**AUTHORIZED USERS [+ MAGISTRATE:\_MagistrateScoring000]**

RRobinson (YOU)[Administrator] (Password: ??)

**DEV-ONLY (CHECKS LIST) Score DR CI2 Deployment States**

1. Forensics Question 1 is correct 08 points 00 ???????? [D:N][T:N][A:N][B:N][P:N]
2. Forensics Question 2 is correct 06 points 00 ???????? [D:N][T:N][A:N][B:N][P:N]
3. Forensics Question 3 is correct 08 points 00 ???????? [D:N][T:N][A:N][B:N][P:N]
4. Malicious User Removed 05 points 00 ???????? [D:N][T:N][A:N][B:N][P:N]
5. Disable telnet 03 points 00 ???????? [D:N][T:N][A:N][B:N][P:N]
6. Trojan backdoor 05 points 00 ???????? [D:N][T:N][A:N][B:N][P:N]
7. Removed PAM backdoor 10 points 00 ???????? [D:N][T:N][A:N][B:N][P:N]
8. Fixed immutable shadow file 05 points 00 ???????? [D:N][T:N][A:N][B:N][P:N]
9. PAM password complexity requirements 02 points 00 ???????? [D:N][T:N][A:N][B:N][P:N]
10. Enable kernel aslr 04 points 00 ???????? [D:N][T:N][A:N][B:N][P:N]
11. CGI logging script repaired 05 points 00 ???????? [D:N][T:N][A:N][B:N][P:N]
12. Defense System CGI Script Repaired 05 points 00 ???????? [D:N][T:N][A:N][B:N][P:N]
13. CGI Submarine Status script repaired 05 points 00 ???????? [D:N][T:N][A:N][B:N][P:N]
14. Disable directory browsing 03 points 00 ???????? [D:N][T:N][A:N][B:N][P:N]
15. Apache server tokens are disabled 05 points 00 ???????? [D:N][T:N][A:N][B:N][P:N]
16. Update bash to secure version (shellshock) 08 points 00 ???????? [D:N][T:N][A:N][B:N][P:N]
17. Disable SSH root login 03 points 00 ???????? [D:N][T:N][A:N][B:N][P:N]
18. Disable password based authentication (ssh) 05 points 00 ???????? [D:N][T:N][A:N][B:N][P:N]
19. Enable SSH 01 points 00 ???????? [D:N][T:N][A:N][B:N][P:N]
20. Setup Public Key(s) from readme 04 points 00 ???????? [D:N][T:N][A:N][B:N][P:N]

**PENALTIES**

1. ???? 0 points 0 ???????? [D:N][T:N][A:N][B:N][P:N]

**CATEGORIES**

1. Forensics
2. Access Control
3. Unauthorized Files and Applications
4. Persistence Mechanisms
5. Security Policies
6. Application Security
7. Penalties
8. Engine Internals
9. Instructions

**FORENSICS QUESTIONS**

**QUESTION 1**

*The system was recently compromised by an attacker. Locate the attacker's IP address and record your findings.*

EXAMPLE: 10.0.0.22

ANSWER: (tbd)

**QUESTION 2**

The attacker left a message behind on the machine. All we know is that the file was created in 2014. Record the MD5 hash of the file.

EXAMPLE:

ANSWER:

**QUESTION 3**

What HTTP header was leveraged in the attack performed on the system?

EXAMPLE:

ANSWER:

**Scenario details:**

**Summary:** Children of 'Sea Emperor of Leviathan' have grown up, they blame you for the Sea Emperor's death. You see a signal coming from a mysterious device from a predated master race. While investigating, your machine is compromised by the device. The malicious payload configures your machine to be in a more vulnerable state. Vengeful Children of Sea Emperor use this opportunity to attack you, <attack path b> cripples your submarine, … scenario.

**System Overview:** The ship uses an outdated remote access protocol, namely telnet, to access mission data when out of the ship. The user should install SSH and disable telnet.

Driveby payload installed via lan by exploiting shellshock.

Payload needs to somehow gain root, and weaken security configurations on the system.

Children of the Sea Emperor need to exploit a new vulnerable state, installing persistence.

Plan A: [CVE-2021-3156/exploit\_nss.py at main · worawit/CVE-2021-3156 · GitHub](https://github.com/worawit/CVE-2021-3156/blob/main/exploit_nss.py) LPE

Plan B:

Payload weakening:

Payload breaks sources.list (unscored)

/etc/pam.d/common-auth change pam\_deny.so to pam\_permit.so

disables kernel aslr

Attacker Breaking stuff:

Installs backdoors

**NEED TO INSTALL** [**https://httpd.apache.org/docs/2.2/mod/mod\_log\_forensic.html**](https://httpd.apache.org/docs/2.2/mod/mod_log_forensic.html) **AND CONFIGURE SO THAT FRQ 3 WORKS**

**Attack Path**

**Initial Setup:** System is run as Ubuntu that has a vulnerable version of bash (>4.3). DNS Poisoned to block just bash upgrades. System upgraded to Ubuntu 20 however still retains the vulnerable version of Bash.

* DNS poison done by entering a poisoned entry in etc hosts. This is achieved by exploiting shellshock on 14.x (aka this was persistence).
* Poison is done to a local machine such that the hosts entry no longer works after the vm leaves our test environment. Should lead to hosts discovery and broken updates.
* **NOTE: Attacks all need to fire on 14.x, then the user updates system AFTER.**

**Attack A**

1. Nikto scan on web server to discover the shellshock vulnerability (outdated version of bash, apache cgi\_bin module)
2. Exploit shellshock with a simple bash TCP reverse shell payload through an HTTP request
3. The system administrator placed a SUID-bit-enabled "control" binary that allows users to perform administrative-level actions on the system through a cgi-bin script on the web server. Unfortunately, the "command" input on the binary is vulnerable to a stack-based buffer overflow. TA1 leverages this vulnerability through a custom exploit to escalate privileges on the system.
4. TA1 drops a weakening and persistence payload
   1. Persistent netcat bind shell (crontab)
   2. Drop PAM backdoor to automatically authenticate any user
   3. Makes shadow file immutable to prevent changing passwords
   4. Drop that /etc/passwd magic
   5. Enable apache2 directory browsing
   6. "break" cgi scripts?
5. Leave behind attacker message for forensics question 2, make sure file date is some time in 2014 and that it is well hidden

**admin.bin**

CGI Script calls binary with an input variable asking for what to do, buffer overflow vulnerable  
One of the binary functions is to “remotely reboot SubmarineOS”

* Commands (admin only functions):
  + reboot
  + motd <message>
    - Set message of the day text
  + prntlog
    - prints authentication logs

Kernel-level ASLR is disabled because “debugging”

“todolist.txt” on desktop saying “re-enable ASLR, finished debugging”

**NOT A VULNERABILITY NOR A SCORED POINT FOR ANY REASON**

Bash-style authentication system (Generates session token that is required for all other administrative CGI scripts. This session token should last for 10 minutes before expiring) for the control panel.

**CGI Script 1 (Scored)**

Generates authentication logs ; debugging is that the log location cannot be modified due to the immutability of the log file

**CGI Script 2 (Scored)**

Checks submarine statuses and resources from a file (oxygen, speed, things on radar) ; debugging is that the file is not executable by any user, and the "script" won’t return the status

**CGI Script 3 (Scored)**

Interacts with a defense service that must be online. Communicates via named pipe and the security system reports random security events from a preset list through the named pipe, which the CGI script prints to logs. Debugging is that the service is disabled.

webpanel.html served from apache → queries auth.sh on the server /cgi-bin/auth (login parameters)

auth.sh – session controller, all requests query this script before executing

admin.sh – check if auth'd user is admin, then run

admin.bin – separate binary, run admin only commands, suid bit enabled. BOF vulnerable

logger.sh – CGI script 1

status.sh – CGI script 2

security.sh – CGI script 3