

LOGIN:  
Security

Mark Wallis

Overview

Execution

Ownership

Network

MAC

Questions

# LOGIN: Security

Mark Wallis

March 17, 2009

# Security in Linux - Overview

LOGIN:  
Security

Mark Wallis

Overview

Execution

Ownership

Network

MAC

Questions

- Execution - root and you
- Ownership - permissions
- Network - software firewalls
- MAC - SELinux

# Execution - root and you

LOGIN:  
Security

Mark Wallis

Overview

Execution

Ownership

Network

MAC

Questions

## Linux is a multi-user environment

- Most Linux installs have 20-30 accounts you don't even realise
- Not all accounts allow interactive login
- Users 'own' objects such as files, directories and processes

# Execution - least privilege

LOGIN:  
Security

Mark Wallis

Overview

Execution

Ownership

Network

MAC

Questions

You should always implement the concept of 'least privilege'

- A user should never have more rights than they need
- Your every-day user should be a personal account
- NOT ROOT
- These days, some distro's enforce this (i.e. ubuntu)

# Execution - whoami ?

LOGIN:  
Security

Mark Wallis

Overview

Execution

Ownership

Network

MAC

Questions

```
[root@qvmmon101 login]# whoami  
root
```

# Execution - escalation

LOGIN:  
Security

Mark Wallis

Overview

Execution

Ownership

Network

MAC

Questions

## su and sudo

- Allow you to temporarily alter the user you are running as
- Either for a single command
  - `sudo apt-get install thingy`
- Or for a whole session
  - `su`
  - `sudo -i`

# Execution - root rulez

LOGIN:  
Security

Mark Wallis

Overview

Execution

Ownership

Network

MAC

Questions

But why ? Security, that's why

- If your local account gets compromised then the offending code cannot alter the kernel
- You can also just delete your local account and recreate it
- Same principal as Microsoft have been implementing in Vista/Windows 7 (UAC)

# Execution - root can still be owned

LOGIN:  
Security

Mark Wallis

Overview

Execution

Ownership

Network

MAC

Questions

- Unfortunately, you still not safe
  - Kernel-level privilege escalation exploits
  - Keyloggers capturing your root PW
- Implementing 'least privilege' is all about limiting the damage

# Ownership - its a democracy baby

LOGIN:  
Security

Mark Wallis

Overview

Execution

Ownership

Network

MAC

Questions

## Users own things:

- Files
- Directories
- Devices
- Processes

# Ownership - you and your friends

LOGIN:  
Security

Mark Wallis

Overview

Execution

Ownership

Network

MAC

Questions

There are two levels of ownership

- Owner (e.g. markw, root)
- Group (e.g. wheel, users)

# Ownership - no you can't come play

LOGIN:  
Security

Mark Wallis

Overview

Execution

Ownership

Network

MAC

Questions

Privileges are assigned at three levels

- Owner
- Group
- Everyone else

# Ownership - file example

LOGIN:  
Security

Mark Wallis

Overview

Execution

Ownership

Network

MAC

Questions

## An example: a file

```
[root@example login]# ls -al  
-rw-r--r--  1 root root   3 Mar 16 14:07 example
```

- r = read
- w = write
- x = execute

# Ownership - masquerade masquerade

LOGIN:  
Security

Mark Wallis

Overview

Execution

Ownership

Network

MAC

Questions

## setuid, setgid

```
[root@example login]# ls -al  
-rwsr--r--  1 root root   3 Mar 16 14:07 example
```

- setuid/setgid = allows privilege escalation
- the above ensures that whoever executes 'example' will have it run with as if root executed it

# Ownership - configuration

LOGIN:  
Security

Mark Wallis

Overview

Execution

Ownership

Network

MAC

Questions

## Changing ownership/permissions

```
[root@example login]# ls -al
-rwxr-xr-x  1 root root   3 Mar 16 14:07 example
[root@qvmmon101 login]# chown markw:markw example
[root@qvmmon101 login]# chmod a-x ./example
[root@example login]# ls -al
-rw-r--r--  1 markw markw   3 Mar 16 14:07 example
```

- chown = allows you to change the owner and group of a file
- chmod = allows you to change the permissions

# Ownership - examples

LOGIN:  
Security

Mark Wallis

Overview

Execution

Ownership

Network

MAC

Questions

chmod examples:

- chmod a+r example (change at all levels)
- chmod o+r example (change at the 'everyone else' level)
- chmod u+r example (change at the owner level)
- chmod g+r example (change at the group level)
- chmod 777 example - octel notation

# Ownership - in the runtime

LOGIN:  
Security

Mark Wallis

Overview

Execution

Ownership

Network

MAC

Questions

## Process ownership:

```
[root@qvmon101 login]# ps aux
USER      PID CPU MEM   VSZ  RSS TTY      STAT START   TIME COMMAND
root         1  0.0  0.0  1708 452 ?        S      Feb17   0:46  init [5]
root     2146  0.0  0.0   2588 728 ?        Ss     Feb17   0:18  syslog-ng -p
root     2581  0.5 27.3 526532 282064 ?      Sl     Feb17 224:33  splunkd -p 808
apache  6535  0.0  1.1 29036 11688 ?        S      11:32   0:03  /usr/sbin/http
```

- 'ps' command shows you the processes running on a system
- -u parameter shows you who owns each process
- root can 'kill' a process based on its PID

# Execution - devices are files too you know

LOGIN:  
Security

Mark Wallis

Overview

Execution

Ownership

Network

MAC

Questions

## Device ownership:

- Users own devices as well
- The example below ensure everyone can use the cdrom
- Not always the case by default for all devices

```
root@qvmon101 login]# ls -al /dev/cdrom  
lrwxrwxrwx  1 root root 3 Feb 17 19:57 /dev/cdrom
```

# Network - big scary words

LOGIN:  
Security

Mark Wallis

Overview

Execution

Ownership

Network

MAC

Questions

## Software Firewalls - terminology

- Software vs Hardware firewall
- Ingress vs Egress vs both
- Protocol, Source IP, Destination IP, Source port, Destination Port

# Network - Fort Knox

LOGIN:  
Security

Mark Wallis

Overview

Execution

Ownership

Network

MAC

Questions

## Software Firewalls - why ?

- Laptops/Travelling
- If you cannot trust your co-inhabitants
- Protection against worm/virus propagation

# Network - RustyCode (tm)

LOGIN:  
Security

Mark Wallis

Overview

Execution

Ownership

Network

MAC

Questions

Linux: Software Firewall = Netfilter + iptables

- Netfilter is the kernel subsystem where the kernel-level software firewall is implemented
- iptables is the command-line tool for configuring the firewall
- Firewall policies are generally written once, saved to disk and loaded upon boot

# Network - in, out and through

LOGIN:  
Security

Mark Wallis

Overview

Execution

Ownership

Network

MAC

Questions

Netfilter allows you to define three basic firewall rulesets

- INPUT - packets entering your network interface destined for your system
- OUTPUT - packets leaving your network interface from your system
- FORWARD - packets passing through your system from one interface to another

# Network - crazyness will ensue

LOGIN:  
Security

Mark Wallis

Overview

Execution

Ownership

Network

MAC

Questions

## Netfilter isn't only a 'firewall'

- QoS, packet marking
- Packet mangling (NAT, PAT, etc)
- Modular framework for extension into crazy places due to the low-level in which Netfilter is integrated into the networking systems of the Linux kernel

# Network - management

LOGIN:  
Security

Mark Wallis

Overview

Execution

Ownership

Network

MAC

Questions

## How to write a software firewall policy

- iptables: by hand, if your policy is small or your crazy
- 'Firewall Builder': if you have a large policy (i.e. your host is actually acting as your Internet gateway)
- Most distro's have a 'standard' policy tool

# Network - remember this

LOGIN:  
Security

Mark Wallis

Overview

Execution

Ownership

Network

MAC

Questions

## Things to keep in mind for firewalls in general

- Peer-to-peer communications
- Strict egress rules
- ICMP - not everything is nasty

# MAC - some days, stuff just doesn't work

LOGIN:  
Security

Mark Wallis

Overview

Execution

Ownership

Network

MAC

Questions

## MAC = Mandatory Access Control

- Super lower-level privileges around EVERYTHING in the system
- SELinux is the most popular implementation
- Implements DoD style access controls and originally developed by the NSA
- Kernel-level security protection
- Reduces the risk to the system if a single application is compromised
- Essentially implements the 'least privilege' principal for all code executing within the system.
- MAC sits ontop of DAC (the standard user/group ownership model)

# MAC - why oh why

LOGIN:  
Security

Mark Wallis

Overview

Execution

Ownership

Network

MAC

Questions

## MAC: why do I care ?

- Most people come across MAC in two different situations
  - 1. Your a hardcore security nazi who wants to lock down a server super-tight
  - 2. Your a not-so-hardcode security nazi that just wants to turn the darn thing off

# MAC - please go away

LOGIN:  
Security

Mark Wallis

Overview

Execution

Ownership

Network

MAC

Questions

## MAC: turning it off

```
[root@qvmmon101 selinux]# cat /etc/selinux/config
# This file controls the state of SELinux on the system.
# SELINUX= can take one of these three values:
# enforcing - SELinux security policy is enforced.
# permissive - SELinux prints warnings instead of enforcing.
# disabled - SELinux is fully disabled.
SELINUX=enforcing
# SELINUXTYPE= type of policy in use. Possible values are:
# targeted - Only targeted network daemons are protected.
# strict - Full SELinux protection.
SELINUXTYPE=targeted
```

# MAC - excellent idea in theory

LOGIN:  
Security

Mark Wallis

Overview

Execution

Ownership

Network

MAC

Questions

## MAC really isn't all that bad

- By default, most distro's have a MAC policy that is well implemented. For example RHEA implements SELinux by default only on certain processes such as Apache and Named, then defaults to DAC for everything else
- It's not easy for an everyday user to wrap their head around though, especially when it comes to altering policy
- Most distro's provide an easy off/transparent/on switch
- Its great on servers when you want to protect against malicious code accessing prohibited data

# MAC - but how does it actually work ?

LOGIN:  
Security

Mark Wallis

Overview

Execution

Ownership

Network

MAC

Questions

Config screen below is from Fedora

The screenshot shows the SELinux Administration interface. The title bar reads "SELinux Administration". The menu bar includes "File" and "Help". On the left, a "Select:" sidebar lists various SELinux options, with "Boolean" selected. The main area shows a table of SELinux booleans for the "apache" process. The table has columns for "Active", "Module", "Description", and "Name".

Active	Module	Description	Name
<input type="checkbox"/>	apache	Allow httpd to act as a FTP server by listening on the	httpd_enable_ftp_server
<input type="checkbox"/>	apache	Allow HTTPD to run SSI executables in the same dom	httpd_ssi_exec
<input type="checkbox"/>	apache	Allow Apache to communicate with avahi service via	allow_httpd_dbus_avahi
<input checked="" type="checkbox"/>	apache	Allow httpd to use built in scripting (usually php)	httpd_built_in_scripting
<input type="checkbox"/>	apache	Allow http daemon to send mail	httpd_can_sendmail
<input type="checkbox"/>	apache	Allow httpd to access nfs file systems	httpd_use_nfs
<input checked="" type="checkbox"/>	apache	Unify HTTPD to communicate with the terminal. Ne	httpd_tty_comm
<input type="checkbox"/>	apache	Allow Apache to use mod_auth_pam	allow_httpd_mod_auth_ntl
<input type="checkbox"/>	apache	Allow HTTPD scripts and modules to connect to the r	httpd_can_network_conne
<input checked="" type="checkbox"/>	apache	Unify HTTPD handling of all content files	httpd_unified
<input type="checkbox"/>	apache	Allow apache scripts to write to public content. Dire	allow_httpd_sys_script_and
<input checked="" type="checkbox"/>	apache	Allow httpd to read home directories	httpd_enable_homedirs
<input checked="" type="checkbox"/>	apache	Allow Apache to modify public files used for public fil	allow_httpd_anon_write
<input type="checkbox"/>	apache	Allow Apache to use mod_auth_pam	allow_httpd_mod_auth_par
<input type="checkbox"/>	apache	Allow httpd to access cifs file systems	httpd_use_cifs
<input checked="" type="checkbox"/>	apache	Allow httpd cgi support	httpd_enable_cgi
<input type="checkbox"/>	apache	Allow HTTPD scripts and modules to network connect	httpd_can_network_conne
<input type="checkbox"/>	apache	Allow httpd to act as a relay	httpd_can_network_relay
<input type="checkbox"/>	bind	Allow BIND to write the master zone files. Generally	named_write_master_zones
<input type="checkbox"/>	cdrecord	Allow cdrecord to read various content. nfs_samba	cdrecord_read_content

# MAC - so the NSA doesn't come to get me ...

LOGIN:  
Security

Mark Wallis

Overview

Execution

Ownership

Network

MAC

Questions

## MAC really isn't all that bad

- By default, most distro's have a MAC policy that is well implemented
- It's not easy for an everyday user to wrap their head around though, especially when it comes to altering policy
- Most distro's provide an easy off/transparent/on switch
- Its great on servers when you want to protect against malicious code accessing prohibited data

# Thanks for listening to me babble

LOGIN:  
Security

Mark Wallis

Overview

Execution

Ownership

Network

MAC

Questions

- Questions ?