Internet security protocols

In this lecture:

- SSH
- Kerberos
- SSL/TLS

SSH protocol is used to mutually authenticate the Client and the Server and to establish a secure channel between them.

It consists of

- Transport Layer Protocol — unilaterally authenticates the Server to the Client. Establishes a channel that the Client deems secure.
- User Authentication Protocol — authenticates the Client to the Server.
- Connection Protocol — multiplexes the secure channel into several logical channels.

Transport layer protocol.

0. Client and Server establish connection.
1. Both sides send to each other the key exchange messages I_C, I_S containing
   - a nonce N_C, N_S;
   - the protocol and software versions;
   - lists of names of accepted key exchange protocols and cryptographic primitives, in order of preference. Primitives are
     - asymmetric primitives
     - symmetric encryption primitives
     - MAC primitives

If D-H key exchange with the group $\langle g \rangle \subseteq \mathbb{Z}_p^*$, $g^q = 1$ chosen, then

2. Client chooses $x \in_R \mathbb{Z}_q$, sends $e = g^x \bmod p$ to Server.
3. Server
   - chooses $y \in_R \mathbb{Z}_q$, computes $f = g^y \bmod p$;
   - computes $k = e^y \bmod p$;
   - computes $H = h(I_C, I_S, \mathsf{pk}(K_S), e, f, k)$;
   - sends $(\mathsf{pk}(K_S), f, [\![\{H\}]\!]_{K_S})$ to Client.
4. Client
   - checks whether it recognizes $\mathsf{pk}(K_S)$;
   - recomputes f, k, H, checks the signature.

The shared secret K and the hash H are used to derive keys and initial vectors for the secure channel:

- IV $C \to S$ is $h(k, H, \text{“A”}, sid)$;
- IV $S \to C$ is $h(k, H, \text{“B”}, sid)$;
- same for encryption keys and MAC keys ($C \to S$ and $S \to C$).

$sid = H$ for H from the initial key exchange.

All further communication is encrypted and MAC-ed.

Both sides may initiate a new exchange of keys.

A payload M is encoded in a packet as

$$P = (packetlen, paddinglen, M, padding)$$

where pad is used to make the length of packet a multiple of the cipher block length.

A packet is encoded as

$$(\{P\}_{K_{\mathrm{enc}}}, \mathsf{MAC}_{K_{\mathrm{mac}}}(seqno, P)) \ .$$

where $seqno \in \mathbb{Z}_{2^{32}}$.

Encryption: actually the stream of packets $P_1 \| P_2 \| P_3 \| \cdots$ is encrypted, not each packet separately. Standard suggests using some block cipher in the CBC-mode.

Exercise. What is the problem here with MACs? With encrypting?

User authentication protocol.

- Password-based:
  - Client sends his name and password.
  - Server checks that (name,password)-pair is valid.
- Signature-based:
  - Client sends his public key and a signature on various things:
    - including the session identifier.
  - Server checks the knowledge of the key and the signature.

Connection protocol.

Not a security protocol.

Kerberos protocol suite provides a single sign-on to various services offered on a “corporate” network.

- corporate — there exists a single authority.

Each user U has a single password (shared key K_U). It is agreed out-of-band.

The intranet of a large corporation:

- Several domains.
  - in different geographic locations
- Each domain contains several servers S.
- Each domain has a ticket-granting server TGS.
- There is a global authentication server AS.

To get a service from a server S, the client C on behalf of the user U first connects the AS:

1. $C \longrightarrow AS : U, TGS, TI_1, N_1$

   Here TI_1 is the desired validity interval (start and end times) of the ticket.

2. $AS \longrightarrow C : U, T_{C,TGS}, TGT_C$, where

   - $T_{C,TGS} = \{U, C, TGS, K_{C,TGS}, TI_2\}_{K_{AS,TGS}}$
   - $TGT_C = \{TGS, K_{C,TGS}, TI_2, N_1\}_{K_U}$
     - TI_2 is not intended as a security feature here.

C then contacts TGS in a similar manner:

3. $C \longrightarrow TGS : S, TI_3, N_2, T_{C,TGS}, \{C, T_{\text{curr}}\}_{K_{C,TGS}}$
   - The last component (the authenticator) shows that the client could decrypt TGT_C.
   - They should be cached to make sure that they're not used twice.

4. $TGS \longrightarrow C : U, T_{C,S}, TKT_C$, where
   - $T_{C,S} = \{U, C, S, K_{C,S}, TI_4\}_{K_{TGS,S}}$
   - $TKT_C = \{S, K_{C,S}, TI_4, N_2\}_{K_{C,TGS}}$

C and S then authenticate using the shared key $K_{C,S}$:

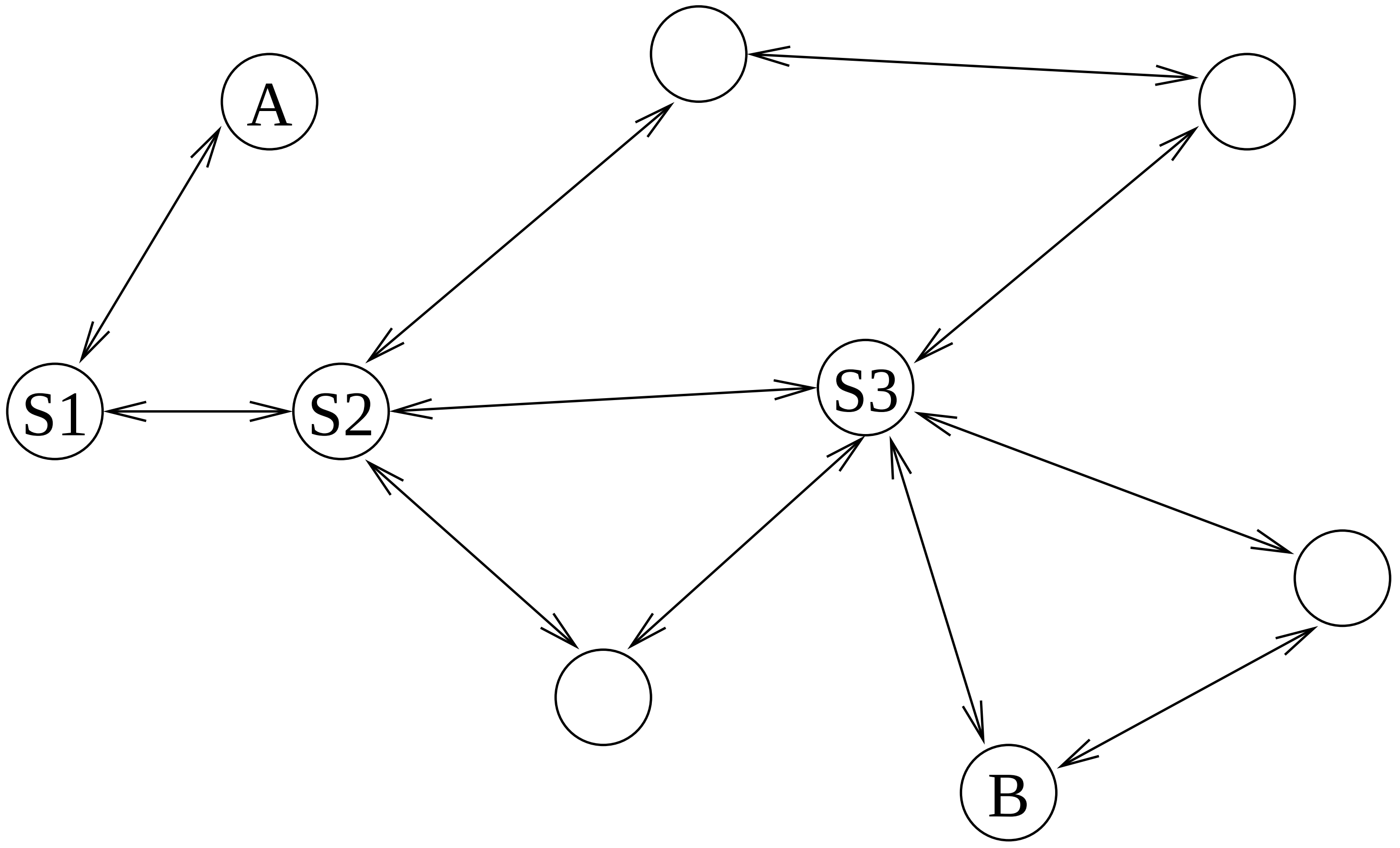
5. $C \longrightarrow S : T_{C,S}, \{C, T_{\text{curr}}\}_{K_{C,S}}$
6. $S \longrightarrow C : \{T_{\text{curr}}\}_{K_{C,S}}$

The key $K_{C,S}$ is also used to secure the channel between C and S.

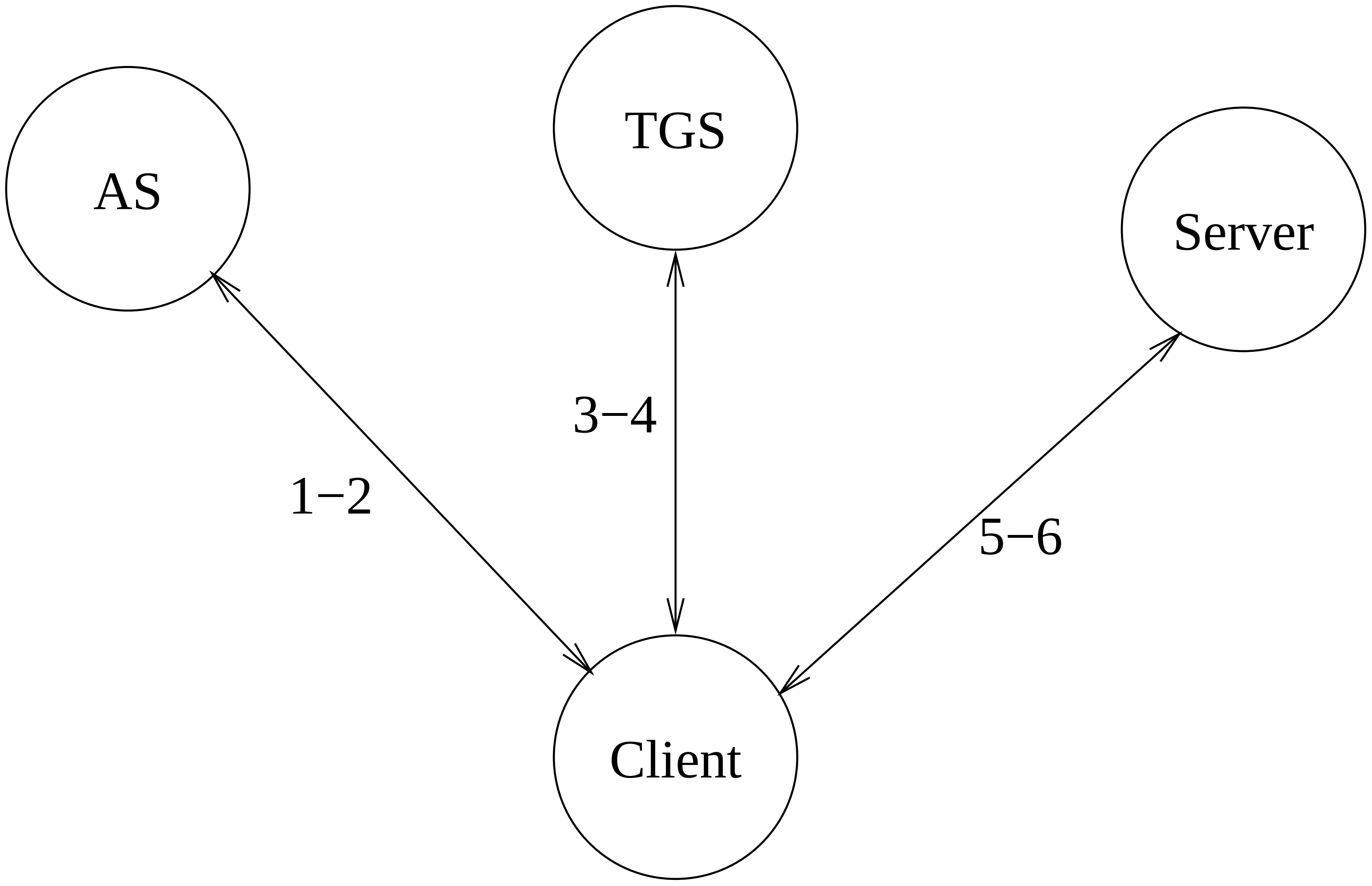
The exchanges 1-2 and 3-4 followed a common pattern:

- A wants to talk to B.
- X and S share a key K_{XS} for $X \in \{A, B\}$.

1. $A \longrightarrow S : A, B, N$
2. $S \longrightarrow A : \{A, B, K_{AB}, TI, N\}_{K_{AS}}, \{A, K_{AB}, TI\}_{K_{BS}}$
3. $A \longrightarrow B : \{A, K_{AB}, TI\}_{K_{BS}}, \{A, T_{\text{curr}}, \ldots\}_{K_{AB}}$
4. $B \longrightarrow A : \{T_{\text{curr}}, \ldots\}_{K_{AB}}$



Somewhat similar to hierarchical PKI. . .



Source of the name “Kerberos”.

TLS consists of

- Handshake protocol
  - Typical public-key protocol
  - Client sends server a secret value encrypted with server's public encryption key.
  - The keys are derived from this secret value.
  - The public keys are found from certificates.
- Record protocol

Record protocol encapsulates the payloads. A payload M is translated to

$$(IV, \{M, \mathsf{MAC}_{K_{\mathrm{mac}}}(seqno, M), pad\}_{K_{\mathrm{enc}}})$$

where pad is used to make the length of the argument of the encryption a multiple of block length.

- Let l be the length of pad in bytes. Then $1 \leqslant l \leqslant 256$ and the bytes in pad are all equal to $l - 1$.

K_{enc} and K_{mac}, as well as encryption and tagging algorithms have been agreed in the handshake protocol.

If a party receives an encrypted packet from the other party, then he

- Decrypts the packet.
- Checks that the padding is correct (at least l last bytes have the value $l-1$ for $l \geqslant 1$).
- If the check fails, then sends an error message, otherwise...
- Checks the MAC.
- If the check fails then sends an error message.
- Otherwise proceeds.

This party may be implementing an oracle that tells whether the padding was correct.

Error message due to incorrect padding and error message due to incorrect MAC may take different amount of time to compute.

Access to such an oracle allows us to decrypt.

Hence the implementation must make sure to insert delays as appropriate.

CBC-mode:

$$c_1 = \mathcal{E}_k(IV \oplus p_1) \qquad c_i = \mathcal{E}_k(c_{i-1} \oplus p_i)$$

Let us be interested in the value of $p_i = c_{i-1} \oplus \mathcal{D}_k(c_i)$.

Let r be a random block. Send $r||c_i$ to the oracle.

If it answers "padding OK" then most probably

$$\mathrm{lsb}_8(r \oplus \mathcal{D}_k(c_i)) = 01_{16} \quad .$$

Exercise. How many tries? How to verify that equation?

We have found $\mathrm{lsb}_8(c_i)$. This tells us $\mathrm{lsb}_8(p_i)$.

Let $r' = r \oplus 03_{16}$. Then $\mathrm{lsb}_8(r' \oplus \mathcal{D}_k(c_i)) = 02_{16}$.

Vary r' (except last 8 bits), until the second last byte of $r' \oplus \mathcal{D}_k(c_i)$ equals 02_{16}.

Etc. Third, fourth, etc. byte. . .