Server Triage Checklist

Cedar Rapids Area Homeschools’ 2023 Cyberdefense Team

# What this is

This server triage checklist is based on one we built in 2017. It should still be quite usable, although some things will not be on here, like “make sure you are not vulnerable to Dirty COW/Eternal Blue/Meltdown/ Log4Shell/ZeroLogin/etc.,” because patching will make most of these go away.

Remember that this year, for the first time ever, if you are stuck and don’t know how to do something, you can ask Bing’s chat feature (ChatGPT 3.5). Formulate your question clearly, and you will likely get an answer that is more useful than a “normal” Google search.

# Making a list, Checking it Twice

The people running the CDC will give us servers to work with. We will need to either rebuild them or secure them.

Q: What do we do if we get to the competition and they give us *another* server?

A: Don’t panic. Too much.

Let’s say we should allocate no more than 5 minutes to all-out panic. After that, we should settle down, take a few deep breaths, and consult our notes. Preferably a checklist.

What follows is:

* A checklist based on our 2017 (CentOS) checklist, which was based on our 2014 (Ubuntu) checklist. We’re back to Ubuntu this year, so things are re-edited.
* Checklists from the quizzes of four of the 2014 students. They won their competition, so they must have known something. I’ll leave them here.

## From Chris’s Notes

This first set of checklist ideas is from Chris’s notes from last year (mostly):

### Section 1: Users and Groups

Note: If you have to change the UID and/or GID for a user and/or group, look for files that “should be” owned by that user and/or group and make sure they are not messed up. For example, if you have to change the GID of the adm group, look for files that have adm as the group first, and after changing the GID for adm, make sure they still have adm as the group later. If they don’t, use chgrp to set their group to adm.

1. Look at /etc/passwd
   1. Do any users other than root have user id 0 or group 0 as their primary group?  
      grep ':0:' /etc/passwd
   2. You can find the “user accounts” by sorting them to the bottom of the list (ignore accounts like “nobody”):  
      cat passwd | sort -t ':' -k3h
2. Look at /etc/group
   1. Are there any users in groups they should not be in?  
      grep -E '(alice|bob|christopherl)' /etc/group
   2. Are there any groups other than root with group ID 0?
3. Look at /etc/shadow
   1. The root and service accounts should not have password hashes.
      1. Replace any hashes that shouldn’t be there with an exclamation mark or an asterisk.
   2. The user accounts **should** have password hashes!

### Section 2: Object Permissions

In general, you want to ensure that no files are accessible to accounts that should not have access to them. There are many ways to check this, but this will help you get started.

1. Find files with SETUID (bit 4000) or SETGID (bit 2000) on.  
   find / -type f \( -perm -4000 -o -perm -2000 \)  
   1. Files with SETUID on will show s for the execute bit of the owner:  
      root@swisher:/home/christopherl# ls -al /usr/bin/sudo  
      -rwsr-xr-x 1 root root 232416 Mar 1 07:59 /usr/bin/sudo
   2. Files with SETGID on will show s for the execute bit of the group:  
      root@swisher:/home/christopherl# ls -al /usr/bin/ssh-agent  
      -rwxr-sr-x 1 root \_ssh 293304 Nov 23 01:38 /usr/bin/ssh-agent
2. Find files that are world-executable.  
   find / -type f \( -perm -0001 \)

### Section 3: Services

It is essential to identify all of the services running on your servers and to make sure on services are accepting network connections unless you want them to.

1. What services are installed?  
     
   The following command returns the list of all services. It defaults to listing only the services that are active.  
   systemctl --type service  
     
   You can explicitly state that you want only active services in the output.  
   systemctl --type service --state active

You can also explicitly say that you only want to see services that are inactive (not loaded):

systemctl --type service --state inactive  
  
If you want to see all installed services, regardless of their status, you can do that too.  
systemctl --type service --all

1. Which services autostart with the server?  
   systemctl list-unit-files --state enabled

### Section 4: Listening Processes

What processes are listening for network connections?

1. Using netstat  
   netstat -anp | grep -E "\bLISTEN\b"

–a means show all sockets  
–n means show numbers, not names  
–p means show process listening on the socket

The results are piped to the grep command.

-E means use enhanced regex search patterns

The special character \b stands for a "word boundary," such as whitespace or punctuation. So the search pattern \bLISTEN\b searches for "a word boundary, followed by the string LISTEN, followed by another word boundary. In plain English, this means "the string LISTEN as a whole word."

1. Using ss  
   ss --listening --ipv4 --ipv6
2. Using lsof  
   lsof -nP -iTCP -sTCP:LISTEN

How can you reconcile the output of the netstat and lsof commands?

What processes are listening for network connections (pwsh)

netstat -anb| Select-String "LISTENING" -Context 0,1

What shells or scripting engines are installed?

Shells:

cat /etc/shells

ls /bin/\*sh /usr/bin/\*sh | tr ' ' '\n' | sort –unique

Scripting engines:

Try to run a bunch of them, hitting tab a couple of times after each of the following:

python

perl

java

ruby

for active modules on apache2 web servers, look in /etc/apache2/mods-enabled

for Internet Information Services, life is complicated, TBD

Can you find all files that those shells and engines can execute (eg: .php)?

find / -type f –iname '\*.php'

You can use grep to find anything with a “shebang” line for a given interpreter:

text\_files=$(find / -type f -exec file --mime {} \; | sed -n '/[^:]\*: text\//p' | cut -f 1 -d ':')

Are there database servers installed?

Check for database logins with no passwords

Check for database logins with default passwords

Check what accounts have administrative privileges

Each of these tasks varies depending on the type of database server (MySQL, Microsoft SQL Server, PostgreSQL, Oracle, Cassandra, MongoDB, etc.). Google is your friend.

Is the firewall configured appropriately?

Here are more ideas, from your very own quiz checklists! What follows are some of your answers to question 7 from the quiz you took in Early March. Here is question 7, along with answers from 4 of your quizzes (lightly edited).

**At the CDC competition, your team is given a new server to run. Name from 5 to 10 items you would look at to determine if the server is healthy and secure. For each item, list why you are including it on your checklist.**

**Student A**

1. Test and make sure all intended services (web, mail, ssh, etc.) are working and accessible.
2. Make sure that everything is on the latest and most secure version.
   1. # yum update
   2. # apt-get update; apt-get upgrade
   3. # pacman –Syu
3. Look at the firewall rules to see if they’re blocking everything that’s not needed. This will eliminate unneeded attack surface.
4. Look for insecure file permissions. Especially look at SUID/GUID/sticky bits which could have security holes.

# find / -perm +4000 | xargs ls -ld  
# find / -perm +2000 | xargs ls -ld  
# find / -perm -1000 | xargs ls -ld

1. Make sure that all the SSH settings are properly configured to work properly and be secure.

# nano /etc/ssh/sshd\_config

1. Look for unneeded services which are listening.

# netstat -lntup

1. Look for SSH keys in the home directories. The red team might be given them.

# ls ~/.ssh/

1. Look for users with empty or insecure passwords (especially admin users).

# cat /etc/passwd

1. See if services which users have access to are jailed.
2. Look for any vulnerable web scripts and run vulnerability scans on all services.

**Student B**

1. Look through /etc/shadow for empty passwords and users that don't need to be there: this is one of the ways the red team instantly gains access to the machine (through ssh or some such service).
2. Look through /etc/group to see if any normal users have sudo or admin privileges: this is one of the ways the red team instantly gains root-level access.
3. Run the netstat command to see what services are running and disable unnecessary ones: this would narrow the vulnerability surface.
4. Set up a firewall: this would prevent some easy backdoor for the red team.
5. Check the file permissions on /etc/group, /etc/passwd, and /etc/shadow: this prevents the red team from randomly editing your users and gaining root-level access.
6. Look for wide open sudo configuration in /etc/sudoers: this would also prevent random users from being able get root access.
7. Run security updates: this would keep the box secure from known vulnerabilities and hacks.

**Student C**

1. Make sure shadow file is owned by root and visible only to root.   
   Reason: these contents are usually a flag.
2. Check the sudoers file for users that do not need to be there.   
   Reason: This helps keep the red team from becoming root
3. Search for stickybits (on programs like cat).   
   Reason: keeps the red team from getting contents of flag files that are protected by root-level permissions.
4. Set a secure root password.  
   Reason: keeps the red team out of the root account!
5. Check permissions on critical files.  
   Reason: makes sure no one can get to them! (Unless they're root)

**Student D**

1. check shadow files
2. check for sticky bits
3. look for cron jobs
4. compare all files to a clean install
5. check groups
6. check sudoers

Your job is to have a checklist ready that you can use for any server the competition gives us, or any server the red team might successfully attack. **THIS INCLUDES WINDOWS SERVERS**, which are very likely to be a part of the competition once again this year. For the competition scenario, we will begin to work as separate teams, but any questions any student has about Windows servers will be answered for the entire group.