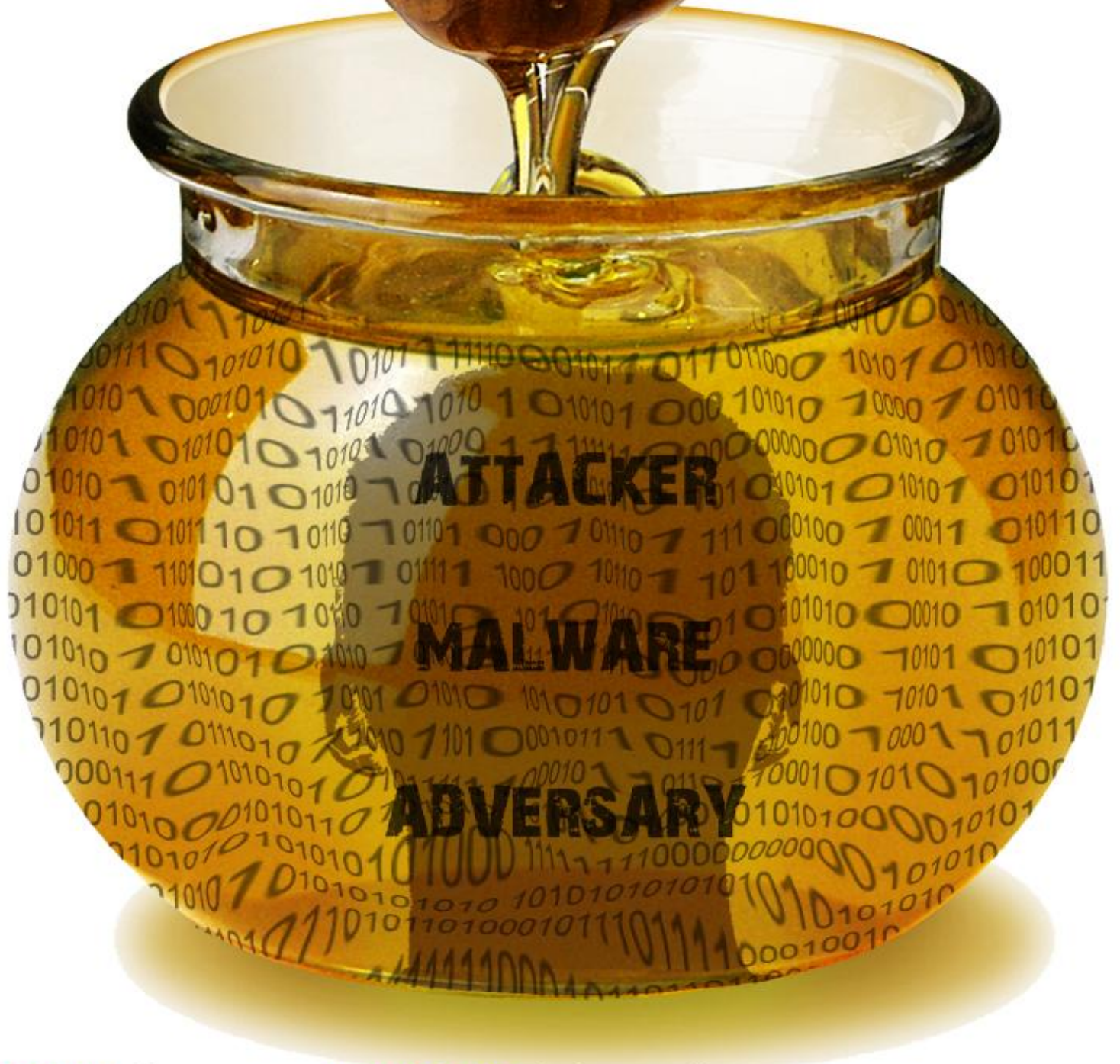


Club **HACK** Mag

Issue 6 | July 2010
www.chmag.in

1st Indian "HACKING" Magazine



TechGyan HoneyPot | **LegalGyan** Software Licenses |
ToolGyan HoneyD | **Mom's Guide** Protecting USB from malware

Hi Friends, I'm Varun Hirve, the moderator and pusher behind the magazine. My work is to push people and get articles in time, compile them & publish issues. Along with that, I sometimes write articles too :)



Varun Hirve

This issue we are dedicating to Indian Honeynet Project and our techGyan this time is about Honeypots. We'll also see how we can setup a simple honeypot at home and support the whole initiative

Along with Honeypot / Honeynet / HoneyD, we'll try to understand software licenses under our legalGyan section and we'll teach Mom's how to save their USB drives from malwares.

Over all, we are here with another issue of CHMag & we hope you are liking it. Please send us your articles and I'll see to it that quality work gets published in future issues. Happy Reading

ClubHACKMag

Issue 6, July 2010.

Team CHmag

Rohit Srivastwa
rohit@clubhack.com

Aarja Bhattacharyya
aarja@chmag.in

Abhijeet R Patil
abhijeet@chmag.in

Abhishek Nagar
abhishek@chmag.in

Deepranjan S More
deepranjan@chmag.in

Pankit Thakkar
pankit@chmag.in

Varun V Hirve
varun@chmag.in

www.chmag.in
info@chmag.in

CONTENTS

Pg 03 **TechGyan**
Honeypots

Pg 07 **ToolGyan**
HoneyD

Pg 10 **Mom'sGuide**
Protecting USB Drives
from Malwares

Pg 13 **LegalGyan**
Understanding
Software Licenses

Pg 17 **Command LineGyan**
A Splash on firewalls



Honeypot

Introduction

A Honeypot or a honey trap is an exciting technology with great potential for security community. It is an information system resource (a monitored decoy) used to attract attackers away from critical resources as well as a tool to analyse an attacker's method and characteristics. Actually, it is a trap set to detect attempts at unauthorized use of information system. The value of a Honeypot lies in unauthorized and illicit use of that resource. Neither any authorised activity runs on these resources nor do they have any production value. In short it has no legitimate activity. Unlike firewalls or Intrusion Detection Systems, Honeypot do not solve a specific problem. Instead, they are a highly flexible tool that comes in many shapes and sizes.

A Honeypot can detect attacks by capturing polymorphic code, capturing a variety of attacks, working with encrypted data and acquiring signatures. It provides a large amount of valuable information for analysis. It is a valuable surveillance and early-warning tool, but it can carry risks to a network, and must be handled with care. It requires a high level of network administration and understanding of protocol and security.

Two or more Honeypot on a network form a HoneyNet. Generally a HoneyNet is used for monitoring a large and/or diverse network in which one Honeypot may not be sufficient.

Modes of Honeypot

Research mode: In this mode the Honeypot characterizes attack environment by collecting data on attacker motivations, attack trends and emerging threats.

Production mode: The Honeypot is used to prevent, detect and respond to attacks. Prevention is accomplished through deterrence i.e. by impeding scans initiated by attackers and diverting an attacker to interact with the Honeypot rather than critical files.

Types of Honeypot:-

Low-Interaction Honeypot: As the name suggests it has limited interaction and it provides a minimal emulation of an operating system running on the target computer. It is easy to deploy because of its limited operating system emulation and ease of maintenance. Generally it involves installing software, selecting the operating system and the services we want to emulate and monitor.

The simplicity of a Low-Interaction Honeypot is its main advantage, but it logs only limited information and is designed to capture known activity, which makes it easier to get detected by a skilled attacker.

High-Interaction Honeypot: This type of Honeypot provides much more realistic target for an attacker as it involves real operating systems and applications. It's not emulation of operating system and services. Its main advantages are that unlike Low-Interaction Honeypot the acquired information is much more detailed and it provides an open environment that captures all activities. However it is more vulnerable to compromise and possibly use it as a platform to launch an attack on the critical information system resource.

Value of Honeypot

Honeypot can help prevent attacks in several ways.

The first is against automated attacks, (such as worms) which are based on tools that randomly scan entire networks looking for vulnerable systems. If vulnerable systems are found, these automated tools will then attack and take over the system. One way that Honeypot can help defend against such attacks is slowing their scanning down. These solutions monitor unused IP space and are called sticky Honeypot. When probed by such scanning activity, these Honeypot interact and slows the attacker down. This is excellent for slowing down or preventing the spread of a worm that has penetrated our internal organization. Sticky Honeypot are most often low-interactive solutions.

Honeypot can also be protecting our organization from human attackers through deception. The Honeypot confuses an attacker, to make him waste his time and resources interacting with Honeypot and not with the original production system. Meanwhile, the detection of the attacker's activity can be done and we have the time to respond and stop the attacker.

The second way Honeypot can help protect an organization is through detection. The purpose of detection is to identify a breakdown. Even if an organisation is super secure, a failure can always be there because no security system is 100% safe. By detecting an attacker, we can quickly react to them, stopping or mitigating the damage they do. Honeypot excel at detection, addressing many of the problems of traditional detection. Honeypot reduce false positives by capturing small data sets of high value, capture unknown attacks such as

new exploits or polymorphic shell code, and work in encrypted and IPv6 environments. In general, low-interaction Honeypot make the best solutions for detection. They are easier to deploy and maintain, than high-interaction Honeypot and have reduced risk.

The third and final way a Honeypot can help protect an organization is in response. One of the greatest challenge organisations face today is how to response to an attack. There is often little information regarding the attacker(s), how they got in, or how much damage they have done. In these situations detailed information on the attacker's activity are critical. The main problem to attack response is that often the compromised system is a production system and is running essential services so it's difficult to take it offline. Even if the system is taken offline, the logs and data entries are so much that it can be difficult to determine what normal day-to-day activity is, and what the attacker activity is.

Honeypots can help address both problems. Honeypots make an excellent incident response tool, as they can quickly and easily be taken offline for a full forensic analysis, without impacting day-to-day production operations. Also, the only activity a Honeypot captures is unauthorized or malicious activity (as already mentioned). This makes hacked Honeypots much easier to analyse than hacked production systems, as any data we retrieve from a Honeypot is most likely related to the attacker. The value Honeypots provide here is quickly giving organizations the in-depth information they need to respond to an attack effectively. Generally high-interaction Honeypots make the best solution for response.

Advantages:

Relevant data set: Honeypots collect small amount of data but almost all of this data is real attack or unauthorized activity.

Reduced false positive: With most detection technologies (IDS, IPS) a large percentage of alerts are false warnings, while with Honeypots this is not the case.

False negatives: It's easy for Honeypots to detect and record attacks which were not seen before.

Cost effective: Honeypot only interacts with malicious activity and do not require high performance resource.

Simplicity: Honeypots are very simple to understand, deploy and maintain.

Disadvantages:

Limited view: Honeypots only see activities that interact with them and do not capture attack, directed against other existing systems.

Risk of being compromised: A Honeypot may be used as a platform to launch further attacks.

Because of these disadvantages, Honeypots do not replace any security mechanisms; they only work with and enhance your overall security

Risk with Honeypots

Exploiting Honeypots

- Human intervention.

- It is expected for attackers to gain privileged control of the Honeypots.
- Step stone to harm other systems.

Identifying Honeypots

- Eliminate fingerprinting.
- Feed Honeypot false or bogus information.
- Chess problem!

Conclusion:

- Honeypots are good resource to track Hackers.
- The value of Honeypots is in being hacked.



Sudhanshu Chauhan.
sudhanshuchauhan007@yahoo.com

Sudhanshu is a B.Tech (CSE) III year student from Amity University, Noida, U.P. Sudhanshu is interested in the field of Cyber Security.



Honeyd: Track the hackers

Introduction

Honeyd is a low interaction Honeypot client that creates virtual hosts (Honeyd) in a network. These Honeyd can be configured to act like a real operating system, in fact there are approximately 1000 personalities of OS's that we can choose. At the same time we can configure those operating systems to activate certain services like FTP, HTTP, Telnet, etc.

Honeyd enables a single host to claim multiple addresses.

Ever wanted to be like those secret agents in Hollywood movies who work with cool looking computers, flashy programs, with sexy assistants and catches the bad guys? Here is your once in a life time chance to be like one. Ok enough jokes, lets come back to reality Honeyd or honey nets are the tools/ systems used by security researchers around the globe to track malicious activity.

In computer terminology, a **Honeyd** is a trap set to detect, deflect, or in some Manner counteracts attempts to

unauthorized use of information systems. Honeyd is a non-interactive, easy to deploy Honeyd. It is a small daemon that creates virtual hosts on a network. The hosts can be configured to run arbitrary services, and their personality can be adapted so that they appear to be running certain operating systems. Honeyd enables a single host to claim multiple addresses.

Installation

Honeyd can be downloaded from <http://www.honeyd.org> and is available for both Windows (Cygwin) and Linux. I will be explaining about installation and configuration in Linux (to be specific in Ubuntu); windows installation follows a similar path. Download the latest release from the website

Before installing the Honeyd, you need to install following libraries

libevent, libdnet, libpcap, libpcrc make sure that you are installing the development packages of the above also.

Once all the dependencies are met, the installation is trivial.

Unzip the tar file and then.

```
./configure
Make
sudo make install
```

Once installation is over Honeyd is ready to play with. Let's run our first Honeyd.

Download the example configuration file `config.localhost` from <http://www.honeyd.org/config/config.localhost> and run the command

```
$honeyd -d -p nmap.prints -f
config.localhost -i lo
```

[For the time being use the `nmap.prints` which came along with the Honeyd package, later you can update it to the latest version.]

In the command you just run, “-d” will ask it not to run as a daemon, so that we can see the log information directly printed to the console and will be able to interact with it. “-p” will specify the *nmap style finger print* used for simulating OS characteristics “-i” will specify the interface used and the IP range to listen, which in our case is the local loop, and “-f” will specify the configuration file which has the structure of our Honeyd.

Configuration Files

Each of the virtual Honeyd you create inside the configuration file is called as a template. A single configuration can have multiple templates (i.e. multiple systems).

An example template configuration taken from `config.localhost`

```
-----
create routerone #create the
template(virtual system) with
name routerone
Set routerone personality
"Cisco 7206 running IOS
11.1(24) "
```

```
#assigns the personality (OS
nature) as CISCO IOS
Set routerone default tcp
action reset
#set the default TCP action
of the system routerone as
RESET
add routerone tcp port 23
"scripts/router-telnet.pl"
#add TCP port 23 to the
system routerone and give
that port the characteristic
defined by the given script
file, here it will emulate a
telnet session
bind 10.0.0.1 routerone
#bind the system routerone
to the particular ip address
-----
```

Honeyd is a really versatile tool that can be used to simulate complex network topologies; if you are interested you can read more on the site.

Experiment

Now that you have a basic understanding about the configuration, let's continue with the experiment.

Once you have started the Honeyd, it will listen on the “lo” interface looking for packets matching the configured hosts. It's our job to make sure that packets are reaching the particular interface.

So now add a route to 10.0.0.0/8 in the “lo”

```
$sudo route -n add -net
10.0.0.0 netmask 255.0.0.0 gw
```

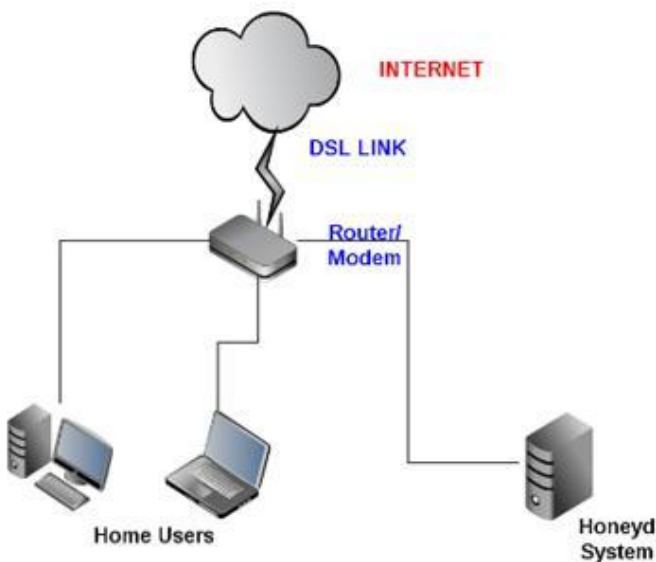
Now try pinging a host we have specified in the config file (eg.10.0.0.1), voila!! You will get ICMP response from that IP. Also if you

But if you are planning to run it to reach the hackers, you need to run it in the daemon mode and store the traces. For that run the command without “-d”, “-l” option will let you log the packet level details and “-s” will let you log at the service level.

For e.g.

```
$sudo honeyd -l log_pkt.log -
s log_srvc.log -p
nmap.prints -f
```

Now you're ready for actual Honeyd deployment. It's advisable to deploy honeyd on a separate system, if you are planning to put it in the wild web. A typical deployment is shown in the figure. Configure the router to forward ports you are planning to monitor to your honeyd PC. Now wait like a spider waiting for the prey [;)], soon enough an attacker will come your way. Happy honey potting



create and share your own diagrams at gliffy.com

gliffy



Rakesh Mukundan

<http://simplified-security.blogspot.com/>

Rakesh has completed his B Tech from NIT Calicut in ECE, now he is working in the field of wireless testing with a leading telecom MNC. Rakesh has fairly good amount of experience in PT, and RCE and is an active member of [Indian Honeydnet project](#). His programming expertise includes PHP, MySQL, Perl, C, Verilog, HTML/CSS and AJAX.



Protecting USB Drives from Malwares

Introduction

The emergence of USB drives has become a blessing for the digital world. It has become very easy to carry and transfer data using USB drives. Unfortunately with the increase in use of USB drives, the nuisance of malwares has also increased.

USB drives are being used by malwares to spread from machine to machine.

The Problem

Malwares use two main techniques to spread through USB drives.

1. Infecting executable files on memory drives so that when they are run on another machine, the infections move with them.
2. This technique uses '**autorun.inf**' file to spread the malware.

The second technique is more dangerous. Because as soon as the USB drive is plugged in, 'autorun.inf' file is automatically executed by Windows operating system which needs no human interaction. Most of the malicious programs use this technique. We can prevent this infection by disabling '**autorun**' feature in Windows. But it requires the client machine to do this, which is not always the case, as most users will not have the technical knowledge to do this.

The solution

We can purchase USB drives with read-only switches. But it has its own disadvantages, like in order to write on to the memory sticks we will have to remove the protection and thus putting USB drive to risk of being infected.

We can solve this problem in very simple and free way and without having to buy memory sticks with read only switches!

So before we start, take the backup of the data on USB drive and make sure it is blank.

Here are the steps to make your USB drives malware safe:-

1. We create a blank '**autorun.inf**' file on the USB drive.
2. Now use a hex editor to open the USB device in read and write mode. Make sure that nothing is accessing the device at that time.
3. In the disk, search for the string – '**AUTORUN**' in a non-Unicode format. You will find it near the beginning of the disk.

This is what we are interested in:-

```
41 55 54 4F 52 55 4E 20 49 4E 46 20
```

```
A U T O R U N   I N F
```

4. The current value of the byte **0x20** has just the archive bit set. We change this bite to **0x40**. This sets the device bit, which is never normally found on disk.

In simple words, what you have to do is just replace '**2**' of **0x20** with **4**, this will make it **0x40**.

The edited block should look like this:-

```
0003DFF0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
0003E000 55 53 42 20 20 20 20 20 20 20 08 00 00 00 USB .....
0003E010 00 00 00 00 00 00 E1 70 55 3C 00 00 00 00 .....épU<.....
0003E020 E5 6D 00 65 00 6E 00 74 00 2E 00 0F 00 9F 74 00 àm.e.n.t....ÿt.
0003E030 78 00 74 00 00 00 FF FF FF FF 00 00 FF FF FF FF x.t...ÿÿÿÿ..ÿÿÿÿ
0003E040 E5 4E 00 65 00 77 00 20 00 54 00 0F 00 9F 65 00 àN.e.w. .T...ÿe.
0003E050 78 00 74 00 20 00 44 00 6F 00 00 00 63 00 75 00 x.t. .D.o...c.u.
0003E060 E5 45 57 54 45 58 7E 31 54 58 54 20 00 8C E8 70 àEWTEX~1TXT .@ép
0003E070 55 3C 55 3C 00 00 E9 70 55 3C 00 00 00 00 00 00 U<U<..épU<.....
0003E080 41 55 54 4F 52 55 4E 20 49 4E 46 40 18 8C E8 70 AUTORUN INF @ép
0003E090 55 3C 55 3C 00 00 E9 70 55 3C 00 00 00 00 00 00 U<U<..épU<.....
0003EOA0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
0003EOB0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
```

```
41 55 54 4F 52 55 4E 20 49 4E 46 40
```

```
A U T O R U N   I N F @
```

5. Save this to the disk, ignoring all the warnings that might appear.

6. Unmount and remount the device. To test if our autorun.inf file is protected or not, try deleting the autorun.inf file. You will get the following popup with an error.



As you will observe, you cannot open, edit, delete or overwrite it. Also its attributes cannot be changed.

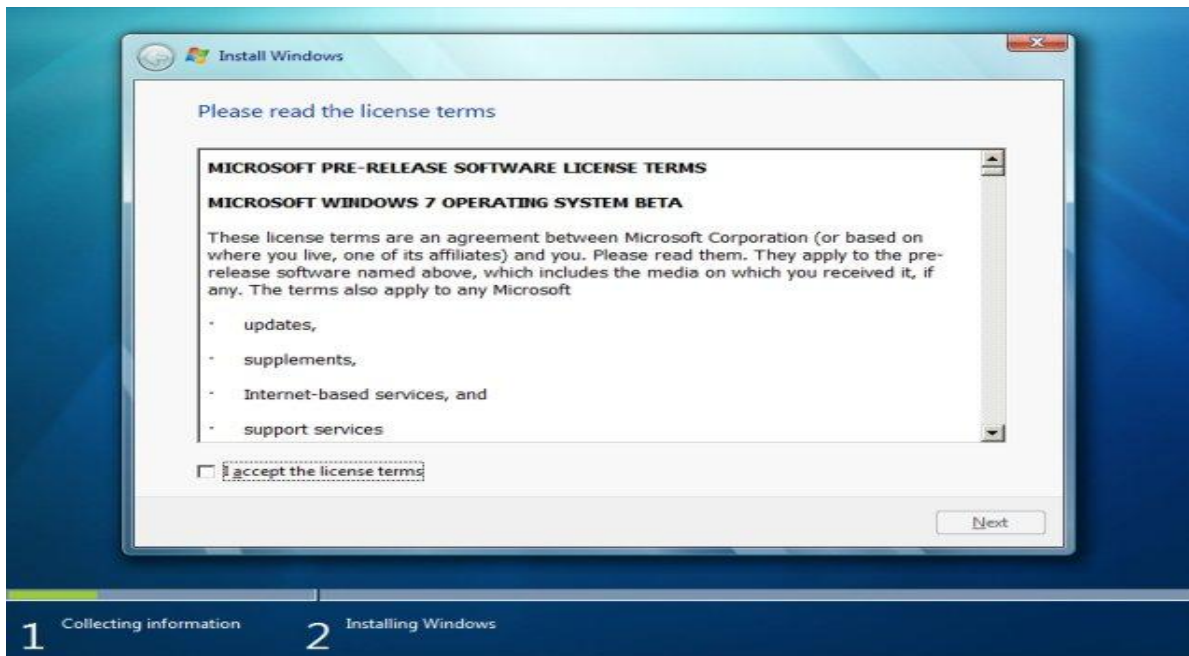
So now feel free to use this protected USB device on any machine.

References:-

<http://www.milworm.com/papers/314>



Abhijeet Patil
abhijeet@chmag.in



Understanding Software License

About License

Licenses are far more prevalent in the “intangible” world as compared to the “real” world. Suppose you buy a car. Once you pay the price of the car to the dealer, you can do almost anything with the car. You can sell it, rent it, make modifications to it and even destroy it!

The situation is not the same when you “buy” software. In fact you hardly ever “buy” software. You buy a “license” to the software. This license sets the terms and conditions subject to which you can use (and sometimes distribute and modify) the software.

A software license usually consists of:

1. **Permissions** granted by the creator to the user,
2. **Rights** granted by the creator to the user, and
3. **Restrictions** placed by the creator,
4. limitations on the creator’s **liability**,
5. **Warranties** and warranty disclaimers, and
6. **Indemnity**,
7. **Term** / duration of the license.

L Violating the terms of the license infringes the legal rights of the creator of the software and can lead to **legal action**.

When software is **mass produced** and sold, the license is usually of a “**take-it-or-leave-it**” type. The software vendor does not give the purchaser an opportunity to negotiate the terms of the license. The purchaser can either accept the license in total and purchase it or reject the license and not purchase it. In **customized**

software that is usually specially developed for a particular customer, the terms of the license are negotiated between the creator and the user.

Software licenses can be of various types such as:

1. **Time-based licenses** where the license expires after a particular time period (e.g. 1 year). The license has to be renewed after that (usually on payment of additional fees).

Illustration 1

AVG is popular anti-virus software. Licenses can be purchased for 2 years at a time. During these two years all updates will be available to the user. On expiry of the 2 year period, the license expires and the user must uninstall the software. If the user wants to continue using the software, he must again pay fees and buy a license for a further 2 years.

Illustration 2

Winhex is popular cyber forensics software. It can be purchased with a 1 year upgrade license. Suppose Sameer purchases this license on 1st January 2008. Till 31st December 2008, he will be eligible to get all upgrades and new versions of Winhex for free. After 1 year this upgrade license will expire. Although Sameer can continue using Winhex, he will not get any further updates for free.

2. **User-based licenses** where the license fee depends upon the number of computers on which the software will be installed (e.g. in case of application software like a word processor). It could also depend upon the number of users who will connect simultaneously to a computer on which the software is

installed (e.g. in case of database software or server operating systems).

3. **Feature-based licenses** where the license fee depends upon the features that are required by the user.

Illustration

Winhex is a popular cyber forensics software. Depending on the features that are required, users can purchase the personal license, professional license, specialist license, or forensic license.

Most software licenses also contain clauses relating to disclaimer of warranties, limitation of liability, privacy policy etc. In this book we will focus on software licenses from an intellectual property rights perspective only.

Windows License

Considering the huge number of users worldwide who use the Microsoft Windows operating system, the Windows End-User License Agreement ("EULA") is discussed here.

This section discusses some of the terms of the EULA for **Microsoft Windows XP Professional Edition**. It may be noted that the EULA for most Microsoft products is similar.

Basic information

The End-User License Agreement ("EULA") is a legal agreement between Microsoft Corporation and the end user. It extends to the Microsoft software accompanying the EULA as well as associated media, printed materials, "online" or electronic documentation, and Internet-based services (collectively known as "Software").

A user is bound by the terms of the EULA by installing, copying or otherwise using the Software. If a user does not agree to be bound by the EULA, he must not install copy or use the software.

The EULA applies to updates, supplements, add-on components, product support services, or Internet-based services components, of the Software obtained from Microsoft after the date of obtaining the initial copy of the Software.

To use Software identified as an upgrade, the user must first be licensed for the software identified by Microsoft as eligible for the upgrade. After upgrading, the user may no longer use the software that formed the basis for the upgrade eligibility.

Grant of license

Rights granted by the EULA are

1. Installation and use

The user can install, use, access, display and run one copy of the Software on a single computer, such as a workstation, terminal or other device ("Workstation Computer"). The Software cannot be used by more than two (2) processors at any one time on any single Workstation Computer.

2. Mandatory Activation

The license rights granted under this EULA are limited to the first thirty (30) days after first installation of the Software unless the user activates the licensed copy through the Internet or telephone. Reactivation is needed if the computer hardware is altered.

3. Device Connections

The EULA permits a maximum of 10 computers or other electronic devices (each a "Device") to connect to the Workstation

Computer to utilize one or more of the following services of the Software: File Services, Print Services, Internet Information Services, Internet Connection Sharing and telephony services.

4. Remote Desktop / Remote Assistance / NetMeeting

Remote Desktop access requires a separate Software license. As an exception, the single primary user of the Workstation Computer may access a Workstation Computer Session from any Device without acquiring an additional Software license for that Device.

5. Storage / Network Use

The EULA permits storage or installation of a copy of the Software on a storage device, such as a network server, used only to install or run the Software on other Workstation Computers over an internal network. An additional license is required for each separate Workstation Computer on or from which the Software is installed, used, accessed, displayed or run. A license for the Software may not be shared or used concurrently on different Workstation Computers.

Reservations of rights and ownership

Microsoft reserves all rights not expressly granted in the EULA. The EULA stresses that the Software is protected by copyright and other intellectual property laws and treaties and that Microsoft or its suppliers own the title, copyright, and other intellectual property rights in the Software. The EULA also expressly states that **the Software is licensed, not sold.**

Reverse Engineering

The EULA prohibits reverse engineering, decompilation and disassembly of the

software unless such activities are permitted by the law.

Exception

Section 52(ab) of the Copyright Act permits this for specific purposes.

Rental / commercial hosting

The EULA prohibits renting, leasing, lending or providing commercial hosting services with the Software.

End user proof of license

A genuine Microsoft "Proof of License" label with a genuine copy of the software identifies a licensed copy of the Software. To be valid, the label must appear on Microsoft software packaging. If the label is received separately, it is invalid. The packaging should be kept as it has the label on it to prove that the user is licensed to use the Software.

Software Transfer

The Software can be transferred to a different Workstation Computer. After the transfer, the Software must be completely removed from the former Workstation Computer.

The initial user of the Software may make a one-time permanent transfer of this EULA and Software to another end user, provided the initial user retains no copies of the Software.

This transfer must include the Software and the Proof of License label. The transfer may not be an indirect transfer, such as a consignment. Prior to the transfer, the end user receiving the Software must agree to all the EULA terms.

Termination

Microsoft can terminate the EULA if the user fails to comply with the terms and conditions of the EULA. In such event, the user must destroy all copies of the Software and all of its component parts.

Applicable Law

If the Software has been acquired in USA, the EULA is governed by the laws of the State of Washington.

If the Software has been acquired in Canada, the EULA is governed by the laws of the Province of Ontario, Canada.

If the Software has been acquired in the European Union, Iceland, Norway, or Switzerland, then local law applies.

If the Software has been acquired in any other country, then local law may apply.



Rohas Nagpal
rn@asianlaws.org

Command LINE

```

c:\tools\c.exe - cmd
Port configuration for LAN:
Port Protocol Mode Name
-----
53 UDP Enable DNS
68 UDP Enable DHCP <68>
67 UDP Enable DHCP <67>
2869 TCP Enable UPnP <TCP>
1900 UDP Enable UPnP <UDP>

VirtualBox Host-Only Network firewall configuration:
Operational mode = Enable

CDMA_1X firewall configuration:
Operational mode = Enable

High Speed Data firewall configuration:
Operational mode = Enable

TATA firewall configuration:
Operational mode = Enable

```

A splash on firewalls

I'm sure you all know that almost all operating system have a kind of inbuilt firewall. Windows has "Windows firewall" whereas the Linux has "iptables".

In this issue of command line gyan, we'll see how can we play around & see allowed/disallowed activities via these firewalls

Windows

On a windows box if we want to see all the ports which are allowed through, all we need to do is type following command

```
C:\> netsh firewall show
```

To see the applications allowed communicating through firewall

```
C:\> netsh firewall show
```

To view all the configuration of windows firewall, try

```
C:\> netsh firewall show
```

```

c:\tools\c.exe - cmd
C:\>netsh firewall show portopening
Port configuration for Domain profile:
Port Protocol Mode Name
-----
26675 TCP Enable ActiveSync Service
Port configuration for Standard profile:
Port Protocol Mode Name
-----
26675 TCP Enable ActiveSync Service
Port configuration for LAN:
Port Protocol Mode Name
-----
53 UDP Enable DNS
68 UDP Enable DHCP <68>
67 UDP Enable DHCP <67>
2869 TCP Enable UPnP <TCP>
1900 UDP Enable UPnP <UDP>

```

This NETSH is really cool & we'll surely cover the whole NETSH sometime soon.

Linux

Ok! as we have always seen Linux is much easier on command line, let's see how we can achieve the same outputs on Linux.

To see a complete config, the easiest way is

```
#iptables -L
```

To make it little fast (and avoid reverse lookup)

```
#iptables -nL
```

```
root@localhost:~# iptables -nL
Chain INPUT (policy ACCEPT)
target prot opt source destination
RR-Firewall-1-INPUT all -- 0.0.0.0/0 0.0.0.0/0

Chain FORWARD (policy ACCEPT)
target prot opt source destination
RR-Firewall-1-INPUT all -- 0.0.0.0/0 0.0.0.0/0

Chain OUTPUT (policy ACCEPT)
target prot opt source destination

Chain RR-Firewall-1-INPUT (2 references)
target prot opt source destination
ACCEPT all -- 0.0.0.0/0 0.0.0.0/0
ACCEPT icmp -- 0.0.0.0/0 0.0.0.0/0 icmp type 255
ACCEPT esp -- 0.0.0.0/0 0.0.0.0/0
ACCEPT ah -- 0.0.0.0/0 0.0.0.0/0
ACCEPT udp -- 0.0.0.0/0 224.0.0.251 udp dpt:5353
ACCEPT udp -- 0.0.0.0/0 0.0.0.0/0 udp dpt:631
ACCEPT tcp -- 0.0.0.0/0 0.0.0.0/0 tcp dpt:631
ACCEPT all -- 0.0.0.0/0 0.0.0.0/0 state RELATED,ESTABLISHED
ACCEPT tcp -- 0.0.0.0/0 0.0.0.0/0 state NEW tcp dpt:25
ACCEPT tcp -- 0.0.0.0/0 0.0.0.0/0 state NEW tcp dpt:22
ACCEPT tcp -- 0.0.0.0/0 0.0.0.0/0 state NEW tcp dpt:443
ACCEPT tcp -- 0.0.0.0/0 0.0.0.0/0 state NEW tcp dpt:80
REJECT all -- 0.0.0.0/0 0.0.0.0/0 reject-with icmp-host-prohibited
root@localhost ~#
```

-n here will stop the command to do a reverse lookup & hence increasing the performance.

To make it more specific

```
# iptables -t nat -nL
# iptables -t mangle -nL
# iptables -t filter -nL
# iptables -t raw -nL
```

These will list (-L) the specific (-t) chain without reverse lookup (n).

More in future issue.

Happy finding your firewall holes ☺



Rohit Srivastwa
rohit@clubhack.com

Honeypots are good way to catch attackers.

Have you deployed one ?



Be a part of Indian HoneyNet Project

<http://honeynet.org.in>

www.clubhack.com