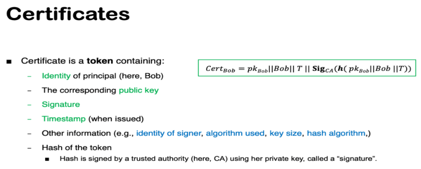
Goal of authentication: bind identity to card/token/password/key

By using a secret key, the AS can verify the identity of Ali



Certificates

- Certificate is a **token** containing:
  - Identity of principal (here, Bob)
  - The corresponding public key
  - Signature
  - Timestamp (when issued)
  - Other information (e.g., identity of signer, algorithm used, key size, hash algorithm,)
  - Hash of the token
    - Hash is signed by a trusted authority (here, CA) using her private key, called a "signature".

$Cert_{Bob} = pk_{Bob}||Bob|| T || \mathbf{Sig}_{CA}(\boldsymbol{h}(pk_{Bob}||Bob ||T))$

ublic-Key Infrastructure (PKI): bind identity to public key

Crucial as people will use key to communicate with principal whose identity is bound to key, Erroneous binding means no secrecy between principals, Assume principal identified by an acceptable name - called Common Name.

A PKI consists of: Cetificates, Certificates Authority (CA), a resposity for retrieving certificates, A method of evaluating a chain of certificates from known public keys to the tartget name, amethod of revoking certificates

PKI Trust Models:

Hierarchical CAs with cross-certification(Multiple root CAs that are cross-certified

Oligarchy model (commonly used in browsers)

Browsers or Operating Systems come pre-configured with multiple trust anchor certificates

New certificates can be added(be careful)

Bad certificate can be revoked

Distributed model

No root CA; instead, users certify each other to build a "web of trust"

PKI Security

What happen if root authority is compromised?

The certificate chain rooted from this CA is corrupted

PKI faces many challenges

Hash collisions: Obsolete hash algorithms

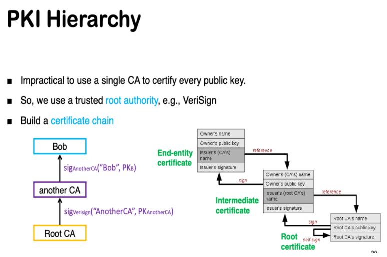
Weak security at CAs: attackes can issue rogue certificates

Users not aware of attacks happening

.Certificate

Certificate Authority (CA)

CA is a trusted third party who issues certificates



PKI Hierarchy

- Impractical to use a single CA to certify every public key.
- So, we use a trusted root authority, e.g., VeriSign
- Build a certificate chain

Certificate Verification PIC3

Certificate Expiration: Certificate holds an expiration date and time, Certificate may need to be **revoked** before expiration, Revocation is **very important** to PKI.

Revocation PIC4

Certificate revocation list (CRL)

A list of revoked certificates

Issued by CA

Signed by CA

Distributed to clients

Clients check CRL before using certificate

Online Certificate Status Protocol (OCSP)

A protocol for checking the status of a certificate

Issued by CA

Signed by CA

Distributed to clients

Clients check OCSP before using certificate

Rogue Certificate PIC5

Password authentication

Authentication is the process of verifying the identity of a user or system.

How do you prove to someone that you are who u claim to be?

Show **credential**

Credential can be:

Something you know (password, certificate,..)

Something you have (token, IP address, hardware/moblie device,..)

Something you are (biometric)

How to steal or exploit passwords?

After a sucessful intrusion

Steal install sniffer or keylogger to steal passwords

Exploit fetch password files and run cracking tools

Use of strong password, why?

Because weak password caused 30% of ransomware infections

Stolen credentials led to nearly 50% of attacks

How to **store** password in the system?

In password files indexed by user ID, in plaintext, Encrypted, hashed.

Hashing, Salting, Encryption, Password managers

assword Hashing

Hashing is the process of converting a password into a unique string of characters that cannot be reversed.

When user enters a password

System computer H(password) and compares with the entry in the password file

System does not store the actual password

Password hash funtion

Onewayness: given H(password), it is hard to deduce password

slow to compute: restrict the speed of brute force attacks

he way to crack password

Brute force attack: after attacker gets ur password file, he tries to hash all possible values and compare the results witht e entries in the password file.

There are 94 candidate characters, 8 characters long password, 94^8 = 6.5*10^14 possible passwords

But since password are not truly randomm. **Dictionary attack** is more effective

Dictionary attack: attacker uses a dictionary of common passwords to crack the password

Attacker pre-computes H(password) for every word in the dictionary.

Pre-computing needs to be done only once and offine

One the password file is obtaioned, cracking is done immediately(search and compare)

Password guessing tools also ultilize frequency of letters, password patterns, etc.

Rainbow table attack: attacker pre-computes H(password) for all possible passwords and stores the results in a table

A space-time tradeoff, can purchase from the Internet

ountermeasures:

Salting is the process of adding a random string to the password before hashing

is a random value chosen for each user

It chosen randomly when password is first set and stored in the password file

password hash = H(salt + password)

Users with the same password have different entries in the password file

Salting adds randomness to password hash, make offline dictionary attack harder

Advantages of Salting

Without salting, attacker can pre-compute hashes of all dictionary words once.

for **ALL** password entries

- – With salting, attacker must pre-compute hashes of all dictionary words once
- for **EACH** password entry (with 12 bit salt, same password can have 2^12 = 4096 different hashes)
- for all hash algorithms, attacker must try all dictionary words for each salt value in the password file

d.Other Password Security Risks

- Weak password, default password, keystroke loggers, broken implementations, social engineering,
- Password strength
- Usability – password manager can help
  - Hard to remember passwords
  - Password management issues
- Password reuse
- Password sharing
- Heavy reuse

e.Way to improve password security

- Password managers are software programs that store and manage passwords
  - What happen when password manager is compromised?
- Password manager is a single point of failure
- Malware, social engineering, brute force attack, insider attack
- Graphical passwords easy to remember, no need to write down
  - Draw a picture, select a point, select a color
  - Side channel attack may reveal the password
- Add biometrics: unique, hard to fake, no need to remember
  - Fingerprint, retina, voice, face, etc.
  - Require special hardware, hard to revoke, false positive, can be stolen
- Multi-factor authentication
  - Levarage **more than one** authentication mechanism for authentication
  - Google: Password + SMS
  - FIDO: Password + hardware

Distributed authentication

Basic concepts

- Potential threats
  - User impersonation: a malicious user with access to a workstation pretends to be another user using the same station.
  - Network impersonation: a malicious user changes the network address of his workstation to impersonate another workstation.
  - Eavesdropping, message modification, and replay attacks
- How to prove user's identity when requesting services from machines on the network?
  - Many to many authentication: m clients, n servers
  - Public-key based solution: need m+n public-private key pairs – PKI
  - Secret-key based solution: mxn secret keys shared between each(client,server)
- Better solution? Kerberos
- What can be expect?
  - Secure against attacks by passive eavesdroppers and active malicious attackers
  - Transparent so that users do not notice authentication and users' effort is minimal.
  - Scalable to serve a number of users and servers

Kerberos

Kerberos Overview

- Key idea: use a trusted third party to authenticate users

Kerberos step

1. Send password to AS - insecure to send plaintext password
   - Convert "password" into client master key: Ka
   - Ka is shared with Key Distribution Center (KDC)
2. Issue ticket - ticket needs to be encrypted. Otherwise, it can be forged.
   - Client -> KDC: "I am Alice, I want to talk to Bob"
   - IDa, IDb, timestamp, lifetime, TGT
   - KDC -> Client: encrypted session key and ticket
   - Eka(Ka-b, IDb, Tb)
   - Ka-b: session key geneerateed by KDC for Alice and Bob
   - Tb = EKb(Ka-b, IDa, IDb)
3. PIC6

4. Kerberos Messages

- PIC7
- PIC8

Kerberos Discussion

- The first single sign-on system - sign-on once, access all resources
- The design goal PIC9
- Scenario PIC10
- The protocal

Kerberos Term keys

- PIC11
- It provides a centralized authentication service.
- It can support mutual authentication
- Entirely based on symmetric cryptography
- Less keys to remember for clients
  - KDC maintains long-term secret keys for each client and server, but servers don't.
  - Client requests short-term session keys(ticket + session key) from KDC and manages them locally.
- Less communication overhead(client sends both ticket and authenticator toserver, so no need to wait)
- More scalable in a large distributed system. #### Kerberos Security
- The protocol, ticket, session key, authenticator
- PIC12
- PIC13 ## DS Security ### Basic concepts
  - CIA ### Inference attacks #### Tracker attack #### Controls for Inference Attacks
- Three paths to follow:
  - Suppress obviously sensitive information (easy to implement, but it hurts database usability)
  - Track what the user knows (costly to implement)
- Used to limit queries accepted and data provided
- Disguise the data using random perturbation, rounding, swapping (cause new problem with precision)
  - Applied only to the released data
  - "Differential Privacy" #### Possible controls
- Query controls – Limit overlap between new and previous queries.
- Item controls – Suppression: query is rejected without sensitive data provided. ■ Limited response, combined results, random sample – Concealing: the answer is close to but not exactly the actual answer.
- Partitioning – Cluster records into exclusive groups and only allow queries on entire groups. ### Access control
- Access control is the process of restricting access to objects in a system
- Access control policy specifies the authorized accesses of a system.
  - Managed by the database administrator (DBA)
- Access control mechanism implements and enforces the policy.
  - Implemented in AC models, enforced by DBMS. #### Access control models
- Subject the active entity that requests access to an object
- Object the passive entity accessed by a subject
- Access Operation how a subject is allowed to access an object
- Similar access control for OS
  - Mandatory access control (MAC)
  - Discretionary access control (DAC)
  - Role-based access control (RBAC)

  #### DAC
- Discretionary access control (DAC) is a form of access control in which access rights are assigned to objects based on the identity of the subject requesting access.
- Widely used in multi-user systems
- What does discretionary mean?
  - Access to data objects(files, directories, etc..) is **permitted based on the identity of the user**.
  - Users can be given the ability of **passing on their privileges** to other users.
  - **granting** and **revoking** privileges is regulated by an administrative policy.
- Subjects
  - A user is referred to by authorization ID(Typically, the login name.)
  - There is an authorization ID: "PUBLIC" (Granting a privilege to PUBLIC makes it available to any authorization ID.)
- Objects (on which privileges exist)
  - In database systems, the objects include stored tables and views.
  - Other privileges are the right to create objects of a type, e.g., triggers.
- Privileges
  - A file system identifies certain privileges on the objects (i.e., files) that it manages, typically, read, write, execute. #### Role-based access control (RBAC)
- Role-based access control (RBAC) is a form of access control in which access rights are assigned to users based on their roles within an enterprise.
- AC is centered around the concept of a role.
- RBAC is a semantic construct.
- Access control is centered around roles
- It provides good flexibility and
  - Access control models: DAC, RBAC
  - DAC: subjects and privileges, GRANT/REVOKE
  - RBAC

OS Security

- OS must protect users from each other - seperation
  - Memory protection: protecting OS kernel, process isolation
  - File protection: access control
  - General control and access to objects: refrence monitor and access control.
  - User authentication

Software Security

- Software flaws(non-malicious) -Buffer overflow: what causes the problem, how to mitigate
  - Incomplete mediation: injection attacks, why they work
  - TOCTTOU: what is the vulnerability

What is the cost of the problem how to solve it Injection attactk Different type of injection attack Risk condition: time of change

III. Software Security

What is the cost of the problem how to solve it Injection attactk Different type of injection attack Risk condition: time of change Software Security

Stack Overflow Dj

The fundamental problem the fundamental of this attack, the fundamental of prevent this attack ask about the paticular tool