Introduction to Cryptology

12.3 - Certificates, PKIs and the TLS Protocol

Federico Pintore

Mathematical Institute, University of Oxford (UK)

Michaelmas term 2020

Certificates

Digital certificates are used to bind entities to public keys.

A trusted party is needed to start the process.

- Consider Catherine, with a pair $(\mathrm{PK}_{CA}, \mathrm{SK}_{CA})$, and Bob, with $(\mathrm{PK}_B, \mathrm{SK}_B)$, and assume Catherine knows PK_B.
- Catherine can issue a certificate for Bob's key, as follows:

$$\mathrm{cert}_{CA \to B} := \mathrm{Sign}(\mathrm{SK}_{CA}, \text{“Bob's key is } \mathrm{PK}_B\text{”}).$$

- More identifying information about the *recipient* (Bob in this example) is usually included.

Certificates and Public Key Infrastructures (PKIs)

Digital certificates are issued within Public Key Infrastractures, which distribute public keys.

A PKI is specified by:

- how users learn PK_{CA};
- how Catherine checks that Bob is the legitimate owner of the public key PK_{B};
- how users decide whether to trust Catherine or not;
- ...

Certificates and Public Key Infrastructures (PKIs)

A Certificate Authority (CA) is a company (or a government agency) that certifies public keys.

In the simplest form of PKI, the CA is only one.

- Every user has to get PK_{CA}.
- The key PK_{CA} can be distributed by physical means, or embedded in the browser.
- When Alice receives $\mathrm{cert}_{CA \to B}$, she will be sure that the certified public key, i.e. PK_B, belongs to Bob.

Certificates and Public Key Infrastructures (PKIs)

A root CA can issue certificates to other CAs, say CA1, CA2, ..., which state:

> "CA1's public key is PK_{CA1} and it is trusted to issue other certificates".

Certificates and Public Key Infrastructures (PKIs)

A root CA can issue certificates to other CAs, say CA1, CA2, ..., which state:

> "CA1's public key is PK_{CA1} and it is trusted to issue other certificates".

The "web of trust" model is a decentralised model which does not rely on a root CA.

Certificates and Public Key Infrastructures (PKIs)

Procedures to invalidate certificates are essential.

Expiration: an expiry date is included in each certificate.

Revocation: a serial number is included in each certificate, and a certificate revocation list is managed and updated regularly.

SSL/TLS

SSL/TLS

The Transport Layer Security (TLS) protocol is used to secure communication over the web.

- Secure Socket Layer (SSL) is the old version, that was developed by Netscape in the mid-1990s.
- TLS is the new version. Major websites support TLS 1.3 (August 2018), although many websites still use TLS 1.0.

A client (web browser) and a server (website) use the TLS protocol to share keys, which are then employed to encrypt and authenticate their communication.

SSL/TLS

The TLS protocol consists of two phases:

- The handshake protocol: it performs an authenticated key-exchange mechanism to establish the shared keys.
- The record layer protocol: it usually uses the shared keys to encrypt/authenticate the communication between parties.

The TLS protocol usually authenticates servers to clients, and then clients can authenticate themselves to servers at the application level by using passwords.

The Handshake Protocol - 1

- Step 1: $C \to S$ - (versions of TLS it supports, ciphersuites, nonce N_C).
- Step 2: $S \to C$ - (Latest version of TLS it supports, ciphersuites, PK_S, $\mathrm{cert}_{i \to S}$, nonce N_S).
- Step 3:
  - C verifies the certificate against the public key of CA_i (it checks the certificate is valid). If the check is positive, C will use PK_S as the server's public key.
  - $(c, \mathrm{pmk}) \leftarrow \mathrm{Encaps}_{\mathrm{PK}_S}(n)$
  - $\mathrm{mk} \leftarrow$ key-derivation function(pmk, N_C, N_S)
  - $(k_C, k'_C, k_S, k'_S) \leftarrow \mathrm{PRG}(\mathrm{mk})$
  - $\tau_C \leftarrow \mathrm{MAC}_{\mathrm{mk}}(\text{transcript : all exchanged messages})$
  - $C \to S$ - (c, τ_C)

The Handshake Protocol - 2

- Step 4:
  - S computes pmk $\leftarrow \text{Decaps}_{\text{SK}_S}(c)$.
  - mk $\leftarrow$ key-derivation function(pmk, N_C, N_S).
  - $(k_C, k'_C, k_S, k'_S) \leftarrow \text{PRG}(\text{mk})$.
  - If $\text{Verify}_{\text{mk}}(\text{transcript}, \tau_C) \neq 1$, then S aborts.
  - Else $\tau_S \leftarrow \text{MAC}_{\text{mk}}(\text{transcript}')$, where transcript$'$ is transcript $\cup$ last message from C.
  - $S \rightarrow C$ - τ_S
- Step 5: If $\text{Verify}_{\text{mk}}(\text{transcript}', \tau_S) \neq 1$, C aborts.

The Handshake Protocol - 2

- Step 4:
  - S computes pmk $\leftarrow \text{Decaps}_{\text{SK}_S}(c)$.
  - mk $\leftarrow$ key-derivation function(pmk, N_C, N_S).
  - $(k_C, k'_C, k_S, k'_S) \leftarrow \text{PRG}(\text{mk})$.
  - If $\text{Verify}_{\text{mk}}(\text{transcript}, \tau_C) \neq 1$, then S aborts.
  - Else $\tau_S \leftarrow \text{MAC}_{\text{mk}}(\text{transcript}')$, where transcript$'$ is transcript $\cup$ last message from C.
  - $S \rightarrow C$ - τ_S
- Step 5: If $\text{Verify}_{\text{mk}}(\text{transcript}', \tau_S) \neq 1$, C aborts.

At the end of the handshake protocol, the client C and the server S share the following symmetric keys: k_C, k'_C, k_S, k'_S.

The Record-Layer Protocol

C will use k_C to encrypt and k'_C to authenticate messages it sends. S will do the same with k_S and k'_S.

S and C will use sequence numbers to prevent replay attacks.

Even TLS 1.2 uses MAC-then-Encrypt paradigm, which is problematic.

Further Reading I


Carlisle Adams and Steve Lloyd.
Understanding PKI: concepts, standards, and deployment considerations.
Addison-Wesley Professional, 2003.


Dan Boneh, Ben Lynn, and Hovav Shacham.
Short signatures from the Weil pairing.
Journal of cryptology, 17(4):297–319, 2004.


Tim Dierks.
The transport layer security (TLS) protocol version 1.2.
2008.


Carl Ellison and Bruce Schneier.
Ten risks of PKI: What you're not being told about public key infrastructure.
Comput Secur J, 16(1):1–7, 2000.

Further Reading II

Amos Fiat and Adi Shamir.
How to prove yourself: Practical solutions to identification and signature problems.
In Advances in Cryptology—CRYPTO'86, pages 186–194. Springer, 1987.

Martin Georgiev, Subodh Iyengar, Suman Jana, Rishita Anubhai, Dan Boneh, and Vitaly Shmatikov.
The most dangerous code in the world: validating SSL certificates in non-browser software.
In Proceedings of the 2012 ACM conference on Computer and communications security, pages 38–49. ACM, 2012.

Further Reading III


Hugo Krawczyk.
Cryptographic extraction and key derivation: The HKDF scheme.
In Annual Cryptology Conference, pages 631–648. Springer, 2010.


Hugo Krawczyk, Kenneth G Paterson, and Hoeteck Wee.
On the security of the TLS protocol: A systematic analysis.
In Advances in Cryptology–CRYPTO 2013, pages 429–448. Springer, 2013.


Leslie Lamport.
Constructing digital signatures from a one-way function.
Technical report, Technical Report CSL-98, SRI International Palo Alto, 1979.