Ingeniería Eléctrica
Facultad de Ciencias Físicas y Matemáticas
Universidad de Chile
EL4107-1 Information and Communication Technologies

PROJECT REPORT 2

Nested Networks and Pivoting Laboratory

Name: Aarón Molina

Professor: Shaharyar Kamal .

Teaching Assistant: Ariel Núñez Lobos

Submission date: 15 de noviembre de 2025
Santiago, Chile

Index

1. Introduction

1.1. Project Context

In contemporary offensive security and penetration testing, professionals increasingly encounter highly segmented corporate environments. The ability to perform lateral movement across multiple network layers has become an essential competency for cybersecurity practitioners. This project directly addresses this challenge through the creation of a specialized laboratory focused on **advanced pivoting** and **nested network** exploitation.

1.2. Real-World Applications and Importance

Nested network architectures represent a fundamental characteristic of modern IT infrastructure rather than merely a theoretical concept. In **cloud environments**, providers implement Virtual Private Clouds (VPCs) within physical data center networks, creating inherent nesting. **Managed service providers** routinely deploy isolated customer networks within their infrastructure, while **enterprise organizations** segment departments and security zones through multiple network layers. This laboratory specifically addresses the security challenges inherent in these real-world nested architectures, including **honeypot deployment** where controlled deception environments detect and analyze attacker behavior across multiple network segments.

1.3. Problem Statement

Modern organizations implement complex network architectures with multiple segmentation levels, yet many security professionals lack practical environments to develop and refine lateral movement techniques. This gap between theoretical knowledge and practical application motivated the development of a comprehensive laboratory that simulates real-world conditions for penetrating stratified network infrastructures commonly found in enterprise environments.

1.4. Technical Objectives

The primary objectives of this project encompass:

- Designing and implementing a multi-tier network architecture replicating realistic corporate scenarios
- Documenting and validating advanced pivoting techniques across firewalls and network segments
- Developing practical exercises for TCP tunneling, service redirection, and proxy chaining
- Establishing a reproducible framework for cybersecurity education and red team training

1.5. Educational Significance

This laboratory serves as a crucial bridge between theoretical network security concepts and hands-on penetration testing experience. It provides telematics and cybersecurity students with

essential practical skills in network persistence, lateral movement, and infrastructure exploitation—competencies vital for modern security assessments and defensive strategy development in increasingly segmented network environments.

2. Key Concepts and Technologies

2.1. Network Tunneling Fundamentals

This laboratory employs essential tunneling techniques that manipulate TCP/IP protocols to overcome network segmentation:

2.1.1. Chisel - Application Layer Tunneling

Creates persistent TCP tunnels with SOCKS5 proxy capabilities, demonstrating protocol encapsulation and reverse connection establishment for bypassing firewall restrictions.

2.1.2. SOCAT - TCP Stream Redirection

Provides simple TCP port forwarding and protocol translation, showcasing direct stream manipulation between network segments.

2.2. Traffic Routing and Monitoring

2.2.1. ProxyChains - Multi-Hop Proxying

Enables traffic routing through multiple proxy servers, illustrating application-layer routing and traffic obfuscation techniques.

2.2.2. ICMP Exfiltration - Covert Channels

Utilizes ICMP protocol for data exfiltration, demonstrating protocol misuse and firewall evasion through ping-based covert communication.

2.3. Remote Code Execution (RCE)

The core objective enabling arbitrary command execution on target systems, facilitating persistence, privilege escalation, and lateral movement within segmented networks.

These concepts provide practical understanding of TCP/IP protocol manipulation, network segmentation challenges, and real-world telematics security applications.

3. Network Architecture and Environment

3.1. Virtualization Platform

All infrastructure operates on VirtualBox virtual machines, with the Debian server configured in **bridge mode** for direct LAN connectivity.

3.2. Network Segmentation

- **Physical LAN (192.168.1.0/24)**: External network for initial access
- **iNET Internal Network (10.0.2.0/24)**: Isolated segment requiring pivoting
- **Docker Virtual Networks**: Application-level isolation within Ubuntu server
  - 172.17.0.0/16 (default bridge)
  - 172.20.0.0/16 (lab-network)
  - 172.25.0.0/16 (lab-network2)

3.3. Component Architecture

- **Debian Gateway**: Dual-homed VM (192.168.1.7 / 10.0.2.2)
- **Ubuntu Server**: iNET host (10.0.2.15) with Docker virtual networks
- **Attacker Machine**: Physical host (192.168.1.4) on external LAN

3.4. Security Implementation

- **Container Firewall**: iptables rules implemented in `infranet-container2384234` restricting outbound TCP traffic to ports 22/80 only
- **Reverse Proxy Configuration**: Ubuntu Server provides service exposure through:
  - Port 80 reverse proxy to container web services
  - SSH port forwarding (22) to container SSH daemon
  - SOCAT-based TCP redirection for service publishing
- **Network Isolation**: Docker containers operate behind Ubuntu Server, requiring explicit port forwarding for external access

This architecture demonstrates progressive isolation from physical LAN through internal networks to container-level segmentation. A better visualization of the architecture can be found at the following link: https://excalidraw.com/#json=Z-M9iLBC1mod8wyPWliWf,A9AFWFELWnlVxrH9QRepuw

4. Services and Server Configuration

4.1. Debian Server Services

- **Web Server**: Apache2 on port 80 with PHP support and virtual hosting
- **File Transfer**: vsftpd service on port 21
- **Remote Access**: SSH service for administrative access
- **Database**: MySQL server supporting web applications
- **Security**: Fail2Ban protection against brute-force attacks

4.2. Ubuntu Server Services

- **Service Redirection**:
  - Port 80: Reverse proxy to container web services
  - Port 22: SOCAT redirection to container SSH
  - Port 2220: Direct SSH access to Ubuntu host
- **Container Management**: Docker hosting multiple containers including distraction containers

4.3. Container Services

- `infranet-container2384234`:
  - Apache web service with PHP execution panel
  - SSH service for remote access
  - iptables firewall restricting outbound traffic
  - Supervisord for service management
- `shellshock-vuln`: Container vulnerable to RCE via Shellshock
- Additional distraction containers for realism

4.4. Automation and Maintenance

- **systemd Services**: Auto-start for firewall, SOCAT, and web services
- **Cron Jobs**: Automated log cleaning and PHP session maintenance
- **Container Persistence**: Auto-restart policies for all critical containers

Linux servers without GUI were selected to minimize resource overhead and optimize performance for networking services.

5. Attack Flow Overview

5.1. Initial Compromise

The attack begins with a reflected XSS vulnerability, allowing session hijacking through cookie theft. A Python bot simulates human interaction to trigger this vulnerability, granting administrative access to the Debian server.

5.2. Privilege Escalation

As administrator, log-poisoning is performed by intercepting HTTP requests with Burp Suite and modifying headers to inject PHP code into system logs, achieving initial Remote Code Execution on the Debian server.

5.3. Network Discovery

Using ping for host discovery and TCP sockets for port scanning, internal network hosts and open ports are identified. The Ubuntu server is found to be exposing a command execution panel that actually belongs to the Docker container (infranet-container2384234), but reverse shell attempts fail due to the container's internal firewall restrictions.

5.4. Lateral Movement

ICMP exfiltration is employed by encoding files in hexadecimal within ping packets, using an ICMP sniffer to extract readable files from the firewalled Docker container. This technique provides access to the container's filesystem, enabling pivoting to the Shellshock-vulnerable container within the same Docker network.

5.5. Final Compromise

Through the Shellshock container, the Ubuntu server's SSH private key (id_rsa) is discovered. Using SOCAT for traffic redirection, the key provides complete access to the Ubuntu host, compromising the entire environment. The steps in this flow are briefly outlined in this link: https://excalidraw.com/#json=yaW3pqDLkx8dL1s4wgwRR,uGVPnMghKMIwneT5eb8CRw If this part is not very clear, the accompanying video explains the intrusion flow well. https://drive.google.com/drive/folders/1ytvDzC8MSkW3FPSdQ8DvI4-QxsTSCAb2?usp=sharing

6. Technical Implementation and Protocol Analysis

6.1. Protocol Manipulation and Network Architecture

This laboratory demonstrates advanced networking concepts through practical protocol manipulation. The implementation successfully created a multi-layered network environment where TCP/IP stack exploitation enabled persistent connections through firewalls using Chisel tunneling, while SOCAT provided precise TCP stream redirection between network segments. At the application layer, techniques including HTTP/S cookie hijacking, SSH port forwarding, and ICMP-based covert channels demonstrated comprehensive protocol understanding and manipulation.

The container networking architecture employed Docker's advanced networking capabilities, creating isolated segments through custom bridge networks with static IP assignment. This allowed for sophisticated service exposure patterns where the Ubuntu server acted as a reverse proxy gateway, managing controlled port publishing while maintaining internal service discovery between containers across different network segments.

6.2. Security Implementation and Infrastructure Management

Multiple security layers were implemented and tested, creating a realistic enterprise environment. Network security controls included iptables firewall rules and service hardening with Fail2Ban, while application security measures provided both protection mechanisms and valuable attack vectors for comprehensive security testing. The infrastructure demonstrated enterprise-grade management through Supervisord service orchestration and systemd integration, ensuring reliable autostart configurations and service resilience through container lifecycle management with stable network persistence.

7. Results and Validation

7.1. Network Segmentation and Traffic Flow

The laboratory successfully validated its core objective of creating an effective nested network architecture. Docker virtual networks operated seamlessly within the internal iNET network, which maintained proper isolation from the external LAN segment. This created a genuine nested environment where virtual container networks existed within physical network segments, demonstrating correct hierarchical network segmentation.

Traffic flow analysis confirmed that network communication occurred only through authorized pathways, with the Debian gateway properly routing between external and internal networks. The Ubuntu server correctly forwarded traffic to Docker containers while maintaining segmentation boundaries across all service redirections, including web traffic on port 80 and SSH connections.

7.2. Pivoting Techniques and Objective Achievement

The attack flow successfully demonstrated that deep TCP/IP protocol knowledge enables effective traffic redirection between segmented networks. Chisel tunnels established reliable SOCKS5 proxies through the Debian gateway, while SOCAT redirections properly exposed container services through the Ubuntu host. ICMP exfiltration proved particularly effective for data movement through firewalled segments, showcasing practical protocol manipulation for lateral movement.

All technical objectives were comprehensively met, including multi-tier network architecture implementation, advanced pivoting technique validation, and practical exercise development for tunneling through progressively restricted network segments. The laboratory confirmed that proper understanding of network protocols enables successful navigation through complex segmented environments.

8. Network Security Analysis and Recommendations

8.1. Security Implications of the Architecture

The nested network architecture used in the laboratory combines physical segmentation, virtual networks and Docker-based container isolation. While this multilayered design establishes a form of defense-in-depth, it also increases the number of interconnected attack surfaces. The attack chain demonstrated that a single weakness in the web layer—specifically a reflected XSS vulnerability—can propagate across these layers and ultimately compromise the underlying host.

The complexity of reverse proxies, SOCAT port forwarding, and container network bridges also creates implicit trust relationships that may be abused if controls are inconsistent. Limited traffic visibility across segments further enables covert channels, such as ICMP-based exfiltration, to pass unnoticed.

8.2. Identified Weaknesses

The compromise revealed several critical deficiencies within the environment:

- **Web Layer Issues:** Weak cookie protection and insufficient input sanitization enabled successful session hijacking.
- **Log Poisoning and Code Execution:** User-controlled data was written to Apache logs without sanitization, enabling log poisoning and remote code execution.
- **Container-Level Security Gaps:** Outbound TCP restrictions were insufficient because ICMP traffic remained unfiltered, enabling data exfiltration despite firewall rules.
- **Credential Exposure:** The presence of the host's SSH private key inside a Docker container facilitated privilege escalation beyond the application layer.
- **Lack of Monitoring:** No mechanisms were present to detect unusual container behavior, suspicious network traffic, or multi-stage attack patterns.

8.3. Recommendations

To enhance the security posture of the laboratory while maintaining functional flexibility, the following measures are recommended:

- **Application Hardening:** Enforce `HttpOnly`, `Secure` and `SameSite=Strict` cookie attributes, implement a restrictive Content Security Policy (CSP), and sanitize all user-controlled input before logging.
- **Safer Log Management:** Store logs outside web-accessible directories, restrict permissions to prevent script execution, and ensure that HTTP headers cannot be written directly into log files without sanitization.
- **Network and Container Controls:** Apply strict egress filtering including ICMP, deploy AppArmor/SELinux profiles for containers, avoid storing private keys inside containers, and apply least-privilege models such as rootless Docker.
- **Monitoring and Detection:** Implement centralized log correlation, anomaly detection for low-level protocols, and continuous auditing of internal container activity.

Overall, while the layered architecture provides segmentation benefits, its effectiveness depends on the consistent application of security controls across all layers. Strengthening monitoring, improving network restrictions, and hardening application components significantly improves resilience without reducing the educational value of the laboratory environment.

9. Conclusion

This project successfully designed, implemented, and validated a sophisticated nested network laboratory that effectively demonstrates advanced pivoting techniques across segmented environments. The multi-layered architecture, combining physical LAN segmentation with virtual Docker networks, provided a realistic platform for practicing lateral movement strategies essential in modern cybersecurity operations.

The laboratory confirmed that comprehensive understanding of TCP/IP protocols and network stack manipulation enables successful navigation through complex segmented infrastructures. Techniques including Chisel tunneling, SOCAT redirection, and ICMP exfiltration proved effective for maintaining persistence and moving through progressively restricted network segments, validating the core hypothesis that protocol knowledge translates to practical penetration capabilities.

Beyond technical implementation, this project serves as an valuable educational bridge between theoretical networking concepts and real-world security challenges. The hands-on experience with service orchestration, container networking, and security control evasion provides students with practical skills directly applicable to contemporary IT environments and cybersecurity roles.

The successful demonstration of end-to-end network penetration—from initial external access through multiple pivoting stages to complete environment compromise—validates both the laboratory design and the educational methodology. This project establishes a foundation for continued development of advanced security training environments that keep pace with evolving network architectures and defense mechanisms.