Web Application Pentesting

RISC - Charlie E // chrlz

RISC Session 0x02 - Week 3 - Semester 1 2019

Agenda

1 Information Gathering

2 Recon & Mapping

3 Discovery

4 Exploitation

Disclaimer

Here at RISC we encourage Ethical Hacking.

Do not hack/test what you do not own or have permission to hack/test.

“As a member of RISC, you agree to use the knowledge gained from workshops in a legal manner. By attending our sessions, you release RISC, the sponsoring company/companies and industry mentors from any liability, and assume any and all risk, liability, cost or damage incurred from the actions and knowledge gained from these sessions. “

Information Gathering

- Enumerate DNS
- Find out what ‘should’ be there
  - Apps/services advertised/linked to
- OSINT - Open Source Intelligence (See Peter’s slides from last week)

Information Gathering Tool - Dig

- DNS query tool useful for finding records registered for a given domain
- Misconfigurations in DNS can allow for information leakage
- Can help to identify scope

Usage:

```
dig [@nameserver] [recordname] [recordtype]# General Form

dig domain.com NS +short# Identify authoritative nameserver for a domain

dig @nameserver.com zonetransfer.com AXFR# List all records for a domain from a server
                                         # This should not be allowed from any source IP/domain

dig [@nameserver] target.domain ANY    # List all records for a domain
                                       # deprecated, likely not accepted
dig [@nameserver] target-ip -x         # Perform a reverse lookup on an IP
```

See: https://linux.die.net/man/1/dig

Mapping & Recon

- How do these applications achieve what they're trying to do?
  - What technologies/frameworks?
  - How do they manage users/authentication/login?
  - How do they handle user input?
  - … and many more
- How can you map out the attack surface?
  - Automatic web mapping (web crawlers eg Burp Spider)
  - Bruteforcing/Fuzzing
  - Manual mapping (simply follow links, look for a sitemap)

Mapping & Recon Tool - NMap

- TCP/IP port scanning tool which provides many different functions including:
    - Identifying running hosts and open ports
    - Fingerprinting Running Services
    - OS Fingerprinting
    - Powerful Scripting Engine

Mapping & Recon Tool - NMap Examples

```
### Basic Usage
nmap -sn [target CIDR Range]# Test network range for hosts which are up
nmap -sn 10.10.10.0/24# test IPs from 10.10.10.0-255
nmap -p [port range] [host]
nmap -p 0-65535 10.10.10.1# scan all ports on 10.10.10.1

### My common usage
nmap -A -T4 [target]# aggressive, quick scan on a target host - good for quick enumeration

nmap -sC -sV -p- [target] -oA [fileprefix]# Scan all ports, enumerating service versions
                                           # and running default scripts
                                          # Additionally, output in all formats to
                                           # fileprefix.{nmap,xml,gnmap}
```

See: https://nmap.org/

Discovery

- Are these functionalities vulnerable in any way?
  - Are there known vulnerabilities/exploits for a given technology?
  - Are there misconfigurations which make an app vulnerable?
  - How does the server validate and/or store data?
  - How are sessions managed?
- Use tools such as vulnerability scanners as a starting point

Discovery - Types of Exploits

- Configuration
  - Default passwords/paths/settings
- Authentication
  - Can it be bruteforced or enumerated?
  - What is in place to prevent this? (MFA, Captcha, retry limit)
  - Password complexity requirements
- Sessions
  - How are sessions handled? Is there a way to exploit stored session data?
- Authorization
  - How is access control implemented?
  - Do users have access to pages/endpoints they shouldn’t?

See: https://www.owasp.org/images/7/72/OWASP_Top_10-2017_%28en%29.pdf.pdf

Discovery - Types of Exploits

- Data Validation
  - How does data get processed by the server?
  - Client side validation is not enough
  - Can lead to:
    - SQL Injection
    - XSS
    - XML Injection
    - Template Injection
  - Does any user input invoke another command or application on the server? (Command injection)
- File inclusion
  - Pages/content loaded by user-provided name can lead to unwanted files shown
- Denial of Service
  - Are there protections against this? (rate limiting, IP restrictions, WAF)

See: https://www.owasp.org/images/7/72/OWASP_Top_10-2017_%28en%29.pdf.pdf

Discovery - Tool - Nikto

- Nikto is a perl script which is used to scan web servers for common vulnerabilities - a class of automated vulnerability scanner
- Automated scanners are good, however manual evaluation is always required
- Automated scanners are usually NOISY

Usage:

```
perl nikto.pl -update # run before use to ensure up to date

perl nikto.pl -h 10.10.10.1 # basic test on default HTTP port (80)

perl nikto.pl -h 10.10.10.1 -output -Format HTML# output report in HTML format
```

See: https://cirt.net/Nikto2

Exploitation

- Finding vulnerabilities is only a starting point
- Understanding how they can be exploited
  - Many tools exist for automated exploitation such as
    - Metasploit
    - BeEF
    - SQLMap
    - Hydra
    - Wfuzz
- What can be achieved through exploiting them?
  - Data Extraction/Dump
  - User account/secrets compromise
  - Code execution
  - Shell access/Host takeover

Exploitation - Practical

- SQLMap is a script for automated detection and exploitation of SQL injection attacks
- It is very, very comprehensive in its features, and can be seen as a swiss army knife of SQL injection

Usage:

```
python sqlmap.py -u [target url]/vulnerable_param=1 --dbs# dump list of databases on host

python sqlmap.py -u [target url]/vulnerable_param=1 --all# retrieve all information from DBMS

python sqlmap.py -r [request file] --all# retrieve all information from DBMS, load URL &
                                          parameters from file
```

See: http://sqlmap.org/

Exercise

Head to: http://10.133.33.147/dvwa

Find what you can!