Jad_AbouNajem_network_safety_solution

Exercise 1:

SMTP (Simple Mail Transfer Protocol)

definition: communication protocol for electronic mail transmission.

Enumeration:

1. Identifying a SMTP Server
2. Enumerate Users
3. DNS Mail Exchange (MX) Record Enumeration
4. Information Disclosure with NTLM Auth

Attack

Open Relay Exploit

SMTP Open Relay occurs when the SMTP server is configured to accept and transfer messages on the network that were neither for nor from local users.

Here is a simple example of how to test for open relay:

```
telnet target.com 25MAIL FROM:<test@example.com>RCPT TO:
<test2@anotherexample.com>DATASubject: Test open relayTest message.QUIT
```

Post-Exploitation

Common SMTP Commands

Command	Description	Usage
HELO	Identifies the client to the server.	HELO example.com
EHLO	Extended HELLO.	EHLO example.com
MAIL FROM:	Specifies the sender's email address.	MAIL FROM: [sender@example.com] (mailto:sender@example.com)

Command	Description	Usage
RCPT TO:	Specifies the recipient's email address.	RCPT TO: [recipient@example.com] (mailto:recipient@example.com)
DATA	Indicates the start of the message body.	DATA
RSET	Resets the session.	RSET
NOOP	No operation; used for testing.	NOOP
QUIT	Terminates the session.	QUIT

source: hackviser.com

mitigations: **disabling open relaying**, limiting connections to prevent abuse, monitoring **logs for suspicious activity**, and **training users** to recognize phishing attempts

SMTP Exploit Example – The “Zeus Email Campaign” (2010s)

- **What happened:**
  Attackers used the **SMTP protocol** to send massive phishing emails that carried the Zeus banking malware. They targeted organizations worldwide.
- **How SMTP was exploited:**
  - Some email servers were **misconfigured as open relays**, meaning anyone could send emails through them without authentication.
  - Attackers sent emails from seemingly legitimate addresses to bypass spam filters.
  - Users clicked on malicious links or attachments, installing malware.

MongoDB

Enumeration

MongoDB Server Information

You can connect to the MongoDB server and gather information about the server, databases, collections, users, etc. using MongoDB commands.

MongoDB Client Tools

Tools like MongoShell and MongoDump can be used for interacting with MongoDB databases and performing enumeration tasks.

Attack Vectors

Default Credentials

MongoDB instances often come with default credentials or no authentication enabled. It's crucial to check for default credentials or weak authentication configurations.

NoSQL Injection

Similar to SQL Injection in relational databases, NoSQL Injection attacks target MongoDB databases by exploiting vulnerabilities in query constructions.

Unprotected MongoDB Instances

MongoDB instances sometimes have no access control or firewall rules, leaving them exposed to unauthorized access from the internet. You can search for MongoDB instances using tools like Shodan and exploit them if they're unprotected.

Post-Exploitation

Default Credentials

MongoDB instances often come with default credentials or no authentication enabled. It's crucial to check for default credentials or weak authentication configurations.

NoSQL Injection

Similar to SQL Injection in relational databases, NoSQL Injection attacks target MongoDB databases by exploiting vulnerabilities in query constructions.

Unprotected MongoDB Instances

MongoDB instances sometimes have no access control or firewall rules, leaving them exposed to unauthorized access from the internet. You can search for MongoDB instances using tools like Shodan and exploit them if they're unprotected.### Common MongoDB Commands

Command	Description	Example
show dbs	List all databases	`show dbs`

Command	Description	Example
use	Switch to a specific database	`use mydatabase`
show collections	List all collections in the database	`show collections`
db.collection.find()	Retrieve documents from a collection	`db.mycol.find().pretty()`
db.collection.insertOne()	Insert a document into a collection	`db.mycol.insertOne({name: "John", age: 30})`
db.collection.deleteOne()	Delete a document from a collection	`db.mycol.deleteOne({name: "John"})`
db.dropDatabase()	Drop the current database	`db.dropDatabase()`

Mitigations: **enabling authentication** with strong passwords, **implementing role based access control (RBAC)**, **encrypting data** in transit (TLS/SSL) and at rest (WiredTiger).

MongoDB Exploit Example – 2017 MongoDB Ransom Attacks

- **What happened:**
  In 2017, attackers scanned the Internet for **unprotected MongoDB instances** (no authentication enabled) and deleted data, replacing it with a ransom note demanding Bitcoin.
- **How MongoDB was exploited:**
  - Default installations were often left **open to the internet** with no password.
  - Attackers used simple scripts to connect to port **27017** (MongoDB default), dump and delete databases, and insert their ransom message.
- **Impact:**
  - Thousands of companies lost access to critical data.
  - Highlighted how **exposing databases to the public Internet without authentication** is extremely risky.

Exercise 2:

1-Windows: Insecure service permissions

This occurs when a service that's running under SYSTEM privileges, but the User has permissions to change the executable binpath to one which could create a reverse shell.

How attackers exploit this

There are several attack techniques:

a) Service Binary Path Manipulation

- **Condition:** Service is running as SYSTEM, but a normal user can change its executable path.
- **Attack:**
  1. Modify the service's executable path to a malicious program (reverse shell, malware).
  2. Restart the service.
  3. The malicious program runs with **SYSTEM privileges**.
- **Tools:** `sc.exe`, `SetACL.exe`, `PowerUp.ps1` (from PowerSploit).

b) Weak Service Permissions

- **Condition:** The service allows users to **stop/start** it.
- **Attack:**
  - Stop a critical service and replace its executable or manipulate its behavior.
  - Can lead to privilege escalation or Denial of Service (DoS).

c) Hijacking Services with Writable Directories

- **Condition:** Service executable resides in a directory where a non-admin user can write.
- **Attack:**
  - Replace legitimate service executable with a malicious one.
  - Restart service → SYSTEM privileges.

Real-life example

1. **CVE-2016-0051** – Windows Task Scheduler Service:
   - Allowed non-admins to modify the service binary path.
   - Could escalate privileges to SYSTEM by replacing the binary with malicious code.

Mitigations

1. Restrict Service Permissions

- Only administrators should have the ability to:
  - Start/stop/restart services.
  - Change service configuration (binary path, startup type).
- Use tools like `sc sdshow <ServiceName>` to check permissions and `sc sdset` to restrict them.

2. Secure Service Executable Locations

- Place service binaries in directories with **strict ACLs**:
  - Only SYSTEM and Administrators should have write permissions.
  - Avoid `C:\Users\` or other writable paths for executables.

2-Linux: Insecure service permissions

This occurs when a service or daemon running with **root privileges** can be modified or influenced by a non-root user, allowing privilege escalation.

How attackers exploit this

a) Writable Service Executables or Scripts

- **Condition:** The service binary or startup script is writable by a normal user.
- **Attack:** Replace the binary/script with malicious code → restart service → code runs as root.
- **Example tools:** `chmod`, `systemctl`, manual replacement.

b) Weak Systemd Service Permissions

- **Condition:** Non-root users can modify service configuration files (`/etc/systemd/system/*.service`).
- **Attack:** Change `ExecStart` to a malicious binary → restart service → gain root privileges.

c) Exploiting Environment Variables

- **Condition:** Services run as root and respect user-writable environment variables.
- **Attack:** Modify variables like `LD_PRELOAD` or `PATH` → inject malicious libraries or binaries.

Real-life example

- Services running scripts from `/tmp` or user-writable directories have been exploited in **local privilege escalation attacks** on Linux servers.

Mitigations

1. **Secure Service Files and Directories**

- Only root should have write permissions for:
  - Service binaries
  - Startup scripts
  - Systemd unit files

2. **Limit Service Privileges**

- Run services as **least-privileged users** when root is not necessary.