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 your 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 sudo and /etc/sudoers.d
- Harden the service for your machine
- Install/Configure a firewall
- Write an audit report containing changes made
- Configure ansible

For all systems:

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

Windows Checklist

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

For each System:

- Change the password for the admin accounts
- Remove any nonessetial 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

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
```

Firewalls and Configuration

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 occaision 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 with the enp0s3 interface. It allows HTTPS traffic defined in the /usr/lib/firewalld/services/https.xml or overwritten 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

```
#!/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 user space 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 user space 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/ 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                       *
```

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
```

Appendix

The MIT License