

Project Report: ZX301 Security Framework (v5.0)



Version: 5.0 (Parallel Edition)

Classification: Automated Network Security Audit Framework

Executive Overview

The **ZX301 Security Framework** is an advanced, automated security auditing solution designed to meet the operational mandate for continuous, self-running network defense. Aligned with the strategic vision to minimize human intervention while maximizing asset protection, ZX301 serves as a "force multiplier" for security teams. It automates the detection of network vulnerabilities, exposed services, and weak authentication vectors, ensuring our data remains protected against unauthorized access.

This project executes the core mission: "**To swiftly and automatically scan networks, identifying and fortifying weak spots against unauthorized access.**"

Strategic Capabilities & Technical Implementation

The framework is engineered to fulfill four key operational objectives. Below are the core functions and the actual code logic driving them.

1. Comprehensive Network Enumeration

- **Objective:** Scan the network for ports and services using a fast, reliable methodology.
- **Implementation:** The framework utilizes a "Double-Tap" hybrid engine. It first uses **Masscan** (for high-speed UDP) and **Nmap** (for TCP) to identify

open ports, then feeds *only* those ports into a deep service scan to save time.

The Hybrid Engine

```
# STAGE A: Fast TCP Scan (Nmap)
# Uses optimized rates and prevents DNS resolution for speed
nmap -sS -p- --min-rate $NMAP_RATE -n -$T_LEVEL $TARGET_IP -oG $T
CP_OUT

# STAGE B: High-Speed UDP Scan (Masscan)
# Scans all 65,535 UDP ports using the detected interface
masscan -pU:1-65535 $TARGET_IP -e $IFACE --rate=$MASS_RATE > $UD
P_OUT

# STAGE C: Deep Analysis (Smart Integration)
# Combines results to scan ONLY open ports for versions (-sV)
if [[ -n "$DISCOVERED_TCP" && -n "$DISCOVERED_UDP" ]]; then
    PORTS_ARG="-p T:${DISCOVERED_TCP},U:${DISCOVERED_UDP}"
fi
nmap -sS -sV -Pn -$T_LEVEL $PORTS_ARG $TARGET_IP -oN $HOST_TXT
```

The screenshot shows the configuration interface of the ZX301 Security Framework. It includes the following sections:

- [?] Select Network Source:
 - 1) Manual Entry (IP, CIDR, or Interface Name)
 - 2) Auto-Detect
 - > Selection: 2
- [*] Auto-Detecting Interfaces ...
 - [OK] Detected: 192.168.72.165/24 (eth0)
- [?] Project Name / Output Dir:
 - > Name [Default: Scan_20260103_1303]: Test 1
- [?] Scan Profile:
 - 1) Stealth (Slow)
 - 2) Normal (Default)
 - 3) Aggressive (Noisy)
 - > Selection [2]: 3
- [?] Analysis Depth:
 - 1) Basic (Ports + Weak Passwords)
 - 2) Full (Basic + Vuln Scripts + Searchsploit)
 - > Selection [1]: ■

2. Vulnerability Mapping & Analysis

- **Objective:** Map vulnerabilities and analyze service versions (Full Mode only).
- **Implementation:** In 'Full Mode', the script activates the Nmap Scripting Engine (NSE) for active checks and pipes the resulting XML into **SearchSploit** to correlate specific software versions with known exploits in the Exploit-DB.

Vulnerability Correlation

```
# 1. Mode Selection (User Input)
case $MODE_OPT in
  2) SCAN_MODE="FULL"; SCAN_FLAGS="$SCAN_FLAGS --script vuln" ;;
  *) SCAN_MODE="BASIC" ;;
esac

# 2. Automated Exploit Lookup (Offline Database)
# Parses Nmap XML output and strips ANSI color codes for clean reporting
if [[ "$SCAN_MODE" == "FULL" && -f "$HOST_XML" ]]; then
    log_audit "[${TARGET_IP}] Running Searchsploit Correlation"
    searchsploit --nmap $HOST_XML | sed -r 's/\x1B\\[[0-9;]*[mK]//g' > "$HOST_DIR/vuln_${HOST_CLEAN}.txt"
fi
```

Exploit Title	Path
Microsoft Windows - 'lsasrv.dll' RPC Remote Buffer Overflow (MS04-011)	windows/remote/293.c
Microsoft Windows - 'RPC DCOM' Long Filename Overflow (MS03-026)	windows/remote/100.c
Microsoft Windows - 'RPC DCOM' Remote (1)	windows/remote/69.c
Microsoft Windows - 'RPC DCOM' Remote (2)	windows/remote/70.c
Microsoft Windows - 'RPC DCOM' Remote Universal	windows/remote/71.c
Microsoft Windows - 'RPC DCOM' Remote Buffer Overflow	windows/remote/74.c
Microsoft Windows - 'RPC DCOM' Scanner (MS03-039)	windows/remote/97.c
Microsoft Windows - 'RPC DCOM2' Remote (MS03-039)	windows/remote/103.c
Microsoft Windows - 'RPC2' Universal / Denial of Service (RPC3) (MS03-039)	windows/remote/109.c
Microsoft Windows - DCE-RPC svccntl ChangeServiceConfig2A() Memory Corruption	windows/dos/3453.py
Microsoft Windows - DCOM RPC Interface Buffer Overrun	windows/remote/22917.txt
Microsoft Windows - DCOM RPC Remote Buffer Overflow (2)	windows/remote/3746.txt
Microsoft Windows - Net-NTLM2 Reflection DCOM/RPC (Metasploit)	windows/local/4592.rb
Microsoft Windows - Net-NTLM2 Reflection DCOM/RPC (Metasploit)	windows/local/47135.txt
Microsoft Windows 10 1903/1809 - RPSS Activation Kernel Security Callback Privilege Escalation	windows/remote/5.c
Microsoft Windows 2000/NT 4 - RPC Locator Service Remote Overflow	windows/local/37768.txt
Microsoft Windows 8.1 - DCOM DCE/RPC Local NTLM Reflection Privilege Escalation (MS15-076)	windows/remote/4745.cpp
Microsoft Windows Message Queuing Service - RPC Buffer Overflow (MS07-065) (1)	windows/remote/4934.c
Microsoft Windows Message Queuing Service - RPC Buffer Overflow (MS07-065) (2)	windows/dos/3737.py
Microsoft Windows Server 2008 - RPC Interface Buffer Overflow	windows/remote/737.c
Microsoft Windows XP/2000 - 'RPC DCOM' Remote (MS03-026)	windows/remote/66.c
Microsoft Windows XP/2000 - RPC Remote Non Exec Memory	windows/remote/117.c
Microsoft Windows XP/2000/NT 4.0 - RPC Service Denial of Service (1)	windows/dos/21951.c

3. Credential Integrity Assurance (The "Breach" Module)

- **Objective:** Identify weak passwords on login services (SSH, FTP, RDP, Telnet).
- **Implementation:** The script uses strict Regex to identify open administrative ports. If found, it launches a targeted brute-force attack using either a built-in "Top 20" list or a user-supplied dictionary.

Automated Brute-Forcing

```
# 1. Strict Service Identification (Regex)
# Identifies SSH ports even if they are not on port 22
local SSH_PORTS=$(grep -E -i "^[0-9]+/tcp.*open.*ssh" "$HOST_TXT" | cut -d'/' -f1)

if [[ -n "$SSH_PORTS" ]]; then
    BRUTE_PORTS="${BRUTE_PORTS}${SSH_PORTS}"
    BRUTE_SCRIPTS="${BRUTE_SCRIPTS}ssh-brute,"
fi

# 2. Active Dictionary Attack
# Uses 'userdb' and 'passdb' arguments to feed wordlists to Nmap
nmap -p $BRUTE_PORTS --script $BRUTE_SCRIPTS \
--script-args userdb=$USER_LIST_PATH,passdb=$PASS_LIST_PATH \
$TARGET_IP -oN "${BRUTE_FILE}_brute"
```

```
HOST: 192.168.72.1
[+] Open Ports: 11
    135/tcp msrpc
    139/tcp netbios-ssn
    445/tcp microsoft-ds?
    902/tcp ssl/vmware-auth
    912/tcp vmware-auth
    5040/tcp unknown
    5357/tcp http
    7680/tcp pando-pub?
    32400/tcp http
    49668/tcp msrpc
    57561/tcp unknown
[+] Web Server Detected
[!] Potential Exploits: 48
```

4. Operational Efficiency, Logging & Archiving

- **Objective:** Log all actions, allow search, and archive data.

- Implementation:** The framework maintains a real-time audit log and provides a post-scan menu to search results or zip the entire engagement folder for secure exfiltration.

Search & Archive Workflow

```
# Search Functionality
grep -r -i --color=always "$SEARCH_QUERY" "$BASE_OUTPUT_DIR"

# Secure Archiving (Zip & Clean)
zip -r -q "${DIR_NAME}.zip" "$DIR_NAME"

# Permission Fix (Root → User)
if [[ -n "$SUDO_USER" ]]; then
    chown "$SUDO_USER":'$SUDO_USER' "${DIR_NAME}.zip"
fi

# Auto-Open functionality for immediate access
if command -v xdg-open &> /dev/null; then
    sudo -u "$SUDO_USER" xdg-open . > /dev/null 2>&1
fi
```

The screenshot shows the ZX301 Security Framework's audit log, generated files manifest, and an actions menu.

Audit Log (Left):

```
[13:05:42] === AUDIT SESSION STARTED ===
[13:05:42] Target: 192.168.72.165/24 | Dir: /home/kali/Desktop/ZX301/d
[13:05:43] Config: Aggressive
[13:05:45] Config: Mode FULL
[13:05:50] Config: Custom wordlist: passlist.txt
[13:05:56] STARTED: Ping Sweep (192.168.72.165/24)
[13:05:52] COMPLETED: Ping Sweep (192.168.72.165/24) (Time: 00:00:02)
[13:05:58] [192.168.72.1] THREAD STARTED
[13:05:58] [192.168.72.2] THREAD STARTED
[13:05:58] [192.168.72.1] Starting Fast TCP Scan...
[13:05:58] [192.168.72.2] Starting Fast TCP Scan...
[13:06:00] [192.168.72.2] Starting Masscan UDP...
[13:06:17] [192.168.72.2] Starting Deep Analysis (Nmap Service Scan)...
[13:06:24] [192.168.72.1] Starting Masscan UDP...
[13:06:36] [192.168.72.2] Running Searchsploit Correlation
[13:06:36] [192.168.72.2] THREAD COMPLETED
[13:06:58] [192.168.72.1] Starting Deep Analysis (Nmap Service Scan)...
[13:12:56] [192.168.72.1] Web Recon Started
[13:13:01] [192.168.72.1] SMB Enumeration Started
[13:13:14] [192.168.72.1] Running Searchsploit Correlation
[13:13:17] [192.168.72.1] THREAD COMPLETED
[13:13:17] Generating final reports. Total Duration: 00:07:39
```

Generated Files (Manifest) (Top Right):

```
- 192_168_72_1/scan_192_168_72_1.txt (8.0K)
- 192_168_72_1/scan_192_168_72_1.xml (12K)
- 192_168_72_1/smb_192_168_72_1.txt (4.0K)
- 192_168_72_1/vuln_192_168_72_1.txt (12K)
- 192_168_72_1/web_192_168_72_1.txt (4.0K)
- 192_168_72_1/web_192_168_72_1.txt.vuln (4.0K)
- 192_168_72_2/scan_192_168_72_2.txt (4.0K)
- 192_168_72_2/scan_192_168_72_2.xml (4.0K)
- 192_168_72_2/vuln_192_168_72_2.txt (0)
* audit_trace.txt (4.0K)
* execution_log.txt (4.0K)
* Final_Report.html (12K)
* Final_Report.txt (4.0K)
* final_report.xml (4.0K)
* live_hosts.txt (4.0K)
```

Actions Menu (Bottom Right):

```
[ACTIONS]
S) Search Logs
Z) Zip & Clean (Recommended)
K) Keep Raw Files
D) Delete All
```

> Choice: █

Mission Statement

To create a self-running system that automatically finds security holes in our networks, keeping our data safe from hackers with minimal human intervention.