Task: Tamago
Operation: S.M.O.K.E.
Team: Adrian

By inspecting the .pcap network capture file using Wireshark, we can determine some information about the communication. Firstly, we see that the communication is done via TLS 1.0. We can also notice during the handshake that the established encryption is AES-256-CBC with SHA signing (TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA). This means that we have a 32 bytes key.

After we extract the application data (server to client and client to server) we can brute-force the keys, knowing that “the symmetric encryption parameters of the TLS conversation are the only ones still present in clear in the dump!”

First, we start with the client to server channel. Because we know it is a HTTPS communication we can expect some HTTP requests of sorts. By brute-forcing every key from the dump and decrypting the application data from client to server using AES-256-CBC, we obtain a set of decryptions. Out of these, we want to filter those who have comprehensible ASCII text. We manage to find a decryption containing “GET doctor_files”. We found our client to server key (‘83b0aead709a83a81bf02f3903c1d82323d2ff63066f506e46be333f51aca4bf’). By decrypting the entire communication between the client and the server we can also see a confirmation message from the client at the end of transmission:

```
GOT 12345 bytes with SHA1 c6fafff4f4997d23edb1379c235e451faca8990c
```

Now we can brute-force every key again from the dump, try to decrypt the application data, and check the SHA1 of the decryption against the SHA1 obtained before.

```python
for i in xrange(len(data) - KEY_LEN):
    pt = dec(e, iv, data[i : i + KEY_LEN])
    for j in xrange(len(pt) - 12345):
        if sha1(pt[j:j+12345]).hexdigest() == 'c6fafff4f4997d23edb1379c235e451faca8990c':
            with open('res/%d' % i, 'w') as f:
                f.write(pt)
```

Flag: CSC2016{5bda403ca5a27b520f818e56be7711f2}