## Introduction

This document provides a comprehensive methodology for modifying Open5GS to support TLS 1.0. The objective is to introduce specific cryptographic vulnerabilities for controlled penetration testing and security evaluation. This process involves modifying Open5GS dependencies, configuring its cryptographic parameters, and verifying the implementation using security testing tools.

## Environment Setup

- **Operating System:** Ubuntu 20.04
- **Open5GS Version:** v2.7.2-149-gbbfd462 (latest at the time of writing)
- **OpenSSL Version:** 1.1.1 (custom-built to enable TLS 1.0)
- **Verification Tool:** TLS-Scanner & TestSSL
- **Testing Utilities:** openssl s_client, curl

## Methodology for TLS Version Downgrade

### 1. Install Required Dependencies

Ensure the system has the essential build tools and libraries:

```
sudo apt update && sudo apt install -y build-essential cmake git pkg-config libgnutls28-dev
sudo apt install meson ninja-build gcc g++ flex bison git libtalloc-dev
```

### 2. Remove Existing OpenSSL

```
sudo apt remove --purge openssl libssl-dev -y
sudo apt autoremove -y
sudo rm /usr/bin/openssl
sudo ln -s /usr/local/openssl/bin/openssl /usr/bin/openssl
sudo ldconfig
sudo rm /usr/lib/x86_64-linux-gnu/libssl.so.1.1
sudo rm /usr/lib/x86_64-linux-gnu/libcrypto.so.1.1
```

### 3. Compile and Install OpenSSL 1.1.1

Since Open5GS is linked by default to OpenSSL 1.1.1f, it must be replaced with a custom-compiled version to permit
the use of TLS 1.0.

```
cd /usr/local/src
wget https://www.openssl.org/source/openssl-1.1.1.tar.gz
mkdir openssl-1.1.1 && tar -xvzf openssl-1.1.1.tar.gz -C openssl-1.1.1 --strip-components=1
cd openssl-1.1.1
./config --prefix=/usr/local/openssl --openssldir=/usr/local/openssl
make -j$(nproc)
sudo make install
```

### 4. Create new symbolic links

```
sudo ln -s /usr/local/openssl/bin/openssl /usr/bin/openssl
sudo ln -s /usr/local/openssl/lib/libssl.so.1.1 /usr/lib/x86_64-linux-gnu/libssl.so.1.1
sudo ln -s /usr/local/openssl/lib/libcrypto.so.1.1 /usr/lib/x86_64-linux-gnu/libcrypto.so.1.1
sudo ldconfig
```

### 5. Update System's CA certificate

```
sudo apt update
sudo apt install --reinstall ca-certificates
sudo update-ca-certificates
```

### 6. Download Open5GS

```
git clone --recurse-submodules https://github.com/open5gs/open5gs.git
cd open5gs
```

### 7. Configure System to Use the Custom OpenSSL Build

Define environment variables to ensure the system prioritizes the newly compiled OpenSSL version:

```
export OPENSSL_ROOT_DIR=/usr/local/openssl
export OPENSSL_LIBRARIES=/usr/local/openssl/lib
export OPENSSL_INCLUDE_DIR=/usr/local/openssl/include
export PKG_CONFIG_PATH=/usr/local/openssl/lib/pkgconfig:$PKG_CONFIG_PATH
export PATH=/usr/local/openssl/bin:$PATH
export LD_LIBRARY_PATH=/usr/local/openssl/lib:$LD_LIBRARY_PATH
```

### 8. Verify OpenSSL Version Alignment

Confirm that the system and Open5GS utilize the expected OpenSSL version:

```
openssl version -a
ldd $(which openssl) | grep ssl
```

Check If Meson Recognizes Your OpenSSL

```
pkg-config --modversion openssl
pkg-config --cflags openssl
pkg-config --libs openssl
```

Expected output:

```
1.1.1
-I/usr/local/openssl/include
-L/usr/local/openssl/lib -lssl -lcrypto
```

### 9. Configure Open5GS

Under /lib/sbi modify the nghttp2-server.c source code to explicitly allow TLS 1.0.

Lines 259-264:

Change “OGS_TLS_MAX_VERSION“ to the highest version you would like to enable, this also counts for communication between the NF’s.

```
#define OGS_TLS_MIN_VERSION TLS1_VERSION
#define OGS_TLS_MAX_VERSION TLS1_3_VERSION
```

Line 277 from:

```
if (SSL_CTX_set_cipher_list(ssl_ctx, DEFAULT_CIPHER_LIST) == 0)
```

To this:

```
if (SSL_CTX_set_cipher_list(ssl_ctx, "ALL:!aNULL:!eNULL:@STRENGTH") == 0)
```

### 10. Build Open5GS to apply these modifications:

```
meson build --prefix=`pwd`/install \
    -Dssl=true \
    -Dlibdir=lib \
    -Ddefault_library=shared \
    -Dcpp_args="-I/usr/local/openssl/include" \
    -Dc_args="-I/usr/local/openssl/include" \
    -Dcpp_link_args="-L/usr/local/openssl/lib -Wl,-rpath,/usr/local/openssl/lib" \
    -Dc_link_args="-L/usr/local/openssl/lib -Wl,-rpath,/usr/local/openssl/lib" \
    -Dpkg_config_path=/usr/local/openssl/lib/pkgconfig
ninja -C build
sudo ninja -C build install
```

### 11. Modify YAML files:

Enable TLS in the NF’s configuration file

```
     - plmn_id:
         mcc: 999
         mnc: 70
@@ -16,4 +16,4 @@
-  sbi:
-    server:
-      - address: 127.0.0.10
-        port: 7777
+# sbi:
+#   server:
+#     - address: 127.0.0.10
+#       port: 7777
```

```
##############################################################################
# HTTPS scheme with TLS
##############################################################################
#  o Set as default if not individually set
@@ -54,12 +54,12 @@
-#  default:
```

```
-#    tls:
-#      server:
-#        scheme: https
-#        private_key: @sysconfdir@/open5gs/tls/nrf.key
-#        cert: @sysconfdir@/open5gs/tls/nrf.crt
-#      client:
-#        scheme: https
-#        cacert: @sysconfdir@/open5gs/tls/ca.crt
-#  sbi:
-#    server:
-#      - address: nrf.localdomain
+  default:
+    tls:
+      server:
+        scheme: https
+        private_key: /home/open5gs/open5gs/install/etc/open5gs/tls/nrf.key
+        cert: /home/open5gs/open5gs/install/etc/open5gs/tls/nrf.crt
+      client:
+        scheme: https
+        cacert: /home/open5gs/open5gs/install/etc/open5gs/tls/ca.crt
+  sbi:
+    server:
+      - address: nrf.localdomain
```

Make sure to edit /etc/hosts to declare the IP address of every localdomain:

```
sudo nano /etc/hosts
# Open5GS Network Functions
192.168.2.197 nrf.localdomain
127.0.0.11 ausf.localdomain
127.0.0.12 udm.localdomain
127.0.0.13 pcf.localdomain
127.0.0.14 nssf.localdomain
127.0.0.15 bsf.localdomain
127.0.0.20 udr.localdomain
127.0.0.200 scp.localdomain
127.0.0.4 smf.localdomain
192.168.2.197 amf.localdomain
```

### 11. Restart Services and Validate Configuration

Initiate the Open5GS NRF component and verify the enabled TLS version:

```
sudo ~/open5gs/install/bin/open5gs-nrfd
```

Perform connectivity tests to confirm TLS 1.0 support:

```
openssl s_client -connect nrf.localdomain:443 -tls1
curl -v --tlsv1.0 --insecure https://nrf.localdomain
```

Perform TLS Testing:

```
java -jar TLS-Server-Scanner.jar -connect 192.168.2.197:443
./testssl.sh 192.168.2.197:443
```

Output from TLS-Scanner which shows TLS1 support:

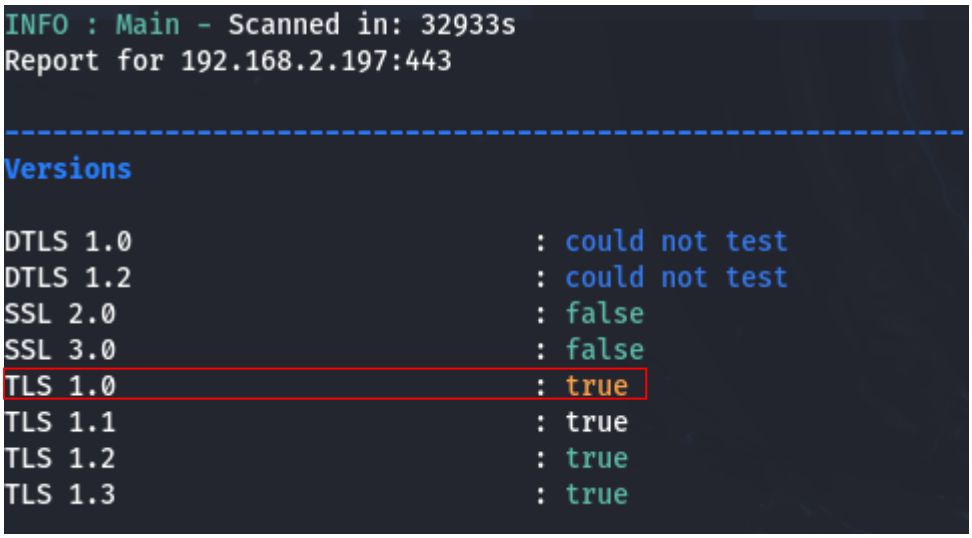


*Figure 1: TLS-Scanner TLS Support*

And the Vulnerabilities:

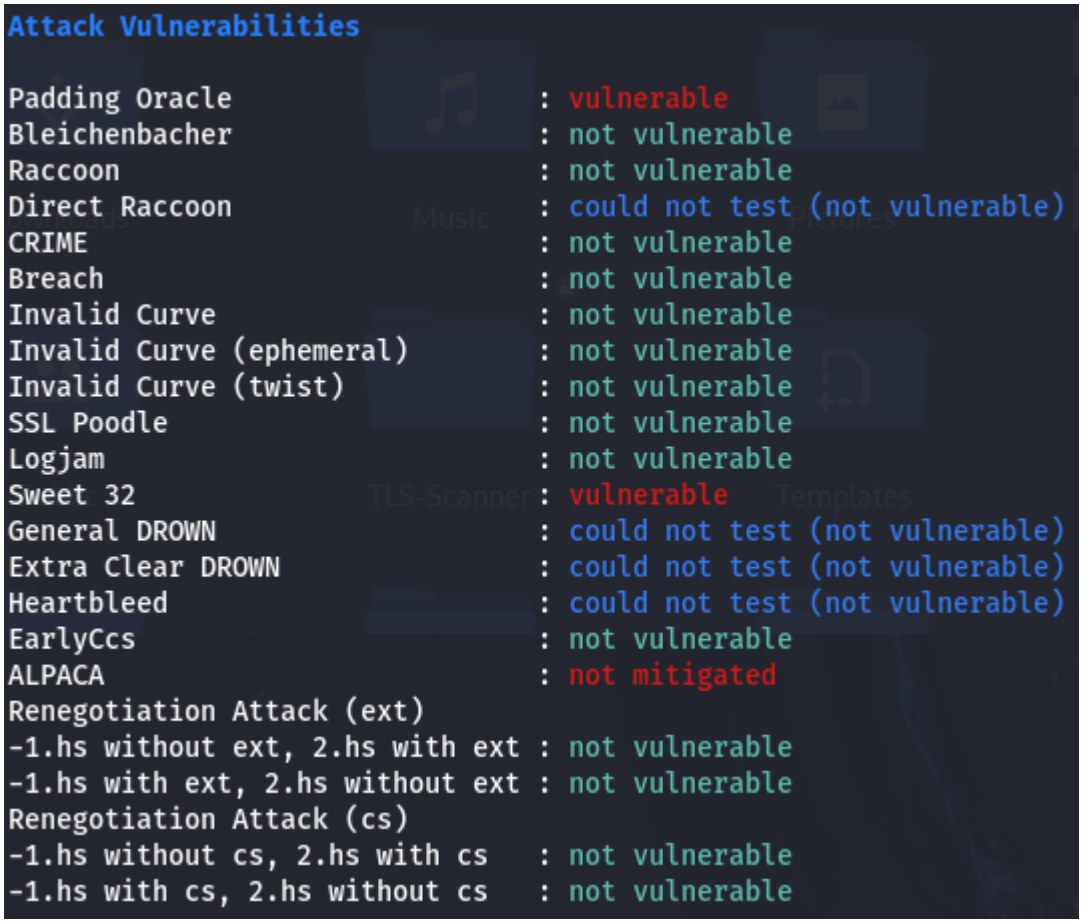


*Figure 2: TLS-Scanner Attack Vulnerabilities*

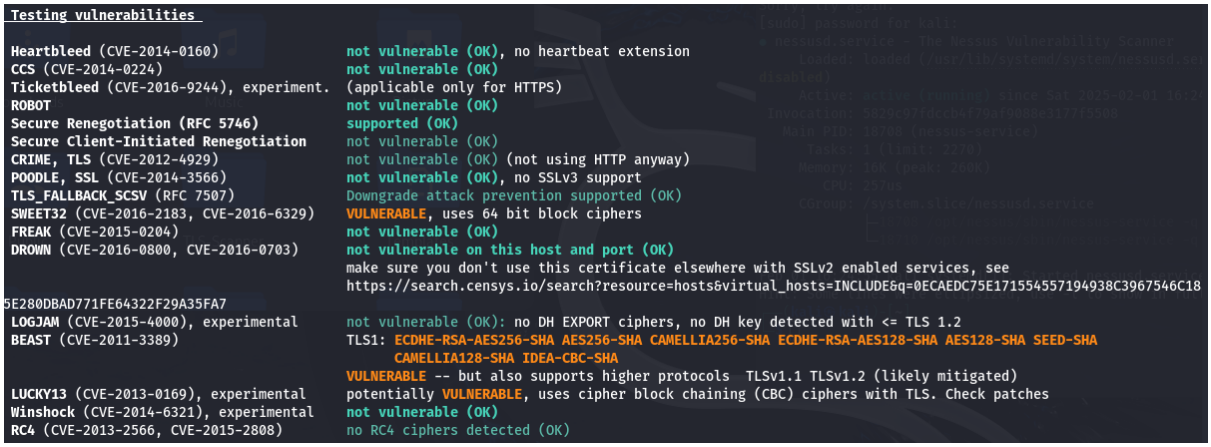


*Figure 3: TestSSL Attack Vulnerabilities*

## Observations and Resolutions

- **Open5GS Defaulting to OpenSSL 1.1.1f:** Resolved by explicitly defining the OpenSSL build path during compilation and deleting previous OpenSSL version.
- **Cipher Suites Not Recognized:** Addressed through adjustments to nghttp2-server.c.
- **TLS-Scanner Detection Issues:** Required complete recompilation of Open5GS following modifications.

**Conclusion**

This procedure successfully downgrades Open5GS to support TLS 1.0, introducing security vulnerabilities relevant for penetration testing and cryptographic analysis. This configuration should never be used in production environments due to its exposure to known attacks such as Padding Oracle, BEAST, SWEET32, LUCKY13, and downgrade-based exploits.