TLS Security Assessment at Scale Using testssl

Author: Rajamohan Reddy

Version: 1.0

Purpose: This SOP defines the standard process to perform bulk TLS/SSL configuration assessment using **testssl** with proper evidence collection in Kali Linux GUI environments. **Scope:** This SOP applies to authorized TLS configuration testing only and does not include exploitation, denial-of-service testing, or scanning outside the approved scope.

1. First Setup SSH

Ensure SSH Service is Enabled on Kali Linux
Enable and start the SSH service on Kali Linux to allow connections from PuTTY using port forwarding for monitoring and file transfer while scans continue to run in the GUI session.

```
sudo apt update
sudo apt install -y openssh-server
sudo systemctl enable ssh
sudo systemctl start ssh
sudo systemctl status ssh #Verify ssh status
```

2. Connect Kali using PuTTY with Port Forwarding

Configure VM Port Forwarding and Connect Using PuTTY

Configure port forwarding on the virtual machine to allow SSH access from the host system to the Kali VM.

VirtualBox (NAT) – Port Forwarding Steps:

1. Open **VirtualBox Manager**.
2. Select the Kali VM → **Settings** → **Network** → **Adapter 1 (NAT)**.
3. Click **Advanced** → **Port Forwarding**.
4. Add a new rule with the following values:
   - **Name:** SSH
   - **Protocol:** TCP
   - **Host IP:** 127.0.0.1
   - **Host Port:** 2222
   - **Guest IP:** (leave blank)
   - **Guest Port:** 22

Download the Putty using below link
https://the.earth.li/~sgtatham/putty/latest/w64/putty.exe

Connect Using PuTTY:

- **Host:** 127.0.0.1
- **Port:** 2222
- **Connection Type:** SSH

Important Note:
Scans that require screenshots must be started from the Kali GUI terminal. SSH (PuTTY) is intended only for monitoring, file transfer, and administrative tasks. Starting screenshot-dependent processes from SSH will not capture screenshots.

3. Create Folders (First-Time Setup)

If running for the first time, create the required folders. If the path already exists, this step can be skipped.

Command:

```
mkdir -p ~/testssl/targets
```

4. Create Domains File

Update the list of domains to be scanned. One target per line (host or host:port). Do not include URL paths.

Command:

```
cat << 'EOF' > ~/testssl/domains.txt
example.com
testphp.vulnweb.com
https://demo.testfire.net
example.com
testphp.vulnweb.com
https://demo.testfire.net
example.com
testphp.vulnweb.com
https://demo.testfire.net
example.com
testphp.vulnweb.com
https://demo.testfire.net
EOF
```

5. Create the Automation Script

Run the following command to create testssl_bulk.sh.

The script scans domains one by one from domains.txt. Outputs are saved as HTML with timestamps. Screenshots are captured every 10 seconds and saved with index, domain, and timestamp.

https://github.com/Rajamohan9/Scripts/blob/main/testssl_bulk.sh

Command:

```
cat << 'EOF' > ~/testssl/testssl_bulk.sh
#!/bin/bash

### CONFIG
TARGET_FILE="$HOME/testssl/domains.txt"
REPORT_DIR="$HOME/testssl/reports"
LOG_DIR="$HOME/testssl/logs"
SCREENSHOT_DIR="$HOME/testssl/screenshots"
TESTSSL_BIN="testssl"

echo "[*] Starting Pre-checks..."

# Ensure folders exist
mkdir -p "$REPORT_DIR" "$LOG_DIR" "$SCREENSHOT_DIR"

# Check testssl
if ! command -v "$TESTSSL_BIN" >/dev/null 2>&1; then
  echo "[!] testssl not found. Installing..."
  sudo apt update && sudo apt install -y git
  if [ ! -d "/opt/testssl.sh" ]; then
    sudo git clone https://github.com/drwetter/testssl.sh.git /opt/testssl.sh
  else
    cd /opt/testssl.sh && sudo git pull && cd - >/dev/null
  fi
  sudo ln -sf /opt/testssl.sh/testssl.sh /usr/local/bin/testssl
fi

# Check scrot
if ! command -v scrot >/dev/null 2>&1; then
  echo "[!] scrot not found. Installing..."
  sudo apt update && sudo apt install -y scrot
fi

# Validate target file
if [ ! -f "$TARGET_FILE" ]; then
  echo "[!] Target file not found: $TARGET_FILE"
  exit 1
fi

echo "[✓] Pre-checks completed"
echo "----------------------------------------------"

i=1
```

```
while read -r domain; do
  [ -z "$domain" ] && continue

  ts=$(date +%Y%m%d_%H%M%S)
  clean=$(echo "$domain" | sed 's/[^a-zA-Z0-9]/_/g')

  echo "=============================================="
  echo "[+] [$i] Scanning: $domain"
  echo "Timestamp: $ts"
  echo "=============================================="

  (
    j=1
    while true; do
      sleep 10
      scrot "$SCREENSHOT_DIR/${i}_${clean}_${ts}_shot${j}.png" 2>/dev/null
      j=$((j+1))
    done
  ) &
  SCREEN_PID=$!

  "$TESTSSL_BIN" --htmlfile "$REPORT_DIR/${i}_${clean}_${ts}.html" "$domain" \
    | tee "$LOG_DIR/${i}_${clean}_${ts}.txt"

  kill "$SCREEN_PID" 2>/dev/null

  echo "[✓] [$i] Completed: $domain"
  echo "----------------------------------------------"

  i=$((i+1))
  sleep 2
done < "$TARGET_FILE"

echo "[✓] All scans completed successfully"
EOF
```

6. Add Execute Permission

Grant execute permissions to the `testssl_bulk.sh` script so it can be run as a program.

Command:

```
sudo chmod +x ~/testssl/testssl_bulk.sh
```

7. Run from Kali GUI (Screenshots Enabled)

Now follow the below steps:

1. Log in to the **Kali Linux GUI** and start the script from a **maximized terminal window** to capture complete terminal output in screenshots.
2. Keep the **GUI terminal session open** for the entire duration of the scan. Closing the terminal or logging out will stop the scan.
3. If parallel work is required, connect to the same Kali VM using **SSH (PuTTY)** for monitoring or file transfers.
4. Do **not start the scan from SSH (PuTTY)** when screenshots are required, as screenshot capture works only for processes started in the GUI session.

Command:

```
cd ~/testssl/
./testssl_bulk.sh
```

Notes & Limitations

Screenshots require a GUI/X display and will not work over SSH/headless sessions. Running the scan in GUI and connecting via SSH later does not impact the running process. Closing the GUI terminal or logging out will stop the scan. Store all evidence securely.