Laboratory_13: SSL/TLS Security Assessment using sslscan

This laboratory covers the usage of sslscan tool for assessing SSL/TLS configuration and identifying potential security vulnerabilities.

Installation

- sudo apt update
- https://github.com/rbsec/sslscan
- sslscan --version

Part 1: Basic SSL/TLS Scanning

Scanning a Local Server

1. First, create a test server with weak SSL/TLS configuration:

   mkdir -p ~/sslscan-lab
   cd ~/sslscan-lab

2. Generate a test certificate:

   openssl req -x509 -newkey rsa:2048 -keyout test.key -out test.crt -days 365 -nodes -subj "/C=US/ST=Test/L=Test/O=Test/CN=localhost"

3. Create a test configuration file for openssl server:

   echo "Certificate = test.crt
   Key = test.key
   Port = 4433
   Cipher = ALL:eNULL" > server.cnf

4. Start the test server:

   sudo openssl s_server -cert test.crt -key test.key -port 4433 -cipher ALL:NULL &

5. Scan the local server:

   sslscan localhost:4433

Scanning External Servers

1. Scan a well-configured server:

   sslscan github.com

Part 2: Advanced Scanning Options

Protocol Version Testing

1. Test only SSLv3 (if supported):

   sslscan --ssl3 localhost:4433

2. Test TLS versions:

   sslscan --tls10 localhost:4433
   sslscan --tls11 localhost:4433
   sslscan --tls12 localhost:4433

Cipher Suite Analysis

1. Show cipher details:

   sslscan --show-ciphers localhost:4433

2. Test specific cipher suites:

   sslscan --cipher=AES256-SHA localhost:4433

Certificate Analysis

1. Show certificate details:

   sslscan --show-certificate --no-ciphersuites github.com

2. Test certificate chain:

   sslscan --show-certificate --show-times github.com

Part 3: Security Vulnerability Detection

Testing for Weak Ciphers

1. Create a server with weak ciphers:

   ```
   echo "Certificate = test.crt
   Key = test.key
   Port = 4433
   Cipher = DES-CBC-SHA:RC4-MD5" > weak-server.cnf
   ```

2. Start the weak server:

   sudo openssl s_server -config weak-server.cnf &

3. Scan for weak ciphers:

   sslscan --no-colour localhost:4433 | grep -E "Accepted|Weak"

Testing for Protocol Vulnerabilities

1. Test for TLS compression (CRIME vulnerability):

   sslscan --compression localhost:4433

2. Test for renegotiation:

   sslscan --renegotiation localhost:4433

3. Test for preferred ciphers:

   sslscan --show-client-cas localhost:4433

Part 4: Generating Reports

XML Output

1. Generate XML report:

   sslscan --xml=scan-report.xml github.com

2. View the XML report:

   cat scan-report.xml | head -50

Creating a Comprehensive Report

1. Perform a full scan with all options:

   sslscan --show-certificate --show-ciphers --show-times --compression --renegotiation github.com > full-scan-report.txt

2. Review the report:

```
less full-scan-report.txt
```

Part 5: Comparing Different Servers

Scanning Multiple Targets

1. Create a list of targets:

```
echo "github.com
google.com
facebook.com" > targets.txt
```

2. Scan multiple targets:

```
while read target; do
  echo "=== Scanning $target ==="
  sslscan --no-colour $target | grep -E "SSL|TLS|Cipher"
  echo ""
done < targets.txt
```

Identifying Best Practices

1. Check for modern TLS versions:

```
sslscan github.com | grep -E "TLSv1.[2-3]"
```

2. Check for strong cipher suites:

```
sslscan github.com | grep -E "AES256|CHACHA20"
```

3. Check for deprecated protocols:

```
sslscan github.com | grep -E "SSLv[2-3]|TLSv1.0"
```

Security Recommendations

Based on sslscan results, ensure:

1. Disable SSLv2, SSLv3, and TLS 1.0
2. Enable TLS 1.2 and TLS 1.3
3. Use strong cipher suites (AES-256, ChaCha20)
4. Disable weak ciphers (RC4, DES, 3DES)
5. Implement proper certificate chains
6. Use certificates from trusted CAs
7. Disable TLS compression
8. Configure secure renegotiation

Cleanup

```
sudo pkill openssl
cd ~
rm -rf ~/sslscan-lab
```