Cyber Defense Hackpack

CU Cyber

Contents

General Information

The Ten Commandments of Cyber Defense

1. Thou shalt NEVER trust the red team
2. Thou shalt trust but verify everything else
3. Thou shalt know thy network
4. Thou shalt patch thy services
5. Thou shalt make frequent backups
6. Thou shalt disable unused services
7. Thou shalt set and use strong passwords
8. Thou shalt always use a firewall
9. Thou shalt log everything
10. Thou shalt get your injects done on time

Linux Checklist

This checklist is designed for the first 30 minutes of competition.

For each system:

- Change the password for the root account
- Check for improper ssh config
- Check for improper sshd config
- Check the crontab (s) for running tasks
- Check with files with wider permissions and setuid
- Create a report of running services and processes and disable unnecessary processes
- Create a report of open ports
- Audit user, groups for invalid entries
- Check mount/nfs if it is running
- Install updates
- Run a full system backup
- Check/Configure '/etc/sudoers' and '/etc/sudoers.d/*'
- Harden the service for your machine
- Install/Configure a firewall
- Write an audit report containing changes made

For all systems:

- Scan the subnet for running servers
- Configure a provisioner such as Salt/Ansible

Windows Checklist

This checklist is designed for the first 30 minutes of competition.

For each System:

- Change the password for the admin accounts
- Remove any nonessential user accounts
- Check the startup list
- Check the event logs
- Create a report of all running services
- Create a report of all open ports
- Install Windows service packages
- Run a full backup of the system

Authentication

Kerberos

Kerberos is a remote login service that allows a set of Linux and Windows servers to share users and groups. The current version of the protocol is version 5, and version 4 is deprecated and should not be used due to weak cryptography.

Installation

Kerberos can be configured to either authenticate against either Linux or a Windows Active Directory server.

Here is how to install and configure Kerberos to run on Linux:

```
#!/bin/sh
# if you can, use realmd (newer servers)
realm discover
realm join "realm_name"
realm permit -a

# which roughly does the following on the backend

#####################################################################
##this section is roughly based on the ArchLinux Wiki documentation##
##Which is available under the GNU Free Document License           ##
#####################################################################
# first install and configure NTP and name resolution for the servers
# next, configure the /etc/krb5.conf file as shown below

# verify that you can now login
kinit administrator@EXAMPLE.COM
klist

# if there are errors regarding a missing pam_winbind make a file called
# 'pam_winbind.conf' with the contents of the next section

# configure Samba as specified in the Samba series section

net ads join -U Administartor

# start and enable the required services
systemctl start smbd nmbd winbindd
```

```
# configure /etc/nsswitch.conf as shown below

# test winbind and nss(Windows Authentication service)
wbinfo -u
wbinfo -g
getent passwd
getent group
net ads info
net ads lookup
net ads status -U administrator

# configure PAM with the following config.

# modify applications as necessary to use Kerberos; see the specific application for documentation

# finally configure smb shares and keytabs if desired using the following
```

/etc/pam_winbind.conf

```
[global]
  debug = no
  debug_state = no
  try_first_pass = yes
  krb5_auth = yes
  krb5_cache_type = FILE
  cached_login = yes
  silent = no
  mkhomedir = yes
```

/etc/pam.d/krb5

```
#%PAM-1.0

auth [success=1 default=ignore] pam_localuser.so
auth [success=2 default=die] pam_winbind.so
auth [success=1 default=die] pam_unix.so nullok
auth requisite pam_deny.so
auth      optional  pam_permit.so
auth      required  pam_env.so


account   required  pam_unix.so
account [success=1 default=ignore] pam_localuser.so
account required pam_winbind.so
account   optional  pam_permit.so
account   required  pam_time.so

password [success=1 default=ignore] pam_localuser.so
password [success=2 default=die] pam_winbind.so
password [success=1 default=die] pam_unix.so sha512 shadow
password requisite pam_deny.so
password  optional  pam_permit.so
```

```
session   required  pam_limits.so
session required pam_mkhomedir.so skel=/etc/skel/ umask=0022
session   required  pam_unix.so
session [success=1 default=ignore] pam_localuser.so
session required pam_winbind.so
session   optional  pam_permit.so
```

/etc/krb5.conf

```
[libdefaults]
    #Default Realm must be unique on the network, by convention it is all caps
    default_realm = EXAMPLE.COM
    #if Windows Server 2008 and older require weak crypto; Think carefully before using
    allow_weak_crypto = true

[realms]
    EXAMPLE.COM = {
        #host where the auth server is running given as a fqdm:port
        admin_server = kerberos_server.example.com:749
        #Name(s) of a host running a Kerberos Key Distribution Server
        #These are nessisary if realm admins don't have SRV records in DNS
        kdc = kerberos_server.example.com:88
        kdc = kerberos_server2.example.com:88
    }
[domain_realm]
    #maps host names to kerberos realms
    #domains beginning with a . include all subdomains of the specified domain
    .example.com = EXAMPLE.COM
    example.com = EXAMPLE.COM

[logging]
    default = FILE:/var/log/krb5libs.log
```

login.defs

'/etc/login.defs' is a configuration file that controls login functionality on Linux machines using encrypted password files. It is one of 3 tools that can control this process (login.defs, pam, systemd-logind).

Best Practices

/etc/login.defs

- set `CONSOLE` to `/etc/securetty`
- set `PASS_MAX_DAYS` to 30
- set `PASS_MIN_DAYS` to 7
- set `PASS_WARN_DAYS` to 8
- set `PASS_MIN_LEN` to 8
- set `MAIL_DIR` to `/var/spool/mail`
- set `UMASK` to 077

/etc/securetty

```
console
tty1
tty2
tty3
tty4
tty5
tty6
```

PAM

PAM is short for Pluggable Authentication Module. It controls the authentication on Linux machines. By default, it reads the '/etc/shadow' file. It can also be configured to use LDAP or Kerberos for remote authentication.

In general the following should be configured:

- use `pam_cracklib retry=3 minlen=14 dcredit=-1 ucredit=-1 ocredit=-1 lcredit=-1`
- use `pam_unix obscure sha512 remember=5`

Configuration

/etc/pam.d/system-auth

```
auth required pam_env.so
auth required pam_unix.so try_first_pass likeauth nullokf
auth required /lib/security/\$ISA/pam_tally.so onerr=fail no_magic_root

account required pam_unix.so
account required /lib/security/\$ISA/pam_tally.so per_user_deny=5 no_magic_root reset

password required pam_cracklib.so retry=3 minlength=8 difok=3
password required pam_unix.so try_first_pass use_authtok sha512 shadow

session required pam_limits.so
session required pam_env.so
session required pam_unix.so
```

Sudo

Sudo is short for super user do. It allows for non-root users to request elevated privileges on linux systems. Sudo has a variety of options that can be configured.

Here are some basic suggestions for managing systems that use sudo:

- Limit users permissions where possible, users should not have `ALL = (ALL) ALL`
- Avoid using groups with root permissions

The '/etc/sudoers' file should be edited as root using the `visudo` command which verifies the syntax before making changes. There are also other configuration files that can be found in '/etc/sudoers.d/'. These can be edited using `visudo -f <filename>`.

Configuration

The following script will create a sane configuration for sudo.

```
#!/bin/sh
cp -Ra /etc/sudoers.d /etc/sudoers.d~
groupdel wheel && groupadd wheel
cp -a /etc/sudoers /etc/sudoers~
cat >/etc/sudoers <<- EOF
root ALL=(ALL) ALL
%wheel ALL=(ALL) ALL
EOF

chown root:root /etc/sudoers
chmod 600 /etc/sudoers

# be sure to add administrators back to 'wheel' and
# merge stuff from '/etc/sudoers~' and '/etc/sudoers.d~/'
```

Backups and Restoration

rsnapshot

Summary

rnapshot is a utility to create incremental snapshot backups using rsync. It has minimal dependencies and should work even on very old Linux distributions.

Dependencies

- perl
- rsync
- openssh

Configuration

/etc/rsnapshot.d/system

```
config_version  1.2

no_create_root  1
lockfile    /var/run/rsnapshot.pid

cmd_cp      /bin/cp
cmd_rm      /bin/rm
cmd_rsync   /usr/bin/rsync
cmd_ssh     /usr/bin/ssh
link_dest   1

one_fs      1

snapshot_root   /mnt/backup/

retain      system  8

exclude     /dev/**
exclude     /proc/**
exclude     /sys/**

exclude     /tmp/**
exclude     /var/cache/**
exclude     /var/lock/**
exclude     /var/run/**
exclude     /var/tmp/**

exclude     /usr/portage/distfiles/**

backup      root@example.local:/    example.local/
```

/etc/rsnapshot.d/application

```
config_version  1.2

no_create_root  1
lockfile    /var/run/rsnapshot.pid

cmd_cp      /bin/cp
cmd_rm      /bin/rm
cmd_rsync   /usr/bin/rsync
cmd_ssh     /usr/bin/ssh
link_dest   1

one_fs      1

snapshot_root   /mnt/backup/

retain      application 8

exclude     /dev/**
exclude     /proc/**
exclude     /sys/**

exclude     /tmp/**

exclude     /var/cache/**
exclude     /var/lock/**
exclude     /var/run/**
exclude     /var/tmp/**

exclude     /usr/portage/distfiles/**

backup      root@example.local:/etc/    example.local/
backup      root@example.local:/opt/    example.local/
backup      root@example.local:/var/    example.local/
```

/etc/cron.d/rsnapshot

```
0,15,30,45 * * * * root rsnapshot -c /etc/rsnapshot.d/application application
8 * * * * root rsnapshot -c /etc/rsnapshot.d/system system
```

Simple Backups

Summary

To create simple archive backups, use the tar command. These backups should be created as the initial backups during the beginning period of the competition.

Dependencies

- tar

Commands

Backup

```
#!/bin/sh
useradd flynn
mkdir -p /home/flynn/
tar cjpf /home/flynn/kevin \
    --exclude={/sys/*,/dev/*,/proc/*,/tmp/*,/run/*} \
    --exclude=/home/flynn/* /
chown flynn:flynn /home/flynn/kevin
chmod 640 /home/flynn/kevin
```

Extract

```
#!/bin/sh
tar xjpf /home/flynn/kevin --wildcards "$@"
```

Restore

```
#!/bin/sh
cd /
tar xjpf /home/flynn/kevin
```

Disaster Recovery

Linux

Here are some of the worst things that they can do, and how to hopefully recover.

Invalid Password

The red team has changed the password to the root account.

1. Reboot and hold shift
2. Press 'e' at the grub prompt to edit the kernel command line
3. Add the option `init=/bin/bash` to the line that says linux
4. Mount the root filesystem, `mount -o remout,rw /`
5. Use 'vi' to edit the `/etc/passwd` file and remove the 'x' between the 2nd and 3rd colon on the root user
6. Write the file
7. Use 'passwd' to create a new root password
8. Reboot

Chmod Not Executable

While this initially might not seem like it is too bad, there is a problem where bash does not provide a built-in to fix this. This can be particularly devious if the entire root directory is marked not executable. If the entire file system is not executable, you are looking at a live CD or a slave mounted drive.

There are a few options here roughly sorted by time to fix:

- Use a programming language like python, perl, or C to write your own chmod command to fix chmod
- Use the linker directly by running `/lib/ld-*.so /bin/chmod 755 /bin/chmod` as root
- Use a shell like busybox, zsh, or ksh that provides a built-in chmod
- Reinstall chmod from repos or a working machine
- Dump the bits of the binary and manually edit the permission bits
- Use a live CD
- Power off the machine, remove the hard drive, slave mount it to a different machine that has a working chroot, mount the drive, and fix it

Chmod Examples

C

```
#include<sys/stat.h>

int main(int argc, char *argv[]) {
    chmod("/bin/chmod", S_IRWXU);
    return 0;
}
```

Perl

```
#!/bin/sh
perl -e 'chmod 0755, "/bin", "/bin/chmod"'
```

Python

```
import os
os.chmod('/bin/chmod', 0755)
```

Firewalls

Firewall Basics

Firewalls are essentially sets of rules that allow network traffic in and out of a machine. In general, firewalls should be configured to allow the minimum required access. For Windows, the firewall is called Windows Firewall. For Linux, iptables is the built-in low-level firewall and ufw and firewalld are the most common high-level firewalls. For BSD, pf, the base of the pfSense enterprise firewall, is the default. For dedicated equipment, such as the Cisco ASA, custom firewalls or firewalls based on Linux and on occasion BSD are common.

In general, there are 3 major elements of firewall security:

- Use a default reject policy to avoid admitting unwanted traffic.
- Open only the required ports to make the services to work.
- Log any unusual traffic that hits the firewall.

FirewallD

Config Files

FirewallD references the following directories of files:

- '/usr/lib/firewalld' - where package default rules reside
- '/etc/firewalld' - where user overrides rules reside

Commands

FirewallD uses only the `firewall-cmd` binary.

Filtering

FirewallD is the new Linux firewall from RedHat. It provides a usability layer on top of iptables by focusing on zones and services. FirewallD therefore inherits iptables's first match rule.

Zones

Zones are affiliated with source addresses or interfaces. Zones have short names they are referenced by.

The following zone example affects all incoming traffic on the enp0s3 interface. It allows HTTPS traffic defined in the '/usr/lib/firewalld/services/https.xml' or overridden in '/etc/firewalld/services/https.xml'. It also blocks traffic on the 10.0.0.0/8 subnet by dropping and logging the packets.

```
<?xml version="1.0" encoding="utf-8"?>
<zone>
    <short>Public</public>
    <description>This is our external interface</description>
    <interface name="enp0s3"/>
    <service name="https"/>
    <rule family="ipv4">
        <source address="10.0.0.0/8"/>
        <log>
            <limit address="5/m"/>
        </log>
        <drop/>
    </rule>
</zone>
```

Services

Services define the ports and protocols that will be used by an application. Services have short names they are referenced by.

The following service example allows traffic on TCP port 21. It uses a kernel module to help track and filter the traffic. This is not required for all modules, but is used for some services such as FTP.

```
<?xml version="1.0" encoding="utf-8"?>
<service>
    <short>FOO</short>
    <description>Foo is a program that allows bar</description>
    <port protocol="tcp" port="21"/>
    <module name="nf_conntrack_foo"/>
</service>
```

Example Configuration

```
#!/bin/bash

# get the name of the device used for the default route
ext_if=$(ip route | head -n 1 | awk '{print $5}')

# list of devices that should be blocked on the external interface
# WARNING! assumes that there are separate interfaces
#          for the external network
# NOTE the 192.168.0.0/16 subnet should be excluded
#      if there is only one interface
broken="224.0.0.22 127.0.0.0/8, 192.168.0.0/16, 172.16.0.0/12, \
        10.0.0.0/8, 169.254.0.0/16, 192.0.2.0/24, \
        192.0.2.0/24, 198.51.100.0/24, 203.0.113.0/24, \
        169.254.0.0/16, 0.0.0.0/8, 240.0.0.0/4, 255.255.255.255/32"

firewall-cmd --zone=public --add-interface=$ext_if
```

```
for addr in $broken; do
    firewall-cmd --zone=public \
        --add-rich-rule="rule family='ipv4' service=ssh \
        source address=\"$addr\" log limit value='5/m' drop"
    firewall-cmd --zone=public \
        --add-rich-rule="rule family='ipv4' service=http \
        source address=\"$addr\" log limit value='5/m' drop"
    firewall-cmd --zone=public \
        --add-rich-rule="rule family='ipv4' service=https \
        source address=\"$addr\" log limit value='5/m' drop"
done

firewall-cmd --zone=public --add-service=ssh
firewall-cmd --zone=public --add-service=http

firewall-cmd --direct --add-rule ipv4 filter INPUT_direct 0 \
    -p tcp --dport ssh -m state --state NEW -m recent --set
firewall-cmd --direct --add-rule ipv6 filter INPUT_direct 0 \
    -p tcp --dport ssh -m state --state NEW -m recent --set
firewall-cmd --direct --add-rule ipv4 filter INPUT_direct 1 \
    -p tcp --dport ssh -m state --state NEW -m recent --update \
    --seconds 30 --hitcount 6 -j REJECT --reject-with tcp-reset
firewall-cmd --direct --add-rule ipv6 filter INPUT_direct 1 \
    -p tcp --dport ssh -m state --state NEW -m recent --update \
    --seconds 30 --hitcount 6 -j REJECT --reject-with tcp-reset

firewall-cmd --runtime-to-permanent
```

iptables

Config Files

iptables stores the majority of its configuration in a series of files:

- '/etc/sysconfig/iptables' - iptables configuration (RedHat-based distributions)
- '/etc/iptables' - iptables configuration (Debian-based distributions)
- '/etc/services' - an optional file that maps service names to port numbers

Commands

iptables uses the following binaries:

- `iptables` - view and modify the firewall
- `iptables-save` - prints the running configuration to stdout; used to save the running configuration to a file
- `iptables-restore` - reads a file and sets the firewall configuration

While a save format exists, iptables is normally configured via shell commands to avoid inconsistencies between save file versions.

Filtering

iptables will stop processing a packet when it matches the first rule. The only exception to this is the LOG target. When the LOG target is matched, matching will continue; but the traffic will be logged in the kernel log.

Example Configuration

```bash
#!/bin/bash

# clear out the current configuration
iptables -F && iptables -X

# allow traffic on the loopback interface
iptables -A INPUT -i lo -j ACCEPT

# ext_if is the device with the default route
ext_if=$(ip route | head -n 1 | awk '{print $5}')

# broken is a list of address that should be blocked on the external interface
# WARNING! Assumes that there is an internal and an external interface
# note that 192.168.0.0/16 should not be blocked if there is only one interface
broken="224.0.0.22 127.0.0.0/8, 192.168.0.0/16, 172.16.0.0/12, \
        10.0.0.0/8, 169.254.0.0/16, 192.0.2.0/24, \
        192.0.2.0/24, 198.51.100.0/24, 203.0.113.0/24, \
        169.254.0.0/16, 0.0.0.0/8, 240.0.0.0/4, 255.255.255.255/32"


# use a default drop policy
iptables -P INPUT DROP

# disable all ipv6 traffic; Syntax is the same as ipv4 if required
ip6tables -P INPUT DROP
ip6tables -P OUTPUT DROP
ip6tables -P FORWARD DROP

# log traffic that is dropped by the firewall
iptables -N LOGDROP
iptables -A LOGDROP -m log --log-level info --log-prefix "IPTABLES" \
    -m limit --limit 5/m --limit-burst 10 -j LOG
iptables -A LOGDROP -j DROP

# block bad packets and http and ssh traffic from broken addresses
iptables -A INPUT -m conntrack --ctstate INVALID -j LOGDROP
iptables -t raw -I PREROUTING -m rpfilter -j LOGDROP
for addr in $broken; do
    iptables -A INPUT -p tcp -i $ext_if -s $addr --dport 80 -j REJECT
    iptables -A INPUT -p tcp -i $ext_if -s $addr --dport 443 -j REJECT
    iptables -A INPUT -p tcp -i $ext_if -s $addr --dport 22 -j REJECT
done

# allow established traffic to applications
iptables -I INPUT 1 -m conntrack --ctstate ESTABLISHED,RELATED -j ACCEPT
```

```
# allow new traffic to applications
iptables -A INPUT -m limit --limit 5/m --limit-burst 10 -m conntrack \
    --ctstate NEW -p tcp --dport 22 -j ACCEPT
iptables -A INPUT -m limit --limit 5/m --limit-burst 10 -m conntrack \
    --ctstate NEW -p tcp --dport 80 -j ACCEPT
iptables -A INPUT -m limit --limit 5/m --limit-burst 10 -m conntrack \
    --ctstate NEW -p tcp --dport 443 -j ACCEPT

# drop all other traffic
iptables -A INPUT -j DROP
```

pf

Config Files

pf references the following files:

- '/etc/rc.conf' - as with all servies, pf must be enabled here
- '/etc/pf.conf' - pf configuration file

Commands

pf uses the following binaries:

- `pfctl -f /etc/pf.conf` - load the firewall configuration
- `pfctl -sa` - see the current configuration status
- `kldload pf` - load the pf kernel module

Filtering

All of the configuration for pf is stored in '/etc/pf.conf'. There is no way to modify the running configuration except to overwrite the running configuration with the saved configuration. Unlike other firewalls, the last rule to match will be the rule that is applied. This behavior can be overridden by using the `quick` keyword.

Example Configuration

```
# adapted from bsdnow tutorial

# variables for convenience
ext_if = "em0"
broken="224.0.0.22 127.0.0.0/8, 192.168.0.0/16, 172.16.0.0/12, \
        10.0.0.0/8, 169.254.0.0/16, 192.0.2.0/24, \
        192.0.2.0/24, 198.51.100.0/24, 203.0.113.0/24, \
        169.254.0.0/16, 0.0.0.0/8, 240.0.0.0/4, 255.255.255.255/32"
# use a default drop policy
set block-policy drop

# skip the local loopback interface
set skip on lo0
```

```
# block invalid packets
match in all scrub (no-df max-mss 1440)
block in all
pass out quick on $ext_if inet keep state
antispoof quick for ($ext_if) inet

# block ipv6 if it is not needed
block out quick inet6 all
block in quick inet6 all

# block any packet we can't find a valid route back to
block in quick from { $broken urpf-failed no-route } to any
block out quick on $ext_if from any to { $broken no-route }

# block bad actors
table <childrens> persist
block in log quick proto tcp from <childrens> to any

# block Chinese address to ssh and web
table <chuugoku> persist file "/etc/cn.zone"
block in quick proto tcp from <chuugoku> to any port { 80 22 }

# allow traffic thought the firewall
pass in on $ext_if proto tcp from any to any port 80 flags S/SA synproxy state
pass in on $ext_if proto tcp from 1.2.3.4 to any port { 137, 139, 445, 138 }
pass in on $ext_if proto tcp to any port ssh flags S/SA keep state \
(max-src-conn 5, max-src-conn-rate 5/5, overload <childrens> flush)
pass inet proto icmp icmp-type echoreq

# adapted from the http://www.bsdnow.tv/tutorials/pf
# which is distrusted under CC-BY-SA
```

Uncomplicated Firewall

Config Files

Uncomplicated Firewall references mainly the following files:

- '/etc/default/ufw' - high level configuration
- '/etc/ufw/sysctl.conf' - kernel tunables

Commands

- ufw enable - enables and reloads the firewall
- ufw default - sets default action
- ufw allow - allows service or port
- ufw deny - blocks a service or port
- ufw limit - allows with connection rate limiting

Filtering

Uncomplicated Firewall is based on iptables and therefore inherits iptables's first match rule.

Example Configuration

```
#!/bin/sh
ufw default deny
ufw allow ssh/tcp
ufw logging on
ufw enable
```

Logging and Investigation

auditd

Linux has a built-in auditing framework that acts in kernel space. This portion of the kernel communicates with the userspace auditd server. It can be configured to monitor files and syscalls.

Installation

```
#!/bin/sh
dnf install audit
systemctl start auditd
```

Config Files

The userspace auditing commands, can be used to configure logs. Audit can stores its rules in ‘/etc/audit/audit.rules’ or in files inside ‘/etc/audit/audit.d/’. The syntax for these files is the same as the userspace commands.

Commands

Viewing the auditlog can be done in a few ways:

- `aureport` - query logs for a specific event
- `ausearch` - view a summary of recent events
- `syslog` - view logs typically stored in ‘/var/log/audit/audit.log’

Example Configuration

```
#!/bin/sh
# remove all rules
auditctl -D

# see a list of rules
auditctl -l

# watch a file for writes
auditctl -w /etc/passwd -p wa -k passwd_access

# watch a directory and all its children for writes
```

```
auditctl -w /etc/ -p wa -k etc_writes

# watch for use of a specific syscalls
auditctl -a always,exit -S stime.* -k time_changes
auditctl -a always,exit -S setrlimit.* -k setrlimits
auditctl -a always,exit -S unlink -S rmdir -k deleting_files

# watch for unsucessful calls
# the -F flag filters out based on various
# options see man auditctl for more details
auditctl -a always,exit -S all -F sucess=0

# make the default audit log buffer larger
auditctl -b 1024

# lock audit rules so that they cannot be edited until reboot
auditctl -e 2
```

Rsyslog

Rsyslog is a client and server that conforms to the syslog protocol. On systems with systemd (i.e. RedHat and newer Debian distributions), journald is typically used instead, but both programs can be used to compliment each other.

Config Files

Rsyslog is generally configured by '/etc/syslog.conf' and '/etc/syslog.conf.d/' like most other syslog daemons.

/etc/syslog.conf

```
*.err;kern.debug                        /dev/console
auth.notice;authpriv.none               /dev/console
*.err;*.crit;*.emerg                    /var/log/critical.log
*.notice                                /var/log/messages
auth,authpriv.none                      /var/log/messages
auth,authpriv.debug                     /var/log/auth.log
cron.info                               /var/log/cron.log
news,kern,lpr,daemon,ftp,mail.info      /var/log/daemon.log
*.err;user.none                         root
*.emerg;user.none                       *
```

Script

The `script` command can be used to record commands run for audit logs.

Usage

```
#!/bin/sh
# use default interactive interpreter
script <file>

# use specified interactive interpreter
script -c bash <file>
```

Operating System Utilities

sh

Backgrounding

Often you will want to be able to run processes in the background during the contest. There are a few ways to do this:

- job control - this is the legacy job control system.
- tmux/screen - if they are installed, these are more full featured tools

```
#!/bin/sh
# run an update with yum in the background writing the log to
# '/var/log/yum_updates' that will continue running even if the user logs out
nohup yum update -y >>/var/log/yum_updates 2>&1 &

# see the running list of jobs
jobs -l

# send a job to the background
bg

# send a job to the foreground
fg

# disown a process after you start it
disown
```

Scripting

```
#!/bin/sh
# for loop that outputs 1 2 3 4 5 6 7 8 9 10 a b c
for i in {1..10} a b c
do
    echo $i
done

# conditional testing for an empty string
foo="bar"
if [ -z "$foo" ]; then
    echo $foo
fi
```

```
# conditional testing equal stings
if [ "$foo" == "bar" ]; then
    echo $foo
fi

# conditional testing numeric values
if [ 1 -eq 2 ]; then
    echo $foo
fi

# example function
foobar(){
    echo $1
}
foobar "this echos this statement"
```

Permissions

/dev

The following command will fix permissions for all device files to reasonable defaults:

```
#!/bin/sh
# find devices with execute permissions and remove the execute permissions
find /dev \( -type c -or -type b -or -type f \) -perm -+x -exec chmod -x {} \;
```

/home

The following command will fix permissions for all home files to reasonable defaults:

```
#!/bin/sh
# barring unusual circumstances, files in a home directory should be 640
sed -i -e "s/^umask [0-9][0-9][0-9]$/umask 027/" /etc/profile
for file in $(find /home/*); do
    if [ -d "$file" ]; then
        #Directories must have execute permissions
        chmod 750 "$file"
    elif [ -x "$file" ]; then
        #Some users will have scripts in there home directory
        read "Should this file be executable ($file) ?" yes
        if [ $yes -eq "y" ]; then
            chmod 750 "$file"
        else
            chmod 640 "$file"
        fi
    else
        #If it is not a file or directory, mark it 640
        chmod 640 "$file"
    fi
done
```

Provisioners

Ansible

Ansible is a lightweight agent-less provisioner that uses Python 2.x and OpenSSH as a backend. It is configured using Playbooks which are files written in a dialect of YAML.

Setting it Up

Managed Windows Machines

Ansible can also manage Windows machines running PowerShell 3.0 or later. For windows machines, you will also need to create an encrypted `host_vars` or `group_vars` file on the control machine that contains the following information:

```
# run 'ansible-vault edit group_vars/windows.yml'
# be sure to specifiy --ask-vault-pass when running ansible

# Will use AD if user name is like username@realm and you are signed into kerberos
# If you are using Ansible 1.x, the ansible_{user,pass,port} were called
# ansible_ssh_{user,pass,port}
ansible_user: WindowsAdministratorUsername
ansible_pass: WindowsAdministratorPassword
ansible_port: 5986
ansible_connection: winrm
```

It is also required to run the 'ConfigureRemotingForAnsible.ps1' from the Ansible source code script on the Windows machines that will be managed.

Managed Linux Machines

Ansible is agent-less, this means that only one machine must have Ansible installed on it. For Linux or BSD managed machines, all you have to have is OpenSSH and Python 2.x. For some Linux distributions where Python 3.x is the default or python is installed in an non-standard location, you may need to set `ansible_python_interpetor = /usr/bin/python2`. For best performance, SFTP must be enabled in '/etc/ssh/sshd_config' as a subsystem with the path to the 'sftp-server' binary

Control Machine

The control machine must have a few more pieces of software. On most distributions, it can be installed from package repositories. It can also be installed from source:

```
#!/bin/sh
# Install the source
git clone git://github.com/ansible/ansible.git --recursive
cd ./ansible
source ./hacking/env-setup

# For really broken machines such as Metaspoitable, install python27 from source
# You may even have to disable certificate checking (-k) but don't do that if you
# don't have to;  You will also need to set ansible_python_interpertor for these machines
curl -LO https://www.python.org/ftp/python/2.7/Python-2.7.tgz
tar -xzvf ./Python-2.7.tgz
cd ./Python-2.7
./configure && make && make install

#For CentOS
sudo yum install epel-release
sudo yum install autoconf gcc python-devel

# or install from pip
sudo pip install ansible

# Install dependencies if installing Ansible from github
sudo pip install paramiko PyYAML Jinja2 httplib2 six

# To manage Windows machines
pip install https://github.com/diyan/pywinrm/archive/master.zip#egg=pywinrm
pip install kerberos
```

Inventory Management

The collections of machines that are managed via Ansible are called the inventory. Here is an example inventory file:

```
[web:children]
webservers
databases

[webservers]
192.168.0.2
192.168.0.[10:20]

[databases]
foo.bar.com
sue.bar.com

[secureservers]
foobar@10.0.0.2:23

[local]
localhost ansible_connection=local
```

In this example we demonstrate,

- A set of hosts specified by a range of ip addresses
- A set of hosts specified by domain name

- A host using a different user and default port.
- Targeting the localhost
- Four groups of hosts called web, webservers, databases, and local
- A group of groups called web that contains all the hosts in webservers and databases.

NOTE, group variables and host variables can also be specified in the hosts file as shown with the `ansible_connection` example, but this format is discouraged because it does not follow a separation of concerns.

Sample Playbooks

Repositories containing Ansible playbooks are generally arranged out as follows:

- 'site.yml' - the primary playbook
- 'hosts' - the primary host inventory
- 'group_vars/' - directory containing encrypted group variables
- 'host_vars/' - directory containing encrypted host variables
- 'roles/' - directory containing roles that will be applied

Here is an example playbook:

```
---
- host: webservers # run this on the webserver group
  become: yes # escalate this play from remote user to super user
              # A user can also be specified via "become_user:"

  # varibles needed in the httpd.conf template
  vars:
    http_port: 80
    max_clients: 200

  # tasks that will be run run on the server
  tasks:

    # demonstrates iteration
    - name: create admins
      user: name={{item.name}} shell={{item.shell}} groups=wheel append=yes
      with_items:
          - { name: 'matthew', shell: '/bin/bash'}
          - { name: 'mark', shell: '/bin/zsh'}
          - { name: 'luke', shell:'/bin/fish'}

    # demonstrates conditionals
    - name: install apache for CentOS
      yum: name=httpd state=latest
      when: ansible_distribution == 'CentOS'

    - name: install apache for Debian
      apt: name=lighttpd state=latest
      when: ansible_os_family == 'Debian'

    # demonstrates handlers and templates
    - name: update the apache config file
      template: src=httpd.j2 dest=/etc/httpd.conf
      notify:
        - restart apache

    # demonstrates starting a service
    - name: ensure apache is running
      service: name=httpd state=started enabled=yes

  # tasks that need to be run when other tasks are run
  handlers:
      - name: restart apache
        service: name=httpd state=restarted
```

These playbooks can also be split into separate sections in what are called roles. Here is how a sample role, in this case a webserver are stored:

- 'roles/webserver' - directory where the webserver role is stored.
- 'roles/webserver/files' - files that would be referenced via copy commands in the role.
- 'roles/webserver/tempates' - templates that would be referenced via template commands in the role.
- 'roles/webserver/tasks' - where tasks for the webserver role are stored.
- 'roles/webserver/handlers' - where handlers that kickoff post processing tasks for the webserver role are stored.
- 'roles/webserver/vars' - where role specific variables for the webserver role are stored.
- 'roles/webserver/defaults' - where role specific default values variables for the webserver role are stored.
- 'roles/webserver/meta' - where role specific meta data for the webserver role are stored such as dependencies could be listed.

Ansible Vault

Ansible has a means of creating AES encrypted files for use of storing configuration. To create a file use `ansible-vault create <filename>` To edit a file use `ansible-vault edit <filename>` which will open the file un-encrypted in the user's `EDITOR` and re-encrypt it after editing. It can be used for file containing variables and files that are part of roles.

Extending Ansible

Somethings Ansible is just not good at, string parsing for instance. You can write modules in Python that do this heavy lifting. Here is a sample module that checks for the sshd version, and sets a variable with the output:

```
"""
Fundamentally, ansible modules simply accept a JSON string as input, do work,
and return a JSON string as output to stdout.  At no time should anything be
printed that is not the final JSON output, or exceptions be returned
This if this module was called site_facts can be included via a play like so:
---
- name: Gather facts
  action: site_facts
  tags:
    - always
"""
import re
import functools

def ssh_facts(module):
    """
    Collect facts for the ssh installations
    """
    #Prior to Python 3, there was not a good subprocess module
    #So ansible includes their own with the necessary options set
    rc, out, err = module.run_command(args=['ssh', '-V'])
    if rc == 0:
        ssh_version = str(err).split(',')[0]
        ssh_version = ssh_version[8:]
    else:
        ssh_version = '0.0p0'

    try:
        major_version, minor_version, patch_version = \
                re.match(r'(\d+)\.(\d+)p(\d+)', ssh_version).groups()
    except AttributeError:
        ssh_version = '0.0p0'
        major_version, minor_version, patch_version = (0, 0, 0)

    return {
        "ssh_version": ssh_version,
        "ssh_major_version": major_version,
        "ssh_minor_version": minor_version,
        "ssh_patch_version": patch_version,
    }

FACTS = {
    "ssh": ssh_facts,
}

def main():
    """
    This is the main method, to extend this module, add an entry to FACTS
    and write a function that gathers the necessary information
    """

    #Here are where the arguments to the module are examined
    #The quotes around str are important
    module = AnsibleModule(argument_spec=dict(
        name=dict(type='str', default='*'),
    ))

    name = module.params['name']
    results = []

    if name == "*":
        results = [FACTS[fact](module) for fact in FACTS]
    else:
        results = [FACTS[name](module)]

    #Unify the dictionaries returned from each command into a single dictionary
    facts = dict(functools.reduce(set.union, map(set, map(dict.items, results))))

    module.exit_json(changed=False, ansible_facts=facts)
```

```
from ansible.module_utils.basic import *
if __name__ == '__main__':
    main()
```

Documentation

For each of the ansible modules, there is documentation that is installed. It can be viewed using the `ansible-doc` command. Use `ansible-doc -l` to get a list of all the available modules and a short description.

Ansible uses Jinja2 for Templates. To access the list of filters as well as extensive examples, run `pydoc jinja2.filters`. These templates can also be used for variables, see `ansible all -m setup` for a list of available facts.

Scripts and Other Useful Snippets

Add Group

The following scripts takes an accounts.csv file with headers 'Firstname,Lastname,SAM,Email' and adds them to the current active directory 'Administrators' group.

```
$Users = Import-Csv -Path "accounts.csv"

ForEach($user in $Users){
    Add-ADGroupMember -Identity Administrators -Member $user.SAM
}
```

Bulk Users

The following scripts takes an accounts.csv file with headers 'Firstname,Lastname,SAM,Email' and adds them to the current active directory.

```
$Users = Import-Csv -Path "accounts.csv"
foreach ($User in $Users)
{
    $Displayname = $User.Firstname + " " + $User.Lastname
    $UserFirstname = $User.Firstname
    $UserLastname = $User.Lastname
    $SAM = $User.SAM
    $UPN = $User.Email
    $Password = "CUCyber9."
    New-ADUser -Name "$Displayname" -DisplayName "$Displayname" `
        -GivenName "$UserFirstname" -Surname "$UserLastname" -SamAccountName $SAM `
        -AccountPassword (ConvertTo-SecureString $Password -AsPlainText -Force) `
        -Enabled $true -ChangePasswordAtLogon $true
}
```

Parallel

The following script takes in a file then command parameter in and for every server in the file, it runs the command on that server.

```
#!/bin/sh
file="$1"
cmd="${*:2}"
```

```
while read server; do
  (ssh $server $cmd) &
done <"$file"
wait
```

Services and Applications

Apache

Apache is a one of the most popular web servers with a large variety of features.

Installation

There is a large variety of steps that are important for securing Apache.

- Install Mod Security either from repos or from www.modsecurity.org
- Configure the Apache to use the Mod Security core rules from the repos or www.modsecurity.org
- Remove unnecessary options and text from Apache's httpd.conf file and '/etc/httpd/conf.d' (sometimes located at '/etc/apache2/conf/extra')
- Remove all unnessisary modules entries from Apache's httpd.conf file
- Create an Apache user and group without a shell
- Configure Apache to run using this user and group
- Restrict access to the webserver via the Order allow,deny line in httpd.conf
- Prevent access to root file system
- Allow only read access to web directory '/var/www/html'
- Disable the following functionality if possible:
  - ExecCGI - Allow scripts to be run by apache from this directory.
  - FollowSymLinks - allow the server to follow symlinks
  - SymLinksIfOwnerMatch - has large performance costs.
  - Includes - permists the execution of server side includes
  - IncludesNOEXEC - same as above except prohibit executing scripts
  - Indexes - create an a directory listing in directories without an index.html
  - AllowOverride - allows overrides in '.htaccess' files
  - Multiviews - allows for the same request to ask for multiple files.
- Use RewriteEngine, RewriteCond, and RewriteRule to force HTTP 1.1
- Configure the web server to only server allowed file types.
- Configure to protect from DoS attacks
  - Timeout - set this to a low value like 10 seconds
  - KeepAlive - set this to on (unless RAM is a problem)
  - KeepAliveTimeout - set to 15
  - AcceptFilter http data - require content to open connection
  - AcceptFilter https data - require content to open connection
- Configure to protect against Buffer Overflows
  - LimitRequestBody 64000 - Limit requests to 10k in size
  - LimitRequestFeilds 32 - Limit number of request fields
  - LimitRequestFeildSize 8000 - Limit size of request lines
  - LimitRequestLine 4000 - Maximum size of the request line
- Use Mod_SSL if possible (see openssl section for generating a sever certificate)

- Set ServerTokens to ProductOnly
- Use custom error pages via the ErrorDocument directive
- Remove default files and cgi-scripts
- Do not keep Apache Source after installation
- Ensure that web sever binaries are owned by root
- Allow only root to read the apache config or logs '/usr/lib/apache/{conf,logs}'
- Move apache to a chroot if possible - see below
- Use Mod_Log_Forensic

Chrooting

```
#!/bin/sh
makedir -p /jail/apache/usr/local
cd /usr/local
mv apache /jail/apache/usr/local

echo "SecChrootDir /jail/apache" >> $HTTPD_CONF
/usr/local/apache/bin/apachectl startssl
```

BIND

BIND is a common, featured DNS server. To make it more secure and less vulnerable to attacks, it is recommended to only run BIND as an authoritative nameserver and not as a recursive nameserver.

Config Files

The configuration for BIND is usually stored in either:

- '/etc/bind/' (Debian-based distributions)
- '/etc/named/' (other distributions)
- '/etc/named.conf' (RedHat-based distributions)
- '/var/named/' (RedHat-based distributions)

Utilize the named-checkconf utility to check configuration before applying it.

Example Configuration

Below is a set of example configuration files for securely configuring BIND as an authoritative nameserver with forward and reverse records.

/etc/named.conf

```
options {
   # disable zone transfers
   allow-transfer { "none"; };
   version "none";
   fetch-glue no;

   # if we have another DNS recursor, disable queries and recursion
   allow-query { "none"; };
   recursion no;

   # if we are a DNS recursor, only allow queries
   # from the local network
   #allow-query { 10.0.0.0/24; localhost; };
};

# if we are a DNS recursor,
# set forwarding addresses to another nameserver
#forwarders {
#    8.8.8.8;
#    8.8.4.4;
#};
```

/var/named/example.com.conf

```
# replace example.com with the actual domain
zone "example.com" {
    type master;
    # rhel puts these in /var/named
    file "/etc/bind/zones/db.example.com";

    # allow queries to this zone from anywhere
    allow-query { any; };
};

# 10.0.0.0/24 subnet, put address octets backwards
zone "0.0.10.in-addr.arpa" {
    type master;
    # rhel puts these in /var/named
    file "/etc/bind/zones/db.10.0.0";

    # allow queries to this zone from anywhere
    allow-query { any; };
};
```

/var/named/db.example.com

```
$ORIGIN example.com.

; TTL of 10 minutes for quick change during competitions
$TTL    600

; hostmaster.example.com. is the email hostmaster@example.com
@       IN      SOA     ns1.example.com. hostmaster.example.com. (
                                         1       ; Serial
                                         600     ; Refresh
                                         600     ; Retry
                                         2419200 ; Expire
                                         600     ; Negative Cache TTL
                                                 ; (how long to cache
                                                 ;  negative (e.g. NXDOMAIN)
                                                 ;  responses)
```

```
                                        )
        IN      NS      ns1         ; this box
        IN      MX  10  mail        ; mail box
        IN      A       10.0.0.103  ; www box (resolve example.com
                                              to the same address as
                                              www.example.com)

ns1     IN      A       10.0.0.101
mail    IN      A       10.0.0.102
www     IN      A       10.0.0.103
```

/var/named/db.10.0.0

```
; put address octets backwards
$ORIGIN 0.0.10.in-addr.arpa.

; TTL of 10 minutes for quick change during competitions
$TTL    600

; hostmaster.example.com. is the email hostmaster@example.com
@       IN      SOA     ns1.example.com. hostmaster.example.com. (
                                        1       ; Serial
                                        600     ; Refresh
                                        600     ; Retry
                                        2419200 ; Expire
                                        600     ; Negative Cache TTL
                                                ; (how long to cache
                                                 negative (e.g. NXDOMAIN)
                                                 responses)
                                        )
        IN      NS      ns1         ; this box

; if on a bigger subnet, put octets backwards (i.e. 101.0.0)
101     IN      PTR     ns1         ; 10.0.0.101
102     IN      PTR     mail        ; 10.0.0.102
103     IN      PTR     www         ; 10.0.0.103
```

MySQL

MySql is a quick database for small to medium size organizations.

Installation

Install from the repositories then use the command `mysql_secure_installation`.

Common Tasks

```
-----------------------------
-- Listing Databases and Tables:
-----------------------------
```

```
SHOW DATABASES; -- lists every db, make sure to DROP the ones you don't need
USE openemr; -- selects a certain database so future operations run on it
SHOW TABLES; -- displays every table in the current database


------------------------------
-- Verify all database users:
------------------------------


# mysql -u root -p -- Login with root
-- Type in root password in prompt. If root doesn't have a password, you should set one now:
UPDATE mysql.User SET Password=PASSWORD('new root password') WHERE User='root';
-- Get list of all users:
SELECT Host, User, Password from mysql.User;

--If there are users that shouldn't be there, delete
--them (remember that % and _ are wildcards, % means 0 or more
--characters and _ means exactly one character).

--Delete all bad users. This should include all anonymous users, and any user
--that has a Host OTHER than 'localhost' (especially root!)

DROP USER 'username'@'hostname';
-- repeat for each undesired user
FLUSH PRIVILEGES;
-- run this after you finish deleting users and/or changing user passwords


--If this is at the beginning of competition, you should delete all non
--root@localhost users and only add them back if you need to. Chances are the
--server is set up to allow an anonymous user or a user with root-like access
--and a weak password full control over the database(s), so the best way to
--prevent an intrusion from Red Team is to outright delete these users. You
--(probably) do not need to worry about copying down password hashes, as if
--some application is using MySQL the password will be stored in plaintext in
--that application, and if not then you should be able to submit a Memo to
--White Team to change a user's password. That said, it might be a good idea
--anyways as long as you store it somewhere that Red Team can't get at and it
--isn't against Policy.


------------------------------
-- Creating new users:
------------------------------
--You should only create users with specific access
--to a specific database (e.g. one user per application that uses a database).
--Additionally, you should restrict the Host as much as possible. If your
--webapp is running on the same box as the db server, make the host localhost,
--otherwise make the host the IP of the box running the webapp. ONLY IF REMOTE
--DATABASE ACCESS IS REQUIRED BY THE INJECT should you open up the host to
--something outside of your team's network (e.g. '%')

--Create the database first
```

```
CREATE DATABASE webapp_name;

--Now add a user to it with a secure password:
--With minimal write access (can add/delete records, but not add/drop tables or
--table structures)

GRANT INSERT, UPDATE, SELECT, DELETE ON webapp_name.* TO
    'database_user'@'hostname per above' IDENTIFIED BY 'password goes here';
--With full write access to the given database
GRANT ALL PRIVILEGES ON webapp_name.* TO
    'database_user'@'hostname per above' IDENTIFIED BY 'password goes here';


-----------------------------
-- Get a user's Perms:
-----------------------------
SHOW GRANTS FOR 'user'@'host';

SELECT * FROM mysql.User where User='user' and Host='host';
--If you see a lot of Y's and the user ISN'T root@localhost, something is wrong.


-----------------------------
-- Backing up and restoring the database:
-----------------------------
--This should be in your list of things to do at
--the beginning of competition, as well as semi-frequently throughout when you
--do installations of new webapps, etc. Each command will prompt you to type
--the root password into the terminal. This is safer than providing the
--password in the command line because it does not get saved in .bash_history
--and possibly other places.

-- # is beginning of shell (Linux):
--Backup:
-- # mysqldump --all-databases -u root -p > backup.sql
--Restore:
-- # mysql -u root -p < backup.sql


-----------------------------
-- Reset root password:
-----------------------------
--Stop MySQL
-- # mysqld -u mysql --skip-grant-tables
-- # mysql -u root --Connect as root

UPDATE mysql.User SET Password=PASSWORD('new root password') WHERE User='root';
FLUSH PRIVILEGES;

--to re-load the grant tables and make root and all other users
--have passwords again
```

Securing

- Make sure it is only listening on localhost unless remote access is required by an inject (or the scoring engine) or you are running the webapps on a different server
- Look for `bind-address` in the `[mysqld]` section and ensure it is set to `127.0.0.1` for allowing local connections only or `0.0.0.0` for allowing remote connections
- Disable the `LOCAL INFILE` queries, which allows someone (i.e. red team) to upload files from their computers into your database, by adding `local-infile = 0` to the `[mysqld]` section of the config file
- Restart MySQL after making any configuration changes

OpenSSL

OpenSSL is a toolkit for the TLS protocol and a general purpose cryptography library.

Generate TLS Certificates

Below is a command to generate a key file and TLS certificate for use in Apache or other server. Copy the generated files to the appropriate place (e.g. '/etc/httpd/conf/ssl.key' and '/etc/httpd/conf/ssl.crt') and make them writable only by root and readable by the web server group (e.g. `chown -R root:httpd /etc/httpd/conf/ssl.{key,crt} && chmod 640 /etc/httpd/conf/ssl.{key,crt}`). The output files are `example.pem`, the key, and `example.crt`, the certificate.

```
#!/bin/sh
openssl req -x509 -newkey rsa:2048 -nodes -sha256 -days 365 \
    -keyout example.pem -out example.crt \
    -subj '/O=Example, Inc./CN=example.com'
```

Samba

Samba is reimplementation of the Serial Message Block Protocol from Windows.

Example Configuration

The following configuration allows the machine to authneticate to a Windows AD via Kerberos.

samba.conf

```
# this config file based on the Archlinux Wiki which is published under the GFDL

[Global]
    #Server information
    netbios name = EXAMPLEHOST
    workgroup = EXAMPLE
    realm = EXAMPLE.COM
    server string = %h  Host

    #Authentication
    security = ads
    encrypt passwords = yes
    password server = ad_server.example.com
    idmap config * : backend = rid
    idmap config * : range = 10000-20000

    #Windows domain authentication
    winbind use default domain = Yes
    winbind enum users = Yes
    winbind enum groups = Yes
    winbind nested groups = Yes
    winbind separator = +
    winbind refresh tickets = yes
    winbind offline logon = yes
    winbind cache time = 300

    #New User Template
    template shell = /bin/bash
    template homedir = /home/%D/%U

    preferred master = no
    dns proxy = no
    wins server = ad_server.example.com
    wins proxy = no

    inherit acls = Yes
    map acl inherit = Yes
    acl group control = yes

    load printers = no
    debug level = 3
    use sendfile = no
```

smb.conf

```
[ExampleShare]
  comment = Example Share
  path = /srv/exports/example
  read only = no
  browseable = yes
  valid users = @NETWORK+"Domain Admins" NETWORK+test.user
```

OpenSSH

SSH or Secure Shell is a remote administration protocol. It allows the user to send remote commands to Linux machines (and soon to versions of Windows 10 or later). It can be a vary powerful tool for system administration, but can also be a powerful exploit target if not secured. It is used with a variety of tools, including the provisioning tool Ansible and back up tool rsnapshot for secure connections. In general there are a few best practices to follow for using ssh.

Best Practices

- Disable root login.

- Disable password authentication (default).
- Disable host-based authentication (default).
- Ensure that ssh is not setuid to prevent host-based authentication.
- Use at least public key authentication and use 2-factor authentication where possible.
- During competition revoke all authorized keys except when required by the scoring engine.
- Use sandbox privilege separation to prevent privilege escalation attacks on the daemon (default).
- Use PAM (pluggable authentication module) (default).
- Block excessive connections to ssh at the firewall.
- Do not forward the SSH Agent to untrusted/compromised servers.

Config Files

Important system level configuration directories and files:

- '/etc/sshd/sshd_config' - daemon configuration
- '/etc/hosts.equiv' - used for insecure host based authentication; remove when found
- '/etc/shosts.equiv' - used for insecure host based authentication; remove when found
- '/etc/ssh/ssh_known_hosts' - system wide list of host keys
- '/etc/ssh/ssh_host_*key' - private keys used for host-based authentication and fingerprints
- '/etc/ssh/sshrc' - commands that are executed when the user logs on

Important user level configuration directories and files:

- '~/.rhosts' - used for insecure host based authentication; remove when found
- '~/.shosts' - used for insecure host based authentication; remove when found
- '~/.ssh/known_hosts' - list of hosts that are not already in /etc/ssh/ssh_known_hosts
- '~/.ssh/authorized_keys' - list of keys that can be used to authenticate as this user
- '~/.ssh/config' - per user configuration options for ssh
- '~/.ssh/environment' - environment options for the user
- '~/.ssh/id*.pub' - public key for the user
- '~/.ssh/id*' - private key for the user

Configuration

/etc/ssh/sshd_config

```
PermitRootLogin no
UsePAM yes
UsePrivilegeSeparation sandbox
AcceptEnv LANG LC_*
ClientAliveInterval 300
ClientAliveCountMax 0
```

Purge User Keys

```
for key_dir in $(awk 'BEGIN { FS=":"} {print $6}' /etc/passwd); do
    if [ -d "$key_dir/.ssh" ]; then
        test -f "$key_dir/.ssh/authorized_keys" && \
            mv  "$key_dir/.ssh/authorized_keys" "$key_dir/.ssh/authorized_keys~" &>/dev/null
        test -f  "$key_dir/.ssh/rhosts" && \
            mv  "$key_dir/.ssh/rhosts" "$key_dir/.ssh/rhosts~" &>/dev/null
        test -f "$key_dir/.ssh/shosts" && \
            mv  "$key_dir/.ssh/shosts" "$key_dir/.ssh/shosts~" &>/dev/null
        ls "$key_dir/.ssh/id*" &> /dev/null && echo "found keys at $key_dir/.ssh"
    fi
done
```

Purge System Keys

```
test -f /etc/hosts.equiv && mv /etc/hosts.equiv /etc/hosts.equiv~ &> /dev/null
test -f /etc/shosts.equiv && mv /etc/shosts.equiv /etc/shosts.equiv~ &> /dev/null
```

Do not forget to restart sshd after configuration.

Apache Tomcat

Apache Tomcat is a web server designed to serve Java Server Page (JSP) web applications.

Installation

- Avoid running tomcat with other services
- Remove the sample server files
- Do not reveal excess information
  - Do not advertise version information
  - Disable X-Powered-By HTTP header by setting `xpoweredBy="false"` in the Connectors
  - Disable Allow Trace HTTP header by setting `allowTrace="false"` in the Connectors
  - Disable Client facing Stack Traces
- Protect shutdown port by either disable by setting port to -1 or setting shutdown value to a random value
- Ensure that file permissions are correct
  - Make `$CATALINA_HOME` owned by `tomcat_admin:tomcat` with permissons 750
  - Make `$CATALINA_BASE` owned by `tomcat_admin:tomcat` with permissons 750
  - Make `$CATALINA_HOME/conf` owned by `tomcat_admin:tomcat` with permissons 770
  - Make `$CATALINA_HOME/logs` owned by `tomcat_admin:tomcat` with permissons 770
  - Make `$CATALINA_HOME/temp` owned by `tomcat_admin:tomcat` with permissons 770
  - Make `$CATALINA_HOME/bin` owned by `tomcat_admin:tomcat` with permissons 750
  - Make `$CATALINA_HOME/webapps` owned by `tomcat_admin:tomcat` with permissons 750
  - Make `$CATALINA_HOME/conf/catalina.policy` owned by `tomcat_admin:tomcat` with permissons 600
  - Make `$CATALINA_HOME/conf/catalina.properties` owned by `tomcat_admin:tomcat` with permissons 600
  - Make `$CATALINA_HOME/conf/logging.properties` owned by `tomcat_admin:tomcat` with permissons 600
  - Make `$CATALINA_HOME/conf/server.xml` owned by `tomcat_admin:tomcat` with permissons 600
  - Make `$CATALINA_HOME/conf/tomcat-users.xml` owned by `tomcat_admin:tomcat` with permissons 600
  - Make `$CATALINA_HOME/conf/web.xml` owned by `tomcat_admin:tomcat` with permissons 600
- Use better authentication
  - Configure Realms to not use MemoryRealm in server.xml
  - Configure Realms to use LockOutRealms
  - If possible use Client-Cert Authentication by setting `clientAuth="True"` in server.xml
- Use SSL where possible
  - Ensure that SSLEnabled is set to True for Sensitive Connectors in server.xml
  - Set the scheme to “https” in connectors in server.xml

  - Ensure that secure is set to false on connectors that are not using SSL in sever.xml.
  - Ensure that the sslProtocol is “TLS” for all connectors using SSLEngine in server.xml.
- Configure Logging
  - Ensure the following lines are in logging.properties `handlers=org.apache.juli.FileHandler, java.util.logging.ConsoleHandler`
  - Ensure the following lines are in logging.properties `org.apache.juli.FileHandler.level=FINEST`
  - Ensure that className is set to org.apache.catalina.valves.FastCommonAccessLogValve in `$CATALINA_BASE/<app name>/META-INF/context.xml`
  - Ensure that directory is set to `$CATALINA_HOME/logs` |in `$CATALINA_BASE/<app name>/META-INF/context.xml`
  - Ensure that pattern is set to `"%t % U %a %A %m %p %q %s"` in `$CATALINA_BASE/<app name>/META-INF/context.xml`
- Prevent unexpected code execution
  - Set package.access = `sun.,org.apache.catalina.,org.apache.coyote.,org.apache.tomcat.,org.apache.ja` in conf/catalina.properties
  - Ensure that Tomcat is started with `-security`
  - Ensure that `autoDeploy="false"` in server.xml
  - Ensure that `deployOnStartup="false"` in server.xml
- Protect the manager application
  - Ensure that the valves with the class RemoteAddrValve is set to allow on `127.0.0.1` only in server.xml
  - Ensure that the valves with the class RemoteAddrValve is set to allow on `127.0.0.1` only in “webapps/host-manager/manager.xml” if it must be used
  - Force SSL to access manager if it must be used
  - Rename the manager application by renaming the xml file and moving the app to a new corresponding directory
- Disable insecure startup settings
  - Ensure that `-Dorg.apache.catalina.STRICT_SERVLET_COMPLIANCE=true` is set in startup script
  - Ensure that `-Dorg.apache.catalina.connector.RECYCLE_FACADES=false` is set in startup script
  - Ensure that `-Dorg.apache.catalina.connector.CoyoteAdapter.ALLOW_BACKSLASH=false` is set in startup script
  - Ensure that `-Dorg.apache.tomcat.util.buf.UDecoder.ALLOW_ENCODED_SLASH=false` is set in startup script
  - Ensure that `-Dorg.apache.coyote.USE_CUSTOM_STATUS_MSG_IN_HEADER=false` is set in startup script
- Do not allow symbolic linking in context.xml by setting `allowLinking="false"`
- Do not run applications as privileged in context.xml by setting `privileged="false"`
- Do not allow cross context requests in context.xml by setting `crossContext="false"`
- Do not allow resolving hosts on logging Valves by setting `resolveHosts="false"`

Configuration

```
# remove sample resources
rm -rf $CATALINA_HOME/webapps/{js-examples,servlet-example,webdav,tomcat-docs,balancer}
rm -rf $CATALINA_HOME/webapps/{ROOT/admin,examples}
rm -rf $CATALINA_HOME/server/webapps/{host-manager,manager}
rm -rf $CATALINA_HOME/conf/Catalina/localhost/{host-manager,manager}.xml

# ensure that only needed connectors are configured remove unused connectors
grep "Connector" $CATALINA_HOME/conf/server.xml

# edit the server properties string to hide properties
#tomcat 5.5
```

```
cd $CATALINA_HOME/server/lib
#tomcat 6.0
cd $CATALINA_HOME/lib
#both
jar xf catalina.jar org/apache/catalina/util/ServerInfo.properties
vim org/apache/catalina/util/ServerInfo.properties
jar uf catalina.jar

# disable client facing stack traces
vim error.jsp
# create a error page with out useful information
vim $CATALINA_HOME/conf/web.xml
# add a section that looks like this in the <web-app> element
# <error-page>
#   <exception-type>java.lang.Throwable</exception-type>
#   <location>/path/to/error.jsp</location>
# </error-page>

# configure LockOutRealms
vim $CATALINA_HOME/conf/server.xml
# add a section that looks like this wrapping the main realm
#  <Realm className="org.apache.catalina.realm.LockOutRealm" failureCount="3"
#         lockOutTime="600" cacheSize="1000" cacheRemovalWarningTime="3600">
#    ... MAIN REALM ...
#  </Realm>

# force SSL when accessing the manager application
vim $CATALINA_HOME/{server/,}webapps/manager/WEB-INF/web.xml
# add lines that look like this
# <security-contraint>
#   <user-data-constraint>
#      <transport-guarantee>CONFIDENTIAL</transport-guarantee>
#   </user-data-constraint>
# </security-contraint>
```

Wordpress

Wordpress is a PHP content management system. It has reasonable security in a new default install, but has a poor track record for remote execution exploits. The best way to secure Wordpress is to update it if possible and remove all unnecessary or old plugins.

Setup

Download the latest tarball available at 'https://wordpress.org/latest.tar.gz' and untar it into the document root (i.e. '/var/www'). Create and configure the necessary using the following SQL commands.

```
CREATE USER wordpress@localhost IDENTIFIED BY 'password';
CREATE DATABASE wordpress;
GRANT SELECT,INSERT,UPDATE,DELETE ON wordpress.* TO wordpress@localhost;
FLUSH PRIVILEGES;
```

Navigate to the setup page at ‘http://localhost/wordpress/’ and follow the setup instructions. Proceed below with how to add a few extra layers of security to a Wordpress installation.

Securing

- Make sure file permissions are restrictive
  - ‘/’ needs to be writable only by the owning user account (e.g. ‘www’)
  - ‘/wp-content’ needs to be writable by web server (e.g. ‘apache’)
  - ‘/wp-content/plugins’ needs to be writable only by the owning user account (e.g. ‘www’)
- Remove unnecessary database permissions
  - reduce database permissions for the SQL user by running the following command, replacing `wordpress.*` with the Wordpress tables and `wordpress@localhost` with the Wordpress user if necessary.
  - `REVOKE ALL PRIVILEGES ON wordpress.* from wordpress@localhost;`
  - `GRANT SELECT,INSERT,UPDATE,DELETE ON wordpress.* TO wordpress@localhost; FLUSH PRIVILEGES;`
- Disable file editing from wp-admin
  - add `define('DISALLOW_FILE_EDIT', true);` to ‘wp-config.php’
- Move ‘wp-config.php’ to the directory above the Wordpress root
- Add ‘AskApache Password Protect’ which enables HTTP authentication preventing wp-admin from being exploited

Sniffers and Information Gathering

Nmap

Nmap is a network exploration tool, port scanner, and service scanner. It is useful for preforming host enumeration on IPv4 networks and auditing ports and services on specific hosts on IPv4 and IPv6 networks.

Common Options

There are several main use cases and common options here:

```
#!/bin/sh
# most common options

# ping scan of a IPv4 network
nmap -sn 192.168.1.0/24

# agressive scan of a IPv4 host
nmap -T4 -A 192.168.1.1/32

# agressive scan of a hostname resolved via dns
nmap -T4 -A www.foobar.com

# agressive scan of a IPv6 host
nmap -6 -T4 -A 2::dead:beaf:cafe/128

# agressive scan of a file of hosts/networks separated by newlines
nmap -T4 -A -iL inputfile.txt
```

Host Discovery Options

For host discovery, the most important flag is `-sn`. It sends an ICMP ECHO to each target host. In IPv4 networks, this is a fast and easy way to enumerate hosts for a deeper scan.

In IPv6 networks, the address space is probably too large to do this effectively. On solution in this case to examine the network switch MAC table or to use tcpdump or wireshark to sniff for packets.

To conduct discovery using different types of packets use the `-P{n,S,A,U,Y}` option which uses no pings, SYN, ACK, UDP, and SCTP packets respectively.

Port Scanning Options

By default, Nmap scans the 1000 most commonly used ports. To use Nmap to scan for specific ports, use the `-p` flag to specify which ports to scan. It accepts hyphen separated ranges and comma separated lists. To scan all ports, use the –allports long option. To use a different type of packets use the `-s{S,T,A,W,U,Y}` option which tests with SYN, TCP connect, ACK, UDP, and SCTP INIT packets respectively.

Service Scanning Options

There are several commons flags to use here:

- `-O` will run OS detection against the target
- `-sV` will run service version detection against the target
- `-sC` will run common default scripts against the target to detect various things
- `--script="<script_name>"` will specify a script or group of scripts to run against the targets
- `-A` will enable OS detection, version detection, script scanning, and traceroute

Scripts that are available can often be found the '/usr/share/nmap' directory. Refer to these for examples on how to write scripts.

Timing and Optimization

Nmap has a series of timing and optimizations that can be run. The most useful is -T[1-5] which specifies how quickly packets are to be sent, 1 is the slowest and 5 is the fastest. You can also specify max retries via the –max-retries long option. You can also specify max timeout via the –host-timeout long option.

Evasive Options

If you are running Nmap offensively, there are several flags that control how evasive Nmap behaves. These allow for spoofing of IP address (`-S`) and MAC address (`--spoof-mac`) and for setting various options for sending custom packets.

Output Options

There are various output options the most important are:

- `-oN <file_name>` will send normal output to a file
- `-oG <file_name>` will send grep-able output to a file
- `-oX <file_name>` will output XML to a file

Appendix

The MIT License

Copyright (c) 2016, CU Cyber cyber@clemson.edu

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the “Software”), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED “AS IS”, WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.