Homework 3 – Xavier Gitiaux

March 2019

1 Exercise 1

1.1 Question 1

The attacker can proceed as follows:

- Relay the "hello" message from the client to the server
- Relay the server's answer (certificate) to the client
- Intercept the cypher $c = Enc_{pke}(K, pk)$, discards it and replace it by $c^{'} = Enc_{pke}(K^{'}, pk)$, $K \neq K$.

The server would accept $c^{'}$ thinking it comes from the client, decrypts it and uses to communicate with the client.

1.2 Question 2

Regardless of the attack, the server never knows to whom he is talking to, since the client does not authenticate himself. Therefore, after the handshake protocol, the server should never assume the identity of the client.

2 Exercise 2

The three-party Diffie Hellman protocol will work as follows:

- Alice takes a random number a (private key) and sends to Bob and Charlie $A = g^a mod(p)$, p is a large prime number and g is a generator of $\mathbb{Z}p$ (publicly known).
- Bob takes a random number b (private key) and sends to Bob and Charlie $B = g^b mod(p)$.
- Charlie takes a random number c (private key) and sends to Bob and Charlie $C = g^c mod(p)$.
- Alice computes $K_A = (BC)^a mod(p)$; Bob computes $K_B = (AC)^b mod(p)$; Charlie computes $K_C = (BA)^c mod(p)$. Note that $K_A = K_B = K_C = g^{abc} mod(p) \equiv K$. K is the shared private key.

3 Question 3

3.1 Question 1

The protocol guarantees that a user is authenticated at any site on campus without having to login username and password when accessing each of the site. The advantage is that there is only one password/username (or its hash) to be stored securely by one trusted party, instead of having to trust each site to store securely users' passwords. That makes key management easier.

3.2 Question 2

A failure or an attack on the central sign-on facility will compromise the whole system: e.g. if an adversary gets the facility's secret key, it can let any unauthorized user use any site on campus.

3.3 Question 3

The malicious site A when it obtains $u, sign(u)$ can use them as parameters in the standardized $HTTPS\ URL$ of site B (e.g. $https\ :\ //redirectsiteB//username\ =\ u\&signature\ =\ sign(u)$), which will believe that it is receiving a valid request from user u, since $verify(pk, u, sign(u))$ will return true.

3.4 Question 4

We can modify the protocol so that:

- Upon receiving the user's request, a site will send the user, an identifier for site A and a nonce n_a to the central facility that will send back a token $u||n_a||A,\ sign(u||n_a||A)$ to site A.
- A would verify the token using the central facility public key: $verify(pk, u||n_a||A,\ sign(u||n_a||A))$

With this modification, A cannot use the token it receives from the central facility to impersonate the user when talking to site B: when verifying the signature, site B expects to see B as an identifier, not A. And A cannot forge the signature sent by the central facility. The nonce protects against replay attacks (avoid an attacker to intercept the token and uses it to the same site, pretending to be the user).

4 Practical Part

4.1 Question 1

(a) The rest of the requests are made via http protocol using cookies to authenticate the user:

$c\_user = 100004451022564; datr = ME9yUFtsro9IZo9Bsvx-mEM1; fr = 09xG7bUTaV3Praqud.AWUl8VnwV$
$Rhm1BbpziSYpwQr9lOfxnanw; xs = 61\%3ATYLvVr8P4xXmM\%3A0\%3A1349668683; sub =$
$3; p = 68; wd = 1366x643;$

Since after the handshake, the data sent between the user and Facebook is sent via http, an attacker could eavesdrop the communication and intercept the cookie.

(b) An attacker can eavesdrop the communication and intercept the cookie. He can then use the same cookie to connect to Facebook, pretending to be the user himself.

(c) The cookie should be encrypted using the key the server and the client have agreed upon during the handshake.

(d) The user clicked on www.aljazeera.com, www.usatoday.com, sent three POST requests (likely message or text in Facebook).

4.2 Question 2

(a) "weblogin.umich.edu"

(b) The domain name can be hidden since the server sent its certificate (an eavedropper could intercept the certificate and use the certificate authority public key to read the certificate and figure out the domain name).

(c) See the list at the end of the homework.

(d) Three cipher suites use RC4_128 which has been proved to be insecure and one suite uses MD5 as a hash function (for which collisions can be found in few minutes, see previous homework).

(e) TLS_DHE_RSA_WITH _AES_256_CBC_SHA.

Cipher Suite
TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA
TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA
TLS_DHE_RSA_WITH_CAMELLIA_256_CBC_SHA
TLS_DHE_DSS_WITH_CAMELLIA_256_CBC_SHA
TLS_DHE_RSA_WITH_AES_256_CBC_SHA
TLS_DHE_DSS_WITH_AES_256_CBC_SHA
TLS_ECDH_RSA_WITH_AES_256_CBC_SHA
TLS_ECDH_ECDSA_WITH_AES_256_CBC_SHA
TLS_RSA_WITH_CAMELLIA_256_CBC_SHA
TLS_RSA_WITH_AES_256_CBC_SHA
TLS_ECDHE_ECDSA_WITH_RC4_128_SHA
TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA
TLS_ECDHE_RSA_WITH_RC4_128_SHA
TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA
TLS_DHE_RSA_WITH_CAMELLIA_128_CBC_SHA
TLS_DHE_DSS_WITH_CAMELLIA_128_CBC_SHA
TLS_DHE_DSS_WITH_RC4_128_SHA
TLS_DHE_RSA_WITH_AES_128_CBC_SHA
TLS_DHE_DSS_WITH_AES_128_CBC_SHA
TLS_ECDH_RSA_WITH_RC4_128_SHA
TLS_ECDH_RSA_WITH_AES_128_CBC_SHA
TLS_ECDH_ECDSA_WITH_RC4_128_SHA
TLS_ECDH_ECDSA_WITH_AES_128_CBC_SHA
TLS_RSA_WITH_SEED_CBC_SHA
TLS_RSA_WITH_CAMELLIA_128_CBC_SHA
TLS_RSA_WITH_RC4_128_SHA
TLS_RSA_WITH_RC4_128_MD5
TLS_RSA_WITH_AES_128_CBC_SHA
TLS_ECDHE_ECDSA_WITH_3DES_EDE_CBC_SHA
TLS_ECDHE_RSA_WITH_3DES_EDE_CBC_SHA
TLS_DHE_RSA_WITH_3DES_EDE_CBC_SHA
TLS_DHE_DSS_WITH_3DES_EDE_CBC_SHA
TLS_ECDH_RSA_WITH_3DES_EDE_CBC_SHA
TLS_ECDH_ECDSA_WITH_3DES_EDE_CBC_SHA
SSL_RSA_FIPS_WITH_3DES_EDE_CBC_SHA
TLS_RSA_WITH_3DES_EDE_CBC_SHA