3- Key Management and Infrastructure

Mathias Ravn Tversted

January 14, 2020

Table of contents

Disposition

- Why do we use session keys
- Key distribution centers
- Certification authorities
- Certificate chains
- Identifying users
- Password attacks
- Hardware and biometrics

Session keys

Two-party communication: Session keys

- If we use the same key for all communication, an adversary has more data to work on.
- We have *long-term keys* K_{AB} between A, B. Then we generate a session key $E_{K_{AB}}(K)$ and use that.
- By switching keys here and there, we only use keys on a limited amount of data.
- However having everyone's keys grows quadratically with number of users. How to fix this?

Key Distribution Centers

Key Distribution Problems

Public key cryptography depends on the distribution of keys. Everyone having to store everyone else's keys scales badly. One way to solve this problem is by having centralised databases of keys. These are called *Key Distribution Centers* or (KDC).

Key Distribution Centers

Key Distribution Centers have the following properties

- Based on Secret-Key Systems
- Every user shares K_A with the KDC
- When A wants to talk to B, KDC generates $E_{K_A}(K)$ for A, $E_{K_B}(K)$ for B.
- All users must trust KDC completely
- If the KDC is down or compromised, the entire network is fåkked.

Certification Authorities

Certification Authorities

We can replace the previous example with the following protocol

- Key K_{AB} is replaced by pair (sk_B, pk_B). Here pk_B is public.
- A sends session key $E_{pk_B}(K)$ to B.

How do we distribute these? We have a *Certification Authority*. Each person has authentic copy of pk_{CA} and then everyone authenticates themselves with the CA giving them their key (this might have to be without cryptography, such as offline methods). CA then signs $S_{sk_{CA}}(ID_A, pk_A)$, that is, signs the ID of A, as well as their public key. Everyone can now check that A's public key is legitimate.

Authenticating yourself with the CA

We cannot just rely on network communications to authenticate users with a CA. *Any secure system using cryptography must make use of one or more keys that are protected by physical, non-cryptographic means*.

Certificate Chains: Or Who Verifies The Verifiers

If two people do not have certificates issued from the same CA, they can use what's called Certificate Chains to solve this problem. If CA's certifiy each other's public keys. If user A has CA_1 and B has CA_2. If A receives $Cert_{CA_2}(B, pk)$ then if he also gets $Cert_{CA_1}(CA_2, pk_{CA_2})$ then A can verify that certificate. A chain has the from

$$Cert_{CA_1}(B, pk_B), Cert_{CA_2}(CA_1, pk_{CA_1}).....$$

It should be noted that this requires one to trust the initial part of the chain in the first place.

How to identify users

How to identify users

There are a number of ways to identify users:

- Something you know (passwords)
- Something you have (hardware tokens)
- Something you are (biometrics)

Who they are? Biometric Security

Biometric Security is when a system makes use of fingerprints, voice, facial recognition and other biological factors unique to an individual to identify them. This has a privacy downsides, and if the information is leaked, it could still be used to attack systems.

Something They Have: Hardware Security

Hardware may have their own keys, such as for RSA encryption, which is only accessible through the CPU. It is also possible to make *tamper evident* hardware, which can tell if it has been tampered with.

Two-factor authentication: Two-factor authentication works by demanding that the user authenticate themselves not only with a password, but also by possession of a piece of hardware, such as a smartphone. This means that even if the password is compromised, it is still possible to deny intruders access.

Password Security

Passwords identity users from what they know. And generally we want to protect passwords from the following 4 attacks

- The adversary can guess and verify their guesses
- The password is transmitted in an insecure manner, and the adversary can intercept it
- The password is stored in an insecure manner, and can be stolen from the user
- The password can be stolen from the verifier

Password security

Guessing passwords: This can be remedied with password rules. However, the more password rules, the fewer combinations people have to guess. Having ample length and characters (but mostly length) and a database of vulnerable passwords that are forbidden seems to be the best policy

Insecure Transmission: Use encryption when communicating over the internet. Without encryption, using a password becomes useless.

Insecure storage by the user: Could be helped by using other forms of authentication, such as two-factor authentication.

Stealing from verifier: Salt and hash passwords, store

Password Security

Number of possible passwords is C^l where l is the length and C the character set. It is therefore better to choose longer passwords.

Restrictive rules are bad, because they reduce the character set. Password rotation rules make people use predictable passwords.

Passphrases are easier to remember and are usually longer.