

# Ubuntu Instance Hardening Guide

(for Ubuntu 12.04 LTS)

**v1.0**



'Ubuntu', the Ubuntu Logo and 'Canonical' are registered trademarks of Canonical. Read Canonical's trademark policy [here](#).

CSS, CSS Corp., and the CSS Corp logos are registered trademarks of CSS Corp. Pvt. Ltd. All other trademarks mentioned in the book belong to their respective owners.

This book is aimed at providing an overview of how public and private cloud instances running Ubuntu can be hardened. This book will be updated periodically based on the suggestions, ideas, corrections, etc., from readers.

# Preface

## Introduction

As more and more corporate infrastructure tends to depend on cloud these days and with the dependence ever increasing, the need for a guide to Ubuntu instance security arose. This document attempts to address that need from a Ubuntu Instance Administrator's perspective.

Security should always be considered when installing, deploying, and using any type of cloud Instance. Although a freshly started Ubuntu Cloud Instance is relatively safe for immediate use on the Internet, it is important to have a balanced understanding of your systems security posture based on how it will be used after deployment.

This document provides an overview of security related topics as pertaining to Ubuntu 12.04 Server Edition, and outlines simple measures you may use to protect your Ubuntu Cloud Instance and network from potential security threats.

## Scope

This document originally evolved from a necessity to harden applications running on Ubuntu instances in the public cloud and from an enlightenment that application security will not be effective nor complete without first focusing on instance hardening.

It is a common agreement in the field of Information System Security (ISS) that Total security is weaker than the weakest component in the system. The point is that there are many points of attack against a complex system, because attackers are not limited to attacking one link; rather, they may attack wherever they please and at multiple points hence selling application security on the cloud will be like selling snake oil without inherently hardening the underlying cloud instance.

## Target Audience

Our aim is to provide a guide to people who want to get started on hardening Ubuntu servers and Ubuntu Cloud Instances. Basic knowledge of System Administration in Ubuntu is expected and familiarity with how system security works on Ubuntu will help though not mandatory.

# Presentation Style

This book is created with an aim to keep it as compatible with existing Ubuntu server hardening techniques, hence some overlap with existing Ubuntu server hardening techniques can be noticed in the initial chapters of this book.

The book starts off with very basic Ubuntu hardening tips and then moves on to advanced topics in instance hardening and we have also explored system and network vulnerability analysis methods so as to help the reader develop expertise in a phased manner as they progress through this book.

# Acknowledgements

Most of the content has been borrowed from web resources like manuals, documentation, white papers and numerous posts on forums; discussions on the IRC Channel and many articles on the web including those of our colleagues at CSS Corp. We would like to thank the authors of all these resources.

# License

Attribution-Noncommercial-Share Alike 3.0 Unported. For the full version of the license text, please refer to <http://creativecommons.org/licenses/by-nc-sa/3.0/legalcode> and <http://creativecommons.org/licenses/by-nc-sa/3.0> for a shorter description.

# Contents

1. User Management.....	6
1.1. Root Account	
1.2. Adding and Deleting Users	
1.3. User Profile Security	
1.4. Password Policy	
1.4.1 Minimum Password Length	
1.4.2 Password Expiration	
1.4.3 Strong Passwords	
a) apg	
b) pwgen	
1.5 Other Security Considerations	
1.5.1. SSH Access by Disabled Users	
1.5.2. External User Database Authentication	
2. Firewall.....	14
2.1 Introduction	
2.2 ufw-Uncomplicated Firewall	
2.2.1	
2.3 IP Masquerading	
3. Apparmor.....	22
4. Public Key Cryptography.....	27
5. Ecryptfs.....	33
6. Automatic Security updates.....	35
7. Shared Memory.....	37
8. Single Packet Authorization.....	38
9. Intrusion Detection Systems.....	44
10. Rootkit detectors.....	47
11. Linux Kernel Hardening.....	53
12. Vulnerability Scanners.....	63
13. Security Analysis Tools.....	74
14. OpenStack Security Groups.....	81

# Chapter 1

## User Management

Any secure cloud instance requires effective user management. Ineffective user management is often the cause for cloud instance compromises. It is vital to understand how to develop simple yet effective user account management techniques to protect your Ubuntu instances.

### 1.1. Root account

The root account has been disabled by default on Ubuntu. Though the root account has not been deleted, it has been disabled by providing it a password which matches no possible encrypted value thus preventing a user from directly logging in as the root user.

Ubuntu uses the sudo tool to carry out system maintenance duties. sudo allows a permitted user to execute a command as the superuser or another user, as specified by the security policy. The real and effective uid and gid are set to match those of the target user, as specified in the password database.

sudo supports a plugin architecture for security policies and input/output logging. Third parties can develop and distribute their own policy and I/O logging modules to work seamlessly with the sudo front end. The default security policy is sudoers, which is configured via the file /etc/sudoers, or via LDAP.

This simple yet effective methodology of sudo provides accountability for all user actions, and gives the administrator granular control over which actions a user can perform with the said privileges.

- If for some reason you wish to enable the root account, simply give it a password:

sudo passwd

Sudo will prompt you for your password, and then ask you to supply a new password for root as shown below:

- To disable the root account, use the following passwd syntax:

```
sudo passwd -l root
```

By default, the initial user created by the Ubuntu installer is a member of the group "admin" which is added to the file /etc/sudoers as an authorized sudo user. If you wish to give any other account full root access through sudo, simply add them to the admin group.

The /etc/sudoers file can also be used to provide

## 1.2 Adding and deleting users

The 'adduser' package is used in Debian and Ubuntu based distributions for account management.

- To add a user account, use the following syntax, and follow the prompts to give the account a password and identifiable characteristics such as a full name, phone number, etc.

```
sudo adduser username
```

- To delete a user account and its primary group, use the following syntax:

```
sudo deluser username
```

Deleting an account does not remove their respective home folder. It is up to the superuser to keep the folder or delete it according to the data retention policies of your organization.

A new user added later on with the same UID/GID as the previous owner will now have access to this folder if you have not taken the necessary precautions.

You may want to change these UID/GID values to something more appropriate, such as the root account, and perhaps even relocate the folder to avoid future conflicts:

```
sudo chown -R root:root /home/username/  
sudo mkdir /home/archived_users/  
sudo mv /home/username /home/archived_users/
```

- To temporarily lock or unlock a user account, use the following syntax, respectively:

```
sudo passwd -l username  
sudo passwd -u username
```

- To add or delete a personalized group, use the following syntax, respectively:

```
sudo addgroup groupname  
sudo delgroup groupname
```

## 1.3 User Profile Security

When a new user is created, the `adduser` utility creates a brand new home directory named `/home/username`, respectively. The default profile is modeled after the contents found in the directory of `/etc/skel`, which includes all profile basics.

If your server will be home to multiple users, you should pay close attention to the user home directory permissions to ensure confidentiality. By default, user home directories in Ubuntu are created with world read/execute permissions. This means that all users can browse and access the contents of other users home directories. This may not be suitable for your environment.

- To verify your current users home directory permissions, use the following syntax:

```
ls -ld /home/username
```

The following output shows that the directory `/home/username` has world readable permissions:

```
drwxr-xr-x  2  username  username  4096  2007-10-02  20:03  username
```

- You can remove the world readable permissions using the following syntax:

```
sudo chmod 0750 /home/username
```

Note: Some people tend to use the recursive option (`-R`) indiscriminately which modifies all child folders and files, but this is not necessary, and may yield other undesirable results. The parent directory alone is sufficient for preventing unauthorized access to anything below the parent.

A much more efficient approach to the matter would be to modify the `adduser` global default permissions when creating user home folders. Simply edit the file `/etc/adduser.conf` and modify the `DIR_MODE` variable to something appropriate, so that all new home directories will receive the correct permissions.

```
DIR_MODE=0750
```

- After correcting the directory permissions using any of the previously mentioned techniques, verify the results using the following syntax:

```
ls -ld /home/username
```

The results below show that world readable permissions have been removed:

```
drwxr-x---  2  username  username  4096  2007-10-02  20:03  username
```



## 1.4 Password Policy

A strong password policy is an inevitable part of your overall corporate security policy. A weak password can be an easy target for dictionary or brute force attacks. Minimum password complexity requirements and setting proper maximum password lifetimes as well as frequent audits of your authentication systems is a must.

### 1.4.1. Minimum Password Length

By default, Ubuntu requires a minimum password length of 6 characters, as well as some basic entropy checks. These values are controlled in the file `/etc/pam.d/common-password`, which is outlined below.

```
password [success=2 default=ignore] pam_unix.so obscure sha512
```

If you would like to adjust the minimum length to 8 characters, change the appropriate variable to `min=8`. The modification is outlined below.

```
password [success=2 default=ignore] pam_unix.so obscure sha512 min=8
```

### 1.4.2. Password Expiration

When creating user accounts, you should make it a policy to have a minimum and maximum password age forcing users to change their passwords when it expires.

- To view the current status of a user account, use the command:

```
sudo chage -l username
```

The output below shows interesting facts about the user account, namely that there are no policies applied:

```
root@shire:~# sudo chage -l frodo
Last password change           : Mar 03, 2006
Password expires               : never
Password inactive              : never
Account expires                : never
Minimum number of days between password change : 0
Maximum number of days between password change : 99999
Number of days of warning before password expires : 7
root@shire:~#
```

- To set any of these values, simply use the following syntax, and follow the interactive prompts:

```
sudo chage username
```

The following is also an example of how you can manually change the explicit expiration date (-E) to 01/28/2013, minimum password age (-m) of 10 days, maximum password age (-M) of 90 days, inactivity period (-I) of 30 days after password expiration, and a warning time period (-W) of 15 days before password expiration.

```
sudo chage -E 01/28/2013 -m 10 -M 90 -I 30 -W 15 username
```

- To verify changes, use the same syntax as mentioned previously:

```
sudo chage -l username
```

The output below shows the new policies that have been established for the account:

```
Last password change : Jan 20, 2008
Password expires : Apr 19, 2008
Password inactive : May 19, 2008
Account expires : Jan 31, 2008
Minimum number of days between password change : 5
Maximum number of days between password change : 90
Number of days of warning before password expires : 14
```

## 1.4.3 Strong Passwords

Applications, and libraries exist for your Ubuntu system to assist in generating, or enforcing strong passwords. A strong password is defined as any password which meets the following criteria:

1. At least *fifteen* (15) characters in length.
2. Does not contain your user name, real name, organization name, family member's names or names of your pets.
3. Does not contain your birth date.
4. Does not contain a complete dictionary word.
5. Is significantly different from your previous password.
6. Should contain three (3) of the following character types.
  - a) Lowercase Alphabetical (a, b, c, etc.)
  - b) Uppercase Alphabetical (A, B, C, etc.)
  - c) Numerics (0, 1, 2, etc.)
  - d) Special Characters (@, %, !, etc.)

# Generating Strong Passwords in Ubuntu

## a) APG

Ubuntu has a Automatic Password Generator called apg. APG can be used to configure strong passwords. apg generates several random passwords. It uses several password generation algorithms and a built-in pseudo random number generator.

### Installing APG

To install APG, ensure you have enabled the Universe Packages, and have a live connection to the Internet, then issue the following command at a command prompt:

```
sudo apt-get install apg
```

You will be prompted for a password. The password being requested is *your user password*, or the same password you use when issuing commands with the sudo command. After authentication, the APG package will be downloaded, and installed. You are now ready to begin using, and further configuring APG to your liking.

For Example: To generate a password of say 25 characters and random characters use the command.

```
apg -m 25 -a 1
```

The -m option allows you to determine the number of characters to be used.

The -a allows you to specify a random character password generation algorithm to be used.

For additional security, we can use the -s and -n option.

The -s option should be used if you wish to always enforce prompting for random data from standard input (keyboard), to ensure the most secure passwords possible, you should explicitly tell APG to do so with the -S option.

The -n option specifies the number of passwords to be generated

I have chosen -n 5 which will generate 5 random passwords for me -m25 will generate a password 25 characters long -a 1 allows random character generation and the -s option will generate a password based on some random data you provide.

For Example: The output of the apg -n 5 -m 25 -a 1 -s is as follows.

```
frodo@shire:~$ apg -n 5 -m 25 -a 1 -s
```

Please enter some random data (only first 16 are significant)

(eg. your old password):>

```
\;QiX2fJS;WxrvIx1-7Yg`X87
m"~3E(H_IY]r9vtNn,rC&6*V(
:k"h_O/K4s]^~HItqAHe\6GzO
Ivwx){;JGzuDv%H%yShD{m{!&
bv+1ekZXq]tk1CC,-$&)nz^J|
frodo@shire:~$
```

## b) Pwgen

The pwgen program generates passwords which are designed to be easily memorized by humans, while being as secure as possible. Human-memorable passwords are never going to be as secure as completely random passwords. In particular, passwords generated by pwgen without the -s option should not be used in places where the password could be attacked via an off-line brute-force attack. On the other hand, completely randomly generated passwords have a tendency to be written down, and are subject to being compromised in that fashion.

The pwgen program is designed to be used both interactively, and in shell scripts. Hence, its default behavior differs depending on whether the standard output is a tty device or a pipe to another program. Used interactively, pwgen will display a screenful of passwords, allowing the user to pick a single password, and then quickly erase the screen. This prevents someone from being able to "shoulder surf" the user's chosen password.

An typical pwgen command is as follows.

```
pwgen -cnysC 10
```

The various options provided in the command perform the following functions:

- c - Include at least one capital letter in the password
- n - Include at least one number in the password
- y - Include at least one special symbol in the password
- s - Generate completely random passwords
- C - Print the generated passwords in columns

The number 10 signifies the password length.

For example the output of `pwgen -cnysC 10` on my system is as follows.

```
LHDY5,T.~b !Ty_wNUy,4 ]/IA9J![qQ ^f01,gO9D\ "1[,|Y@B<J b+23319YNi q9:J1fT7k`  
nEJdQayb`3 xzO,A8Y5Cp |Z8pA{5!-) Axt8QzE<b' m7[?.;Hu;* ._1%={qW:p j&4CNbNv2
```

## 1.5. Other Security Considerations

Many applications use alternate authentication mechanisms that can be easily overlooked by even experienced system administrators. Therefore, it is important to understand and control how users authenticate and gain access to services and applications on your server.

### 1.5.1. SSH Access by Disabled Users

Simply disabling/locking a user account will not prevent a user from logging into your server remotely if they have previously set up RSA public key authentication. They will still be able to gain shell access to the server, without the need for any password. Remember to check the users home directory for files that will allow for this type of authenticated SSH access. e.g. `/home/username/.ssh/authorized_keys`.

Remove or rename the directory `.ssh/` in the user's home folder to prevent further SSH authentication capabilities.

Be sure to check for any established SSH connections by the disabled user, as it is possible they may have existing inbound or outbound connections. Kill any that are found.

Restrict SSH access to only user accounts that should have it. For example, you may create a group called "sshlogin" and add the group name as the value associated with the `AllowGroups` variable located in the file `/etc/ssh/sshd_config`.

`AllowGroups sshlogin`

Then add your permitted SSH users to the group "sshlogin", and restart the SSH service.

```
sudo adduser username sshlogin  
sudo /etc/init.d/ssh restart
```

Sometimes it is necessary to allow root logins when doing automated tasks such as backups. To disallow normal logins but allow forced commands, you can use:

`PermitRootLogin forced-commands-only`

# Chapter 2

## Firewall

### 2.1. Introduction

iptables is the userspace command line program used to configure the Linux 2.4.x and 2.6.x IPv4 packet filtering ruleset. The Kernel space firewall application is called Netfilter. Since Network Address Translation is also configured from the packet filter ruleset, iptables is used for this, too.

### 2.2 ufw-Uncomplicated Firewall

ufw is the default firewall configuration tool for Ubuntu developed to enable ease iptables firewall configuration, ufw provides a user friendly way to create an IPv4 or IPv6 host-based firewall.

The following are some examples of how to use ufw:

- To enable ufw

```
sudo ufw enable
```

- To open a port (smtp in this example):

```
sudo ufw allow 25
```

- To add rules in a numbered format:

```
sudo ufw insert 1 allow 443
```

- Similarly, to close an opened port:

```
sudo ufw deny 443
```

- To remove a rule, use delete followed by the rule:

```
sudo ufw delete deny 443
```

- It is also possible to allow access from specific hosts or networks to a port. The following example allows ssh access from host 10.10.1.1 to any ip address on this host:

```
sudo ufw allow proto tcp from 10.1.1.1 to any port 22
```

Replace 10.10.10.1 with 10.10.0.0/24 to allow ssh access from the entire subnet.

- Adding the --dry-run option to a ufw command will output the resulting rules, but not apply them. For example, the following is what would be applied if opening the HTTP port:

```
sudo ufw --dry-run allow http
```

\*filter

```
:ufw-user-input - [0:0]
```

```
:ufw-user-output - [0:0]
```

```
:ufw-user-forward - [0:0]
```

```
:ufw-user-limit - [0:0]
```

```
:ufw-user-limit-accept - [0:0]
```

```
### RULES ###
```

```
### tuple ### allow tcp 80 0.0.0.0/0 any 0.0.0.0/0
```

```
-A ufw-user-input -p tcp --dport 80 -j ACCEPT
```

```
### END RULES ###
```

```
-A ufw-user-input -j RETURN
```

```
-A ufw-user-output -j RETURN
```

```
-A ufw-user-forward -j RETURN
```

```
-A ufw-user-limit -m limit --limit 3/minute -j LOG --log-prefix "[UFW LIMIT]: "
```

```
-A ufw-user-limit -j REJECT
```

```
-A ufw-user-limit-accept -j ACCEPT  
COMMIT  
Rules updated
```

- ufw can be disabled by:

```
sudo ufw disable
```

- To see the firewall status, enter:

```
sudo ufw status
```

- And for more verbose status information use:

```
sudo ufw status verbose
```

- To view the numbered format:

```
sudo ufw status numbered
```

Note: If the port you want to open or close is defined in `/etc/services`, you can use the port name instead of the number. In the above examples, replace 22 with `ssh`.

This is a quick introduction to using ufw. Please refer to the ufw man page for more information.

## 2.2.1. ufw Application Integration

Applications that open ports can include an ufw profile, which details the ports needed for the application to function properly. The profiles are kept in `/etc/ufw/applications.d`, and can be edited if the default ports have been changed.

- To view which applications have installed a profile, enter the following in a terminal:

```
sudo ufw app list
```

- Similar to allowing traffic to a port, using an application profile is accomplished by entering:

```
sudo ufw allow Samba
```

- An extended syntax is available as well:



ufw allow from 192.168.0.0/24 to any app Samba

Replace Samba and 192.168.0.0/24 with the application profile you are using and the IP range for your network.

Note: There is no need to specify the protocol for the application, because that information is detailed in the profile. Also, note that the app name replaces the port number.

- To view details about which ports, protocols, etc are defined for an application, enter:

```
sudo ufw app info Samba
```

## 2.3 IP Masquerading

IP Masquerade is a networking function in Linux similar to NAT (Network Address Translation) servers found in many firewalls and network routers. Linux IP Masquerading allows for this functionality even though the internal machines don't have an external assigned IP address.

Masquerade allows a set of machines to invisibly access the Internet via the masqueraded gateway. To other machines on the Internet, the outgoing traffic will appear to be from the Linux server itself. In addition to the added functionality, IP Masquerade also helps create a very secured networking environment.

### 2.3.1. ufw Masquerading

IP Masquerading can be achieved using customized ufw rules. This is done via the current back end for ufw iptables-restore with the rules files located in /etc/ufw/\*.rules. These files are used to add legacy iptables rules used without ufw, and rules that are more network gateway or bridge related.

The rules are split into two different files, rules that should be executed before ufw command line rules, and rules that are executed after ufw command line rules.

- First, packet forwarding needs to be enabled in ufw. Two configuration files will need to be

adjusted, in /etc/default/ufw change the DEFAULT\_FORWARD\_POLICY to “ACCEPT”:  
DEFAULT\_FORWARD\_POLICY="ACCEPT"

```
DEFAULT_FORWARD_POLICY="ACCEPT"
```

Then edit /etc/ufw/sysctl.conf and uncomment:

```
net/ipv4/ip_forward=1
```

Similarly, for IPv6 forwarding uncomment:

```
net/ipv6/conf/default/forwarding=1
```

Now we will add rules to the `/etc/ufw/before.rules` file. The default rules only configure the filter table, and to enable masquerading the nat table will need to be configured. Add the following to the top of the file just after the header comments:

```
# nat Table rules
*nat
:POSTROUTING ACCEPT [0:0]
```

```
# Forward traffic from eth1 through eth0.
-A POSTROUTING -s 192.168.0.0/24 -o eth0 -j MASQUERADE
```

```
# don't delete the 'COMMIT' line or these nat table rules won't be processed
COMMIT
```

The comments are not strictly necessary, but it is considered good practice to document your configuration. Also, when modifying any of the rules files in `/etc/ufw`, make sure these lines are the last line for each table modified:

```
# don't delete the 'COMMIT' line or these rules won't be processed
COMMIT
```

For each Table a corresponding COMMIT statement is required. In these examples only the nat and filter tables are shown, but you can also add rules for the raw and mangle tables.

Note: In the above example replace `eth0`, `eth1`, and `192.168.0.0/24` with the appropriate interfaces and IP range for your network.

- Finally, disable and re-enable ufw to apply the changes:

```
sudo ufw disable && sudo ufw enable
```

IP Masquerading should now be enabled. You can also add any additional FORWARD rules to the `/etc/ufw/before.rules`. It is recommended that these additional rules be added to the `ufw-before-forward` chain.

## 2.3.2. iptables Masquerading

iptables can also be used to enable Masquerading.

- Similar to ufw, the first step is to enable IPv4 packet forwarding by editing /etc/sysctl.conf and uncomment the following line .

```
net.ipv4.ip_forward=1
```

If you wish to enable IPv6 forwarding also uncomment:

```
net.ipv6.conf.default.forwarding=1
```

- Next, execute the sysctl command to enable the new settings in the configuration file:

```
sudo sysctl -p
```

- IP Masquerading can now be accomplished with a single iptables rule, which may differ slightly based on your network configuration:

```
sudo iptables -t nat -A POSTROUTING -s 192.168.0.0/16 -o ppp0 -j MASQUERADE
```

The above command assumes that your private address space is 192.168.0.0/16 and that your Internet-facing device is ppp0. The syntax is broken down as follows:

- -t nat -- the rule is to go into the nat table
- -A POSTROUTING -- the rule is to be appended (-A) to the POSTROUTING chain
- -s 192.168.0.0/16 -- the rule applies to traffic originating from the specified address space
- -o ppp0 -- the rule applies to traffic scheduled to be routed through the specified network device
- -j MASQUERADE -- traffic matching this rule is to "jump" (-j) to the MASQUERADE target to be manipulated as described above.

• Also, each chain in the filter table (the default table, and where most or all packet filtering occurs) has a default policy of ACCEPT, but if you are creating a firewall in addition to a gateway device, you may have set the policies to DROP or REJECT, in which case your masqueraded traffic needs to be allowed through the FORWARD chain for the above rule to work:

```
sudo iptables -A FORWARD -s 192.168.0.0/16 -o ppp0 -j ACCEPT
```

```
sudo iptables -A FORWARD -d 192.168.0.0/16 -m state --state ESTABLISHED,RELATED -i ppp0 -j ACCEPT
```

The above commands will allow all connections from your local network to the Internet and all traffic related to those connections to return to the machine that initiated them.

- If you want masquerading to be enabled on reboot, which you probably do, edit /etc/rc.local and add any commands used above. For example add the first command with no filtering:

```
iptables -t nat -A POSTROUTING -s 192.168.0.0/16 -o ppp0 -j MASQUERADE
```

## 2.4. Logs

Firewall logs are essential for recognizing attacks, troubleshooting your firewall rules, and noticing unusual activity on your network. You must include logging rules in your firewall for them to be generated, though, and logging rules must come before any applicable terminating rule (a rule with a target that decides the fate of the packet, such as ACCEPT, DROP, or REJECT).

If you are using ufw, you can turn on logging by entering the following in a terminal:

```
sudo ufw logging on
```

If using iptables instead of ufw, enter:

```
sudo iptables -A INPUT -m state --state NEW -p tcp --dport 80 -j LOG --log-prefix  
"NEW_HTTP_CONN: "
```

A request on port 80 from the local machine, then, would generate a log in dmesg that looks like this:

```
[4304885.870000] NEW_HTTP_CONN: IN=lo OUT=  
MAC=00:00:00:00:00:00:00:00:00:00:00:08:00 SRC=127.0.
```

The above log will also appear in /var/log/messages, /var/log/syslog, and /var/log/kern.log. This behavior can be modified by editing /etc/syslog.conf appropriately or by installing and configuring ulogd and using the ULOG target instead of LOG. The ulogd daemon is a userspace server that listens for logging instructions from the kernel specifically for firewalls, and can log to any file you like, or even to a PostgreSQL or MySQL database. Making sense of your firewall logs can be simplified by using a log analyzing tool such as fwanalyze, fwlogwatch, or lre.

## 2.5. Other Tools

There are many tools available to help you construct a complete firewall without intimate knowledge of iptables. For the GUI-inclined:

- Firestarter<sup>1</sup> is quite popular and easy to use.
- fwbuilder<sup>2</sup> is very powerful and will look familiar to an administrator who has used a commercial firewall utility such as Checkpoint FireWall-1.

If you prefer a command-line tool with plain-text configuration files:

- Shorewall3 is a very powerful solution to help you configure an advanced firewall for any network.
- ipkungfu4 should give you a working firewall "out of the box" with zero configuration, and will allow you to easily set up a more advanced firewall by editing simple, well-documented configuration files.
- fireflifer5 is designed to be a desktop firewall application. It is made up of a server (fireflifer-server) and your choice of GUI clients (GTK or QT), and behaves like many popular interactive firewall applications for Windows.

## 2.6. References

- The Ubuntu Firewall6 wiki page contains information on the development of ufw.
- Also, the ufw manual page contains some very useful information: `man ufw`.
- See the packet-filtering-HOWTO7 for more information on using iptables.
- The nat-HOWTO8 contains further details on masquerading.
- The IPTables HowTo9 in the Ubuntu wiki is a great resource.

-----  
<http://www.fs-security.com/>

<http://www.fwbuilder.org/>

<http://www.shorewall.net/>

<http://www.linuxkungfu.org/>

<http://fireflifer.sourceforge.net/>

<https://wiki.ubuntu.com/UncomplicatedFirewall>

<http://www.netfilter.org/documentation/HOWTO/packet-filtering-HOWTO.html>

<http://www.netfilter.org/documentation/HOWTO/NAT-HOWTO.html>

<https://help.ubuntu.com/community/IptablesHowTo>

# Chapter 3

## AppArmor

AppArmor is kernel enhancement to confine programs to a limited set of resources. AppArmor is a kernel enhancement to confine programs to a limited set of resources.

AppArmor's unique security model is to bind access control attributes to programs rather than to users

On an Ubuntu Cloud Instance, Apparmor is installed and loaded by default. The files and permissions that an application requires is determined by the corresponding Apparmor profile. Some Ubuntu packages are known to install their own profiles while additional profiles found in a separate package named apparmor-profiles.

AppArmor can operate in two modes: enforcement, and complain or learning:

- enforcement - Profiles loaded in enforcement mode will result in enforcement of the policy defined in the profile as well as reporting policy violation attempts to syslogd.
- complain - Profiles loaded in "complain" mode will not enforce policy. Instead, it will report policy violation attempts. This mode is convenient for developing profiles. To manage complain mode for individual profiles the utilities `aa-complain` and `aa-enforce` can be used. These utilities take a program name as an argument.

To install the apparmor-profiles package from a terminal prompt:

```
sudo apt-get install apparmor-profiles
```

AppArmor profiles have two modes of execution:

- Complaining/Learning: profile violations are permitted and logged. Useful for testing and developing new profiles.
- Enforced/Confined: enforces profile policy as well as logging the violation.

## 3.1. Using AppArmor

The `apparmor-utils` package contains command line utilities that you can use to change the AppArmor execution mode, find the status of a profile, create new profiles, etc.

- `apparmor_status` is used to view the current status of AppArmor profiles.

```
sudo apparmor_status
```

- `aa-complain` places a profile into complain mode.

```
sudo aa-complain /path/to/bin
```

- `aa-enforce` places a profile into enforce mode.

```
sudo aa-enforce /path/to/bin
```

- The `/etc/apparmor.d` directory is where the AppArmor profiles are located. It can be used to manipulate the mode of all profiles.

Enter the following to place all profiles into complain mode:

```
sudo aa-complain /etc/apparmor.d/*
```

To place all profiles in enforce mode:

```
sudo aa-enforce /etc/apparmor.d/*
```

- `apparmor_parser` is used to load a profile into the kernel. It can also be used to reload a currently loaded profile using the `-r` option. To load a profile:

```
cat /etc/apparmor.d/profile.name | sudo apparmor_parser -a
```

To reload a profile:

```
cat /etc/apparmor.d/profile.name | sudo apparmor_parser -r
```

- /etc/init.d/apparmor can be used to reload all profiles:

```
sudo /etc/init.d/apparmor reload
```

- The /etc/apparmor.d/disable directory can be used along with the apparmor\_parser -R option to disable a profile.

```
sudo ln -s /etc/apparmor.d/profile.name /etc/apparmor.d/disable/  
sudo apparmor_parser -R /etc/apparmor.d/profile.name
```

- To re-enable a disabled profile remove the symbolic link to the profile in /etc/apparmor.d/disable/. Then load the profile using the -a option.

```
sudo rm /etc/apparmor.d/disable/profile.name  
cat /etc/apparmor.d/profile.name | sudo apparmor_parser -a
```

- AppArmor can be disabled, and the kernel module unloaded by entering the following:

```
sudo /etc/init.d/apparmor stop  
sudo update-rc.d -f apparmor remove
```

- To re-enable AppArmor enter:

```
sudo /etc/init.d/apparmor start  
sudo update-rc.d apparmor defaults
```

Note: Replace profile.name with the name of the profile you want to manipulate. Also, replace /path/to/bin/ with the actual executable file path. For example for the ping command use /bin/ping

## 3.2. Profiles

AppArmor profiles are simple text files located in /etc/apparmor.d/. The files are named after the full path to the executable they profile replacing the "/" with ".". For example /etc/apparmor.d/bin.ping is the AppArmor profile for the /bin/ping command.



There are two main type of rules used in profiles:

- Path entries: which detail which files an application can access in the file system.
- Capability entries: determine what privileges a confined process is allowed to use.

As an example take a look at `/etc/apparmor.d/bin.ping`:

```
#include <tunables/global>
/bin/ping flags=(complain) {
#include <abstractions/base>
#include <abstractions/consoles>
#include <abstractions/nameservice>
capability net_raw,
capability setuid,
network inet raw,

/bin/ping mixr,
/etc/modules.conf r,
}
```

- `#include <tunables/global>`: include statements from other files. This allows statements pertaining to multiple applications to be placed in a common file.
- `/bin/ping flags=(complain)`: path to the profiled program, also setting the mode to complain.
- `capability net_raw`,: allows the application access to the CAP\_NET\_RAW Posix.1e capability.
- `/bin/ping mixr`,: allows the application read and execute access to the file.

Note: After editing a profile file the profile must be reloaded. See Section 4.1, “Using AppArmor” for details.

### 3.2.1. Creating a Profile

- Design a test plan: Try to think about how the application should be exercised. The test plan should be divided into small test cases. Each test case should have a small description and list the steps to follow.

Some standard test cases are:

- Starting the program.
- Stopping the program.
- Reloading the program.
- Testing all the commands supported by the init script.
- Generate the new profile: Use aa-genprof to generate a new profile. From a terminal:

```
sudo aa-genprof executable
```

For example:

```
sudo aa-genprof slapd
```

- To get your new profile included in the apparmor-profiles package, file a bug in Launchpad against the AppArmor10 package:
  - Include your test plan and test cases.
  - Attach your new profile to the bug.

## 3.2.2. Updating Profiles

When the program is misbehaving, audit messages are sent to the log files. The program aa-logprof can be used to scan log files for AppArmor audit messages, review them and update the profiles. From a terminal:

```
sudo aa-logprof
```

## 3.3. References

- See the AppArmor Administration Guide<sup>11</sup> for advanced configuration options.
- For details using AppArmor with other Ubuntu releases see the AppArmor Community Wiki<sup>12</sup> page.
- The OpenSUSE AppArmor<sup>13</sup> page is another introduction to AppArmor.

# Chapter 4

## Public Key Cryptography

In this section we will explore the concept of public key cryptography. Public key cryptography uses an asymmetrical scheme that uses a *pair* of keys for encryption: a *public key*, which encrypts data, and a corresponding *private*, or *secret key* for decryption. One can publish their public key while keeping your private key secret. You can share your public key so that anyone having a copy of your public key can encrypt information that only you can read.

Public key encryption has two primary uses, encryption and digital signatures. In this system the necessity to trust the security of some means of communication is eliminated. The only requirement is that public keys need to be associated with their users in a trusted (authenticated) manner (say for example, in a trusted online vault). Furthermore, public-key cryptography can be used not only for privacy (encryption), but also for authentication (digital signatures) and other various techniques.

### Encryption

When Sarah wants to send a confidential message to Mathew, she looks up Mathew's public key from a directory and uses it to encrypt the message and sends it across to Mathew. Mathew then decrypts the message using his private key. Even if the encrypted message is intercepted mid way by say a man in the middle attack, the encrypted message would look gibberish to them unless they possess the private key to decrypt the message. Mathew, and only Mathew can read the message (because only Mathew knows Mathew's private key).

### Digital Signatures

To sign a message, Sarah does a computation involving both her private key and the message itself. The output is called a digital signature and is attached to the message. To verify the signature, Mathew does a computation involving the message, the purported signature, and Sarah's public key. If the result is correct according to a simple, prescribed mathematical relation, the signature is

verified to be genuine; otherwise, the signature is fraudulent, or the message may have been altered.

## Public Key Cryptography

A common use for public-key cryptography is encrypting application traffic using a Secure Socket Layer (SSL) or Transport Layer Security (TLS) connection. For example, configuring Apache to provide HTTPS, the HTTP protocol over SSL. This allows a way to encrypt traffic using a protocol that does not itself provide encryption.

A Certificate is a method used to distribute a public key and other information about a server and the organization who is responsible for it. Certificates can be digitally signed by a Certification Authority or CA. A CA is a trusted third party that has confirmed that the information contained in the certificate is accurate.

## 4.1. Generating a certificate

To set up a secure server using public-key cryptography, in most cases, you send your certificate request (including your public key), proof of your company's identity, and payment to a CA. The CA verifies the certificate request and your identity, and then sends back a certificate for your secure server. Alternatively, you can create your own self-signed certificate.

Note, that self-signed certificates should not be used in most production environments.

Continuing the HTTPS example, a CA-signed certificate provides two important capabilities that a self-signed certificate does not:

- Browsers (usually) automatically recognize the certificate and allow a secure connection to be made without prompting the user.
- When a CA issues a signed certificate, it is guaranteeing the identity of the organization that is providing the web pages to the browser.

Most Web browsers, and computers, that support SSL have a list of CAs whose certificates they automatically accept. If a browser encounters a certificate whose authorizing CA is not in the list, the browser asks the user to either accept or decline the connection. Also, other applications may generate an error message when using a self-signed certificate.

The process of getting a certificate from a CA is fairly easy. A quick overview is as follows:

1. Create a private and public encryption key pair.
2. Create a certificate request based on the public key. The certificate request contains information

about your server and the company hosting it.

3. Send the certificate request, along with documents proving your identity, to a CA. We cannot tell you which certificate authority to choose. Your decision may be based on your past experiences, or on the experiences of your friends or colleagues, or purely on monetary factors.

Once you have decided upon a CA, you need to follow the instructions they provide on how to obtain a certificate from them.

4. When the CA is satisfied that you are indeed who you claim to be, they send you a digital certificate.

5. Install this certificate on your secure server, and configure the appropriate applications to use the certificate.

### Generating a Certificate Signing Request(CSR)

Whether you are getting a certificate from a CA or generating your own self-signed certificate, the first step is to generate a key.

If the certificate will be used by service daemons, such as Apache, Postfix, Dovecot, etc, a key without a passphrase is often appropriate. Not having a passphrase allows the services to start without manual intervention, usually the preferred way to start a daemon.

This section will cover generating a key with a passphrase, and one without. The non-passphrase key will then be used to generate a certificate that can be used with various service daemons.

Note: Running your secure service without a passphrase is convenient because you will not need to enter the passphrase every time you start your secure service. But it is insecure and a compromise of the key means a compromise of the server as well.

To generate the keys for the Certificate Signing Request (CSR) run the following command from a terminal prompt:

```
openssl genrsa -des3 -out server.key 2048
```

```
openssl genrsa -des3 -out server.key 2048
Generating RSA private key, 2048 bit long modulus
.....++++++
.....++++++
e is 65537 (0x10001)
Enter pass phrase for server.key:
```

You can now enter your passphrase. For best security, it should at least contain eight characters. The minimum length when specifying -des3 is four characters. It should include numbers and/or punctuation and not be a word in a dictionary. Also remember that your passphrase is case-sensitive.

Re-type the passphrase to verify. Once you have re-typed it correctly, the server key is generated and stored in the server.key file.

Now create the insecure key, the one without a passphrase, and shuffle the key names:

```
openssl rsa -in server.key -out server.key.insecure
mv server.key server.key.secure
mv server.key.insecure server.key
```

The insecure key is now named server.key, and you can use this file to generate the CSR without passphrase.

To create the CSR, run the following command at a terminal prompt:

```
openssl req -new -key server.key -out server.csr
```

It will prompt you enter the passphrase. If you enter the correct passphrase, it will prompt you to enter Company Name, Site Name, Email Id, etc. Once you enter all these details, your CSR will be created and it will be stored in the server.csr file.

You can now submit this CSR file to a CA for processing. The CA will use this CSR file and issue the certificate. On the other hand, you can create self-signed certificate using this CSR.

## 4.2. Creating a Self-Signed Certificate

To create the self-signed certificate, run the following command at a terminal prompt:

```
openssl x509 -req -days 365 -in server.csr -signkey server.key -out server.crt
```

The above command will prompt you to enter the passphrase. Once you enter the correct passphrase, your certificate will be created and it will be stored in the server.crt file.

**Note:** If your secure server is to be used in a production environment, you probably need a CA-signed certificate. It is not recommended to use self-signed certificate.

## 4.3 Installing the Certificate

You can install the key file server.key and certificate file server.crt, or the certificate file issued by your CA, by running following commands at a terminal prompt:

```
sudo cp server.crt /etc/ssl/certs
sudo cp server.key /etc/ssl/private
```

Now simply configure any applications, with the ability to use public-key cryptography, to use the certificate and key files. For example, Apache can provide HTTPS, Dovecot can provide IMAPS and POP3S, etc.

## 4.4. Certification Authority

If the services on your network require more than a few self-signed certificates it may be worth the additional effort to setup your own internal Certification Authority (CA). Using certificates signed by your own CA, allows the various services using the certificates to easily trust other services using certificates issued from the same CA.

1. First, create the directories to hold the CA certificate and related files:

```
sudo mkdir /etc/ssl/CA
sudo mkdir /etc/ssl/newcerts
```

2. The CA needs a few additional files to operate, one to keep track of the last serial number used by the CA, each certificate must have a unique serial number, and another file to record which certificates have been issued:

```
sudo sh -c "echo '01' > /etc/ssl/CA/serial"
sudo touch /etc/ssl/CA/index.txt
```

3. The third file is a CA configuration file. Though not strictly necessary, it is very convenient when issuing multiple certificates. Edit `/etc/ssl/openssl.cnf`, and in the `[ CA_default ]` change:

<code>dir</code>	<code>= /etc/ssl/</code>	<code># Where everything is kept</code>
<code>database</code>	<code>= \$dir/CA/index.txt</code>	<code># database index file.</code>
<code>certificate</code>	<code>= \$dir/certs/cacert.pem</code>	<code># The CA certificate</code>
<code>serial</code>	<code>= \$dir/CA/serial</code>	<code># The current serial number</code>
<code>private_key</code>	<code>= \$dir/private/cakey.pem</code>	<code># The private key</code>

4. Next, create the self-signed root certificate:

```
openssl req -new -x509 -extensions v3_ca -keyout cakey.pem -out cacert.pem -days 3650
```

You will then be asked to enter the details about the certificate.

5. Now install the root certificate and key:

```
sudo mv cakey.pem /etc/ssl/private/
sudo mv cacert.pem /etc/ssl/certs/
```

6. You are now ready to start signing certificates. The first item needed is a Certificate Signing Request (CSR), see Section 5.2, “Generating a Certificate Signing Request (CSR)” for details. Once you have a CSR, enter the following to generate a certificate signed by the CA:

```
sudo openssl ca -in server.csr -config /etc/ssl/openssl.cnf
```

After entering the password for the CA key, you will be prompted to sign the certificate, and again to commit the new certificate. You should then see a somewhat large amount of output related to the certificate creation.

7. There should now be a new file, `/etc/ssl/newcerts/01.pem`, containing the same output. Copy and paste everything beginning with the line: `-----BEGIN CERTIFICATE-----` and continuing through the line: `-----END CERTIFICATE-----` lines to a file named after the hostname of the server where the certificate will be installed. For example `mail.example.com.crt`, is a nice descriptive name.

Subsequent certificates will be named `02.pem`, `03.pem`, etc.

Note : Replace `mail.example.com.crt` with your own descriptive name.

8. Finally, copy the new certificate to the host that needs it, and configure the appropriate applications to use it. The default location to install certificates is `/etc/ssl/certs`. This enables multiple services to use the same certificate without overly complicated file permissions.

For applications that can be configured to use a CA certificate, you should also copy the `/etc/ssl/certs/cacert.pem` file to the `/etc/ssl/certs/` directory on each server.

## 4.5. References

- For more detailed instructions on using cryptography see the SSL Certificates HOWTO<sup>15</sup> by `tldp.org`
- The Wikipedia HTTPS<sup>16</sup> page has more information regarding HTTPS.
- For more information on OpenSSL see the OpenSSL Home Page<sup>17</sup>.
- Also, O'Reilly's Network Security with OpenSSL<sup>18</sup> is a good in depth reference.

---

<http://tldp.org/HOWTO/SSL-Certificates-HOWTO/index.html>

<http://en.wikipedia.org/wiki/Https>

<http://www.openssl.org/>

<http://oreilly.com/catalog/9780596002701/>



# Chapter 5

## eCryptfs

eCryptfs is a cryptographic filesystem for Linux. Cryptographic metadata is stored by eCryptfs in the header of each file written, so that the encrypted files can be copied between hosts; the file will be decryptable with the proper key.

During installation of any Ubuntu 12.04 server, there is an option to encrypt the /home partition. This will automatically configure everything needed to encrypt and mount the partition. On an Ubuntu Instance you can install ecryptfs and remount the partition to enable eCryptfs for that particular partition.

As an example, this section will cover configuring /srv to be encrypted using eCryptfs.

### 5.1. Using eCryptfs

First, install the necessary packages. From a terminal prompt enter:

```
sudo apt-get install ecryptfs-utils
```

Now re-mount the partition to be encrypted:

```
sudo mount -t ecryptfs /srv /srv
```

You will then be prompted for some details on how ecryptfs should encrypt the data.

To test that files placed in /srv are indeed encrypted copy the /etc/default folder to /srv:

```
sudo cp -r /etc/default /srv
```

Now unmount /srv, and try to view a file:

```
sudo umount /srv  
cat /srv/default/cron
```

Remounting /srv using ecryptfs will make the data viewable once again.

## 5.2. Automatically Mounting Encrypted Partitions

There are a couple of ways to automatically mount an ecryptfs encrypted filesystem at boot. This example will use a `/root/.ecryptfsrc` file containing mount options, along with a passphrase file residing on a USB key.

First, create `/root/.ecryptfsrc` containing:

```
key=passphrase:passphrase_passwd_file=/mnt/usb/passwd_file.txt
ecryptfs_sig=5826dd62cf81c615
ecryptfs_cipher=aes
ecryptfs_key_bytes=16
ecryptfs_passthrough=n
```

```
ecryptfs_enable_filename_crypto=n
```

Note: Adjust the `ecryptfs_sig` to the signature in `/root/.ecryptfs/sig-cache.txt`.

Next, create the `/mnt/usb/passwd_file.txt` passphrase file:

```
passphrase_passwd=[secrets]
```

Now add the necessary lines to `/etc/fstab`:

<code>/dev/sdb1</code>	<code>/mnt/usb</code>	<code>ext3</code>	<code>ro</code>	<code>0 0</code>	
<code>/srv</code>	<code>/srv</code>	<code>ecryptfs</code>	<code>defaults</code>	<code>0 0</code>	

Make sure the USB drive is mounted before the encrypted partition.  
Finally, reboot and the `/srv` should be mounted using eCryptfs.

# Chapter 6

## Automatic Security Updates

Automatic Security Updates in an Ubuntu Image can be handled via the `unattended-upgrade` package.

This program can download and install security upgrades automatically and unattended, taking care to only install packages from the configured APT source, and checking for `dpkg` prompts about configuration file changes. All output is logged to `/var/log/unattended-upgrades.log`.

First install the `unattended-upgrades` package

```
sudo apt-get install unattended-upgrades
```

To enable `unattended-upgrades`, do:

```
sudo dpkg-reconfigure -plow unattended-upgrades
```

(it's an interactive dialog) which will create `/etc/apt/apt.conf.d/20auto-upgrades` with the following contents:

```
APT::Periodic::Update-Package-Lists "1";  
APT::Periodic::Unattended-Upgrade "1";
```

To configure unattended upgrades via cron and aptitude, edit `/etc/cron.weekly/apt-security-updates`

Copy the following text into this new file, save, and exit:

```
echo "*****" >> /var/log/apt-security-updates  
date >> /var/log/apt-security-updates  
aptitude update >> /var/log/apt-security-updates
```

```
aptitude safe-upgrade -o Aptitude::Delete-Unused=false --assume-yes --target-release `lsb_release  
-cs`-security >> /var/log/apt-security-updates  
echo "Security updates (if any) installed"
```

Once you are complete, you want to make the file executable. So, via the terminal, type the following line:

```
sudo chmod +x /etc/cron.weekly/apt-security-updates.
```

This script will run once weekly and it installs all available packages from the security repository. It also generates a log in `/var/log/apt-security-updates` for later inspection in case something goes wrong.

This script will output information to a log file, so to prevent this log file from getting too large we need to make sure it gets rotated out. To do this, we'll use the logrotate utility, which comes with Ubuntu.

Edit `/etc/logrotate.d/apt-security-updates` and add the following lines.

```
/var/log/apt-security-updates {  
    rotate 2  
    weekly  
    size 250k  
    compress  
    notifempty  
}
```

# Chapter 7

## Shared Memory

Linux Shared memory is memory that may be simultaneously accessed by multiple programs with an intent to provide communication between them.

Linux distributions based on the 2.6 kernel have started to offer `/dev/shm` as shared memory in the form of a RAM disk, more specifically as a world-writable directory (a directory in which every user of the system can create files) that is stored in memory. Ubuntu based distributions include it by default.

`/dev/shm` is nothing but implementation of traditional shared memory concept. It is an efficient means of passing data between programs. One program will create a memory portion, which other processes (if permitted) can access. This will result into speeding up things on Linux.

By default `/dev/shm` is mounted read/write with permissions to execute programs. In recent years, `/dev/shm` has been increasingly used in an attacks a service which is running like `httpd`.

In most cases `/dev/shm` need not be mounted read/write in servers and desktops as not real time configuration of hardware is required on any commodity hardware or even server class hardware.

## Securing Shared Memory

To mount `/dev/shm` as read only, edit the `/etc/fstab` to include the following line.

```
tmpfs /dev/shm tmpfs defaults,ro 0 0
```

This mount `/dev/shm` in read-only mode. If you have a good reason to keep it writable, put this line in `/etc/fstab` instead:

```
tmpfs /dev/shm tmpfs defaults,noexec,nosuid 0 0
```

This will mount `/dev/shm` writable, but without permission to execute programs and without permission to change the UID of running programs.

The changes will take effect the next time you reboot, unless you remount `/dev/shm` with the `sudo mount -o remount /dev/shm` command.

# Chapter 8

## Single Packet Authorization

Single Packet Authorization (SPA) using fwknop is one of the most important recent innovations in server and network access control technology. fwknop implements an authorization scheme known as Single Packet Authorization (SPA) for Linux systems running iptables.

The main application of this program is to use iptables or ipfw in a default-drop stance to protect services such as SSH with an additional layer of security in order to make the exploitation of vulnerabilities (both 0-day and unpatched code) much more difficult.

An authorization server fwknopd passively monitors authorization packets via libpcap and hence there is no "server" to which to connect in the traditional sense. Any service protected by fwknop is inaccessible (by using iptables or ipfw to intercept packets within the kernel) before authenticating; anyone scanning for the service will not be able to detect that it is even listening.

Although the default data collection method in Single Packet Authorization mode is to use libpcap to sniff packets off the wire, fwknop can also read packets out of a file that is written by the iptables ulogd pcap writer (or a separate sniffer process that is writing packet data to a file).

In its simplest form, your Linux server can have an inbound firewall rule that by default drops all access to any of its listening services. Nmap scans will completely fail to detect any open ports, and zero-day attacks will not have any effect on vulnerable services since the firewall is blocking access to the applications.

An authorized user sends a single encrypted UDP packet that is passively sniffed and analyzed by the fwknopd service running on the server using pcap. If successfully authenticated, fwknopd dynamically creates an iptables firewall rule, granting the source IP address of the authorized client access to the service for a defined period of time (default is 30 seconds).

# Installing fwknop

We will first install fwknop and all of its prerequisites.

Since we are using SSH, make sure you have installed the OpenSSH server. If you have already done so, you may skip this step.

```
$ sudo apt-get install openssh-server
```

## fwknop installation prerequisites

```
$ sudo apt-get install build-essential libpcap-dev mailx
```

## fwknop installation

Download the latest version of fwknop from the official website, and install. Please check for latest version of fwknop at <http://www.cipherdyne.org/fwknop/download/>.

```
$ wget http://www.cipherdyne.org/fwknop/download/fwknop-1.9.5.tar.gz
$ tar zxvf fwknop-1.9.5.tar.gz
$ cd fwknop-1.9.5
$ sudo ./install.pl
```

The installer will ask you a few questions which you must answer appropriately. For a typical server installation the following answers will work nicely. Note: The output has been snipped and slightly modified for brevity.

```
In which mode will fwknop be executed on the local system?
(client/[server]): server
```

```
Which of the following data acquisition methods would you like to use?
([pcap], file_pcap, ulogd, syslogd, syslog-ng): pcap
```

```
Which network interface would you like fwknop to sniff packets from? eth0
```

```
fwknop access alerts will be sent to: root@localhost
Would you like access alerts sent to a different address ([y]/n)? n
```

Enable fwknop at boot time ([y]/n)? y

fwknop has been installed! To start in server mode, run  
"/etc/init.d/fwknop start"

Before starting the fwknop service, you need to configure authentication. The following steps will outline the usage of GnuPG for fwknop authentication.

## GnuPG Authentication

### Generate Client-side GnuPG key pairs

From the workstation you are using as a client, generate your client side key pair, and export the public key to a text file.

```
$ gpg --gen-key
```

```
$ gpg --list-key fwknop-client@localhost
pub 1024D/2FBEA691 2007-11-17
uid          fwknop client key <fwknop-client@localhost>
sub 2048g/C3FD544F 2007-11-17
```

```
$ gpg -a --export 2FBEA691 > fwknop-client.asc
```

Upload the client public key to the SSH server.

```
$ scp fwknop-client.asc <server-ip-address>:
```

### Generate Server-side GnuPG key pairs

From the server, generate the server side key pair, and export the public key to a text file.

```
$ gpg --gen-key
```

```
$ gpg --list-key fwknopd@localhost
```

```
pub 1024D/3F89D02C 2007-11-17
uid          fwknop server key <fwknopd@localhost>
sub 2048g/C6AABDF0 2007-11-17
```

```
$ gpg -a --export 3F89D02C > fwknop-server.asc
```

From the client, download the server's exported public key.

```
$ scp <server-ip-address>:fwknop-server.asc .
```

### Import and sign GnuPG key pairs

From your client side workstation, import the server's public key into your keyring and sign it.



```
$ gpg --import fwknop-server.asc
```

```
$ gpg --sign-key fwknopd@localhost
```

From your server, import the clients public key into your keyring and sign it.

```
$ gpg --import fwknop-client.asc
```

```
$ gpg --sign-key fwknop-client@localhost
```

## Finishing the installation

fwknopd configuration

You will need to edit the fwknop configuration file “/etc/fwknop/access.conf”. An example of this configuration is shown below.

```
SOURCE: ANY;  
OPEN_PORTS: tcp/22;  
DATA_COLLECT_MODE: PCAP;  
GPG_HOME_DIR: /root/.gnupg;  
GPG_DECRYPT_ID: SERVER_KEY_ID;  
GPG_DECRYPT_PW: PASSWORD_HERE;  
GPG_REMOTE_ID: CLIENT_KEY_ID;  
FW_ACCESS_TIMEOUT: 30;
```

Note: Be sure to replace the GPG\_HOME\_DIR variable with the path to the correct key ring. The other variables are fairly self explanatory.

## Optional Fall back Authentication

Although NOT recommended, in addition to the use of GnuPG, you have the option to configure a static password as a fall back authentication mechanism. This is especially useful in scenarios where you need access to your server but do not have access to your GnuPG key pair. The fwknop client requires that you use a minimum of 8 characters for your password, but you should use additional best practices to reduce the threat of password theft and brute force attacks.

If you would like to use this form of authentication, add your selected password to “/etc/fwknop/access.conf”.

```
KEY: YourPasswordHere;
```

Start fwknop

```
$ sudo /etc/init.d/fwknop start
```

### Testing fwknop

You should now be ready to test things out using another computer with the fwknop client. You install everything exactly the same as the server, with the exception of specifying that the installer should run fwknop as a client.

The typical authorization process from client to server can be completed as follows,

```
$ fwknop -A tcp/22 --gpg-recv SERVER_KEY --gpg-sign CLIENT_KEY -w -k SERVER_IP
```

The “-w” flag queries [www.whatismyip.com](http://www.whatismyip.com) from the clients real ip address and uses that as the source I.P Address. This is useful when you are behind a NAT firewall, since the source address on the SPA packet would otherwise be a local address.

If you are on the same network as the server, or simply do not have to worry about NAT, the syntax would be as follows:

```
$ fwknop -A tcp/22 --gpg-recv SERVER_KEY --gpg-sign CLIENT_KEY -s -k SERVER_IP
```

The “-s” flag specifies that the server should use the source address from which the SPA packet originates.

Upon issuing the command, the fwknop client will ask you to enter your client side GnuPG key password. The output will look similar to the following.

```
$ fwknop -A tcp/22 --gpg-recv 3F89D02C --gpg-sign 2FBEA691 -s -k 192.168.200.131
```

```
[+] Starting fwknop client (SPA mode)...
```

```
[+] Enter the GnuPG password for signing key: 2FBEA691
```

GnuPG signing password:

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

```
[+] Packet fields:
```

```
Random data: 4498932332071523
Username:    yourusername
Timestamp:   1212948230
Version:     1.9.5
Type:        1 (access mode)
Access:      0.0.0.0,tcp/22
SHA256 digest: UBgkOqX60ILVFUjH/BbQE/wGW3/nMp1pHyh7f5bQgAk
```

```
[+] Sending 1040 byte message to 192.168.200.131 over udp/62201...
```

If you have opted to configure a symmetric key password for fall back authentication, you can test this out by simply omitting the gpg options:

```
$ fwknop -A tcp/22 -s -k 192.168.200.1
```

```
[+] Starting fwknop client (SPA mode)...
```

```
[+] Enter an encryption key. This key must match a key in the file
```

```
/etc/fwknop/access.conf on the remote system.
```

Encryption Key:

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

```
[+] Packet fields:
```

```
Random data: 2259603590509959
```

```
Username:    yourusername
Timestamp:   1212948275
Version:     1.9.5
Type:        1 (access mode)
Access:      0.0.0.0,tcp/22
SHA256 digest: 9BUOw7cNHacvTOfZwCicv3GGgkaT2V14n822N6LH3WM
```

[+] Sending 182 byte message to 192.168.200.1 over udp/62201...

If successful, your server adds the appropriate access list entry for you to connect using your ssh client. You will have 30 seconds to make the connection, after which the access list is dynamically removed.

```
$ ssh username@SERVER_IP
```

The following is a great way to view what is going on in the background. Run this command while using the client to see the action in real time.

```
$ tail -f /var/log/syslog | grep fwknop
```

```
Jun  8 11:03:55 ubuntu-server fwknopd: received valid GnuPG encrypted packet (signed with required key ID: "2FBEA691") from: 192.168.200.1, remote user: yourusername, client version: 1.9.5 (SOURCE line num: 22)
```

```
Jun  8 11:03:55 ubuntu-server fwknopd: add FWKNOP_INPUT 192.168.200.1 -> 0.0.0.0/0(tcp/22) ACCEPT rule 30 sec
```

```
Jun  8 11:04:26 ubuntu-server fwknopd(knoptm): removed iptables FWKNOP_INPUT ACCEPT rule for 192.168.200.1 -> 0.0.0.0/0(tcp/22), 30 sec timeout exceeded
```

It is important to note that the SPA packet is sent to the servers IP address using the destination port of UDP/62201. You must ensure that this port number is allowed outbound from the network you are connecting from, and that no router or firewall is blocking it from reaching your server.

It should also be noted that the time stamp embedded in the SPA packet must fall within 120 seconds of the servers clock. You should make sure that both the server and client are using NTP to keep their clocks as close as possible. This will be apparent when you syslog gives you the following error messages.

```
Jun  8 11:00:30 ubuntu-server fwknopd: remote time stamp is older than 120 second max age.
```

It's also helpful to view your iptables output in real time while testing.

```
$ sudo watch -n1 iptables -L -n
```

# Chapter 9

## Intrusion Detection Systems

A host-based IDS analyzes several areas to determine misuse (malicious or abusive activity inside the network) or intrusion (breaches from the outside). Host-based IDSes consult several types of log files (kernel, system, server, network, firewall, and more), and compare the logs against an internal database of common signatures for known attacks.

A host-based IDS can also verify the data integrity of important files and executables. It checks a database of sensitive files (and any files added by the administrator) and creates a *checksum* of each file with a message-file digest utility such as md5sum (128-bit algorithm) or sha1sum (160-bit algorithm). The host-based IDS then stores the sums in a plain text file and periodically compares the file checksums against the values in the text file. If any of the file checksums do not match, the IDS alerts the administrator by email or cellular pager.

### 9.1 AIDE

aide is an host based intrusion detection system for checking the integrity of files.

#### 9.1.1 Install AIDE

To install aide on Ubuntu 12.04, type the following command:

```
# apt-get update && apt-get install aide
```

You need to customize `/etc/aide/aide.conf` to meet your requirements. The default configuration is acceptable for many environments.

`/etc/aide/aide.conf`, `/etc/aide/aide.conf.d`: Default AIDE configuration files

`/var/lib/aide/aide.db`: Default location for AIDE database

`/var/lib/aide/aide.db.new`: Default location for newly-created AIDE database

## 9.1.2 Build Store and Test a Database.

`aideinit` creates a new AIDE database. It will initialize an AIDE database in the default `database_out` location. It will then prompt you to replace your existing AIDE database. In most cases you will want to check for any problems before doing this.

`aideinit` attempts to automatically detect the correct locations of your database and `database_out` files based on your `aide.conf` settings. These settings may be overridden on the command line, as may the prompts.

```
root@shire:~# aideinit
Overwrite existing /var/lib/aide/aide.db.new [Yn]? Y
Running aide --init...
```

Once the database is initialized, the location of the database is displayed as follows.

```
AIDE, version 0.15.1

### AIDE database at /var/lib/aide/aide.db.new initialized.

Overwrite /var/lib/aide/aide.db [yN]? You have new mail in /var/mail/root
root@shire:~# █
```

Finally, install the newly-generated database, enter:

```
root@shire:~# cp /var/lib/aide/aide.db.new /var/lib/aide/aide.db
You have new mail in /var/mail/root
root@shire:~# █
```

Next, run a manual check:

```
root@shire:~# aide -c /etc/aide/aide.conf --check
```

If this check produces any unexpected output, investigate. You need to move the database, as well as the configuration file `/etc/aide/aide.conf` and the `aide` binary to a secure offsite readonly location. This should be done to improve overall security. If attacker can modify the binary then you would not spot anything, so move it out or burn the files to the CD-ROM and use that for the checking.

You can also use hashes of these files. Move files to offsite server.

```
# scp /var/lib/aide/aide.db* /usr/bin/aide /etc/aide/aide.conf /etc/aide/aide.conf.d/*
```

user@offsite.server.com:/path/to/dir

Use tools such as cdfrecord to write the files on CDROM.

### 9.1.3 Testing the integrity of the Binary

Run the command (note: usually you only need to run `aide -c /etc/aide/aide.conf --check`):

```
# touch /bin/date
```

```
# aide -c /etc/aide/aide.conf --check
```

### 9.1.4 Cron To Implement Periodic Execution of Integrity Checking

By default, AIDE install itself for periodic execution at `/etc/cron.daily/aide`. This script will get executed once a day, which may be suitable for many server environments. If there is any problem with installed binaries (modified by you or a system update program such as `apt-get` or by an attacker), you will get an email (default sent to root user). You can customize email by editing

`/etc/default/aide` file. You need to set `MAILTO` variable. This is the email address reports get mailed.

```
MAILTO=vivek.cherian@csscorp.com
```

### 9.1.5 A Note About System Changes

AIDE mail may be an indication of an attack against your server. However, sometime you update system and configuration change or a software update. The following steps should be repeated when configuration changes or software updates necessitate:

```
# aideinit
```

```
# cp /var/lib/aide/aide.db.new /var/lib/aide/aide.db
```

```
# aide -c /etc/aide/aide.conf --check
```

Finally, move the files (database and aide binary) to readonly media or offsite server using `scp`

# Chapter 10

## RootKit Detectors

Most rootkits are self-hiding toolkits used by blackhats, crackers and script kiddies, to avoid the eye of the system administrator. It is often a collection of tools (programs) that enable administrator level access to a computer or computer network. The term *rootkit* is a concatenation of “root” (the traditional name of the privileged account on Unix operating systems) and the word "kit" (which refers to the software components that implement the tool).

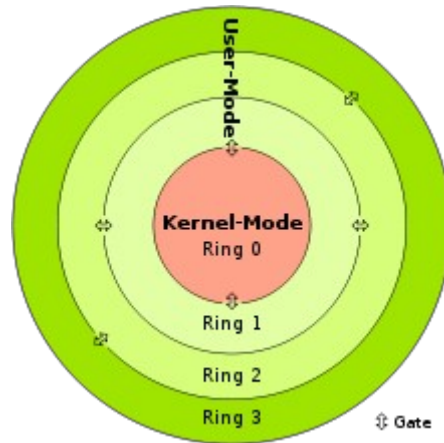
Typically a cracker gains user level access to instance after first obtaining user-level access, either by exploiting a known vulnerability or cracking a password and then performs privilege escalation.

Typically a rootkit may consist of spyware and other programs that monitor traffic and keystrokes and creates backdoors into the system for further and repeated exploitation by the cracker, alter log files, attack other systems on the network and alter existing system programs to evade detection.

## Types of Rootkits

Rootkits can be classified into at least 5 major types

- 1) User Mode Rootkits
- 2) Kernel Mode Rootkits
- 3) Hypervisor Level Rootkits
- 4) Firmware Level Rootkits
- 5) Bootkits.



**The ring model of rootkit classification** (Image courtesy: Wikipedia)

## User-mode rootkits

User mode rootkits run with user privileges. They mostly work by altering or disrupting the behavior of APIs. Userland rootkits also modify processes, network connections, files and events. User mode rootkits are the least privileged of rootkits running often at Ring 3 in the ring terminology and run with only user level access.

Ring3 is where user applications run, and since hence even untrustworthy programs can run here, operating systems give this layer the least privilege that makes monitoring/detection and prevention easier as opposed to kernel rootkits.

Detection techniques in userland rootkits falls under few main categories such as heuristics based, anomaly based, signature based, cross-view based, etc.

Before a userland rootkit is installed on the system, the attacker would have already breached into the system and would have performed privilege escalation to obtain administrator privileges and finally install rootkit, which retains the root/admin privilege.

## Kernel mode rootkits

In the ring terminology, kernel rootkits run on ring0. Since, ring0 is under all the other layers, monitoring kernel rootkits is almost impossible if the monitoring software is run on the same system, except for certain detection techniques that compares the different states or interacts with kernel using different stimulus signals and compares the response. Though once again, since the kernel rootkit modifies the entire operating system properties, functions and anything and everything possible to deceive a user from noticing the compromise by completely hiding itself,



detection is quite hard in case of kernel land rootkits.

Kernel rootkits run in ring 0 and have the highest operating system privileges. Kernel rootkits have posed serious security threats due to their stealthy manner. To hide their presence and activities, many rootkits hijack control flows by modifying control data or hooks in the kernel space.

Kernel Rootkits typically work by hijacking a number of "hooks," or control data, in a computer's operating system. "By taking control of these hooks, the rootkit can intercept and manipulate the computer system's data at will, essentially letting the user see only what it wants the user to see. As a result, the rootkit can make itself invisible to the computer user and any Anti Virus software. Furthermore, the rootkit can install additional malware, such as programs designed to steal personal information, and make them invisible as well.

The only safe guard against a Kernel level rootkit is to install a customized and hardened kernel at install time itself in the form of choosing only those modules that are absolutely necessary and relying on building your own hardened linux from scratch.

## **Firmware Level rootkits**

Firmware is small static pieces of code that runs on devices ranging from consumer electronics to anything that controls heavy machinery. In computers, chips a.k.a. the real interactive hardware, such as the BIOS, CPU, GPU, etc. runs firmware code to perform specific functions. Network devices ranging from firewalls, routers, switches, IDS/IPS, etc. runs firmware a.k.a. microcode on their chipsets. Firmware is tiny and not modified often like the user land or kernel land software. Thus, integrity checks are very rarely done at this layer. A firmware rootkit has to go through kernel land to go to reach the firmware layer hence removal of firmware level root kits by an is virtually impossible.

Since firmwares are often proprietary and the source code of firmware is not available for open source review, the only option is to wait till the firmware manufacturer comes out with a security patch.

## **Hypervisor level rootkits**

By exploiting hardware virtualization features such as Intel-VT or AMD-V , this type of rootkit runs in Ring 1 and hosts the target operating system as a virtual machine, thereby enabling the rootkit to intercept hardware calls made by the original operating system. Unlike normal hyper visors, they do not have to load before the operating system, but can load into an operating system before promoting it into a virtual machine.

# Rootkit Scanners

We will learn about 2 common rootkits that are available in a Ubuntu Server 12.04 image namely rkhunter and chkrootkit.

## rkhunter

rkhunter is a shell script which carries out various checks on the local system to try and detect known rootkits and malware. It also performs checks to see if commands have been modified, if the system startup files have been modified, and various checks on the network interfaces, including checks for listening applications.

Two common options that are used with rkhunter is -c and --update

**The -c, --check option.**

```

root@shire:~# rkhunter -c
[ Rootkit Hunter version 1.3.8 ]

Checking system commands...

Performing 'strings' command checks
  Checking 'strings' command [ OK ]

Performing 'shared libraries' checks
  Checking for preloading variables [ None found ]
  Checking for preloaded libraries [ None found ]
  Checking LD_LIBRARY_PATH variable [ Not found ]

Performing file properties checks
  Checking for prerequisites [ Warning ]
    /sbin/depmod [ OK ]
    /sbin/fsck [ OK ]
    /sbin/ifconfig [ OK ]
    /sbin/ifdown [ OK ]
    /sbin/ifup [ OK ]
    /sbin/init [ Warning ]
    /sbin/insmod [ OK ]
    /sbin/ip [ OK ]
    /sbin/lsmode [ OK ]
    /sbin/modinfo [ OK ]

```

This `command` option tells `rkhunter` to perform various checks on the local system. The result of each test will be displayed on `stdout`. If anything suspicious is found, then a warning will be displayed. A log file of the tests and the results will be automatically produced. It is suggested that this command option is run regularly in order to ensure that the system has not been compromised.

The output of the `rkhunter -c` command is quite verbose and it would be impractical to show the entire output in this manual. Broadly the `-c` checks system commands, system startup files. Possible rootkits, malware, hidden ports, checks `passwd` and `group` file changes for anomalies, `ssh` level checks and it also checks `syslog`.

## The `-update` option

This command option causes `rkhunter` to check if there is a later version of any of its text data files. A command-line web browser, for example `wget` or `lynx`, must be present on the system when using this option. It is suggested that this command option is run regularly in order to ensure that the data files are kept up to date.

```

root@shire:~# rkhunter --update
[ Rootkit Hunter version 1.3.8 ]

Checking rkhunter data files...
  Checking file mirrors.dat           [ No update ]
  Checking file programs_bad.dat      [ Updated   ]
  Checking file backdoorports.dat     [ No update ]
  Checking file suspscan.dat          [ No update ]
  Checking file i18n/cn               [ No update ]
  Checking file i18n/de               [ No update ]
  Checking file i18n/en               [ No update ]
  Checking file i18n/zh               [ No update ]
  Checking file i18n/zh.utf8          [ No update ]
root@shire:~# █

```

## chkrootkit

chkrootkit is a tool to locally check for signs of a rootkit.

Type the following command to install chkrootkit

```
$ sudo apt-get install chkrootkit
```

Start looking for rootkits, enter:

```
$ sudo chkrootkit
```

```

root@shire:~# chkrootkit
ROOTDIR is '/'
Checking `amd'...           not found
Checking `basename'...     not infected
Checking `biff'...         not found
Checking `chfn'...         not infected
Checking `chsh'...         not infected
Checking `cron'...         not infected
Checking `crontab'...      not infected
Checking `date'...         not infected
Checking `du'...           not infected
Checking `dirname'...      not infected
Checking `echo'...         not infected
Checking `egrep'...        not infected
Checking `env'...          not infected

```

Look for suspicious strings, enter:

```
$ sudo chkrootkit -x | less
```

You need to specify the path for the external commands used by chkrootkit such as awk, grep and others. Mount /mnt/safe using nfs in read-only mode and set /mnt/safe binaries PATH as trusted one, enter:

```
$ sudo chkrootkit -p /mnt/safe
```

# Chapter 11

## Linux Kernel Hardening

The 'linux' kernel can be configured with certain advanced features that can prevent certain kinds of attacks.

The Linux kernel provides for a multiuser operating system. Since it is a multiuser operating system, the kernel also provides mechanisms for user authentication and access control to prevent unauthorized intervention with other users or applications. The access control built into the Linux kernel is based on the traditional Discretionary Access Control (DAC) model.

### Hardening the Linux kernel with Grsecurity

An administrator who does not wish to migrate their Linux hosts to a new distribution in order to take advantage of their hardened kernels will need to look at building a custom kernel.

An administrator wishing to secure the address space on a Linux box in addition to replacing discretionary access control might not have the time necessary to try out all the combinations of patches to find what works best. The Grsecurity kernel patch is a more comprehensive patch that addresses the need for security enhancements in both areas.

### Downloading grsecurity

Point your browser to <http://grsecurity.net/>. Click on the “Download” link and then “Stable”. For the purposes of this document, we will be installing the latest stable grsecurity for kernel 3.2.33. Therefore the patch file will be called “grsecurity-2.9.1-3.2.33-201206201812.patch”.

### Downloading the Linux Kernel

The grsecurity patches can only be applied to a vanilla kernel. Many distributions modify the official kernel with additional patches, which means that any kernel source packages acquired through their package manager is very likely incompatible with grsecurity.

For this reason we will download the official unmodified kernel from <http://www.kernel.org/>.

Download the full kernel source package and its signature (the “.sign” file), and make sure its version matches the version of the grsecurity patch you downloaded. In this document the version is 3.2.21. The required version is most likely not the latest, so you need to get it from the linux kernel archives at [kernel.org](http://kernel.org)

## Patching Your Kernel with grsecurity

In this document the compressed kernel source package is called “linux-3.2.23.tar” and the matching grsecurity patch “grsecurity-2.9.1-3.2.33-201206201812.patch”. Both files are in the same directory.

Change to the root user and run the following commands in the directory you downloaded the files to. The first command decompresses the Linux source package, and the second one applies the patch to the kernel.

```
# tar -xf linux-3.2.33.tar
# cd linux-3.2.21
# patch -p1 < ../grsecurity-2.9.1-3.2.33-201206201812.patch
```

## Configuring the Kernel

The kernel source package contains a generic configuration file that should work without any significant modifications. Your distribution may have its own process and tools for configuring and building the kernel, in which case you should consult their documentation. Nonetheless you should go through the options and make sure they match your hardware and current setup.

To configure the kernel using the default configuration as a base, change into the kernel source directory (e.g. “/usr/src/linux-3.2.33”), and execute the below command.

```
# make menuconfig
```

```

root@shire:/usr/src# ls
linux-3.2.33                linux-headers-3.2.0-27      linux-headers-3.2.0-32-generic
linux-headers-3.2.0-23      linux-headers-3.2.0-27-generic linux-headers-3.2.0-33
linux-headers-3.2.0-23-generic linux-headers-3.2.0-29      linux-headers-3.2.0-33-generic
linux-headers-3.2.0-25      linux-headers-3.2.0-29-generic virtualbox-4.1.12
linux-headers-3.2.0-25-generic linux-headers-3.2.0-31      virtualbox-guest-4.1.12
linux-headers-3.2.0-26      linux-headers-3.2.0-31-generic virtualbox-guest.tar.bz2
linux-headers-3.2.0-26-generic linux-headers-3.2.0-32      virtualbox.tar.bz2
root@shire:/usr/src# cd linux-3.2.33/
root@shire:/usr/src/linux-3.2.33# ls
arch      Documentation  include  kernel      mm          scripts    virt
block     drivers        init     lib          net         security
COPYING   firmware       ipc      localversion-grsec README      sound
CREDITS   fs             Kbuild   MAINTAINERS  README     tools
crypto    grsecurity     Kconfig  Makefile     samples    usr
root@shire:/usr/src/linux-3.2.33# make menuconfig

```

## .config - Linux/x86\_64 3.2.33 Kernel Configuration

Linux/x86\_64 3.2.33 Kernel Configuration

Arrow keys navigate the menu. <Enter> selects submenus --->. Highlighted letters are hotkeys. Pressing <Y> includes, <N> excludes, <M> modularizes features. Press <Esc><Esc> to exit, <?> for Help, </> for Search. Legend: [\*] built-in [ ] excluded <M> module < > module capable

```

[*] DMA memory allocation support
    General setup --->
[*] Enable loadable module support --->
[*] Enable the block layer --->
    Processor type and features --->
    Power management and ACPI options --->
    Bus options (PCI etc.) --->
    Executable file formats / Emulations --->
-* Networking support --->
    Device Drivers --->
    Firmware Drivers --->

```

v(+)

<Select> < Exit > < Help >

```

Processor type and features --->
Power management and ACPI options --->
Bus options (PCI etc.) --->
Executable file formats / Emulations --->
-* Networking support --->
Device Drivers --->
Firmware Drivers --->
File systems --->
Kernel hacking --->
[*] Security options --->
-* Cryptographic API --->

```

Grsecurity

Arrow keys navigate the menu. <Enter> selects submenus --->. Highlighted letters are hotkeys. Pressing <Y> includes, <N> excludes, <M> modularizes features. Press <Esc><Esc> to exit, <?> for Help, </> for Search. Legend: [\*] built-in [ ] excluded <M> module < > module capable

```

[*] Grsecurity
    Configuration Method (Automatic) --->
    Usage Type (Server) --->
    Virtualization Type (None) --->
    Required Priorities (Performance) --->
    Default Special Groups --->
    Customize Configuration --->

```

<Select>    < Exit >    < Help >

```

PaX --->
Memory Protections --->
Role Based Access Control Options --->
Filesystem Protections --->
Kernel Auditing --->
Executable Protections --->
Network Protections --->
Sysctl Support --->
Logging Options --->

```



☒ Enable various PaX features

PaX Control --->  
Non-executable pages --->  
Address Space Layout Randomization --->  
Miscellaneous hardening features --->

☒ Enable various PaX features

☒ PaX Control --->

Non-executable pages --->  
Address Space Layout Randomization --->  
Miscellaneous hardening features --->

☐ Support soft mode

☒ Use legacy ELF header marking  
☒ Use ELF program header marking  
☒ Use filesystem extended attributes marking  
MAC system integration (direct) --->

☒ Enable various PaX features

PaX Control --->

☒ Non-executable pages --->

Address Space Layout Randomization --->  
Miscellaneous hardening features --->

☒ Enforce non-executable pages

☒ Paging based non-executable pages  
☐ Emulate trampolines  
☒ Restrict mprotect()  
☐ Use legacy/compat protection demoting (read help)  
☐ Allow ELF text relocations (read help)

```
[*] Enable various PaX features
    PaX Control --->
    Non-executable pages --->
    Address Space Layout Randomization --->
    Miscellaneous hardening features --->
```

```
[*] Address Space Layout Randomization
[*] Randomize kernel stack base
[*] Randomize user stack base
[*] Randomize mmap() base
```

```
[*] Enable various PaX features
    PaX Control --->
    Non-executable pages --->
    Address Space Layout Randomization --->
    Miscellaneous hardening features --->
```

```
[ ] Sanitize kernel stack
[*] Prevent various kernel object reference counter overflows
[*] Harden heap object copies between kernel and userland
[*] Prevent various integer overflows in function size parameters
[*] Generate some entropy during boot
```

```
PaX --->
Memory Protections --->
Role Based Access Control Options --->
Filesystem Protections --->
Kernel Auditing --->
Executable Protections --->
Network Protections --->
Sysctl Support --->
Logging Options --->
```

```
☒ Deny reading/writing to /dev/kmem, /dev/mem, and /dev/port
[*] Disable privileged I/O
[*] Harden ASLR against information leaks and entropy reduction
[*] Deter exploit bruteforcing
[*] Harden module auto-loading
[*] Hide kernel symbols
[*] Active kernel exploit response
```

```
PaX --->
Memory Protections --->
☒ Role Based Access Control Options --->
Filesystem Protections --->
Kernel Auditing --->
Executable Protections --->
Network Protections --->
Sysctl Support --->
Logging Options --->
```

```
☐ Disable RBAC system
[ ] Hide kernel processes
(3) Maximum tries before password lockout
(30) Time to wait after max password tries, in seconds
```

```
PaX --->
Memory Protections --->
Role Based Access Control Options --->
☒ Filesystem Protections --->
Kernel Auditing --->
Executable Protections --->
Network Protections --->
Sysctl Support --->
Logging Options --->
```

#### **[\*] Proc restrictions**

```
[ ] Restrict /proc to user only
[*] Allow special group
(1001) GID for special group
[*] Additional restrictions
[*] Linking restrictions
[*] Kernel-enforced SymlinksIfOwnerMatch
(1006) GID for users with kernel-enforced SymlinksIfOwnerMatch
[*] FIFO restrictions
[*] Sysfs/debugfs restriction
[ ] Runtime read-only mount protection
```

#### **[\*] Chroot jail restrictions**

```
[*] Deny mounts
[*] Deny double-chroots
[*] Deny pivot_root in chroot
[*] Enforce chdir("/") on all chroots
[*] Deny (f)chmod +s
[*] Deny fchdir out of chroot
[*] Deny mknod
[*] Deny shmat() out of chroot
[*] Deny access to abstract AF_UNIX sockets out of chroot
[*] Protect outside processes
```

```
[*] Protect outside processes
[*] Restrict priority changes
[*] Deny sysctl writes
[*] Capability restrictions
```

```
PaX --->
Memory Protections --->
Role Based Access Control Options --->
Filesystem Protections --->
Kernel Auditing --->
Executable Protections --->
Network Protections --->
Sysctl Support --->
Logging Options --->
```

```
[ ] Single group for auditing
[ ] Exec logging
[*] Resource logging
[ ] Log execs within chroot
[ ] Ptrace logging
[ ] Chdir logging
[ ] (Un)Mount logging
[*] Signal logging
[ ] Fork failure logging
[*] Time change logging
[*] /proc/<pid>/ipaddr support
```

```
[*] Resource logging
[ ] Log execs within chroot
[ ] Ptrace logging
[ ] Chdir logging
[ ] (Un)Mount logging
[*] Signal logging
[ ] Fork failure logging
[*] Time change logging
[*] /proc/<pid>/ipaddr support
[*] Denied RWX mmap/mprotect logging
[ ] ELF text relocations logging (READ HELP)
```

```
PaX --->
Memory Protections --->
Role Based Access Control Options --->
Filesystem Protections --->
Kernel Auditing --->
Executable Protections --->
Network Protections --->
Sysctl Support --->
Logging Options --->
```

**[\*] Dmesg(8) restriction**

- [\*] Deter ptrace-based process snooping
- [\*] Require read access to ptrace sensitive binaries
- [\*] Enforce consistent multithreaded privileges
- [\*] Trusted Path Execution (TPE)
- [ ] Partially restrict all non-root users
- [ ] Invert GID option
- (1005) GID for TPE-untrusted users

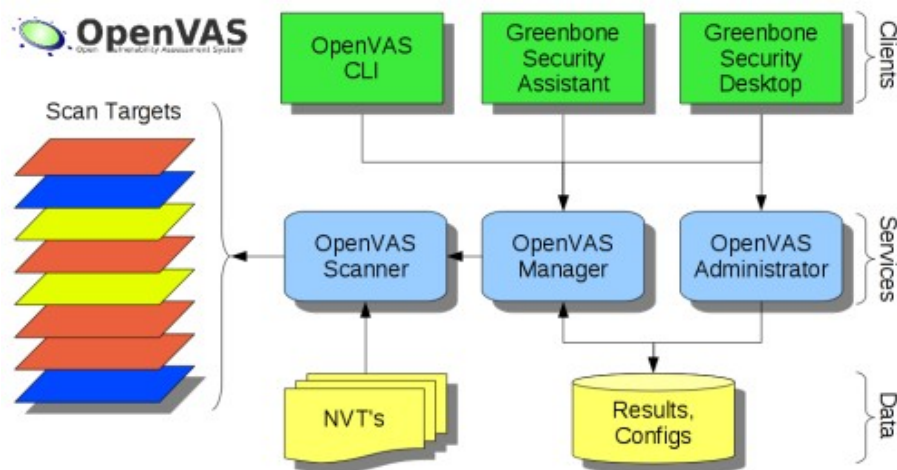
## Chapter 12

# Vulnerability Scanners

### OpenVAS

OpenVAS is a framework of several services and tools offering vulnerability scanning and vulnerability management solution. OpenVas also contains a daily updated feed of Network Vulnerability Tests (NVTs).

OpenVAS contains the following major components



### OpenVAS Scanner

The core of this SSL-secured service-oriented architecture is the OpenVAS Scanner. The scanner executes the actual Network Vulnerability Tests (NVTs) which are served with daily updates via the OpenVAS NVT Feed or via a commercial feed service.

### OpenVAS Manager

The OpenVAS Manager is the central service that consolidates plain vulnerability scanning into a full vulnerability management solution. The Manager controls the Scanner via OTP (OpenVAS Transfer Protocol) and itself offers the XML-based, stateless OpenVAS Management Protocol (OMP). All intelligence is implemented in the Manager so that it is possible to implement various

lean clients that will behave consistently e.g. with regard to filtering or sorting scan results. The Manager also controls a SQL database (sqlite-based) where all configuration and scan result data is centrally stored.

## OpenVAS Administrator

The OpenVAS Administrator acts as a command line tool or as a full service daemon offering the OpenVAS Administration Protocol (OAP). The most important tasks are the user management and feed management.

## OpenVAS Installation on Ubuntu

The steps to install OpenVAS from the openSUSE Build Service(OBS) for Ubuntu 12.04 is as follows.

### Step 1: Configure OBS Repository

```
sudo apt-get -y install python-software-properties
sudo add-apt-repository "deb
http://download.opensuse.org/repositories/security:/OpenVAS:/UNSTABLE:/v5/xUbuntu_12.04/ ."
sudo apt-key adv --keyserver hkp://keys.gnupg.net --recv-keys BED1E87979EAFD54
sudo apt-get update
```

### Step 2: Quick-Install OpenVAS

```
sudo apt-get -y install greenbone-security-assistant gsd openvas-cli openvas-manager openvas-scanner openvas-
administrator sqlite3 xsltproc
```

### Step 3: Quick-Start OpenVAS

(copy and paste whole block, during first time you will be asked to set a password for user "admin")

```
test -e /var/lib/openvas/CA/cacert.pem || sudo openvas-mkcert -q
sudo openvas-nvt-sync
test -e /var/lib/openvas/users/om || sudo openvas-mkcert-client -n om -i
sudo /etc/init.d/openvas-manager stop
sudo /etc/init.d/openvas-scanner stop
sudo openvassd
sudo openvasmd --migrate
sudo openvasmd --rebuild
sudo killall openvassd
sleep 15
sudo /etc/init.d/openvas-scanner start
sudo /etc/init.d/openvas-manager start
sudo /etc/init.d/openvas-administrator restart
sudo /etc/init.d/greenbone-security-assistant restart
test -e /var/lib/openvas/users/admin || sudo openvasad -c add_user -n admin -r Admin
```

### Step 4: Log into OpenVAS as "admin"

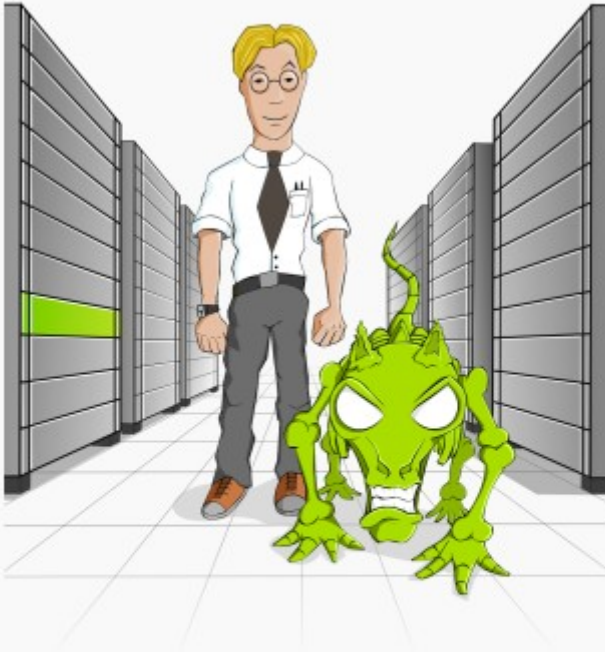
Open <https://localhost:9392/> or start "gsd" on a command line as a regular user (not as root!).

## Running OpenVAS Vulnerability Scans

Open <https://localhost:9392/> in a web browser and you will be greeted with the login screen, enter your admin username and password.



**Greenbone Security Assistant**



Username

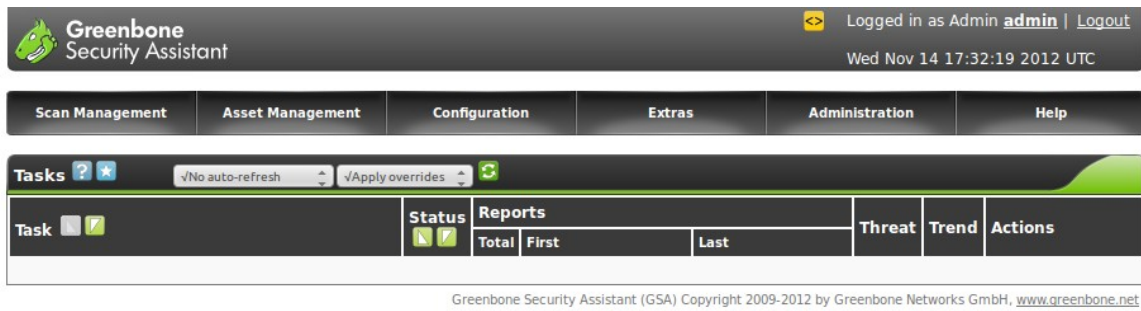
admin

Password

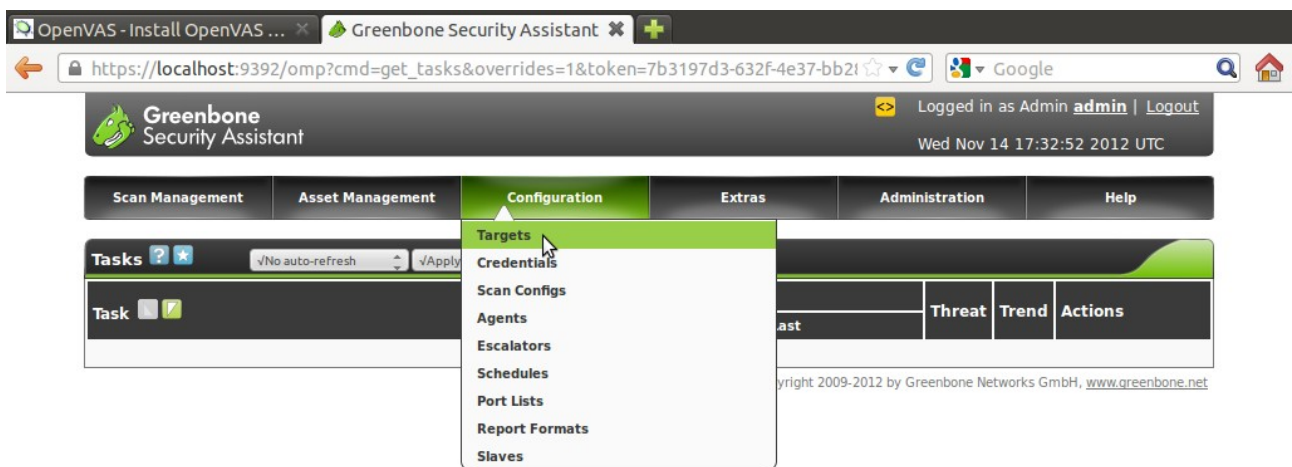
.....

Login

Once you login, you will be greeted with the Greenbone Security Assistant screen as the one shown below.




The next step is to configure the target we want to scan. For this purpose choose the Targets option under the Configuration menu as follows.



[https://localhost:9392/omp?cmd=get\\_targets&token=7b3197d3-632f-4e37-bb28-de3d047b4dee](https://localhost:9392/omp?cmd=get_targets&token=7b3197d3-632f-4e37-bb28-de3d047b4dee)

You can specify a new target by choosing the new target name, the hosts to be scanned can be specified by choosing the manual option and providing the I.P Address of the machine

which needs to be scanned. There is also a available list of ports that OpenVAS will scan, these ports can be specified by choosing the appropriate entry in the drop down menu that appears to the left of the Port List option.

 **Greenbone**  
Security Assistant

Logged in as Admin [admin](#) | [Logout](#)  
Wed Nov 14 17:37:30 2012 UTC

Scan Management

Asset Management

Configuration

Extras

Administration

Help

New Target ?

Name

unnamed

Hosts

☒ Manual

localhost

☐ From file

Browse...

Comment (optional)

Port List

All IANA assigned TCP 2012-02-10

SSH Credential (optional)



on port 

22

SMB Credential (optional)

Create Target


Targets ?


Name	Hosts	IPs	Port List	SSH Credential	SMB Credential	Actions
Localhost	localhost	1	<a href="#">OpenVAS Default</a>			 

Greenbone Security Assistant (GSA) Copyright 2009-2012 by Greenbone Networks GmbH, [www.greenbone.net](http://www.greenbone.net)

In our case, we are choosing a target with the hostname 'shire' having the I.P Address 10.10.27.19 and we prefer to scan the target via the OpenVAS Default port list.

The Targets which are already specified can be viewed under the Targets tab. Here you can view a scan on localhost using the OpenVAS Default port list.


**Greenbone**  
 Security Assistant
 


 Logged in as Admin **admin** | [Logout](#)  
 Thu Nov 15 08:12:06 2012 UTC

Scan Management
 Asset Management
 Configuration
 Extras
 Administration
 Help

**New Target ?**

Name 
  
 Hosts
 

☒ Manual 
  
☐ From file

  
 Comment (optional) 
  
 Port List 



OpenVAS Default

  
 SSH Credential (optional) 

on port




  
 SMB Credential (optional)

**Targets ?**

Name	Hosts	IPs	Port List	SSH Credential	SMB Credential	Actions
Localhost	localhost	1	<a href="#">OpenVAS Default</a>			 

Greenbone Security Assistant (GSA) Copyright 2009-2012 by Greenbone Networks GmbH, [www.greenbone.net](http://www.greenbone.net)

Once we specify our new target, it appears in the Targets tab with the Host name, I.P Address and Port List.

Targets ?						
Name	Hosts	IPs	Port List	SSH Credential	SMB Credential	Actions
Localhost	localhost	1	<a href="#">OpenVAS Default</a>			 
shire (Scan on shire)	10.10.27.19	1	<a href="#">OpenVAS Default</a>			 

Greenbone Security Assistant (GSA) Copyright 2009-2012 by Greenbone Networks GmbH, [www.greenbone.net](http://www.greenbone.net)

After the new target is specified, we have to specify the vulnerability assessment tasks that have to be performed on the target. This can be done by choosing the 'New Task' option under the 'Scan

Management' drop down menu.

Greenbone Security Assistant - Mozilla Firefox

OpenVAS - Install OpenVAS ... Greenbone Security Assistant

https://localhost:9392/omp?cmd=get\_tasks&overrides=1&token=2091d0fc-80a7-48c4-b6fe

Greenbone Security Assistant

Logged in as Admin **admin** | Logout

Thu Nov 15 08:15:17 2012 UTC

Scan Management Asset Management Configuration Extras Administration Help

Tasks

New Task

Notes

Overrides

✓Apply overrides

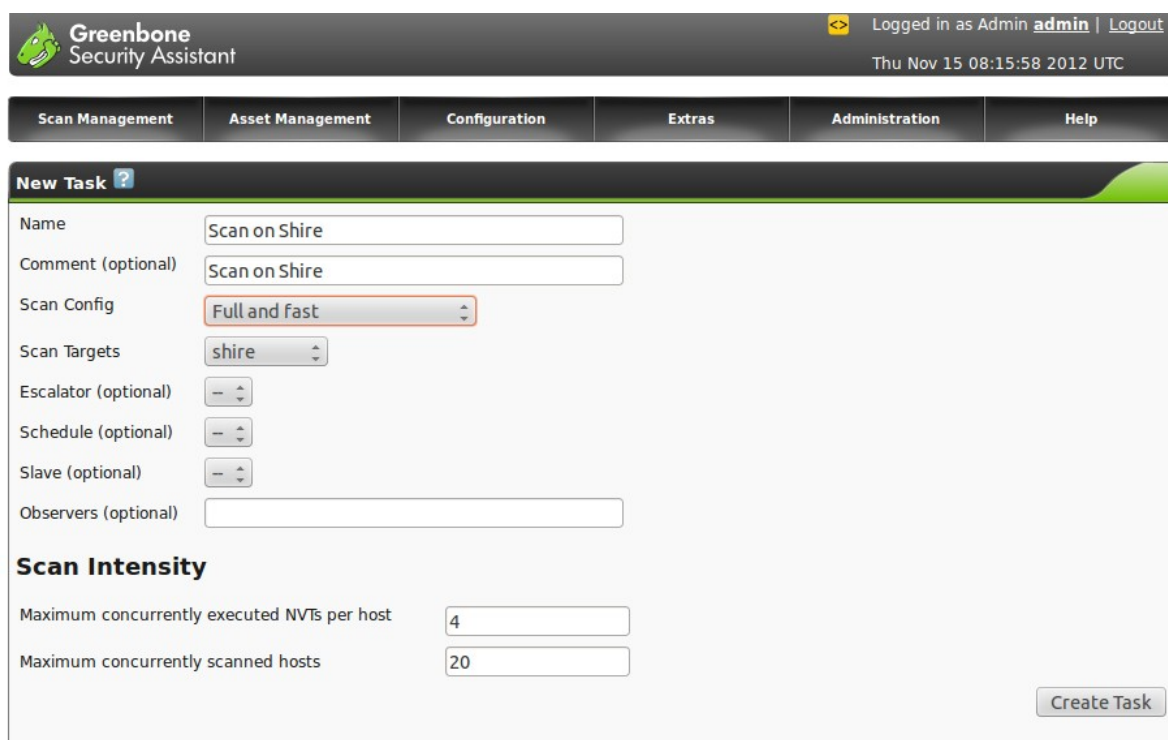
Status	Reports	Threat	Trend	Actions
Total	First	Last		

Greenbone Security Assistant (GSA) Copyright 2009-2012 by Greenbone Networks GmbH, [www.greenbone.net](http://www.greenbone.net)

https://localhost:9392/omp?cmd=new\_task&overrides=1&token=2091d0fc-80a7-48c4-b6fe-ae60b9aef99e

To set up a New Task, the 'Name' option should be chosen and the name of the Task should be specified. The 'comment' option is not mandatory, the 'Scan Config' option is chosen as 'Full and

Fast and the name of the target host is provided for the 'Scan Targets' option.



Greenbone Security Assistant

Logged in as Admin [admin](#) | [Logout](#)

Thu Nov 15 08:15:58 2012 UTC

Scan Management Asset Management Configuration Extras Administration Help

### New Task ?

Name:

Comment (optional):

Scan Config:

Scan Targets:

Escalator (optional):

Schedule (optional):

Slave (optional):

Observers (optional):

#### Scan Intensity

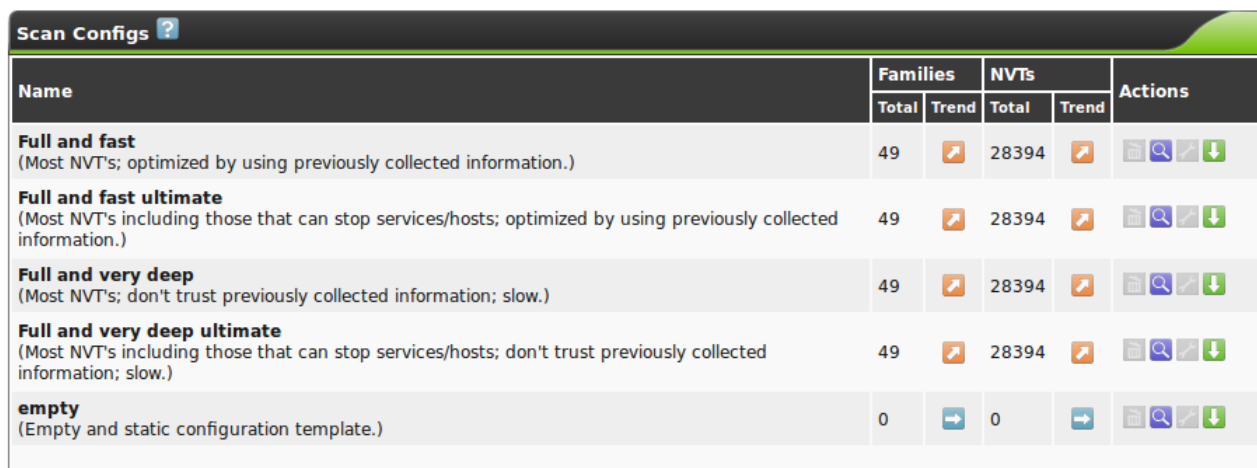
Maximum concurrently executed NVTs per host:

Maximum concurrently scanned hosts:

[Create Task](#)

The various types of Scan Configs on OpenVAS is as follows

- 1) Full and fast
- 2) Full and fast ultimate
- 3) Full and very deep
- 4) Full and very deep ultimate



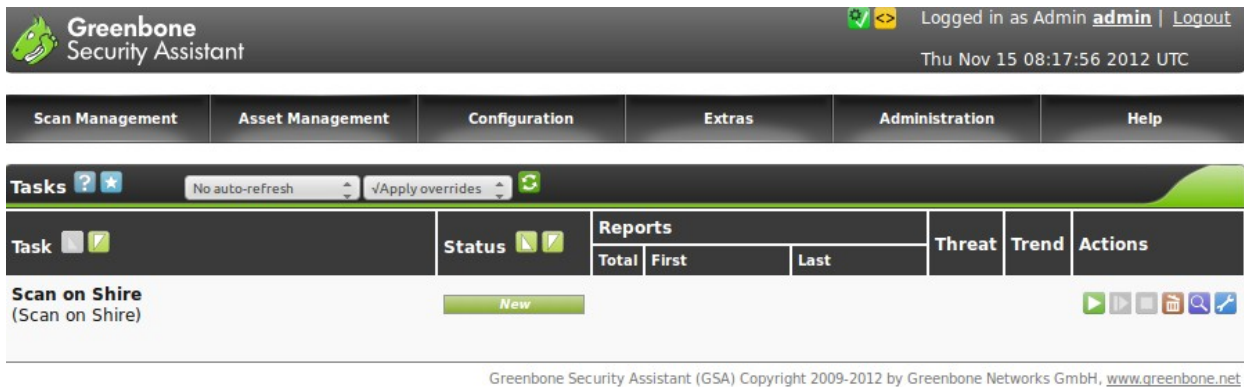
Name	Families		NVTs		Actions
	Total	Trend	Total	Trend	
<b>Full and fast</b> (Most NVTs; optimized by using previously collected information.)	49		28394		
<b>Full and fast ultimate</b> (Most NVTs including those that can stop services/hosts; optimized by using previously collected information.)	49		28394		
<b>Full and very deep</b> (Most NVTs; don't trust previously collected information; slow.)	49		28394		
<b>Full and very deep ultimate</b> (Most NVTs including those that can stop services/hosts; don't trust previously collected information; slow.)	49		28394		
<b>empty</b> (Empty and static configuration template.)	0		0		

Greenbone Security Assistant (GSA) Copyright 2009-2012 by Greenbone Networks GmbH, [www.greenbone.net](http://www.greenbone.net)

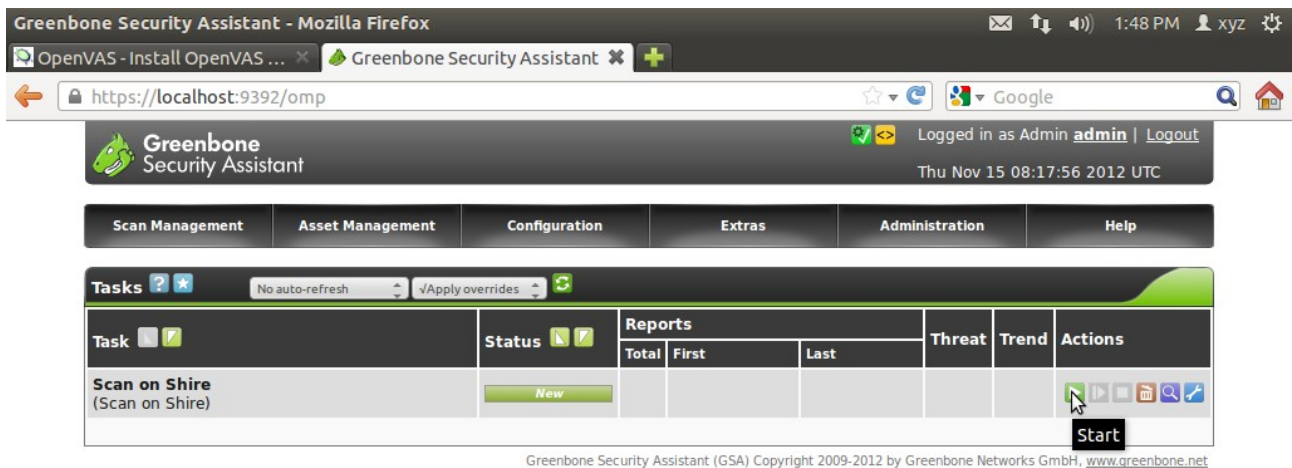
**Note:** NVTs stand for Network Vulnerability Tests. OpenVAS contains more than 25,000 NVTs growing on a daily basis and the latest NVTs can be downloaded periodically to update the NVT

list.

Once the task is chosen, it shows up on the Tasks screen.





To start the scan choose the green arrow symbol that points to the right under the 'Actions' section as shown below.



Once the scan is started, you will be taken to a screen where the 'Status' of the scan will be shown as requested.










**Greenbone**  
 Security Assistant
 


 Logged in as Admin [admin](#) | [Logout](#)  
 Thu Nov 15 08:18:36 2012 UTC


Scan Management
 Asset Management
 Configuration
 Extras
 Administration
 Help


**Tasks**
?
★
√No auto-refresh
√Apply overrides
↺

Task	Status	Reports			Threat	Trend	Actions
		Total	First	Last			
<b>Scan on Shire</b> (Scan on Shire)	<div>Requested</div>	0					     

Greenbone Security Assistant (GSA) Copyright 2009-2012 by Greenbone Networks GmbH, [www.greenbone.net](http://www.greenbone.net)







Once the scan is complete, the status of the scan will be shown as complete. In this case the 'Treat' section shows 'None' which means OpenVAS did not detect any treat while scanning the 'target' shire using the 'Full and Fast' Scan Configuration


**Greenbone**  
 Security Assistant
 


 Logged in as Admin [admin](#) | [Logout](#)  
 Thu Nov 15 08:30:34 2012 UTC

Scan Management
 Asset Management
 Configuration
 Extras
 Administration
 Help

**Tasks**
?
★
√No auto-refresh
√Apply overrides
↺


Task	Status	Reports			Threat	Trend	Actions
		Total	First	Last			
<b>Scan on Shire</b> (Scan on Shire)	<div>Done</div>	1		<a href="#">Nov 15 2012</a>	None		     

Greenbone Security Assistant (GSA) Copyright 2009-2012 by Greenbone Networks GmbH, [www.greenbone.net](http://www.greenbone.net)

To view further details of the scan choose the link under 'Total' which again falls under 'Reports'.



You will be greeted by a screen as shown below.


**Greenbone**  
 Security Assistant
 
 Logged in as Admin **admin** | [Logout](#)  
 Thu Nov 15 08:32:57 2012 UTC

Scan Management
 Asset Management
 Configuration
 Extras
 Administration
 Help

**Task Summary**
?
▶
▶
▶
▶
▶


**Name:** Scan on Shire [Tasks](#)  
**Comment:** Scan on Shire  
**Scan Config:** [Full and fast](#)  
**Escalator:**  
**Schedule:** (Next due: over)  
**Target:** [shire](#)  
**Slave:**  
**Status:** Done  
**Reports:** 1 (Finished: 1)  
**Observers:**

**Scan Intensity**  
 Maximum concurrently executed NVTs per host: 4  
 Maximum concurrently scanned hosts: 20

**Reports for "Scan on Shire"**
?
√Apply overrides
↺
↻

Report	Threat	Scan Results					Actions
		High	Medium	Low	Log	False Pos.	
<b>Thu Nov 15 08:18:36 2012</b> Done	None	0	0	0	7	0	<a href="#">A</a> <a href="#">Q</a> <a href="#">X</a>

The next screen shot shows a series of scans that I have conducted on the host 'Shire' which show that no threat has been discovered on the host.


**Greenbone**  
 Security Assistant
 
 Logged in as Admin **admin** | [Logout](#)  
 Thu Nov 15 10:28:40 2012 UTC

Scan Management
 Asset Management
 Configuration
 Extras
 Administration
 Help

**Tasks**
?
★
√No auto-refresh
√Apply overrides
↺
↻

Task	Status	Reports			Threat	Trend	Actions
		Total	First	Last			
<b>Full and fast Scan on Shire</b> (Full and Fast Scan on Shire)	Done	1		Nov 15 2012	None		<span>▶</span> <span>▶</span> <span>▶</span> <span>▶</span> <span>▶</span> <span>▶</span>
<b>Full and very deep ultimate scan on shire</b> (Full and very deep ultimate scan on shire)	50 %	0					<span>  </span> <span>▶</span> <span>▶</span> <span>▶</span> <span>▶</span> <span>▶</span>
<b>full and very deep scan on shire</b> (full and very deep scan on shire)	Done	1		Nov 15 2012	None		<span>▶</span> <span>▶</span> <span>▶</span> <span>▶</span> <span>▶</span> <span>▶</span>
<b>shire scan full and fast ultimate</b> (shire scan full and fast ultimate)	Done	1		Nov 15 2012	Low		<span>▶</span> <span>▶</span> <span>▶</span> <span>▶</span> <span>▶</span> <span>▶</span>

Greenbone Security Assistant (GSA) Copyright 2009-2012 by Greenbone Networks GmbH, [www.greenbone.net](http://www.greenbone.net)

# Chapter 13

## Security Analysis Tools

### Nmap

Nmap also known as “Network Mapper” is a tool for network analysis and security auditing. It can scan single hosts as well as entire network ranges. Nmap achieves this by using raw I.P packets in multiple ways to explore what hosts, services and its versions those hosts are running, operating system versions, firewalls they use and host of other characteristics. Though nmap is used mostly for security audits, it is also useful for routine maintenance tasks such as monitoring hosts, checking uptime, upgrade scheduling and also network inventory.

In addition to the interesting ports table, Nmap can provide further information on targets, including reverse DNS names, operating system guesses, device types, and MAC addresses.

**Disclaimer:** This information is provided to assist administrators to scan their own networks for which they have an explicit permission to perform such a scan. This is not intended for assisting remote scanning with an intention of cracking systems or breaking into networks or exploiting services or any illegal activities.

#### Common TCP Flags used by Nmap

**SYN (Synchronise) :** SYN packets include a TCP sequence number, which lets the remote system know what sequence numbers to expect in subsequent communication.

**ACK (Acknowledge):** ACK acknowledges receipt of a packet or set of packets.

**FIN (Finished) :** FIN is sent when a communication is finished, requesting that the connection be closed,

**RST (Reset):** RST is sent when the connection is to be reset (closed immediately).

To initiate a TCP connection, the initiating system sends a SYN packet to the destination, which

will respond with a SYN of its own, and an ACK, acknowledging the receipt of the first packet (these are combined into a single SYN/ACK packet)

Various types of scans can be performed via Nmap, they are as follows,

### **TCP Connect Scanning**

TCP connect Scanning is performed using the **-sT** option. In this kind of scan, Nmap requests the operating system to establish a connection with the target machine and port by issuing the connect system call connect(). This scan is a basic port scan which connects to every port one after the other and tries to establish a connection. This scan provides an output which lists ports to which a connection could be established as 'open' and the ones to which connection failed as 'closed'.

Once the connect scan is complete, any port listed as open can be connected to. The drawback of this kind of scan is that systems being scanned can easily detect the scan. In case of a Firewall or IDS running on the victim machine, this will clearly trigger a warning and this scan will be logged by most modern firewalls. The source I.P Address and connection logs on the Firewall and IDS lead to easily detecting the source of such connect() scans.

To overcome the drawbacks of connect scan, the TCP Stealth Scan was developed.

### **TCP Stealth Scan**

The TCP Stealth Scan is performed by using the **-sS** option in Nmap, it also known as SYN scanning works by sending SYN packets and checking responses. If a SYN/ACK is sent back, it indicates an open port at the remote host trying to open a TCP connection. Nmap then sends an RST request to terminate the connection before it is established.

These days, most modern firewalls and IDS can even detect a SYN scan, but when used with various other combinations of nmap like altering timing and a few other options, a virtually undetectable SYN Scan can be created.

### **FIN, Null and Xmas Tree Scans**

There are 3 more TCP scan types FIN, Null and Xmas denoted by the **-sF**, **-sN**, **-sX** options respectively.

The FIN scan sends a packet with only the FIN flag set, the Xmas Tree scan sets the FIN, URG and PUSH flags and the Null scan sends a packet with no flags switched on.

These scans works on the principle that a closed port should respond with an RST upon receiving packets, whereas an open port should just drop them because they listen for packets with SYN set.

If the SYN scan shows open ports, and the FIN/NUL/XMAS does not, chances are you're looking at a Windows box since Microsoft Windows does not follow RFC 793

### **Ping Scan**

The Ping scan is denoted by the **-sP** option and is used to list the hosts within the specified range that responding to ping and shows which hosts are online.

This scan type lists the hosts within the specified range that responded to a ping. It allows you to detect which computers are online, rather than which ports are open.

## **Decoys**

nmap permits you to specify decoys while performing scans. The -D option is used to specify Decoys. The use of a Decoy scan makes it appear to the target that the host(s) provided as decoys are scanning. IDSs will report scans from different I.P Addresses but will often be unable to identify the real source of the scan from the suite of decoys.

Each Decoy host can be separated by a comma and the use of the term ME in the decoy host range will represent the position of your real I.P Address.

Decoys can be used both in the initial ping scan (using ICMP, SYN, ACK, or whatever) and during the actual port scanning phase. Decoys are also used for remote OS detection.

## **Fragmenting**

Fragmenting breaks down I.P Packets into tiny chunks and is specified by the -f option and is used to make it hard for a firewall or Packet Filter to identify the packet type. Fragmenting works with the SYN, FYN. NULL and XMAS scans.

## **UDP Scan**

The UDP scan is performed by the -sU option. In this can, Nmap sends 0-byte UDP packets to target ports on the the victim. If an ICMP Port Unreachable message is received, the port is closed else it is open.

## **IP Scans**

The IP Scan is performed by the -sO option. IP Scans attempt to determine the IP protocols supported on a target

## **Idle Scanning**

Idle scanning is denoted by the -sI option. Idle scan is considered to be an extremely stealthy scan performed via a zombie host so that packets sent to any victim cannot be traced back to origin.

## **Version Detection**

Version Detection denoted by the -sV option collects information about services running on an open port, which includes application name name and version number.

## **Ack Scan**

This scan is denoted by the -sA option and is performed to distinguish between stateful and stateless packer filtering. This scan sends ACK packets and if a RST response is obtained the the port is said

to be unfiltered. If no response is received, the port is assumed to be filtered.

## Nmap Scan Timing

This scan is denoted by the -T option and Nmap can be configured to adjust the scan timings according to various parameters like the response time of the victim, network connectivity etc. The timing can be used to control if a quick scan should be performed or a more detailed scan which consumes more time should be performed.

The timings are Paranoid(0), Sneaky(1), Polite(2), Normal(3), Aggressive(4) and Insane(5).

The first two are for IDS evasion. Polite mode slows down the scan to use less bandwidth and target machine resources. Normal mode is the default and so -T3 does nothing. Aggressive mode speeds scans up by making the assumption that you are on a reasonably fast and reliable network. Finally insane mode. assumes that you are on an extraordinarily fast network or are willing to sacrifice some accuracy for speed.

Each -T0 scan or Paranoid scan is generally performed after an interval of 5 mins. A gap of 5 minutes makes it undetectable for a firewall due to the large time interval between each scan. The timing of the scan will make it appear even in logs as routine network traffic and most traffic analysis tools and humans will miss it completely.

The -T5 scan will attempt a very fast scan and is generally less accurate than the other scans.

## Operating System Detection

A very important feature of nmap is the Operating System detection option specified by the -O option. The -O option along with the -v option provides you the following information.

Device Type, Operating System Running, Operating System Version, Uptime guess, Network distance, TCP sequence prediction, IP ID sequence generation.

# Tcpdump

Tcpdump is a utility for monitoring network traffic and data acquisition and can be compared to a swiss army knife for network troubleshooting.

## Various options of Tcpdump

**-D option** - The -D option lists the network interfaces available on the system on which tcpdump can capture packets.

**-n Option** – The -n option displays numerical I.P Addresses rather than symbolic I.P Addresses.

**-i option** – The -i option captures traffic of a particular interface.

Ex: tcpdump -i eth0 captures the traffic of eth0

**-c options** - The -c option specifies the number of packets to capture.

Ex: tcpdump -c 20 -i eth0, this command captures the first 20 packets from eth0.

**-w option** - The -w option option causes the output of the tcpdump to directed to a file for later analysis.

Ex: tcpdump -w tcpdumplog1.log causes the output of tcpdump to be re-directed to the file tcpdump.log

```
root@shire:~# tcpdump -i eth1 -w tcpdump.log
tcpdump: listening on eth1, link-type EN10MB (Ethernet), capture size 65535 bytes
^C182 packets captured
182 packets received by filter
0 packets dropped by kernel
root@shire:~#
```

**-r option** - The -r option is used to read the output from a tcpdump log file.

Ex: tcpdump -r tcpdump.log is used to read the content of the the tcpdum.log file

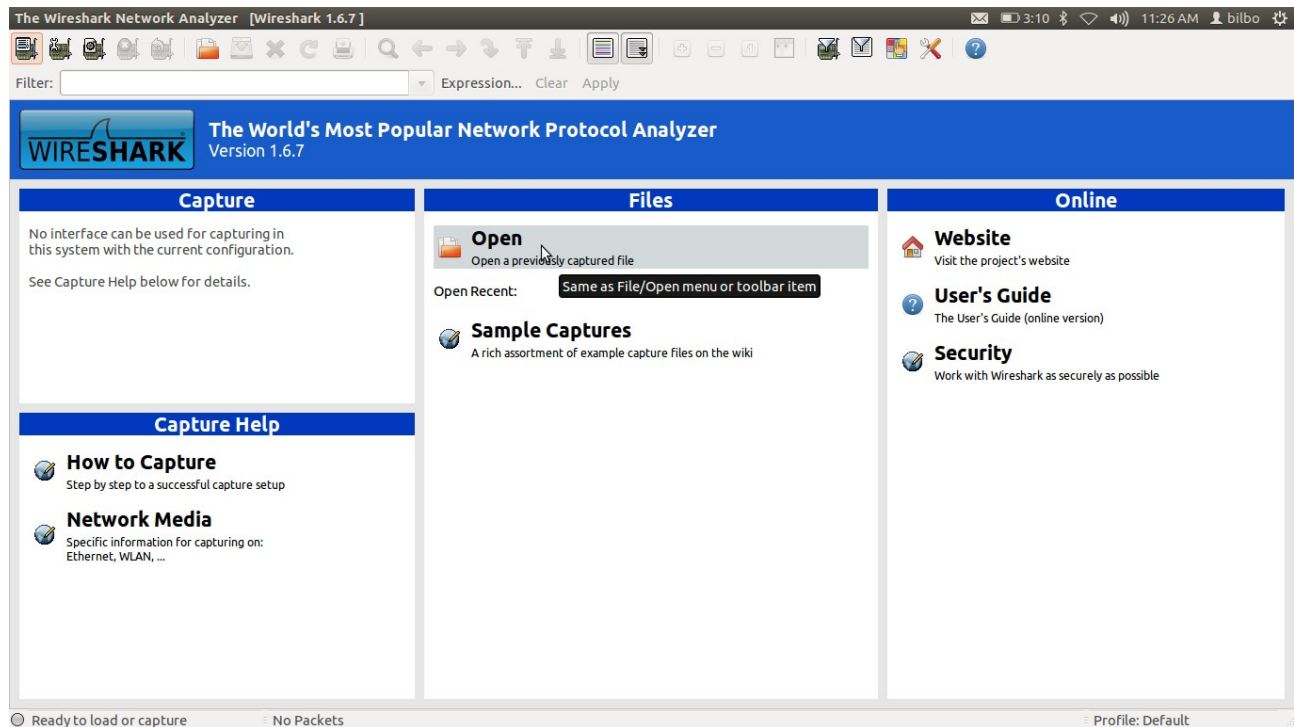
```
root@shire:~# tcpdump -r tcpdump.log
reading from file tcpdump.log, link-type EN10MB (Ethernet)
09:24:06.662462 IP shire.local.15980 > host-178-251-137-202.dssv.ru.12821: UDP, length 48
09:24:09.663001 IP shire.local.15980 > ip-212-117-183-170.as5577.net.14885: UDP, length 48
09:24:11.665933 ARP, Request who-has 192.168.1.1 tell shire.local, length 28
09:24:11.666387 ARP, Reply 192.168.1.1 is-at 00:08:5c:57:b4:81 (oui Unknown), length 46
09:24:12.663254 IP shire.local.15980 > li459-70.members.linode.com.18251: UDP, length 48
09:24:15.663642 IP shire.local.15980 > 60-240-86-246.static.tpgi.com.au.16317: UDP, length 48
09:24:16.869722 IP shire.local.15980 > 37-144-196-37.broadband.corbina.ru.11705: UDP, length 48
09:24:16.876412 IP shire.local.15980 > broadband-82-140-203-101.atc.tver.ru.65204: UDP, length 304
09:24:17.231966 IP broadband-82-140-203-101.atc.tver.ru.65204 > shire.local.15980: UDP, length 368
09:24:17.237341 IP shire.local.15980 > broadband-82-140-203-101.atc.tver.ru.65204: UDP, length 480
09:24:17.593134 IP broadband-82-140-203-101.atc.tver.ru.65204 > shire.local.15980: UDP, length 64
09:24:17.595402 IP broadband-82-140-203-101.atc.tver.ru.65204 > shire.local.15980: UDP, length 592
09:24:17.595601 IP shire.local.15980 > chello084115129021.5.graz.surfer.at.49322: UDP, length 80
09:24:17.596754 IP shire.local.15980 > broadband-82-140-203-101.atc.tver.ru.65204: UDP, length 592
09:24:17.596817 IP shire.local.15980 > broadband-82-140-203-101.atc.tver.ru.65204: UDP, length 4
```

## Wireshark

Wireshark is a GUI network protocol analyzer. A network packet analyzer will try to capture network packets and tries to display that packet data as detailed as possible. Wireshark lets you interactively browse packet data from a live network or from a previously saved capture file. Wireshark's native capture file format is libpcap format, which is also the format used by tcpdump and various other tools.

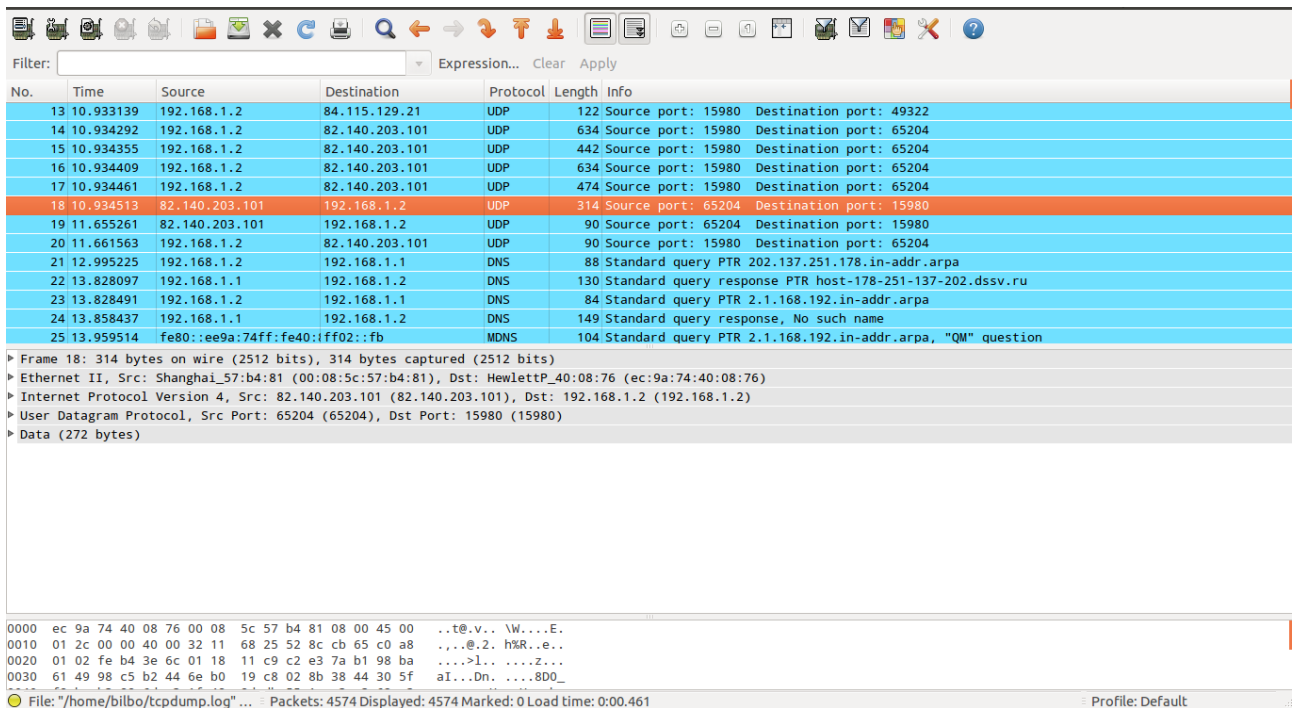
Wireshark also has a console version called tshark which can be used on servers which do have GUI.

To open the file tcpdump.log via wireshark, choose the 'Open' option shown in the screenshot below and specify the path of the tcpdump.log file.

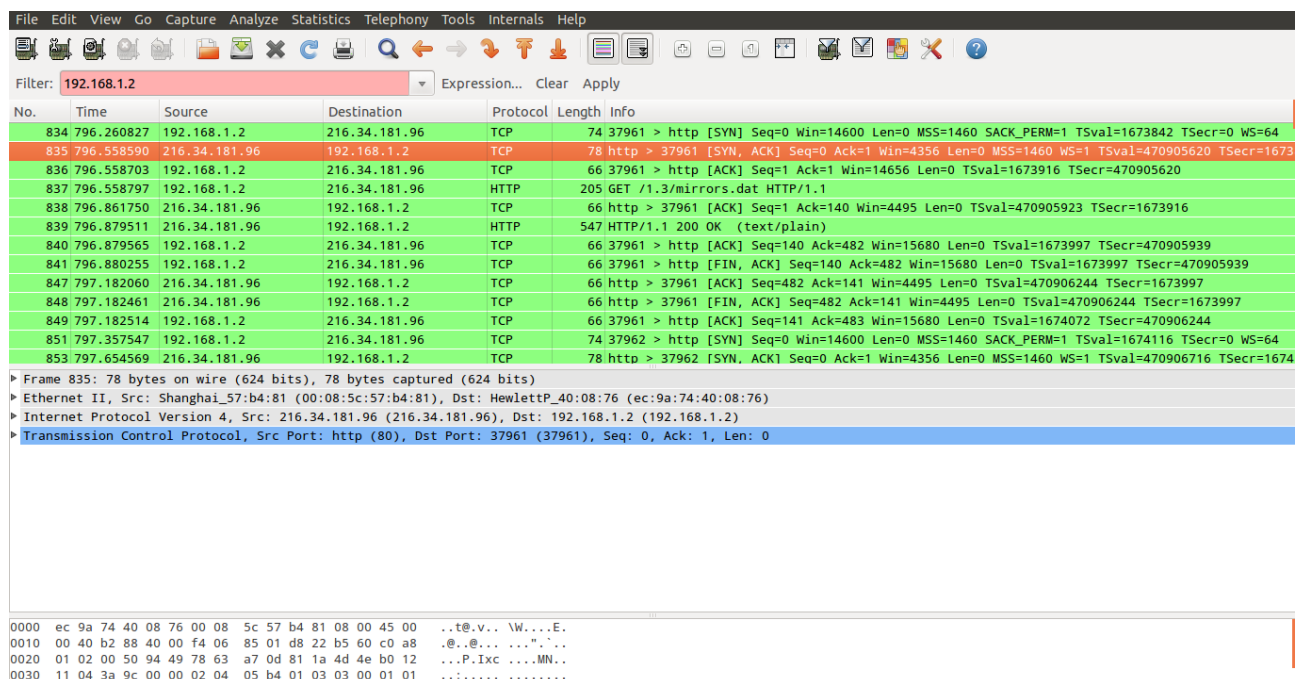


Once The tcpdump.log file is opened with wireshark, you can view the details of the transaction in a GUI.

Like other protocol analyzers, Wireshark's main window shows 3 views of a packet. It shows a summary line, briefly describing what the packet is. A packet details display is shown, allowing you to drill down to exact protocol or field that you interested in. Finally, a hex dump shows you exactly what the packet looks like when it goes over the wire.



In addition, Wireshark has some features that make it unique. It can assemble all the packets in a TCP conversation and show you the ASCII (or EBCDIC, or hex) data in that conversation. Display filters in Wireshark are very powerful; more fields are filterable in Wireshark than in other protocol analyzers, and the syntax you can use to create your filters is richer.





# Chapter 14

## OpenStack Security

In this chapter, we will try to explore the Security Groups concept on Ubuntu 12.04 Cloud Instance running on OpenStack

The entire steps to install and configure a single node OpenStack Essex setup is available at [http://cssoss.files.wordpress.com/2012/05/openstackbookv3-0\\_csscorp2.pdf](http://cssoss.files.wordpress.com/2012/05/openstackbookv3-0_csscorp2.pdf) . Once Openstack is setup log into the OpenStack Dashboard.

On the OpenStack Dashboard choose the 'Access and Security' option and then choose the 'Create Security Group' tab.

The screenshot shows the Ubuntu OpenStack Dashboard interface. The top header is orange with the Ubuntu logo and 'OpenStack Dashboard' text. On the right, it says 'Logged in as: vivekvc' with links for 'Settings' and 'Sign Out'. The left sidebar has a 'Project' dropdown set to 'Vivek Cherian'. Below this are sections for 'Manage Compute' (Overview, Instances & Volumes, Images & Snapshots), 'Access & Security' (highlighted), and 'Object Store' (Containers). The main content area is titled 'Access & Security' and contains three sections: 'Floating IPs' with an 'Allocate IP To Project' button and an empty table; 'Security Groups' with 'Create Security Group' and 'Delete Security Groups' buttons, and a table with one row 'default'; and 'Keypairs' with 'Create Keypair' and 'Import Keypair' buttons.

ubuntu<sup>®</sup> OpenStack Dashboard

Logged in as: vivekvc Settings Sign Out

Project

PROJECT Vivek Cherian

Manage Compute

- Overview
- Instances & Volumes
- Images & Snapshots
- Access & Security

Object Store

- Containers

### Access & Security

#### Floating IPs

Allocate IP To Project

<input type="checkbox"/>	IP Address	Instance	Floating IP Pool	Actions
No items to display.				
Displaying 0 items				

#### Security Groups

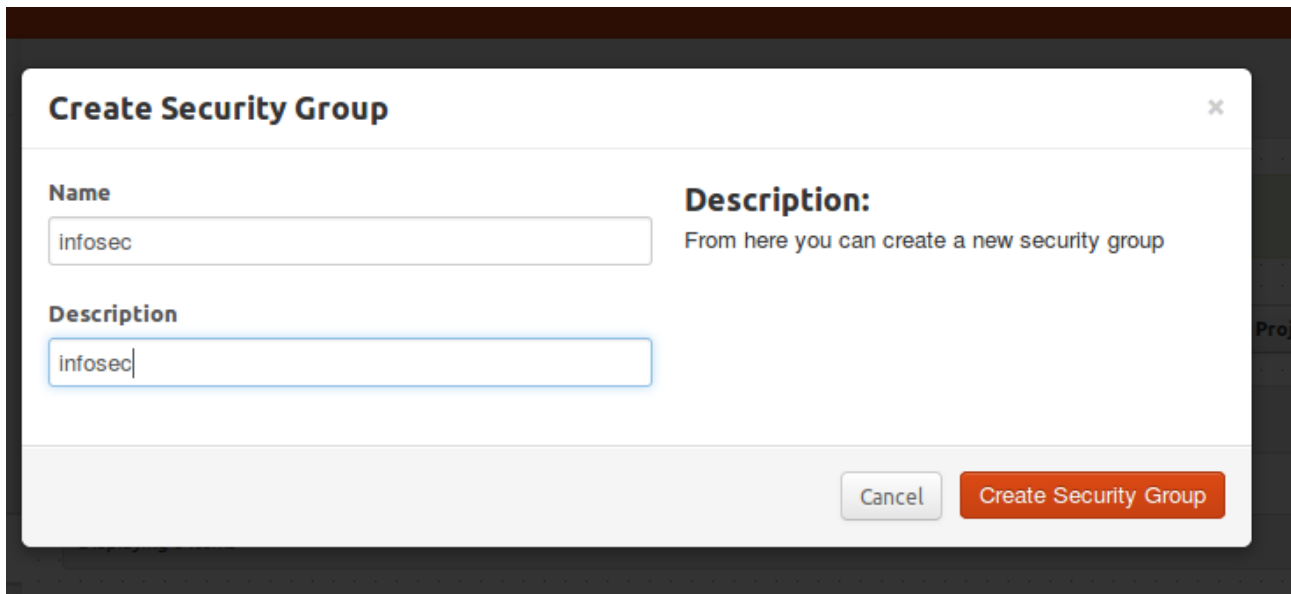
Create Security Group Delete Security Groups

<input type="checkbox"/>	Name	Description	Actions
<input type="checkbox"/>	default	default	Edit Rules
Displaying 1 item			

#### Keypairs

Create Keypair Import Keypair

On this screen you will be prompted for a name and description of the security group you wish to create. Once done choose the 'Create Security Group' option.

A modal dialog box titled "Create Security Group" with a close button (X) in the top right corner. It contains two input fields: "Name" with the value "infosec" and "Description" with the value "infosec". To the right of the description field, there is a text label "Description:" followed by the text "From here you can create a new security group". At the bottom right, there are two buttons: "Cancel" and "Create Security Group".

**Create Security Group** [X]

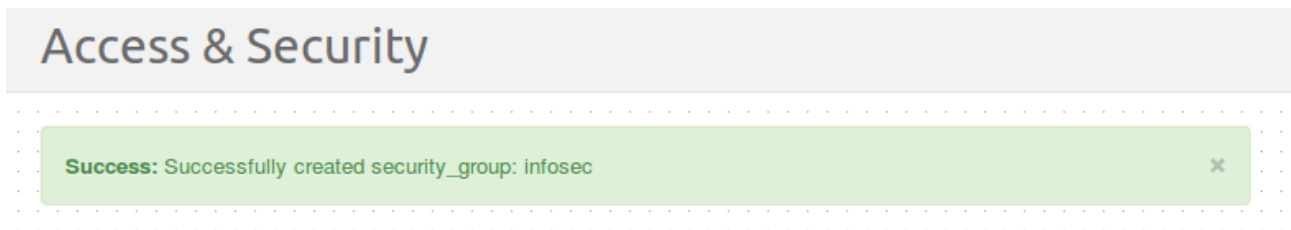
**Name**  
infosec

**Description:**  
From here you can create a new security group

**Description**  
infosec

Cancel Create Security Group

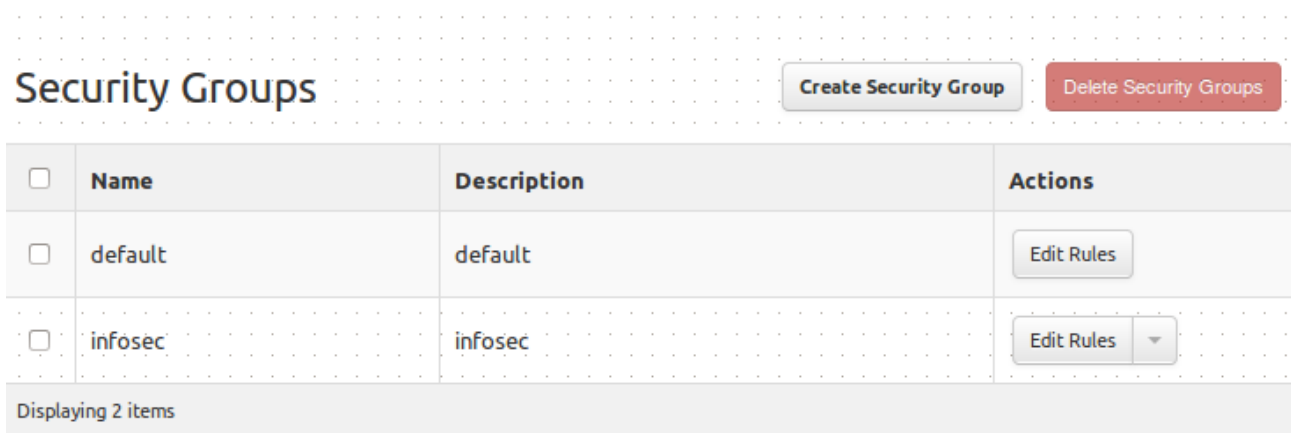
Once the Security group is created you will get the message as seen below.

A section titled "Access & Security" with a light gray background. Below the title is a green success message box with a close button (X) in the top right corner. The message text is "Success: Successfully created security\_group: infosec".

**Access & Security**

Success: Successfully created security\_group: infosec [X]

Your new security group will be displayed below the default security group.

A table titled "Security Groups" with two buttons at the top right: "Create Security Group" and "Delete Security Groups". The table has four columns: a checkbox column, "Name", "Description", and "Actions". It contains two rows of data: "default" and "infosec". The "infosec" row is highlighted. Below the table, it says "Displaying 2 items".

**Security Groups** Create Security Group Delete Security Groups

<input type="checkbox"/>	Name	Description	Actions
<input type="checkbox"/>	default	default	Edit Rules
<input type="checkbox"/>	infosec	infosec	Edit Rules [v]

Displaying 2 items

Once the security groups is created, add rules to be applied to the groups. In the example below a rule to permit ssh traffic from anywhere is added.

### Edit Security Group Rules

#### Security Group Rules

IP Protocol	From Port	To Port	Source	Actions
No items to display.				
Displaying 0 items				

#### Add Rule

IP Protocol

From Port

To Port

Source Group

CIDR

TCP

22

22

CIDR

0.0.0.0/0

Cancel

Add Rule

Once the rule is added you will get a acknowledgement tab as follows.

## Access & Security

Success: Successfully added rule: ALLOW 22:22 from 0.0.0.0/0

Now Under the Project tab choose the 'Images and Snapshots' option, a list of Images which can be launched appears.

ubuntu<sup>®</sup> OpenStack Dashboard

Logged in as: vivekvc Settings Sign Out

Project

PROJECT Vivek Cherian

Manage Compute

Overview

Instances & Volumes

Images & Snapshots

Access & Security

Object Store

Containers

Images & Snapshots

Images

Delete Images

<input type="checkbox"/>	Image Name	Type	Status	Public	Container Format	Actions
<input type="checkbox"/>	QA	Image	Active	Yes	OVF	<div>Launch</div>
<input type="checkbox"/>	Symrise-indexing-server	Image	Active	Yes	OVF	<div>Launch</div>
<input type="checkbox"/>	ubuntu-symrise	Image	Active	Yes	OVF	<div>Launch</div>
<input type="checkbox"/>	pcloud	Image	Active	Yes	OVF	<div>Launch</div>
<input type="checkbox"/>	Centos-hapx-revision4	Image	Active	Yes	OVF	<div>Launch</div>
<input type="checkbox"/>	Centos-monitoring	Image	Active	Yes	OVF	<div>Launch</div>
<input type="checkbox"/>	Centos-Hapx-revision3	Image	Active	Yes	OVF	<div>Launch</div>

In this example a bare minimum Ubuntu Pristine image is being launched.

☒

ubuntu-pristine

Image

Active

Yes

OVF

Launch

Displaying 10 items

Prior to the launching of any instance choose the Server Name. In our case, let us choose 'ubuntu-harden' as the Server name.

## Launch Instances

Server Name

ubuntu-harden

User Data

Flavor

m1.tiny (1VCPU / 0GB Disk / 512MB Ram )

Keypair

No keypairs available.

Description:

Specify the details for launching an instance. The chart below shows the resources used by this project in relation to the project's quotas.

Project Quotas

Instance Count (0)2 Available

VCPUs (0)2 Available

Disk (0 GB)10 GB Available

Memory (0 MB)2048 MB Available

Also under the Security Groups section, choose the group you created to be attached to your instance and launch your instance.

Instance Count

1

Security Groups

☐ default

☒ infosec

Cancel

Launch Instance

Once an instance is launched it appears instantly on the Dashboard as follows

Instances							
				<a href="#">Launch Instance</a>		<a href="#">Terminate Instances</a>	
<input type="checkbox"/>	Instance Name	IP Address	Size	Status	Task	Power State	Actions
<input type="checkbox"/>	ubuntu-harden	192.168.4.36	512MB RAM   1 VCPU   0 Disk	Active	None	Running	<a href="#">Edit Instance</a> ▼
Displaying 1 item							

Double Click on the Instance name, 'ubuntu-harden' in our case.

The Instance Detail screen shows up as follows,

ubuntu<sup>®</sup> OpenStack Dashboard

Logged in as: vivekvc Settings Sign Out

Project

PROJECT Vivek Cherian

Manage Compute

Overview

Instances & Volumes

Images & Snapshots

Access & Security

Object Store

Containers

Instance Detail: ubuntu-harden

Overview Log VNC

Instance Overview

Info

Name

ubuntu-harden

ID

cee2f514-0e9c-4625-9448-074033b66a88

Status

Build

Specs

RAM

512MB

VCPUs

1 VCPU

Disk

0GB

Under the Security Groups section, Security Groups associated with the instance and the permissions are displayed.

IP Addresses	
Private	192.168.4.36
Security Groups	
Infosec	ALLOW 22:22 from 0.0.0.0/0
Meta	
Key Name	None
Image Name	ubuntu-pristine