ACCESS CONTROL

Prepared By

-Anooja Joy

ACCESS CONTROL BASICS

- **"Access"** is a key concept that implies **flow of information** from a **subject to an object** or from an **object to a subject.**
- **Eg:**
  - When a **user (a subject) updates a data set (an object)**, the information flows from the subject to the object.
  - When **a user reads a record from a data set**, the information flows from the object to the subject.
- **Access control policies** make sure of fundamental requirements of a secure system through **authentication** and **authorization.**
- It is typically provided to **Operating System** and **Databases.**

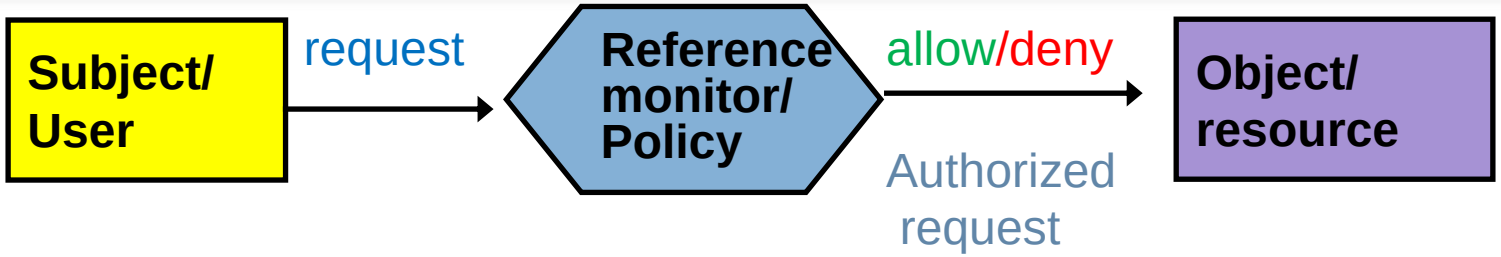
Access Control

- **Access control** is the collection of mechanisms for selective restriction of access to a **place or other resource** ie, determining who has access to specific files, or other system resources. **Its function is to control which (active) subject have access to which (passive) object with some specific access operation.**
- **Examples**
- Passwords, Encryption, Antivirus software, Firewalls, Routers, Protocols to preserve confidentiality and integrity of data.
- The main aim of access conbtrol mechanism is **to control operations executed by subjects** in order **to prevent actions that could damage data and resources.**

Goals of access control

1. Granting access
2. Limiting access
3. Preventing access
4. Revoking access

Abstract Access Control Model



- The very nature of access control suggests that there is an ***active subject*** requiring access to a ***passive object*** to perform some specific access operation.
- In any access control model, the entities that can **perform actions/ inititates requests in the system** are called subjects. Subjects Issues requests to access the object. Subjects can be considered as software entities/ human users/processes, modules, roles and Objects can be files, processes, etc.

Eg: Users, Process, Groups

- The **entities representing resources that hold data** to which access may need to be controlled are called objects.

Eg: memory, files, directories, nodes on a network, I/O devices, interprocess messages etc.

- A reference monitor **grants or denies access.** Reference monitor **makes authorization decisions.** It knows which subjects are allowed to issue which requests.

Eg: Firewall, Operating System, Database, JVM

Abstract Access Control Model

- The **access control mechanism** uses some access control policies to decide **whether to grant or deny a subject access to a requested resource.**
- Access control policies map principals, objects and access rights.
- Access rights refer to the user's ability to access a system resource.
  - read
  - write
  - append
  - execute

Access Modes

- 2 access modes

1. **Observe:** Subject may only look at the content of the object.

   Example: Read, Search

2. **Alter:** Subject may change the content of the object

   Example: Write, Append, Delete

- Any access right can be performed within each access mode.
- The owner of the resource sets the access rights to the resource.

Basic Operations in Access Control

1. **Grant permissions:** Inserting values in the matrix's entries
2. **Revoke permissions:** Remove values from the matrix's entries
3. **Check permissions:** Verifying whether the entry related to a subject *s* and an object *o* contains a given access mode

Access Control Models

- **Mandatory access control (MAC)**
- In this nondiscretionary model, people are granted access based on an information clearance. A central authority regulates access rights based on different security levels. It's common in government and military environments.
- Mandatory Access Control begins with security labels assigned to all resource objects on the system. These security labels contain two pieces of information - a classification (top secret, confidential etc) and a category (which is essentially an indication of the management level, department or project to which the object is available).
- When a user attempts to access a resource under Mandatory Access Control the operating system checks the user's classification and categories and compares them to the properties of the object's security label. If the user's credentials match the MAC security label properties of the object access is allowed. It is important to note that both the classification and categories must match. A user with top secret classification, for example, cannot access a resource if they are not also a member of one of the required categories for that object.

Access Control Models

Discretionary access control (DAC)

- In this method, the owner or administrator of the protected system, data, or resource sets the policies for who is allowed access.
- Discretionary Access Control (DAC) allows each user to control access to their own data. DAC is typically the default access control mechanism for most desktop operating systems.
- Instead of a security label in the case of MAC, each resource object on a DAC based system has an Access Control List (ACL) associated with it. An ACL contains a list of users and groups to which the user has permitted access together with the level of access for each user or group. For example, User A may provide read-only access on one of her files to User B, read and write access on the same file to User C and full control to any user belonging to Group 1.
- Discretionary Access Control provides a much more flexible environment than Mandatory Access Control but also increases the risk that data will be made accessible to users that should not necessarily be given access.

Access Control Models

- **Role-based access control (RBAC)**
- RBAC grants access based on defined business functions rather than the individual user's identity. The goal is to provide users with access only to data that's been deemed necessary for their role within the organizations. This widely used method is based on a complex combination of role assignments, authorizations, and permissions.
- Role Based Access Control (RBAC), also known as Non discretionary Access Control, takes more of a real world approach to structuring access control. Access under RBAC is based on a user's job function within the organization to which the computer system belongs
- RBAC assigns permissions to particular roles in an organization. Users are then assigned to that particular role.
- For example, an accountant in a company will be assigned to the Accountant role, gaining access to all the resources permitted for all accountants on the system. Similarly, a software engineer might be assigned to the developer role.
- **Attribute-based access control (ABAC)**
- In this dynamic method, access is based on a set of attributes and environmental conditions, such as time of day and location, assigned to both users and resources.

Access Control Mechanism

- Access control is usually taken care of in following steps, which are authentication, and authorization
  - Ø **Authentication:** Are you who you say you are? Who are you?
    - v Determine whether access is allowed or not
    - v Determining the identity of a user ie, May also involve verification of IP address, machine, time, etc…
    - Enforcement Mechanism: (password/crypto/smartcard/biometricetc.)
  - Ø **Authorization:** What are you allowed to do? Are you allowed to do that?
    - v Once you have access, what can you do?
    - v Enforces limits on actions
    - Enforcement Mechanism: (Acess control)
- Verification of access rights -->***Access Control***
- Granting of access rights --> ***Authorization***

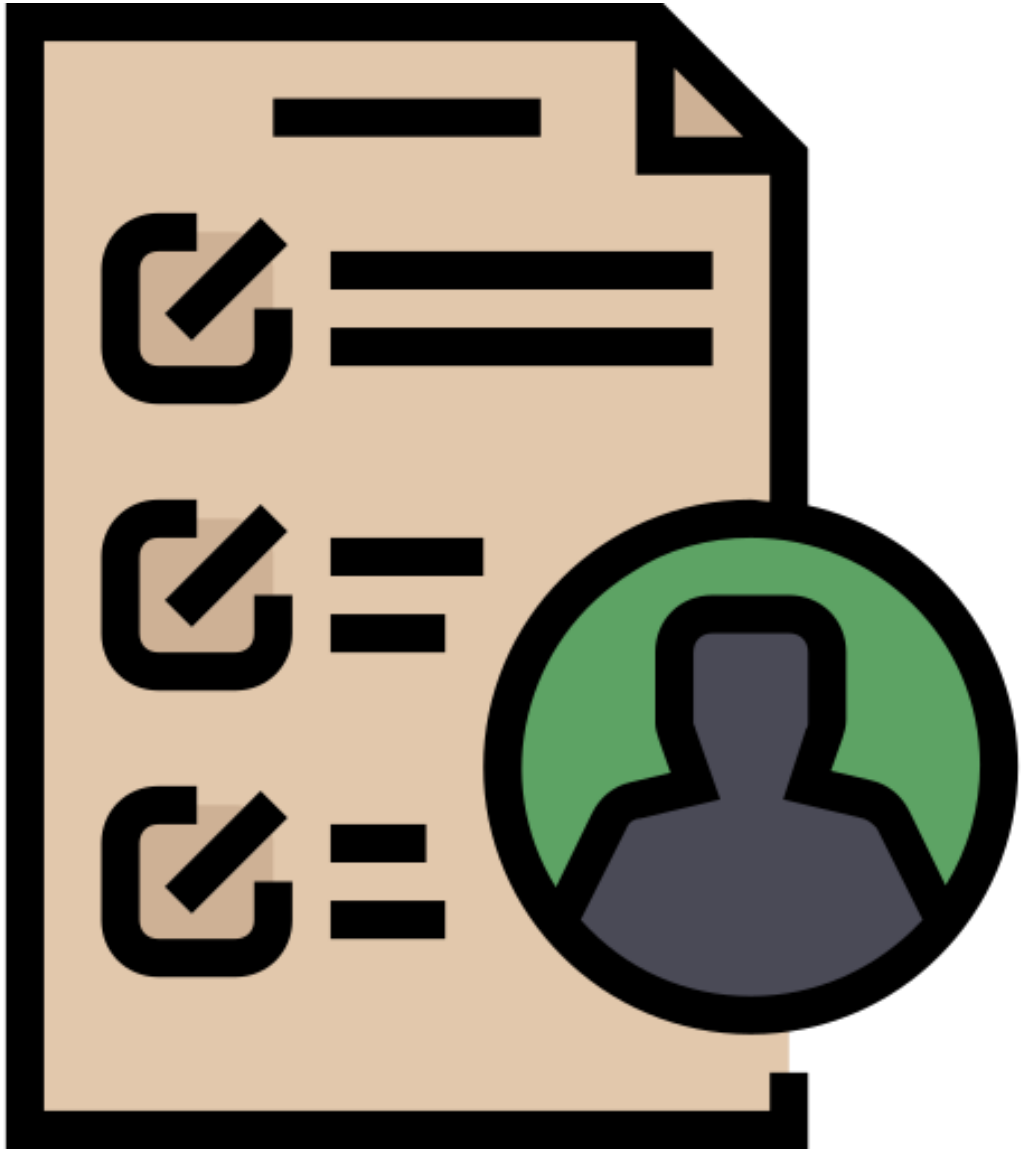
AUTHENTICATION

- Verifies credentials to determine whether users are who they claim to be.
- Users or persons are verified.
- Works through passwords, biometrics, one-time pins, or apps
- Eg: Logins, OTP or a magic link, 2FA/MFA, Single sign-on (SSO)
- It is **visible to the user**
- It is **partially changeable by the user.**
- **Data move** through **ID tokens**

AUTHORIZATION

- Grants or denies permissions to determines what users can and cannot access.
- Users or persons are validated.
- Works through settings maintained by security teams
- Eg: Role-based access controls (RBAC), JSON web token (JWT),SAML(a standard SSO format), OAuth
- It is **not visible to the user**
- It is **not changeable by the user.**
- **Data move** access tokens

AUTHORIZATION



Access Control Structures

- Access control structures are mechanisms for implementing access policies or used for implementation of protection model:

1. Access Control Matrix
2. Capability Tables
3. Access Control Lists
4. Role-Based Access Control
5. Rule-Based Access Control
6. Restricted Interfaces
7. Content-Dependent Access Control

Requirements for access control structures:

- an ability to express control policies
- verifiability of correctness.
- scalability and manageability

Access Control Matrix

- Access control matrix is a basic control structure to implement protection model.
- The *access control matrix* is a matrix with each subject/domain(user/process/procedure) represented by a row, and each object(resources) represented by a column
- Characterizes possible rights of each subject with respect to each object ie, access allowed by subject S to object O is stored in intersection. The entry *M[s, o]* lists the operations that subject *s* may carry out on object *o*.
- *ACM enforces restriction on accounting data.*
- The access control matrix provides a useful way to think about the rights in the system.
- **Examples:** read/write/execute/etc

Access Control Matrix

- Subjects (users) index the rows
- Objects (resources) index the columns

	OS	Accounting program	Accounting data	Insurance data	Payroll data
Bob	rx	rx	r	—	—
Alice	rx	rx	r	rw	rw
Sam	rwx	rwx	r	rw	rw
Accounting program	rx	rx	rw	rw	rw

Example

Objects

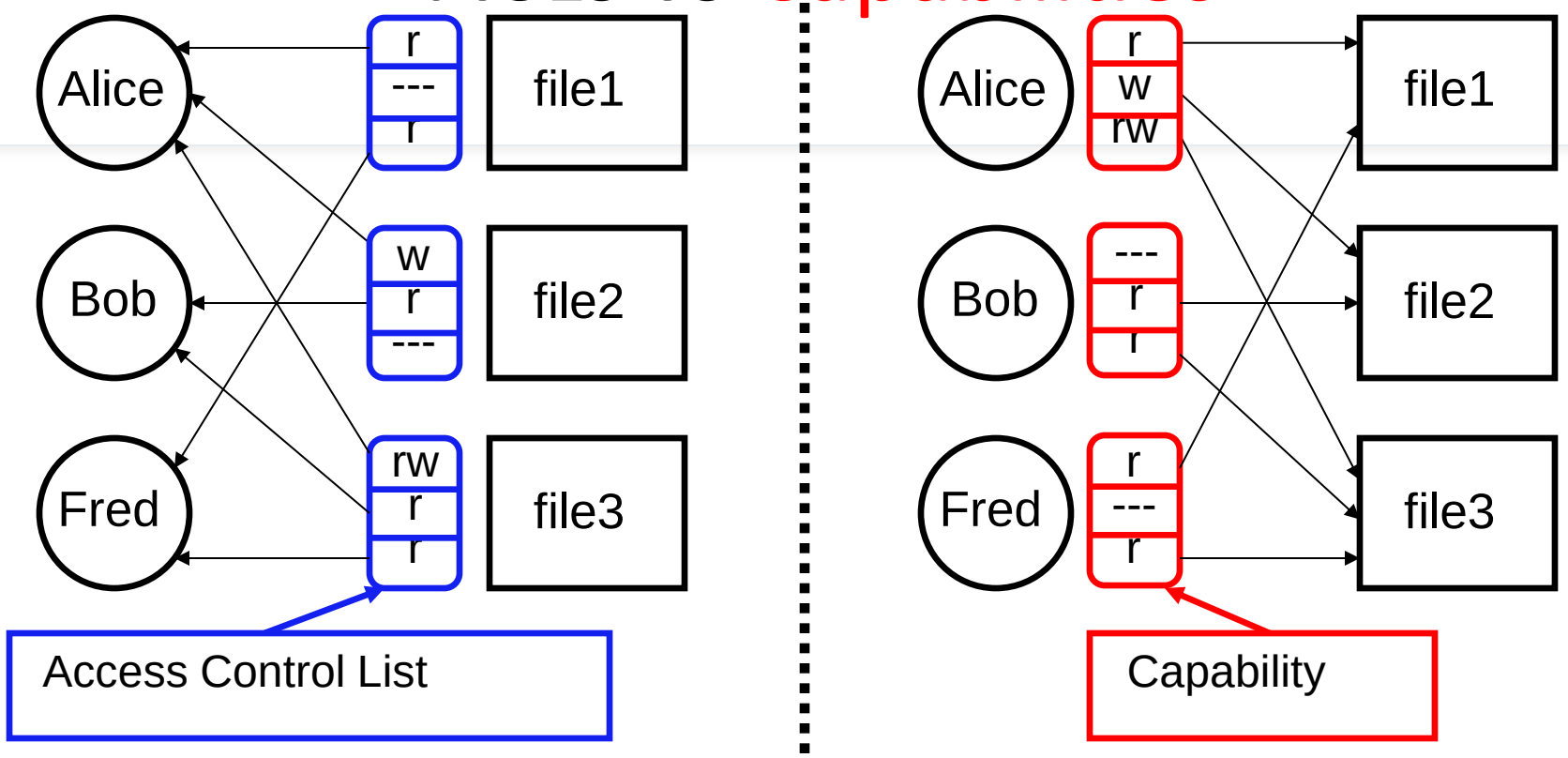
Subjects

	File 1	File 2	File 3	…	File n
User 1	{r,w}	{w}			{r,w}
User 2	{w}	{w}	{r,w}		
User 3				{r}	{w}
…					
User k	{r}	{r}	{r,w}	{r}	{w}

Access Control Matrix

- **Advantages:**
  - clarity of definition
  - easy to verify
- **Disadvantages**:
  - Poor **Scalability.** "Small' change can result in "many" changes to the access control matrix
  - **Poor handling of changes.** One central matrix modified every time subjects/objects are created/deleted, or rights are modified
  - Number of subjects/objects is very large. Could be 100's of users, 10,000's of resources, Then matrix has 1,000,000's of entries. How to manage such a large matrix?

ACLs vs Capabilities



- Note that arrows point in opposite directions...
- With ACLs, still need to associate users to files

2. Access control lists (ACLs)

- ACL is an **object-centered description of access rights**: Can be viewed as storing the columns of the access control matrix with the appropriate object, leaving out empty entries
- Ideally, one list per object showing all subjects with access and their rights
- Missing subjects given “default” access. Easy to make an object public

EXAMPLE:

- bill.doc: {Bob: read,write}
- exit.exe: {Alice: execute}, {Bob: execute}
- fun.com: {Alice: execute, read}, {Bob: execute, read,write}

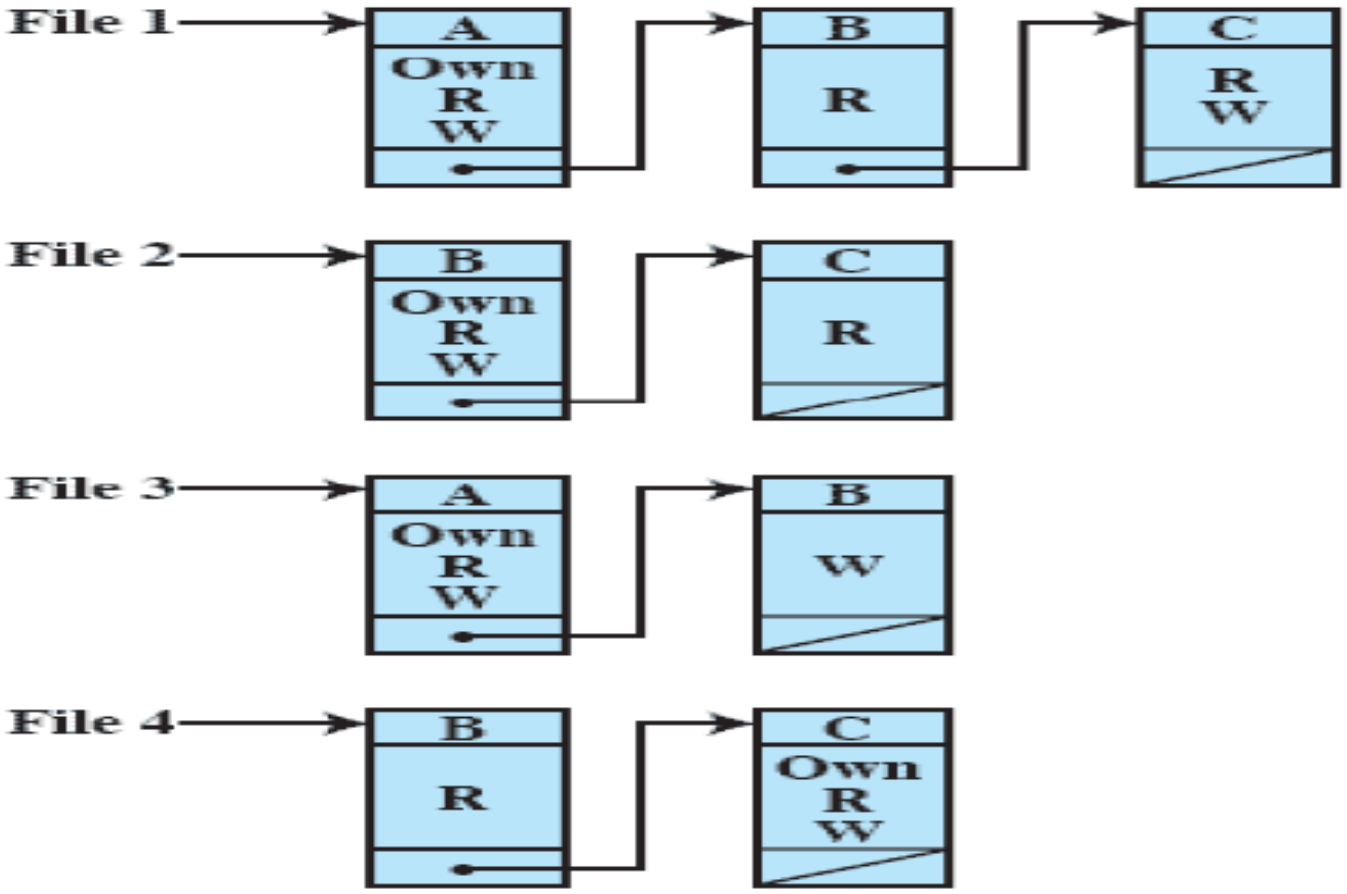
ACL

For example, the ACL corresponding to insurance data

	OS	Accounting program	Accounting data	Insurance data	Payroll data
Bob	rx	rx	r	—	—
Alice	rx	rx	r	rw	rw
Sam	rwx	rwx	r	rw	rw
Accounting program	rx	rx	rw	rw	r

(Bob, —), (Alice, rw), (Sam, rw), (accounting program, rw).

ACL



(b) Access control lists for files of part (a)

ACL

- **Advantages:**
  - easy access to object access rights
- **Disadvantages:**
  - poor overview of access rights per subject
  - difficulty of revocation
  - difficulty of sharing
  - Need a mechanism for handling conflicts

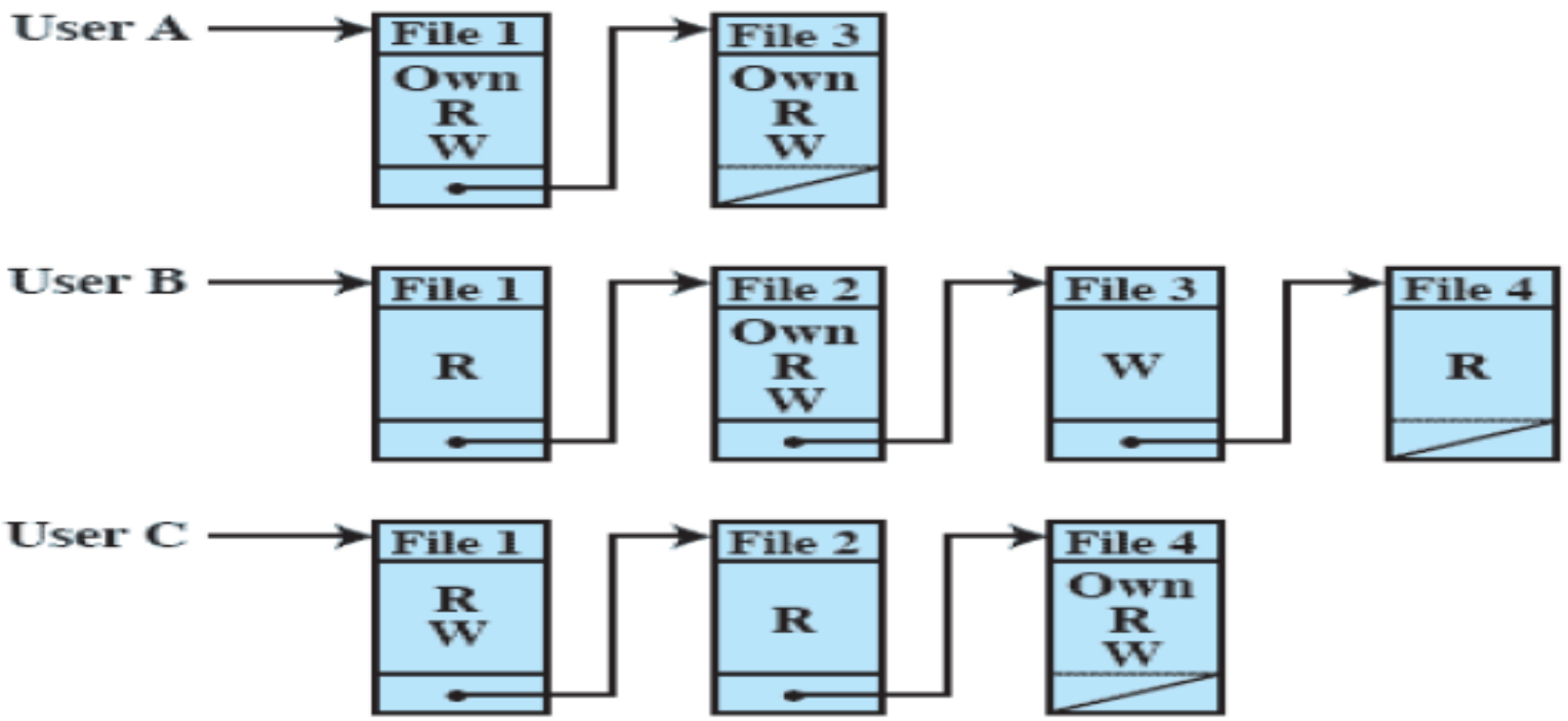
3. Capabilities

- Can be viewed as storing the rows of the access control matrix with the corresponding subject, leaving out empty entries.
- Whenever a subject tries to perform an operation, consult its row of the access control matrix to see if the operation is allowed.
- A capability is an unforgeable token giving user access to an object and describing the level of allowable access
- Capabilities can specify new types of rights

EXAMPLES:

- Alice: {edit.exe: execute}, {fun.com: execute, read}
- Bob: {bill.doc: read,write}, {edit.exe: execute}, {fun.com: execute, read,write}

Capabilities



(c) Capability lists for files of part (a)

Capabilities (or C-Lists)

- Store access control matrix by **row**
- Example: Capability for **Alice** is in **red**

Subjects \ Objects	OS	Accounting program	Accounting data	Insurance data	Payroll data
Bob	rx	rx	r	—	—
Alice	rx	rx	r	rw	rw
Sam	rwx	rwx	r	rw	rw
Accounting program	rx	rx	rw	rw	rw

Capabilities: two approaches

Advantages:

- easy ownership transfer
- easy inheritance of access rights

Disadvantages:

- poor overview of access rights per object
- difficulty of revocation
- need for extra integrity protection

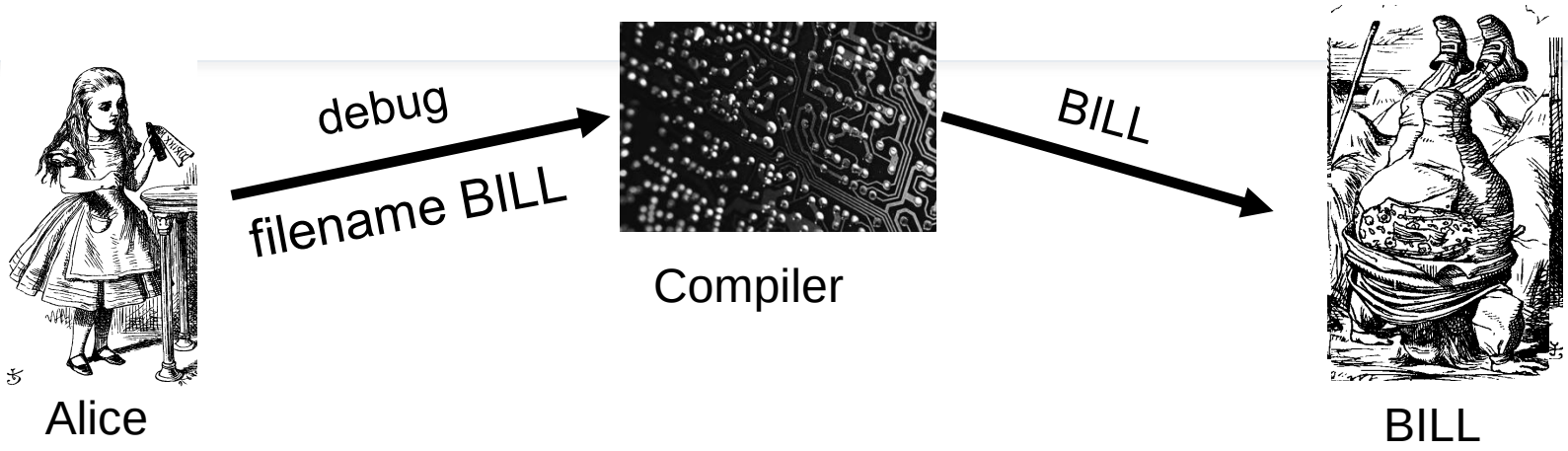
- Ticket is held by the OS, which returns to the subject a pointer to the ticket
- Ticket is held by the user, but protected from forgery by cryptographic mechanisms
  - How?
  - Ticket can then be verified by the OS, or by the object itself

Confused Deputy

- Consider a system with:
  - **2 System resources:** Compiler and BILL file (billing info) that contains critical billing information
  - **1 user:** Alice.
- Compiler can write to any file, while Alice can invoke the compiler and she can provide a filename where debugging information will be written.
- However, Alice is not allowed to write to the file BILL, since she might corrupt the billing information.

	Compiler	BILL
Alice	x	—
Compiler	rx	rw

Confused Deputy



- Suppose that Alice is invoking compiler, and she provides BILL as the debug filename. This command should fail, as Alice doesn't have privilege to access file BILL, does have the privilege to overwrite BILL. The result of Alice's command should be the trashing of the BILL.
- Ccompiler is **deputy** acting on behalf of Alice and Compiler is **confused** since its acting based on it's own privileges or should be acting based on ALice's proivileges.

Confused Deputy

- There has been a separation of **authority** from the **purpose** for which it is used
- With ACLs, more difficult to prevent this
- With Capabilities, easier to prevent problem
  - Must maintain association between authority and intended purpose
- Capabilities — easy to **delegate** authority

ACLs vs Capabilities

- ACLs
  - Preferable when users manage their own files and when protection is data oriented.
  - Protection is data-oriented
  - Easy to change rights to a particular resource
  - more difficult (but not impossible) to avoid the confused deputy
  - ACLs are used in practice far more often than capabilities
- Capabilities
  - Easy to delegate avoid the confused deputy
  - Easy to add/delete users
  - More complex to implement and they have somewhat higher overhead
  - Due to the ability to delegate, easy to avoid the confused deputy when using capabilities

COVERT CHANNEL

Covert Channel

- **Covert channel**: a communication path not intended as such by system's designers.
- It arises in **network communications** and are **impossible to eliminate**.
- **Covert channel** is any communications channel that can be exploited by a process to transfer information in a manner that violates the system security policy.
- Method of communication that is not part of the actual systems design but can be used to transfer information to outside sources
- MLS designed to restrict legitimate channels of communication.
- May be other ways for information to flow. For example, resources shared at different levels could be used to "signal" information

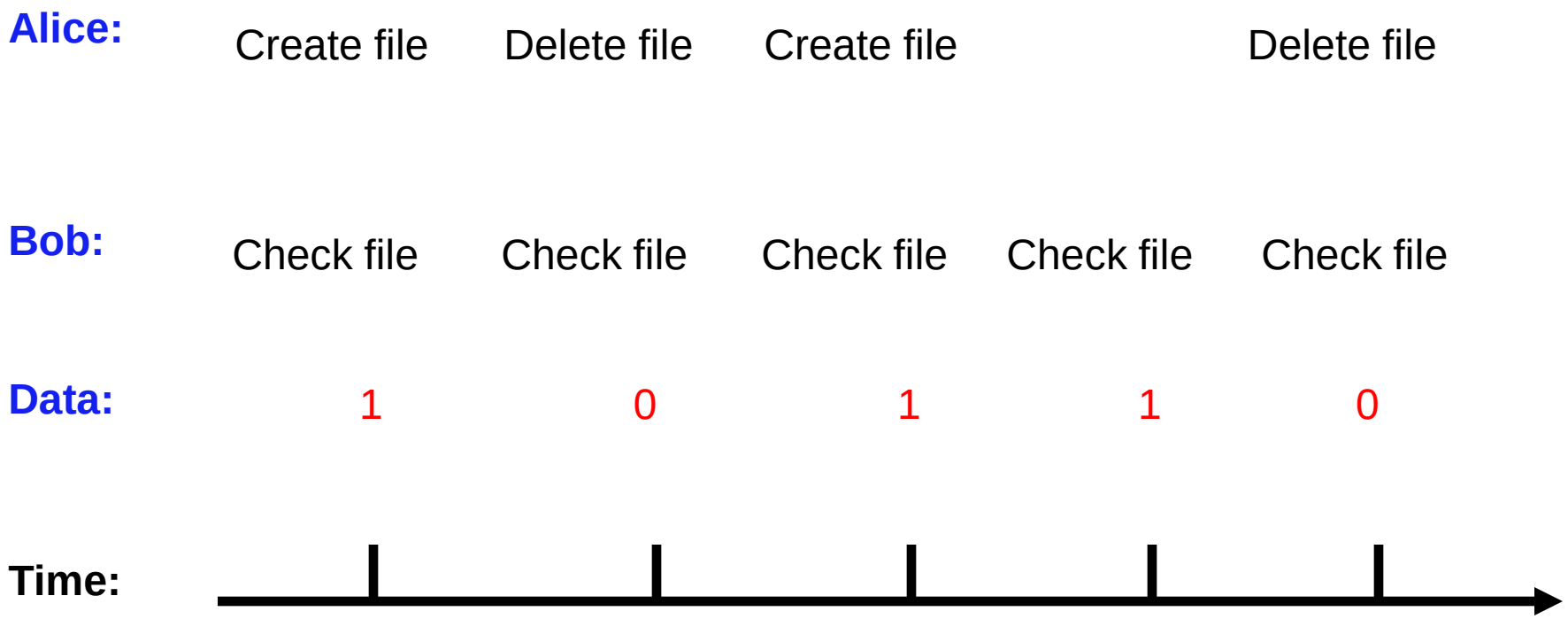
Covert Channel

- Potential covert channels are everywhere
- But, it's easy to eliminate covert channels:
  - "Just" eliminate all shared resources and all communication!
- Virtually impossible to eliminate covert channels in any **useful** information system
  - DoD guidelines: **reduce covert channel capacity** to no more than 1 bit/second
  - Implication? DoD has given up on *eliminating* covert channels

Covert Channel Example

- Alice has **TOP SECRET** clearance, Bob has **CONFIDENTIAL** clearance
- Suppose the file space shared by all users
- **Alice** creates file **FileXYzW** to signal **"1"** to **Bob**, and removes file to signal **"0"**
- Once per minute Bob lists the files
  - If file FileXYzW does not exist, Alice sent 0
  - If file FileXYzW exists, Alice sent 1
- Alice can leak **TOP SECRET** info to Bob

Covert Channel Example



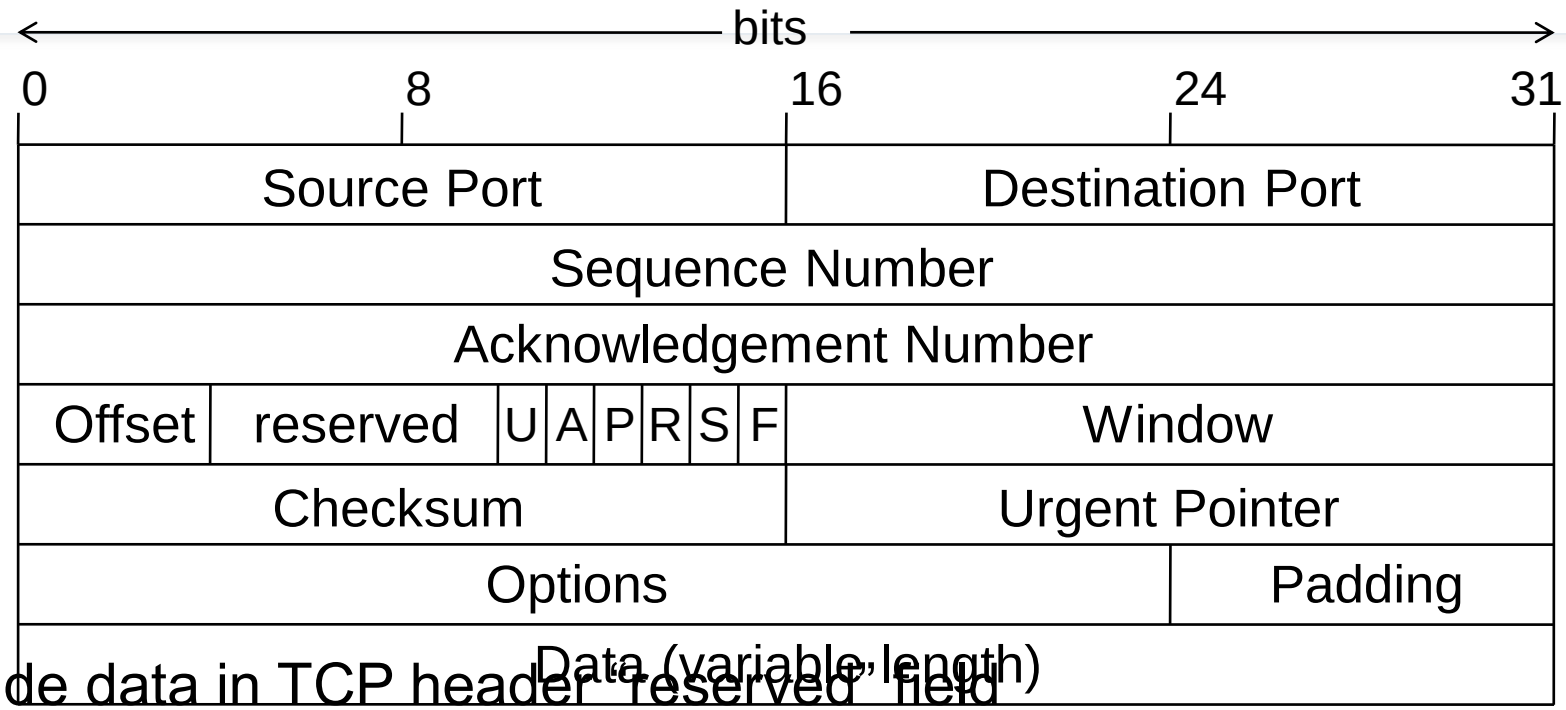
Covert Channel

- **Other possible covert channels?**
  - Print queue
  - ACK messages
  - Network traffic, etc.
- **When does covert channel exist?**
  1. Sender and receiver have a shared resource
  2. Sender able to vary some property of resource that receiver can observe
  3. "Communication" between sender and receiver can be synchronized

Covert Channel

- Consider 100MB **TOP SECRET** file
  - Plaintext stored in **TOP SECRET** location
  - Ciphertext encrypted with AES using 256-bit key stored in **UNCLASSIFIED** location
- Suppose we reduce covert channel capacity to 1 bit per second
- It would take more than 25 years to leak entire document thru a covert channel
- But it would take less than 5 minutes to leak 256-bit AES key thru covert channel!

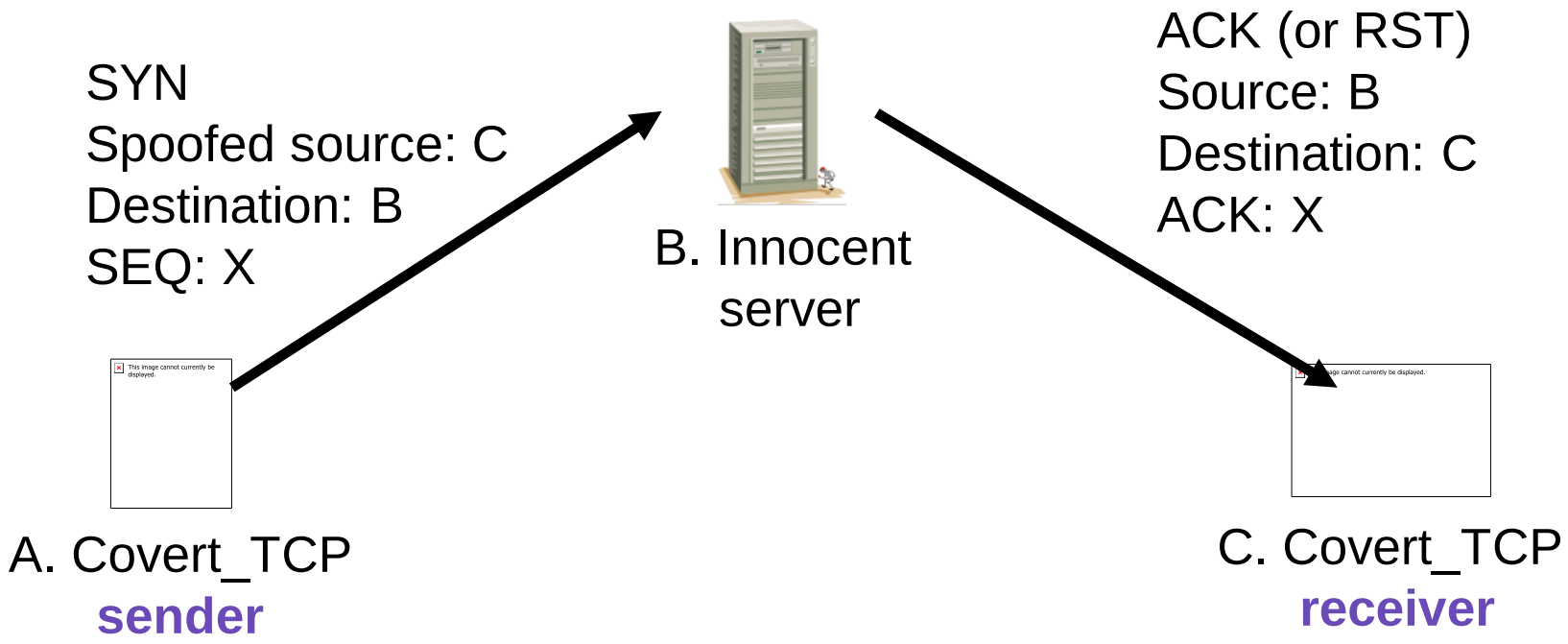
Real-World Covert Channel



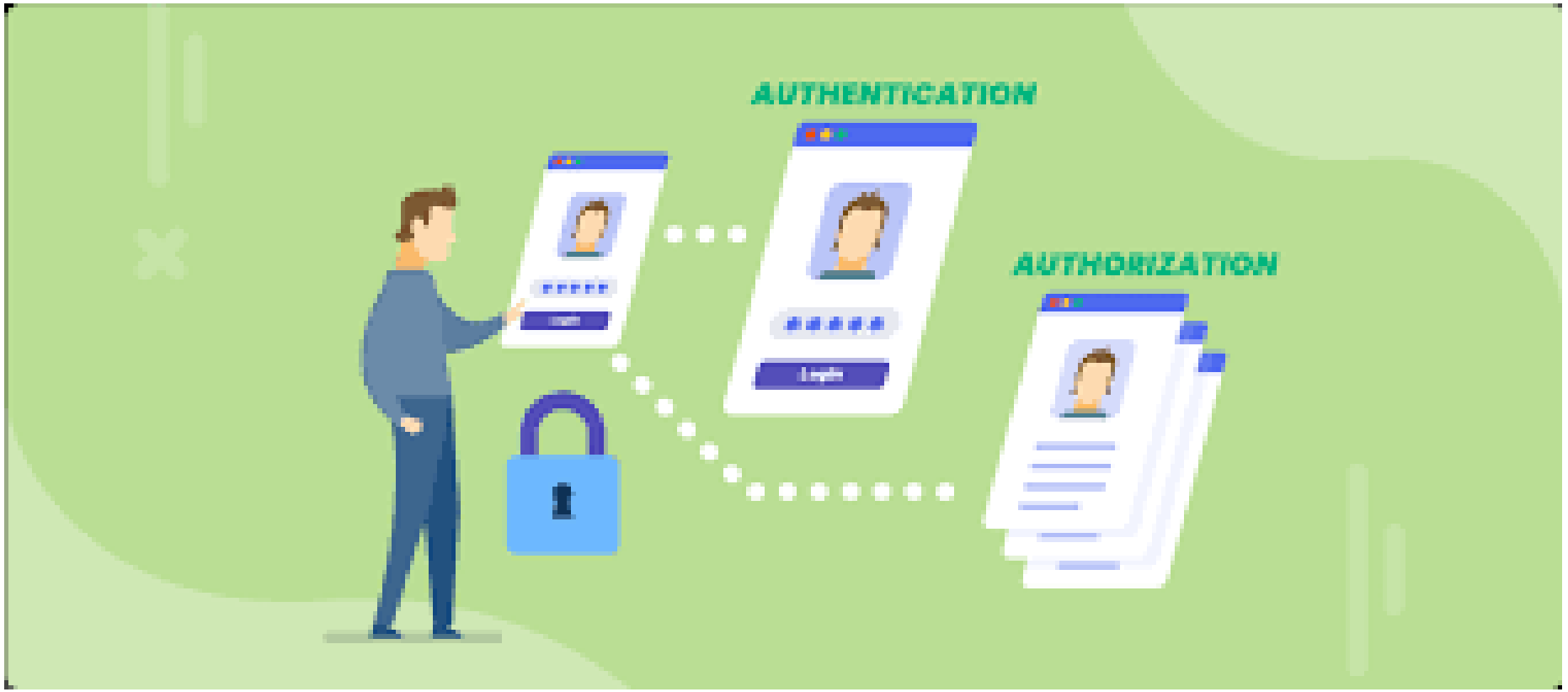
- Hide data in TCP header "reserved" field
- Or use covert_TCP, tool to hide data in
  - Sequence number
  - ACK number

Real-World Covert Channel

- Hide data in TCP sequence numbers
- Tool: covert_TCP
- Sequence number X contains covert info



AUTHENTICATION



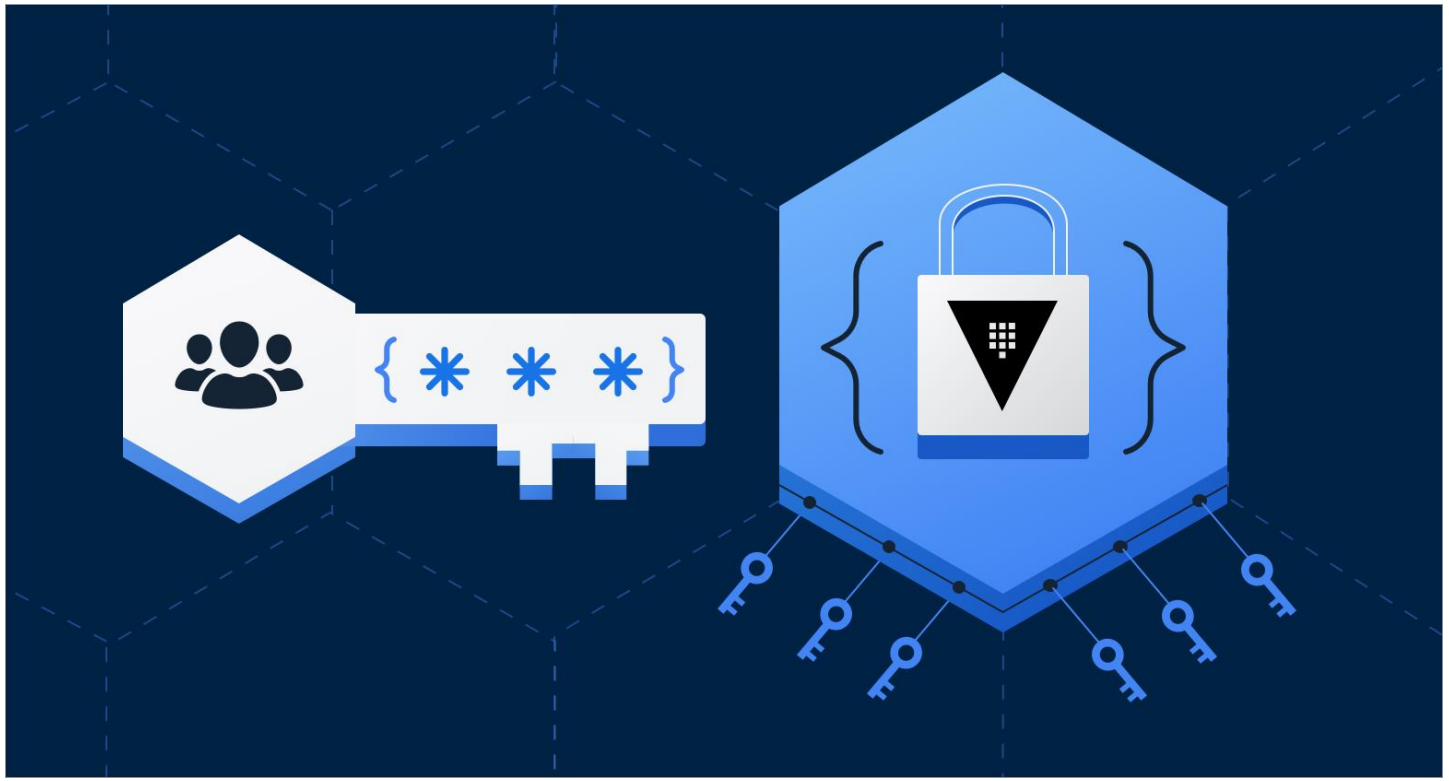
AUTHENTICATION PROTOCOLS

Designing authentication protocols and attack-based analysis

AUTHENTICATION PROTOCOLS

- As a network administrator, you need to log into your network devices. To do this, of course, you need a login ID and a password. And that raises the question of how you'll manage this account information. There are many options. A better alternative is to use an authentication protocol to allow the devices to get the account information from a central server.
- An authentication protocol is a type of cryptographic protocol specifically designed for transfer of authentication data between two entities(client and server)
- Eg: TACACS+, RADIUS, LDAP,KERBEROS

Key Management

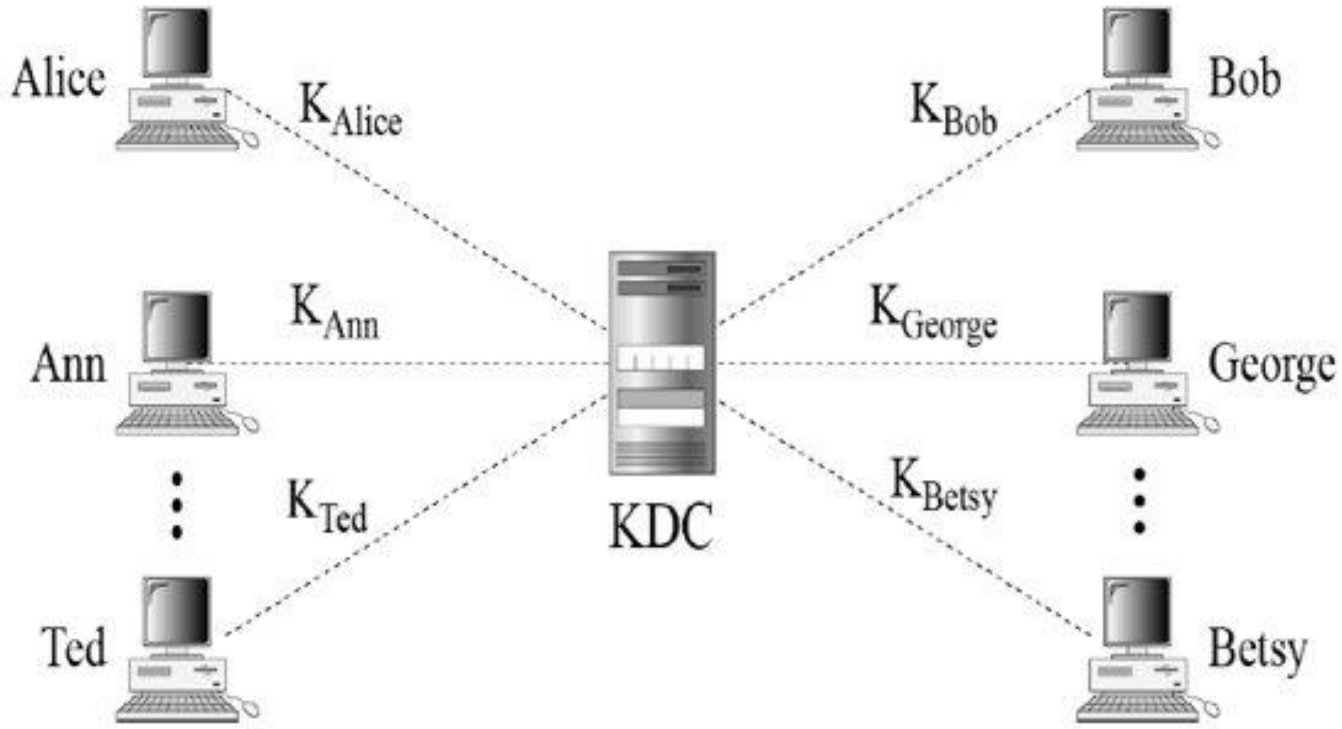


Key management

- Secret keys in symmetric key cryptography and public keys in Asymmetric key cryptography need to be distributed and maintained.
- **Symmetric Key Distribution:** *Symmetric-key cryptography is more efficient than asymmetric-key cryptography for enciphering large messages. Symmetric-key cryptography, however, needs a shared secret key between two parties. The distribution of keys is another problem.*
- **Distribution of symmetric keys done by:**
  - **Using a TTP***(Kerberos)*
  - **Without using TTP***(Diffie Hellman)*
  - **Using KDC***(Needham Schroder)*
  - **Using CA's**
  - **PKI**

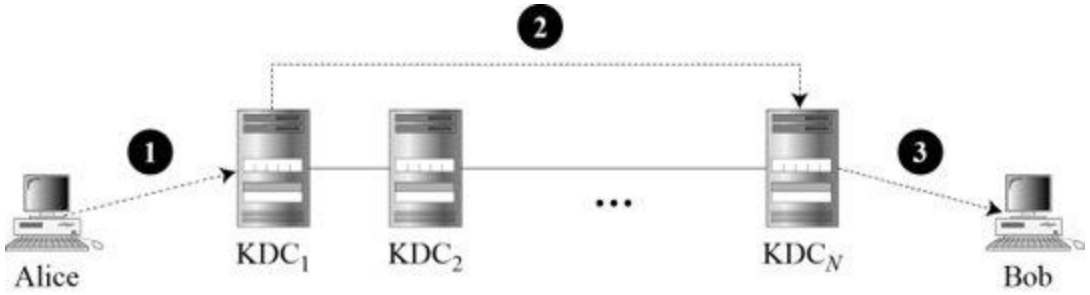
Key-Distribution Center: KDC

- A ***KDC creates a secret key for each member***. *This* ***secret key(K_{AS}, K_{BS})*** *can be used only between the member and the KDC, not between two members.*
- A **session symmetric key(K_{AB})** is established with KDC with Bob's(server) agreement between two parties is **used only once.**

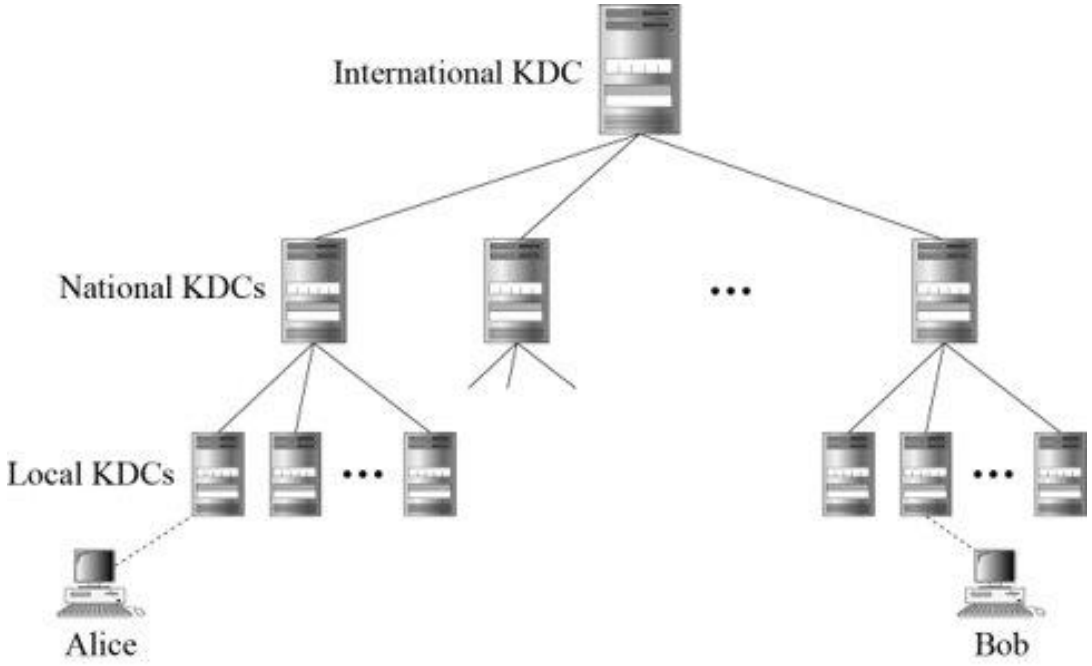


KDC Types

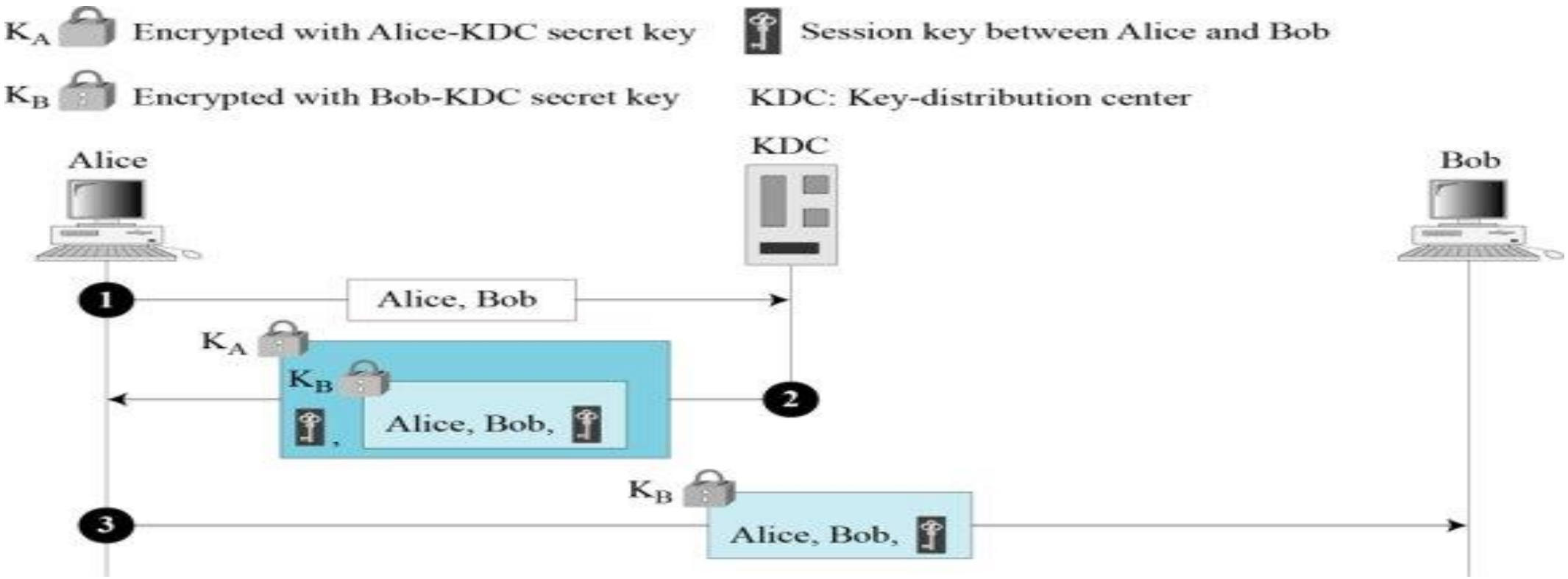
Flat Multiple KDCs



Hierarchical Multiple KDCs



A Simple protocol using KDC



Needham-Schroeder Protocol

- **Needham–Schroeder protocol** is one of the two key transport protocols intended for use over an insecure network, proposed by Roger Needham and Michael Schroeder
- This protocol is intended to provide **mutual authentication** between two parties communicating on a network, but in its proposed form is insecure.
- N-S is protocol **aims to establish a session key** between two parties on a network, typically to protect further communication.
- There are **two types** of Needham-Schroeder protocol.
  - **Needham-Schroeder protocol with symmetric key**
  - **Needham-Schroeder protocol with asymmetric key**
- The *Needham–Schroeder Symmetric Key Protocol*, based on a **symmetric encryption algorithm** forms the basis for the **Kerberos protocol.**
- Needham-Schroeder protocol allows to **prove the identity of the end users** communicating, and also **prevents a middle man from eavesdropping.**

Protocol Identities

- Alice (A) initiates the communication to Bob (B). K_{AB} is a server trusted by both parties. In the communication:
  - **A** and **B** are identities of Alice and Bob respectively
  - **K_{AS}** is a symmetric key known only to A and S
  - **K_{BS}** is a symmetric key known only to B and S
  - It is assumed that A and B already have secure symmetric communication with S using keys K_{AS} and K_{BS}
  - **N_A** and **N_B** are nonces generated by A and B respectively.
  - **K_{AB}** is a symmetric key, which will be the session key generated for the session between A and B

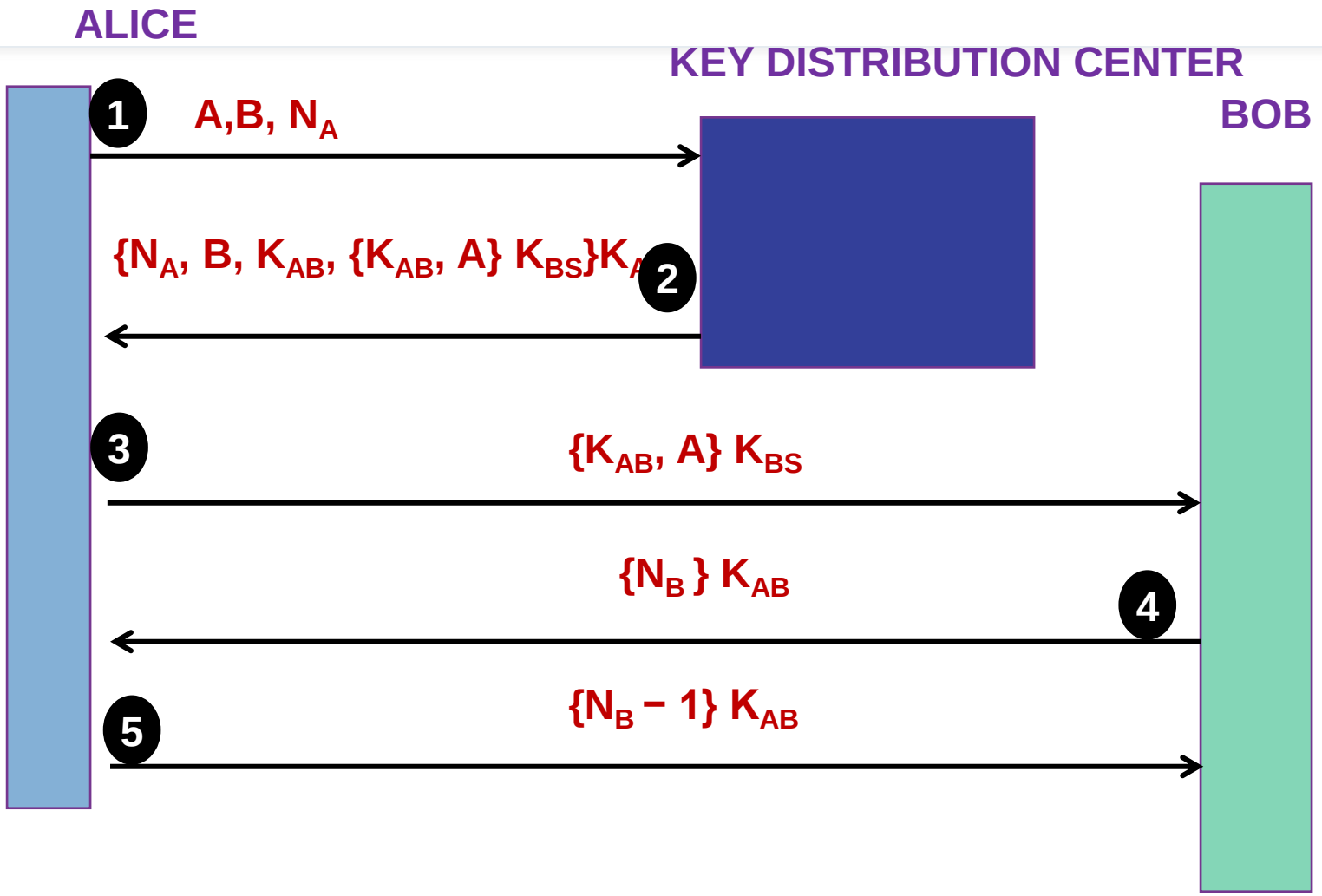
Nonces

- N-S protocol uses **nonces** (short for **“numbers used once”),** randomly generated values included in messages. This is used in encryption protocols to **prevent replay attack**.
  - For example if somebody captures a packet during the communication between me and a shopping website, he can resend the packet without decrypting it, and the server can accept the packet and do operations on it. To prevent this, nonce(the random value generated) is added to the data, so as the server can check if that nonce is valid, or expired.
- If a nonce is generated and sent by A in one step and returned by B in a later step, A knows that B’s message is fresh and not a replay from an earlier exchange.
- Note that a nonce is not a timestamp. The only assumption is that it has not been used in any earlier interchange, with high probability

NS PROTOCOL(with symmetric keys) STEPS

1. Alice sends a message to KDC that includes her identity, Bob's identity and her Nonce N_A.
   - $A \rightarrow KDC : A,B, N_A$
2. KDC generates K_{AB} the session key. The session key K_{AB} is appended with Alice's identity and encrypted with Bob's secret key K_{BS} which is known as **ticket to Bob**. The ticket along with session key K_{AB} , Bob's identity and Alice's nonce NA is encrypted with Alice's secrete key K_{AS} and sends to Alice. The nonce assures Alice that the message is fresh and that the server is replying to that particular message and the inclusion of Bob's name tells Alice who she is to share this key with.
   - $S \rightarrow A : \{N_A, B, K_{AB}, \{K_{AB}, A\} K_{BS}\}K_{AS}$
3. Alice forwards the Bob's ticket.
   - $A \rightarrow B : \{K_{AB}, A\} K_{BS}$
4. Bob can decrypt the ticket with the key K_{BS} he shares with KDC, thus authenticating the data. Bob sends Alice a nonce encrypted under K_{AB} to show or confirm that he has the symmetric key or session key provided by the middle man server K_{AB}.
   - $B \rightarrow A : \{N_B\} K_{AB}$
5. Alice performs a simple operation on the nonce provided by Bob, re-encrypts it and sends it back to Bob just to verify that she is still alive and that she holds the key.
   - $A \rightarrow B : \{N_B - 1\} K_{AB}$

Needham-Schroeder Protocol



KERBEROS

Motivation for KERBEROS

- Users wish to access services on servers.
- Three threats exist:
  1. User pretend to be another user.
  2. User alter the network address of a workstation.
  3. User eavesdrop on exchanges and use a replay attack.
- **Solution:** Authentication using public keys
  - N users⇒ N key pairs
- Authentication using symmetric keys
  - N users requires (on the order of) N^2 keys
- Symmetric key case **does not scale**
- Kerberos based on symmetric keys but only requires N keys for N users
  - Security depends on TTP (trusted third parties) & No PKI is needed

KERBEROS

- In Greek mythology, Kerberos is 3-headed dog with a snake for a tail that guards entrance to Hades
- Kerberos is an authentication protocol based on symmetric key crypto that provides mutual authentication
  - Originated at **MIT**
  - Based on **Needham and Schroeder protocol**
  - Relies on a **Trusted Third Party (TTP)**
  - uses **UDP port 88** by default.
  - Several versions of the protocol exist. Two versions in use: 4 & 5- **Version 4 makes use of DES**
  - Used by **Red Hat 7.2** and **Windows Server 2003** or higher
  - Kerberos has also become a standard for websites and **Single-Sign-On implementations** across platforms.
  - Kerberos Consortium maintains Kerberos as an open-source project.

KERBEROS Working Parties

- Kereberos has separated user verification from issuing of tickets. The main working parties are:
  - **Client Workstation(CWS):** Alice is the client workstation
  - **Authentication Server (AS):** Each user(client as well as server) register with AS and are granted with a user identity and a password. AS has a database with user identities and passwords. AS verifies user, issues a secret key to be used between Alice and TGS.
  - **Ticket-Granting Server (TGS):** TGS issues a ticket for real server Bob. It also provides session key KAB between Alice and Bob.
  - **Real Server:** The real server (Bob) provides services for the user (Alice).
- AS and TGS together constitutes KDC
- The security of the protocol relies heavily on participants maintaining loosely synchronized time and on short-lived assertions of authenticity called Kerberos tickets.
- In Kerberos KDC acts as the TTP. TTP is trusted, so it must not be compromised

KERBEROS Login Process

1. Alice enters her username which is received by the authentication server(AS) in plain text.
2. **AS** creates a package for user Alice which contains **a randomly generated session Keys(K_{A-TGSS} =SA(Mark stamp)), user id of Alice and expiration time** and the package is encrypted using **symmetric key(K_{AS-TGS} =KKDC) that AS & TGS shares**. The output is known as **Ticket Granting Ticket(TGT),** which can be opened only by TGS. **AS encrypts TGT and K_{A-TGS} using symmetric key (K_{A-AS}) derived from password of Alice.** TGT is issued when user logs in and acts as user credentials which is later on used to obtain tickets to access network resources. TGT ensures statelessness of Kerberos.

   Symmetric Key K_{A-AS} = h(Alice's password)

   TGT=(K_{A-TGS},"Alice",timestamp)K_{AS-TGS}

Alice does not know K_{A-AS} , but can be created from Alice's password and can extract session key(K_{A-TGSS}) and **TGT**. Client machine immediately destroys the password that Alice has entered from its memory to prevent its stealing by the attacker.

((TGT)K_{AS-TGS}, K_{A-TGS})K_{A-AS}

KERBEROS Login Process

3. Alice has successfully logged in and want to access Bob, the email server. To communicate with Bob Alice needs a ticket and sends TGT to TGS. Client machine creates a message for TGS, which includes following**: ID of server(BOB), Expiration time(curent time stamp) encrypted with session key** K_{A_TGS} which is known as **authenticator, TGT** which already it received. Timestamp **encrypted with session key** K_{A_TGS}/authenticator prevents replay attack by Eve.

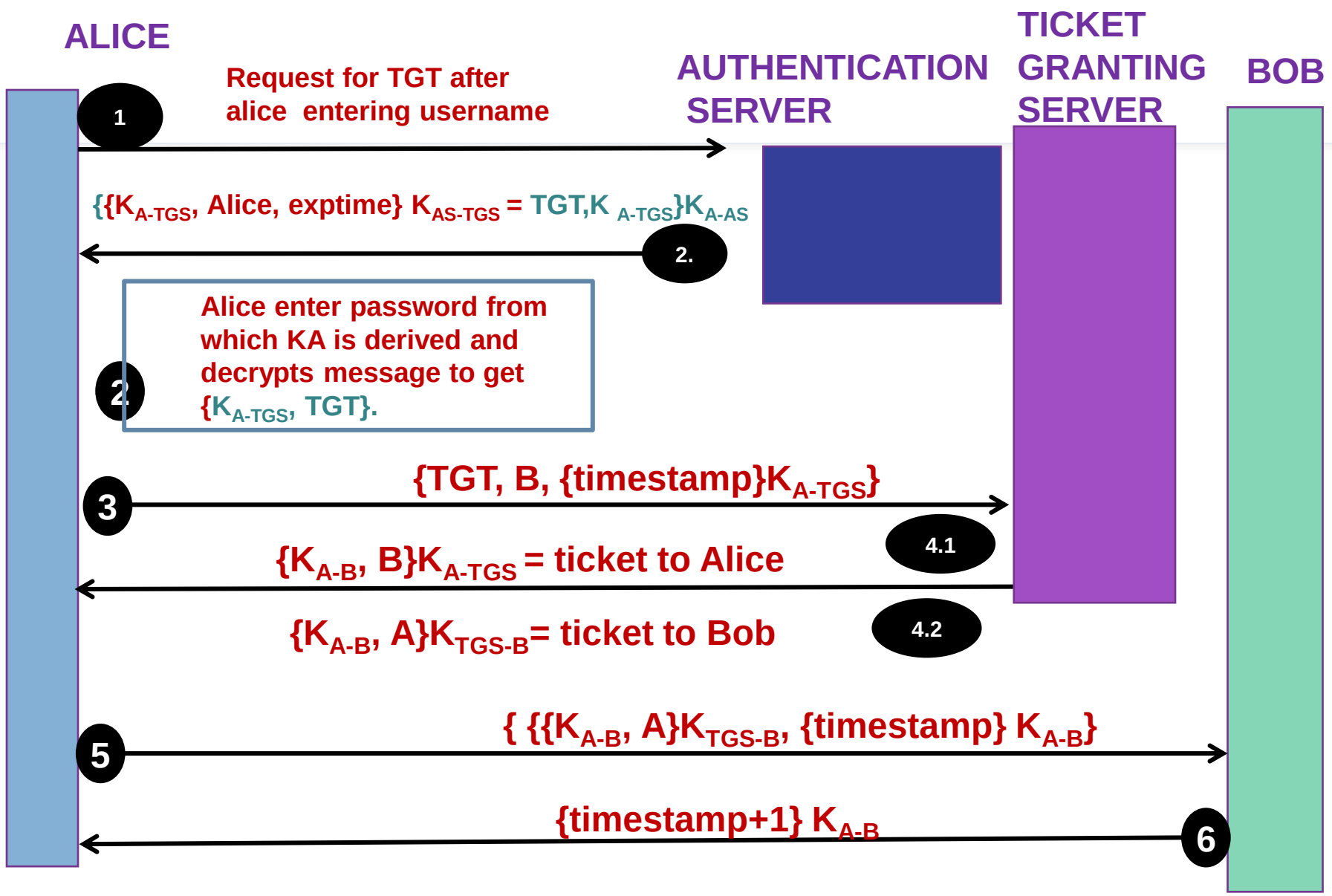
4. **TGS obtains** $K_{A\text{-}TGS}$ **from TGT**; since only TGS knows how to decrypt TGT using $K_{AS\text{-}TGS}$. TGS verifies TGT and timestamp. TGS creates a session key $K_{A\text{-}B}$ for Alice to have secure communication with BOB. TGS sends 2 tickets to Alice.

   1. $K_{A\text{-}B}$ combined with **Bob's id and encrypted with key** $K_{A\text{-}TGS}$ It is known as **ticket to Alice.**
   2. $K_{A\text{-}B}$ combined with **Alice's id and encrypted with BOB's secret Key(**$K_{TGS\text{-}B}$**).** It is known as **ticket to Bob.**

5. Alice forwards BOB's ticket $\{K_{A\text{-}B}, A\}K_{TGS\text{-}B}$ which she had received from TGS. To guard replay attacks Alice adds time stamp encrypted with $K_{A\text{-}B}$.

6. Bob generates KA-B and decrypts time stamp using KA-B. For acknowledgement BOB adds 1 to time stamp and encrypts the result with KA-B and send it to Alice.

KERBEROS PROTOCOL



Kerberos Security

ADVANTAGES

- Statelessness

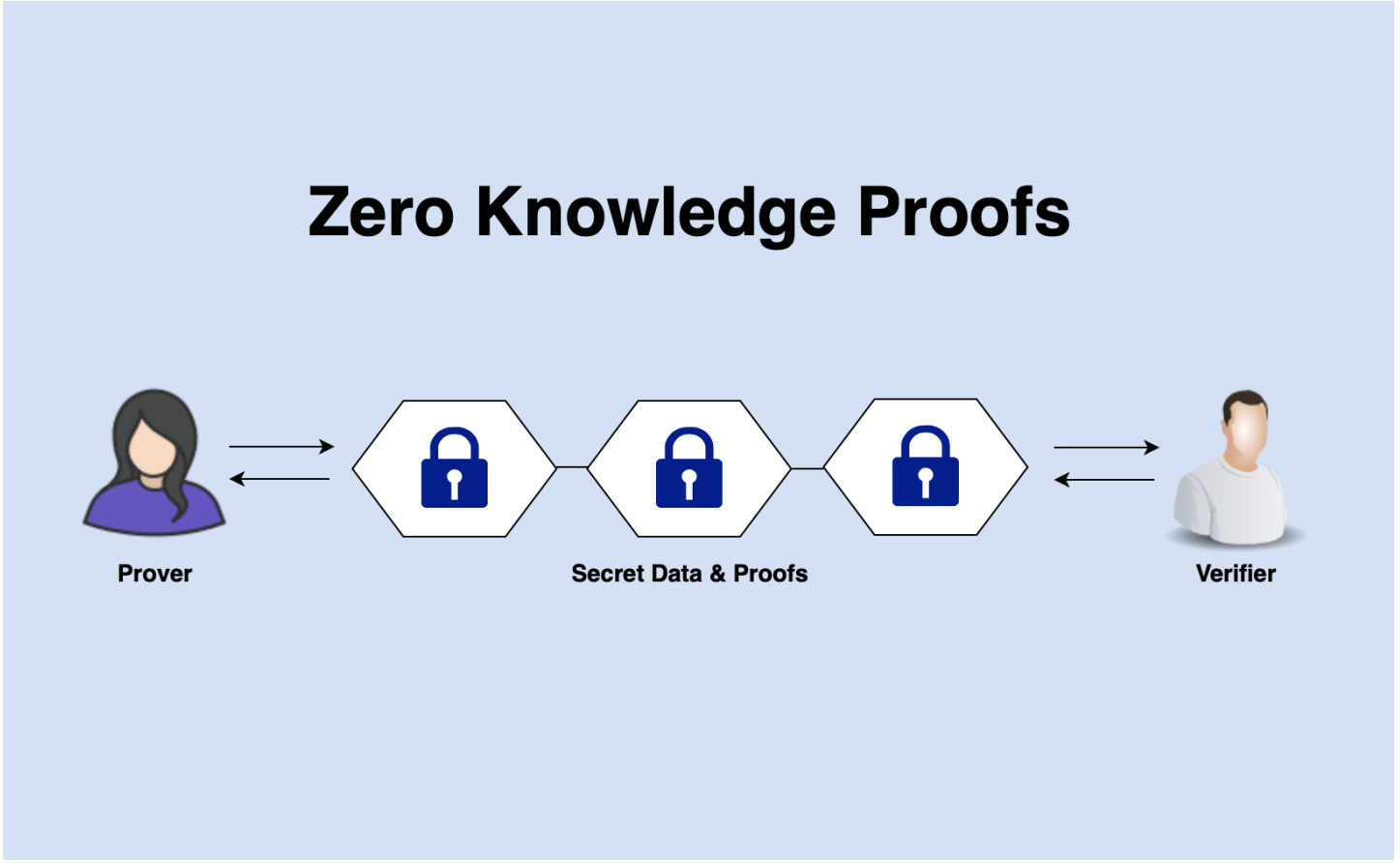
K_{AB} is sent from Alice to Bob. If it was done by KDC to Bob then Bob has to save key and maintain state for further communication.
Also $K_{AS\text{-}TGS}$ is generated by KDC(TTP) without intervention of Alice a nd Bob. So KDC remains stateless. Eliminates DOS attack.

- Anonymity of Alice: TGS doesn't need to know who is making request to issue session key and to decrypt the message as it verifies key $K_{AS\text{-}TGS}$
- Replay Attack preventin
- Less symmetric keys
- Authentication assured

DISADVANTAGES

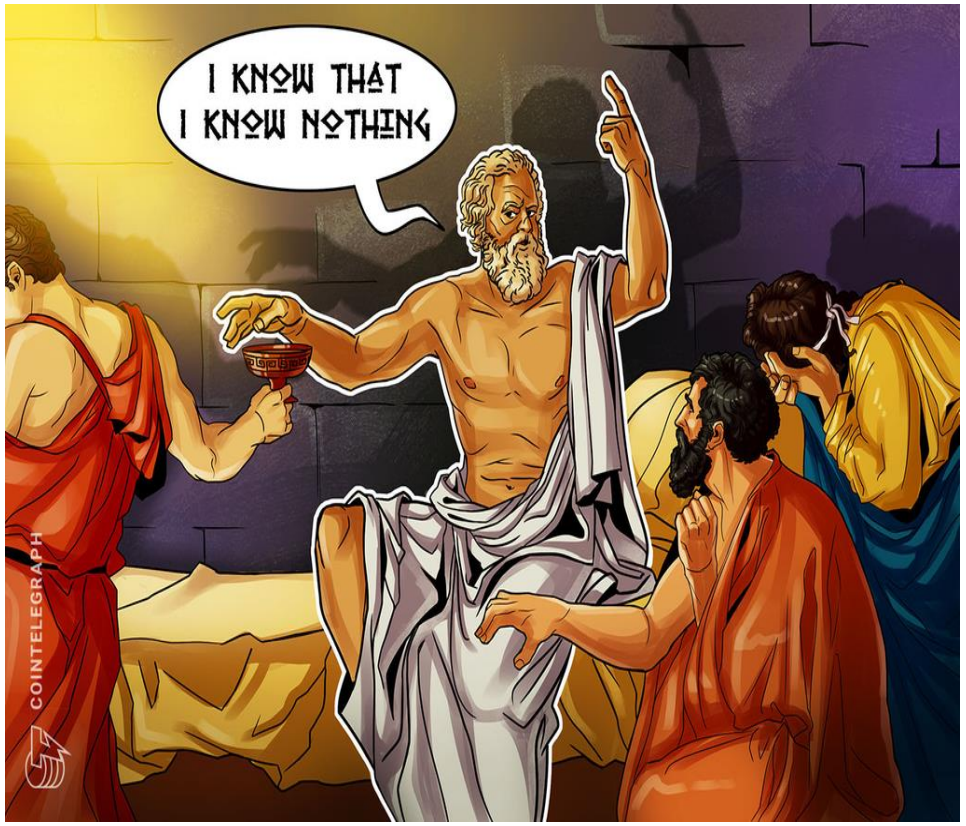
- Entire security depends on security of KDC/TTP. If its compromised entire security is compromised
- Kerberos uses timestamp for mutual authentication and replay attack prevention, the drawback is time becomes a critical security parameter. All clocks need to be synchronized and clock skew must be tolerated(typically 5 minutes)
- The administration protocol is not standardized and differs between server implementations.
- Single point of failure: It requires continuous availability of a central server. When the Kerberos server is down, no one can log in. This can be mitigated by using multiple Kerberos servers and fallback authentication mechanisms.

ZERO KNOWLEDGE PROOFS



ZERO KNOWLEDGE PROOFS

- An **authentication scheme** developed by **Fiege, Fiat** and **Shamir.** Usually run in several rounds and have high costs.
- In ZKP Alice wants to prove Bob that she knows a secret without revealing any information about secret to anyone, even to Bob.Bob must be able to verify that Alice knows secret, even though he gains no information on secret.
- *A zero-knowledge proof is a digital protocol that allows for data to be shared between two parties without the use of a password or any other information associated with the transaction.*



Relevance of ZKP in Authentication

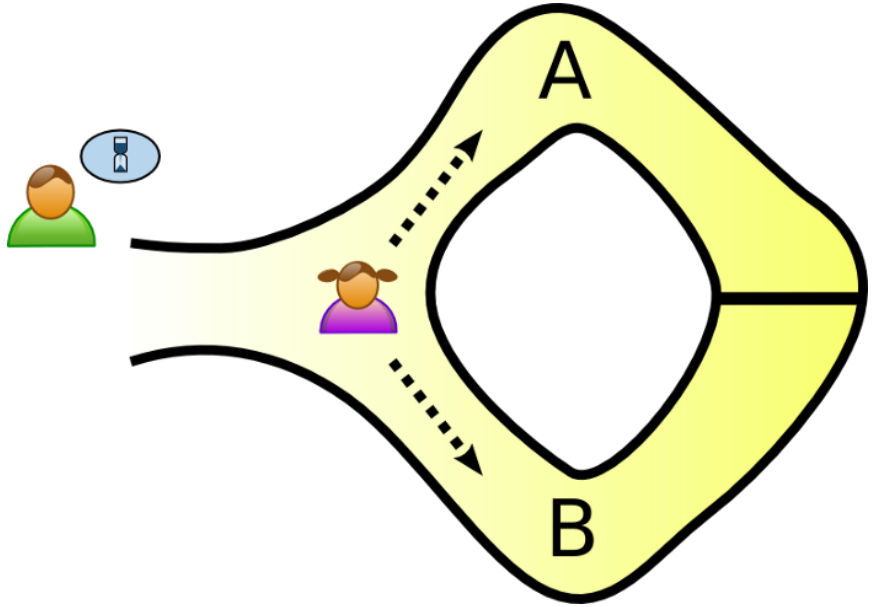
- In password authentication claimant needs to send her password to verifier. The isssues associated are
  - Evesdropping by Trudy
  - Dishonest verifier can reveal secret.
  - Use thepassowrd to impersonate user
- In zeroknowledge authentication, claimant doesnt reveal anything that endanger confidentiality of secret.
- From the interaction between claimant and verifier,the verifier knows claimant have secret or not.

Usecases

- *Zero-knowledge proofs offer a lot of benefits to blockchain systems help in making crypto transaction's extremely secure.* cryptocurrency transactions can be done without disclosing any data related to it — such as where the transactions originated from, where it went or how much money was transferred. **Eg:** Zcash, Quorum(a native blockchain ecosystem by JP Morgan Chase). ZoKrates using Solidity.
- Governments can also use ZKPs to determine the nuclear capabilities of various militaries without having to spy on or inspect their inventories. Eg: Defense Advanced Research Projects Agency, or DARPA, released a statement in which it claimed that it was working on a new project called SIEVE — i.e., Securing Information for Encrypted Verification and Evaluation — that makes use of ZKPs to determine the origin of highly secure data without the U.S. government having to reveal the way in which it was acquired.
- digital identification mechanisms, a secure alternative to the fog of birth certificate photocopies and smartphone photos of passports. Those ID schemes could also allow people to prove that they meet a minimum age requirement without sharing their date of birth, or that they have a valid driver's license without handing over their number.

Alibaba Cave

- There is a cave which has two entries inside called A and B. Both entries are connected inside the cave by a door that can open if correct secret spell words known("open "sesame"). Peggy says Victors that she knows the secret spell to open the door. Victor wants to know whether Peggy knows the secret word; but Peggy, being a very private person, does not want to reveal her knowledge and also does not like to reveal which path she entered. How Peggy supposed to prove that she knows the secret spell without telling Victor the spell and without telling from which entrance she entered.

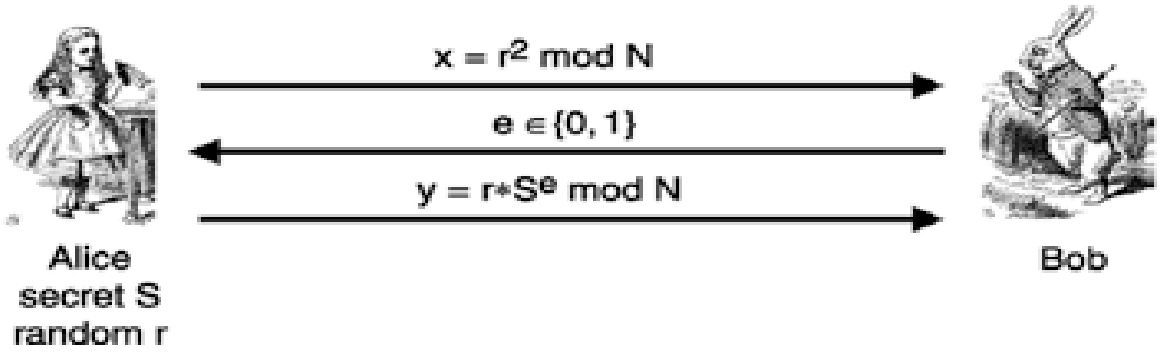


Alibaba Cave

- First, Victor waits outside the cave as Peggy goes in. Peggy takes either path A or B; Victor is not allowed to see which path she takes.
- Then Victor enters the cave and chooses an exit path for peggy. If she really does know the magic word, this is easy: she opens the door, if necessary, and returns along the desired path.
- suppose she did not know the word. Then, she would only be able to return by the named path if Victor were to give the name of the same path by which she had entered. Since Victor would choose A or B at random, she would have a 50% chance of guessing correctly. If they were to repeat this trick many times, say 20 times in a row, her chance of successfully anticipating all of Victor's requests would become vanishingly small (about one in a million).Probability that Peggy can trick Victor will be $(1/2^n)$

Achieving Cave Effect-Fiat Shamir Protocol

- To set up scheme and prove its secure on RSA, the trusted third paty or verifier or Bob choses **two very large prime numbers p** and **q** and calculate **N**, as **N=p*q**, where N is known to public and p and q are secret.
- Alice the claimant choses a secret **S** such that **S** is coprime to **N** and value of **S** is **1<=S<=N-1.** Alice computes **$v= S^2 \bmod N$**. Alice keeps **S** as her private key and **v** as her public key with TTP/Bob. Alice must convince Bob that she knows **S** without revealing any information on**S**. It can be done in 5 steps.

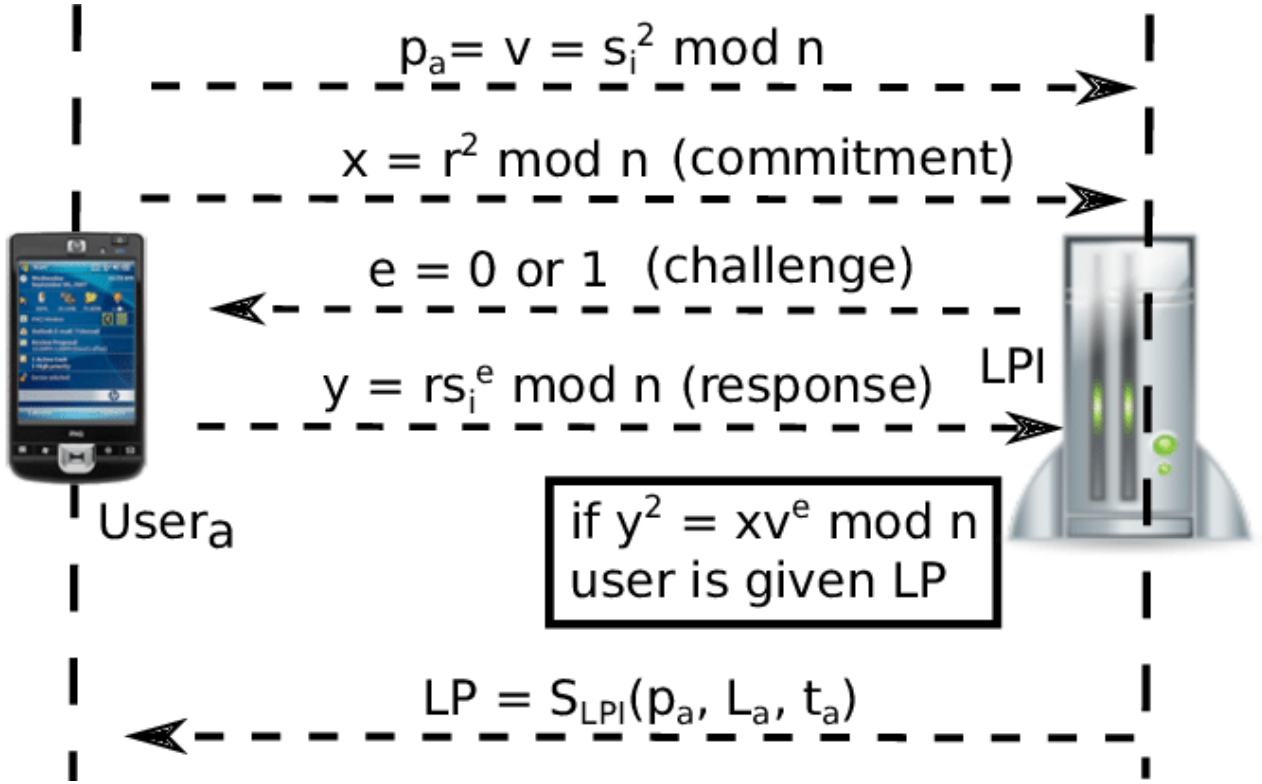


Fiat Shamir Protocol

1. Alice selcts a random number **r** between **0** and **N-1** and sends **$x = r^2 \bmod N$** to Bob. Here **x** is called as witness.
2. Alice sends **x** to Bob as witness.
3. Bob sends the challenge **e** to Alice**,** which is either **0** or **1**.
4. Alice calculates **$y = r * S^e \bmod N$** to Bob, where **r** is the random number selected by Alice in step 1 and**S** is her private key. Alice send **y** to Bob to show that she knows her private value **S.**
5. Bob calculates **y^2** and **$x * v^e \bmod N$**. If the 2 values are congruent, then Alice knows secret key **S**or she has calculated value of **y** in some other ways.

$y^2 = (r * S^e)^2 = r^2 s^{2e} = r^2(s^2)^e = xv^e \bmod N$

Fiat Shamir Protocol



Verification of Fiat Shamir Protocol

- **Verification** is repeated several times with value of **e** equal to 0 or 1.
- Alice must pass the test for every round even if she fails in one round she is not authenticated.
- If alice is dishonest 2 cases can happen:
  - Assume Bob has selected e as 1, then Alice must know secret S.
  - If **e=1** then Alice responds with **$y=r*S \bmod N$** in msg 4 and **$y^2= r^{2*}.s^2=x.v \bmod n$**. So **Alice must know secret S**
  - If **e=0** then Alice responds with **$y=r*S^0 \bmod N=r \bmod N$** in msg 4 and **$y^2= r^{2*}=x \bmod n$.** So **Alice must not know secret S**.
- If Trudy expects Bob to send e = 1, she can send $x = r^{2*}v^{-1}$ in msg 1 and $y = r$ in msg 3. If Bob chooses $e \in \{0,1\}$ at random, Trudy can only fool Bob with probability ½.

Fiat Shamir Facts

- Trudy can fool Bob with prob 1/2, but after n iterations, the probability that Trudy can fool Bob is only $(½)^n$
- Just like Bob's cave Bob's e $\in$ {0,1} must be unpredictable
- Alice must use new r for each iteration or else
  - If e = 0, Alice sends r in message 4
  - If e = 1, Alice sends r*S in message 4
  - Anyone can find S given **both** r and r*S
- **Security Benefits**
- It allows authentication with anonymity. Both Alice and Bob knows public Key V but there is nothing in V that

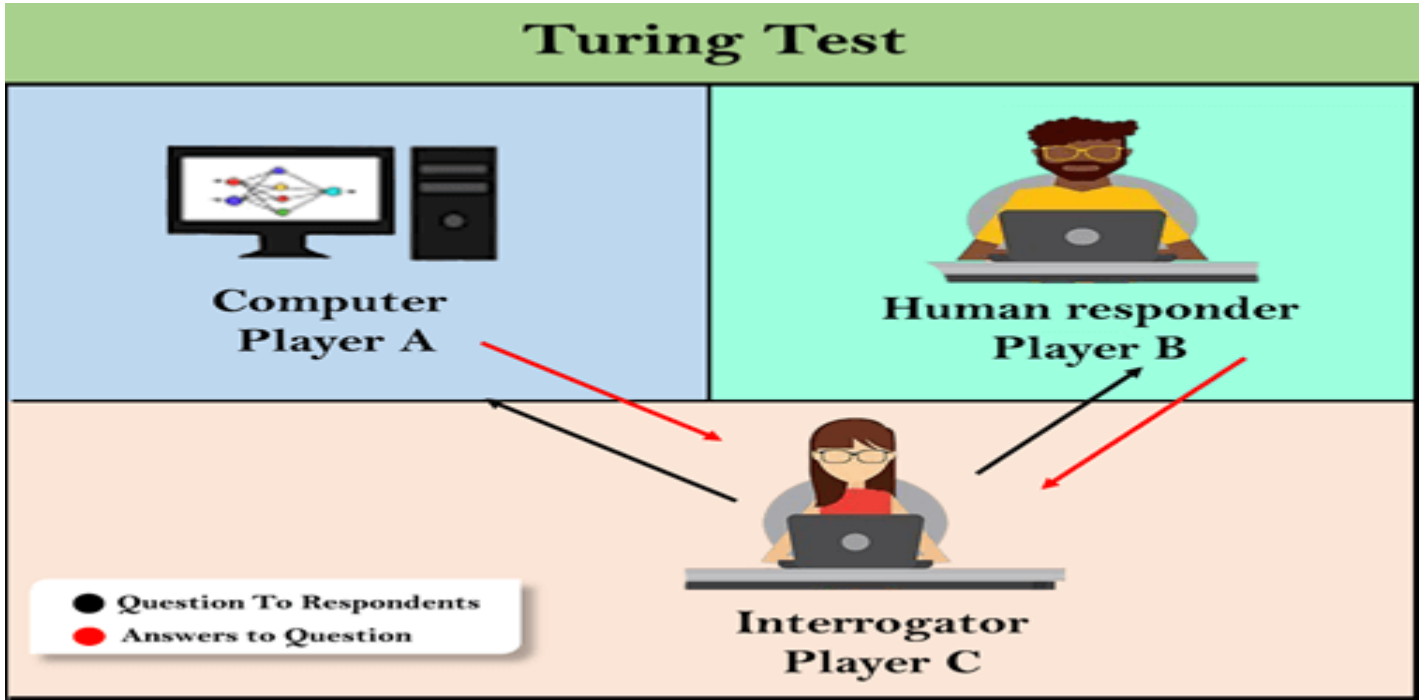
Fiat-Shamir Zero Knowledge?

- **Is Fiat-shamir protocol really "zero knowledge"? Can Bob Learn anything about Alice's secret S?**
- Zero knowledge means that Bob learns **nothing** about the secret
- V= S^2 Mod N is public. Bob sees **r^2mod N** in message 1 and assuming **e=1** Bob sees **r * S mod N** in message 4. If Bob can find r from **r^2Mod N** then he can find **S** . Here security relies on finding a modulus square root which is computationally infeasible.
- **ZKP in the Real World? ZKP is not just fun and games for mathematicians!**
- If Public key certificates are used to identify users then No anonymity .
- ZKP is supported in Microsoft's Next Generation Secure Computing Base (NGSCB) used to authenticate software "without revealing machine identifying data"

CAPTCHA

ACTING HUMANLY: TURING TEST

- Turing Test is the **gold standard** in AI
- No computer can pass this today
  - But some claim they are close to passing



Alan Turing

- If interrogator cannot reliably distinguish human from computer, then computer possess intelligence.

CAPTCHA

- CAPTCHA (Completely Automated Public Turing test to tell Computers and Humans Apart). Inverse Turing Test
- CAPTCHA is a Completely Automated test which is generated and scored by a computer that a human can pass, but a machine cannot pass(even if it has access to source code to generate test) with probability better than guessing.
- CAPTCHA **is a program that can generate a test and grade it, but itself cannot pass.**(similar to some professors)
- CAPTCHA **is an access control mechanism to restrict access to resources to humansie,** Only humans get access (not bots/computers)

CAPTCHA

- Computing problems that must be solved to break CAPTCHA(eg: Automatic Recognition, Distorted Text, Distorted Audio) are considered difficult problems in AI.
- The requirements for a CAPTCHA include
  - It must be **easy** for most **humans** to pass.
  - It must be **difficult** or **impossible** for a machines to pass, (even if the machine has access to the CAPTCHA software)
  - **Randomness** in generating CAPTCHA..
  - Have **different types of CAPTCHA** depending on individual requirement

FEATURES TO BE POSSESSED BY CAPTCHA

1. **Accessibility.** CAPTCHAs must be accessible. CAPTCHAs based solely on reading text — or other visual-perception tasks — prevent visually impaired users from accessing the protected resource. Such CAPTCHAs may make a site incompatible with Section 508 in the United States. Any implementation of a CAPTCHA should allow blind users to get around the barrier, for example, by permitting users to opt for an audio or sound CAPTCHA.
2. **Image Security.** CAPTCHA images of text should be distorted randomly before being presented to the user. Many implementations of CAPTCHAs use undistorted text, or text with only minor distortions. These implementations are vulnerable to simple automated attacks.
3. **Script Security.** Building a secure CAPTCHA code is not easy. In addition to making the images unreadable by computers, the system should ensure that there are no easy ways around it at the script level. Common examples of insecurities in this respect include: (1) Systems that pass the answer to the CAPTCHA in plain text as part of the web form. (2) Systems where a solution to the same CAPTCHA can be used multiple times (this makes the CAPTCHA vulnerable to so-called "replay attacks"). Most CAPTCHA scripts found freely on the Web are vulnerable to these types of attacks.
4. **Security Even After Wide-Spread Adoption.** There are various "CAPTCHAs" that would be insecure if a significant number of sites started using them. An example of such a puzzle is asking text-based questions, such as a mathematical question ("what is 1+1"). Since a parser could easily be written that would allow bots to bypass this test, such "CAPTCHAs" rely on the fact that few sites use them, and thus that a bot author has no incentive to program their bot to solve that challenge. True CAPTCHAs should be secure even after a significant number of websites adopt them.

APPLICATIONS OF CAPTCHA

1. **Preventing Comment Spam in Blogs.** Most bloggers are familiar with programs that submit bogus comments, usually for the purpose of raising search engine ranks of some website. This is called comment spam. By using a CAPTCHA, only humans can enter comments on a blog. There is no need to make users sign up before they enter a comment, and no legitimate comments are ever lost!
2. **Protecting Website Registration.** Several companies (Yahoo!, Microsoft, etc.) offer free email services. Up until a few years ago, most of these services suffered from a specific type of attack: "bots" that would sign up for thousands of email accounts every minute. The solution to this problem was to use CAPTCHAs to ensure that only humans obtain free accounts. In general, free services should be protected with a CAPTCHA in order to prevent abuse by automated scripts.
3. **Protecting Email Addresses From Scrapers.** Spammers crawl the Web in search of email addresses posted in clear text. CAPTCHAs provide an effective mechanism to hide your email address from Web scrapers. The idea is to require users to solve a CAPTCHA before showing your email address. A free and secure implementation that uses CAPTCHAs to obfuscate an email address can be found at reCAPTCHA MailHide.
4. **Search Engine Bots.** It is sometimes desirable to keep webpages unindexed to prevent others from finding them easily. There is an html tag to prevent search engine bots from reading web pages. The tag, however, doesn't guarantee that bots won't read a web page; it only serves to say "no bots, please." Search engine bots, since they usually belong to large companies, respect web pages that don't want to allow them in. However, in order to truly guarantee that bots won't enter a web site, CAPTCHAs are needed.

APPLICATIONS OF CAPTCHA

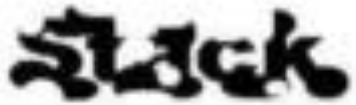
1. **Online Polls.** In November 1999, *http://www.slashdot.org* released an online poll asking which was the best graduate school in computer science (a dangerous question to ask over the web!). As is the case with most online polls, IP addresses of voters were recorded in order to prevent single users from voting more than once. However, students at Carnegie Mellon found a way to stuff the ballots using programs that voted for CMU thousands of times. CMU's score started growing rapidly. The next day, students at MIT wrote their own program and the poll became a contest between voting "bots." MIT finished with 21,156 votes, Carnegie Mellon with 21,032 and every other school with less than 1,000. Can the result of any online poll be trusted? Not unless the poll ensures that only humans can vote.
2. **Preventing Dictionary Attacks.** CAPTCHAs can also be used to prevent dictionary attacks in password systems. The idea is simple: prevent a computer from being able to iterate through the entire space of passwords by requiring it to solve a CAPTCHA after a certain number of unsuccessful logins. This is better than the classic approach of locking an account after a sequence of unsuccessful logins, since doing so allows an attacker to lock accounts at will.
3. **Worms and Spam.** CAPTCHAs also offer a plausible solution against email worms and spam: "I will only accept an email if I know there is a human behind the other computer." A few companies are already marketing this idea.

CAPTCHA TYPES

- **TEXT BASED**
  - Gimpy, ez-Gimpy
  - Gimpy-R
  - Simard’s HIP
- **GRAPHIC BASED**
  - Bongo
  - Pix
  - 3D
- **AUDIO BASED**
- **Video based**

TEXT BASED CAPTCHA's

(a) reCAPTCHA

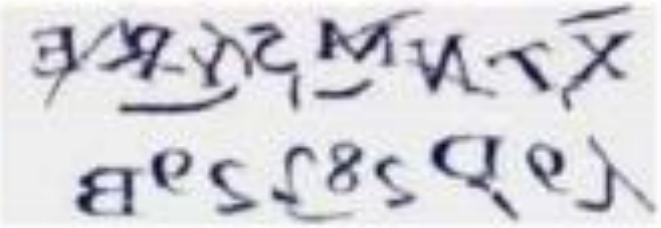


(b) buffle Text

(c) Gimpy

(a) E-Z gimpy

(b) Drag&Drop



(c) MSN

Gimpy

- Designed by Yahoo and CMU (Carnegie Mellon University)
- **Gimpy** works by choosing a certain number of words from a dictionary, and then displaying them corrupted, distorted and overlapped manner. **Gimpy** then asks the user enter a subset of the words in the image.
- Gimpy is based on the human has the ability to read extremely distorted text otherwhile the computer programs don‘t have such type of ability to do the same.

ez-gimpy

- Modified version of Gimpy. Randomly picks a single word from a dictionary and applies distortion to the text. The user is then asked to identify the text correctly.
- Used by Yahoo in Messenger
- Not a good implementation, already broken by OCRs

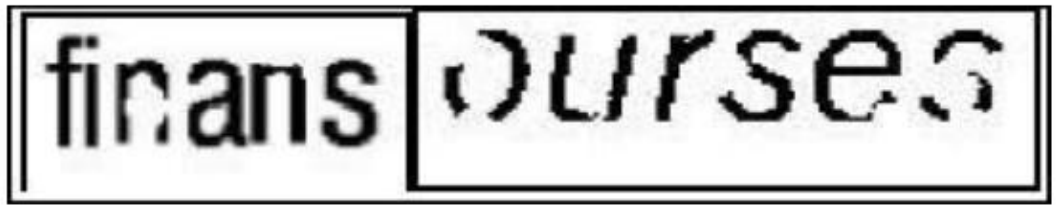
Gimpy-r

- Google CAPTCHA- Instead of complete word individual letters are noised.
- Pick random letters , Distort them , add noise and background

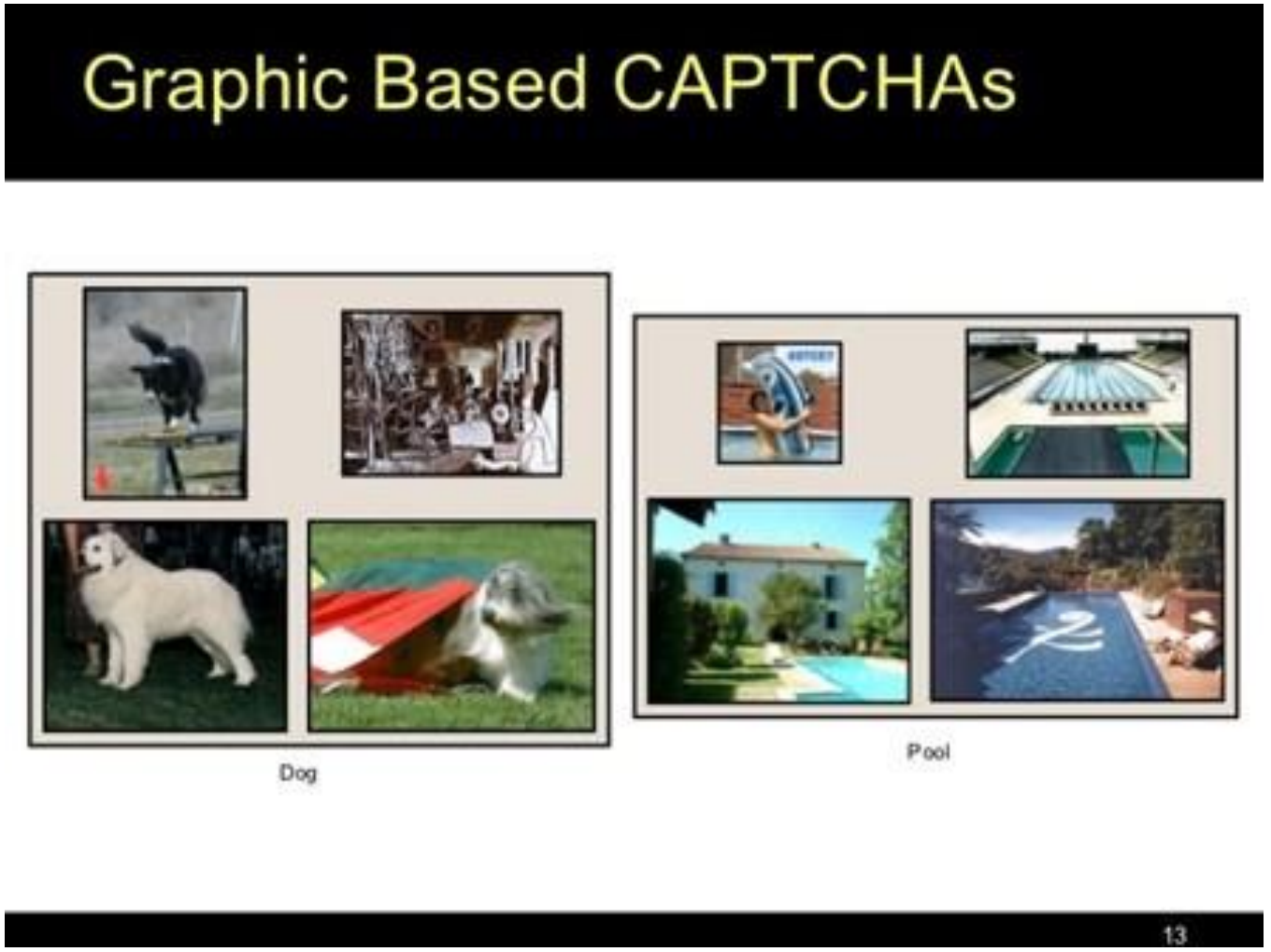
- **Simcard's HIP (Human Interaction Proofs) (MSN)-**
- Distort using arcs
  - Pick random letters and numbers
  - Distort them and add arcs

BAFFLE TEXT

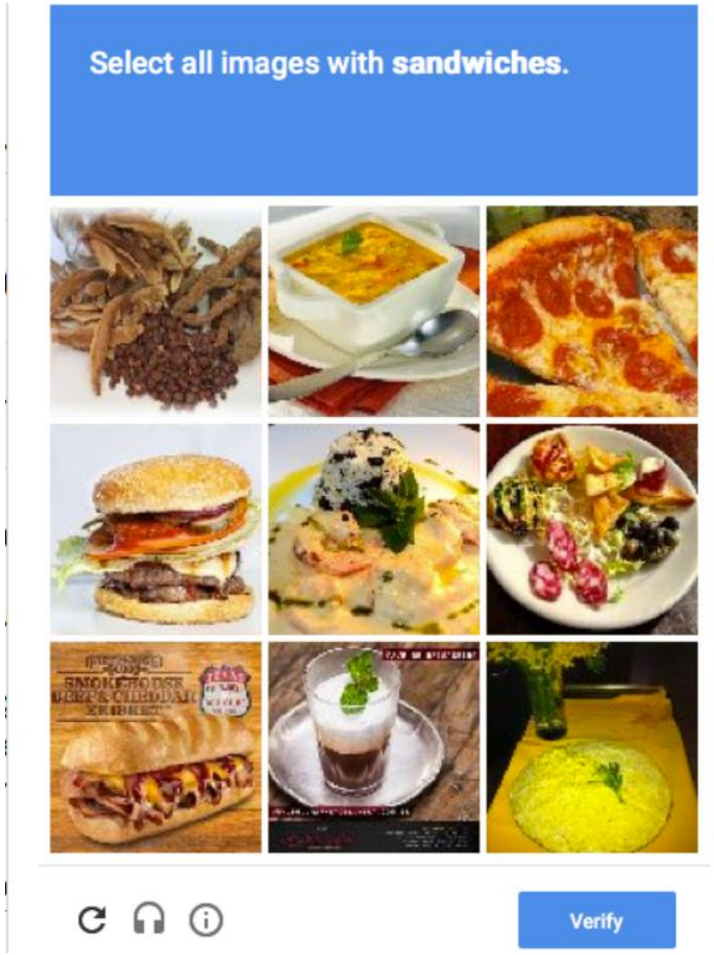
- This doesn’t contain dictionary words, but it picks up random alphabets to create a nonsense but pronounceable text. Distortions are then added to this text and the user is challenged to guess the right word.
- This technique overcomes the drawback of Gimpy CAPTCHA because, Gimpy uses dictionary words and hence, clever bots could be designed to check the dictionary for the matching word by brute-force.



GRAPHIC BASED CAPTCHAS



Pix



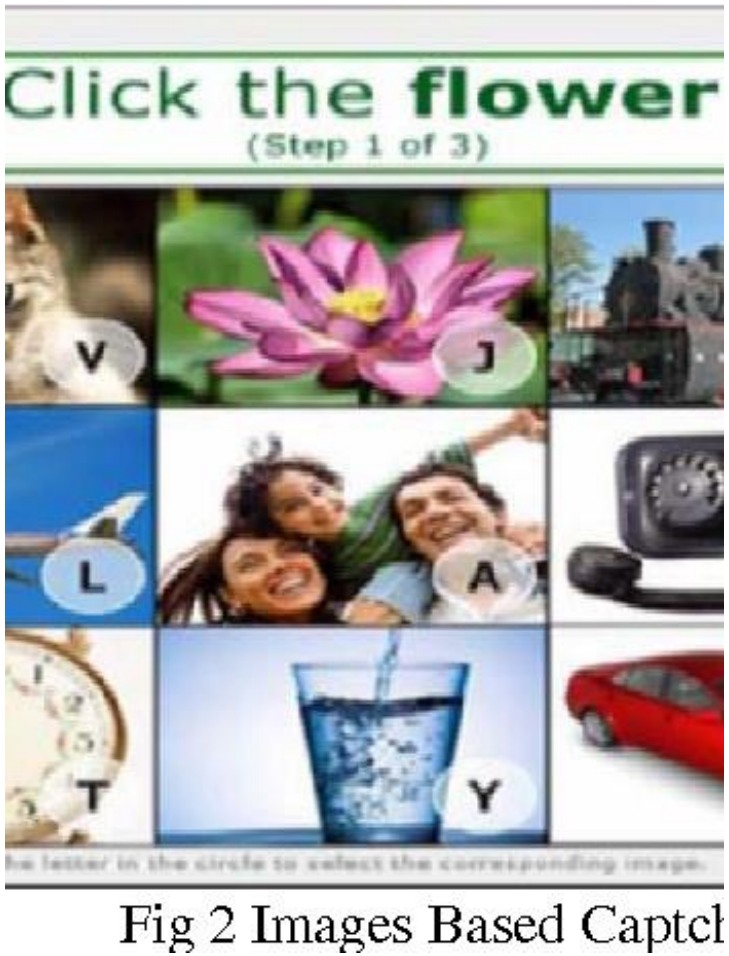
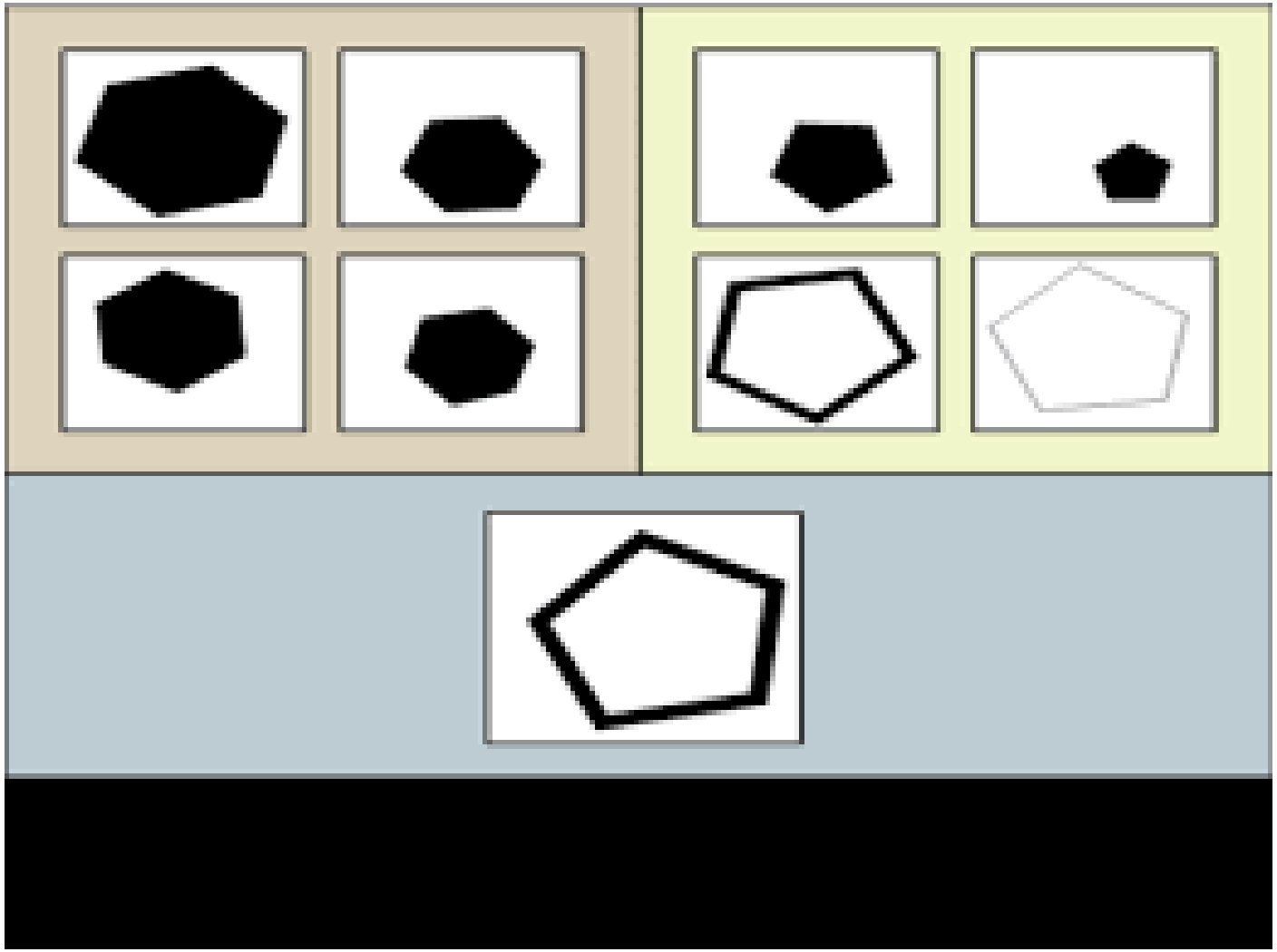


Fig 2 Images Based Captcl

- Uses a large database of labeled images
- Shows a set of images, user has to recognize the common feature among those

Bongo

- Display 2 series of blocks
- User must find characteristics that sets 2 apart
- User is asked to which series given block belongs to



Audio Based

- Consists of downloadable audio clip. picks a word or a sequence of numbers at random, renders the word or the numbers into a sound clip and distorts the sound clip; it then presents the distorted sound clip to the user and asks users to enter its contents
- User listens and enters the spoken word
- Helps visually disabled users

AUTHENTICATION METHODS

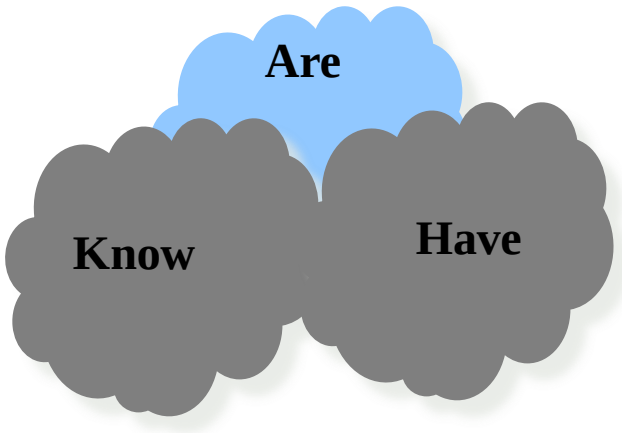
User Authentication Methods

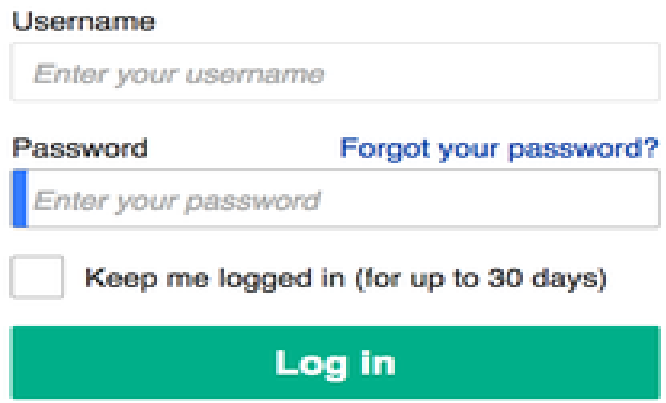
- Using a method to validate users who attempt to access a computer system or resources, to ensure they are authorized.
- Authenticating a Human to a machine to convince someone or something claiming to be.
- Types of user authentication
  - Something you know

    E.g: user account names and passwords
  - Something you have

    Eg: Smart cards or other security tokens
  - Something you are

    Eg: Biometrics





PASSWORD AUTHENTICATION

- ***An ideal password is something user knows so that a machine can verify that user know and attacker doesn't know or can't guess even with the access to unlimited computing resources.***

Eg: PIN for an ATM, password of web applications, Social security number, Mother's maiden name, Date of birth, Name of your pet

Why Passwords are more preferred or popular?

- **Cost:** passwords are free compared to smart cards and biometric devices
- **Convenience:** easier for sysadmin to reset a compromised password than to issue and configure a new smart card.

Password Variants

- **Cognitive Passwords:** created through several experience-based questions. Fact or opinion-based questions

Eg: favorite color, college name, birthplace

- **One-time passwords:** used in sensitive cases

Eg: Token devices, prepaid cards

- **Passphrases:** sequence of characters ie no longer than password. User enters phrase into application which transforms value into password.

Eg: FSa7Yago(four score and seven years ago)

- **Personal identification number (PIN):** an arbitrary string of characters where the permissible characters are constrained to be numeric

Eg:347865

Cryptographic keys

- If cryptographic key is 64 bits, then 2^{64} possible keys where the attacker must try about 2^{63} keys to find correct one.

Passwords

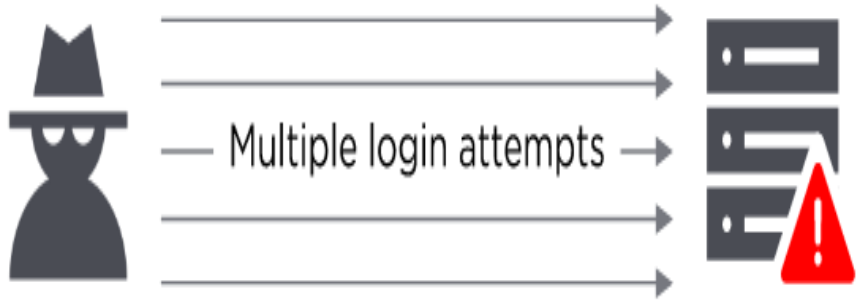
- If password is 8 characters long, with 256 possible choices for each character, then there are $256^8 = 2^{64}$ possible passwords.
- But users do not select passwords at random since they must memorize and a clever attacker can make fewer than 2^{63} guesses to crack it.(Dictionary attack)

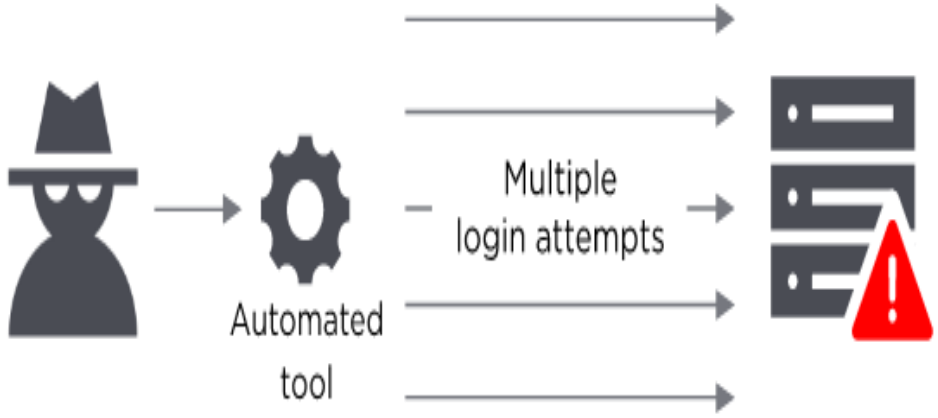
DICTIONARY ATTACK FAILURE FACTORS

- The average number of guesses the attacker must make to guess the correct password is determined by how unpredictable the password is or randomness of password, including **how long the password is, what set of symbols it is drawn from, and how it is created.**
- The ease with which an attacker can check the validity of a guessed password is determined by **how the password is stored, how the checking is done, and any limitation on trying passwords**

COMMON ATTACKS TO PASSWORDS

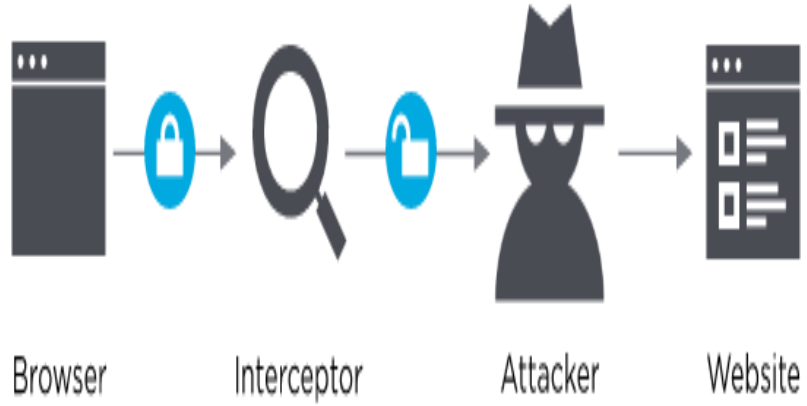
- **Dictionary attack:** An attack that takes advantage of the fact people tend to use common words and short passwords. The hacker uses a list of common words, the dictionary, and tries them, often with numbers before and/or after the words, against accounts in a company for each username. Trudy pre-computes h(x) for all x in a dictionary of common passwords. Suppose Trudy gets access to password file containing hashed passwords. She only needs to compare hashes to her pre-computed dictionary.
- **Brute force:** Using a program to generate likely passwords or even random character sets. These attacks start with commonly used, weak passwords like Password123 and move on from there. The programs running these attacks usually try variations on upper and lowercase characters, as well.

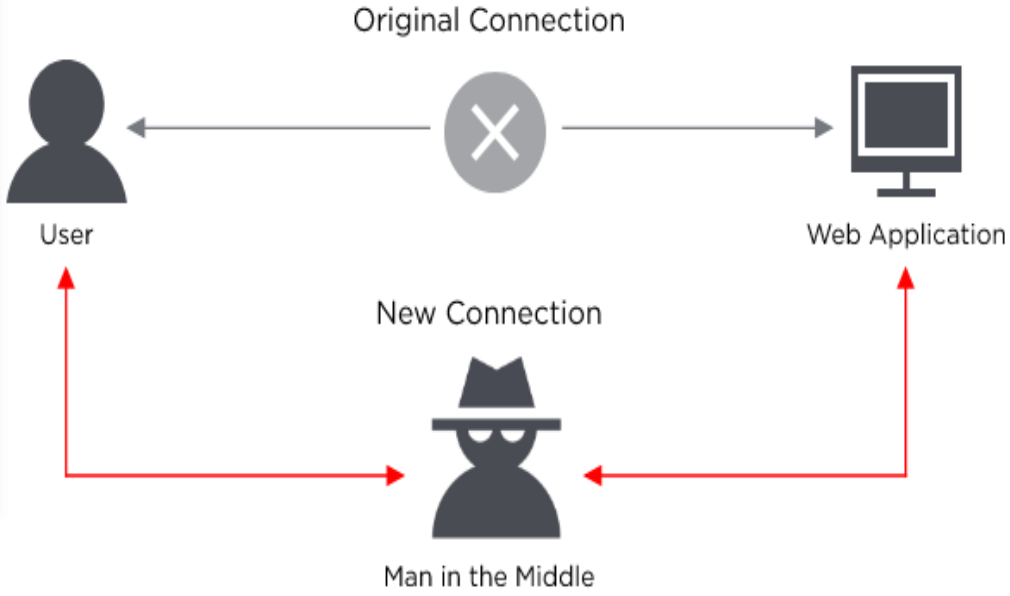




COMMON ATTACKS TO PASSWORDS

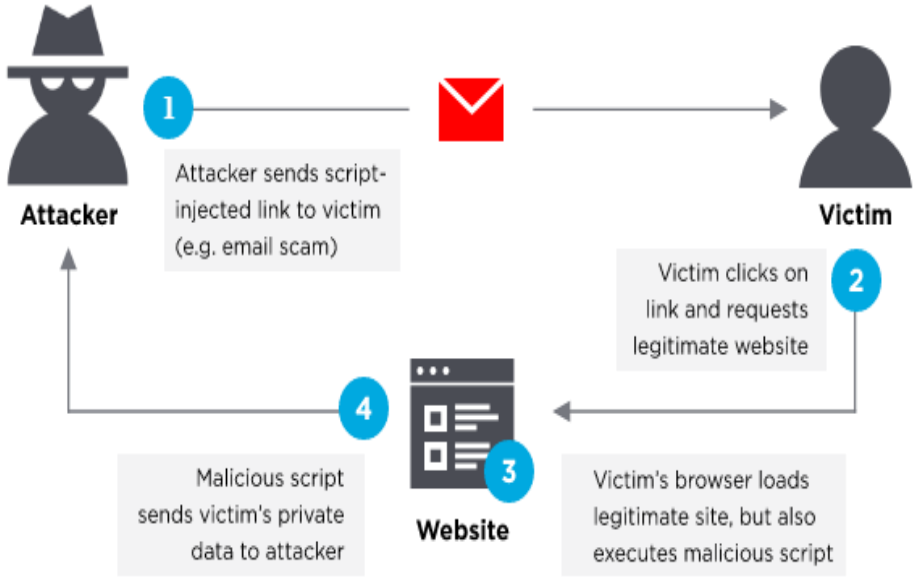
- **Traffic interception/ Eavesdropping:** In this attack, the cyber criminal uses software such as packet sniffers to monitor network traffic and capture passwords as they're passed. In this attack the software monitors and captures critical information. Obviously, if that information such as passwords is unencrypted, the task is easier. But even encrypted information may be decryptable, depending on the strength of the encryption method used.
- **Man In the Middle/ Social Engineering:** In this attack, hacker's program doesn't just monitor information being passed but actively inserts itself in the middle of the interaction, usually by impersonating a website or app. This allows the program to capture the user's credentials and other sensitive information, such as account numbers, social security numbers, etc. MITM attacks are often facilitated by social engineering attacks which lure the user to a fake site.





COMMON ATTACKS TO PASSWORDS

- **Key logger attack:** A cyber criminal manages to install software that tracks the user's keystrokes, enabling the criminal to gather not only the username and password for an account but exactly which website or app the user was logging into with the credentials. This type of attack generally relies on the user first falling prey to another attack that installs the malicious key logger software on their machine. It is also known as electronic monitoring.
- Accessing password file
- Login spoofing (human errors)



COMMON ATTACK PATH

- Outsider → normal user → administrator
- May only require **one** weak password to crash entire system!
- Normally attacker tries to grab one account and tries for privilege escalation.

RESPONSE TO A PASSWORD ATTACK

- If an attack happens with a password what is the proper response mechanism?
- Lock after 3 failed attempts
- How long should system lock?
- 5 seconds(Attacker can easily surge for next round) or 5 minutes(aattacker can cause denial of service attack)

Bad passwords

- **Default passwords** (as supplied by the system vendor and meant to be changed at installation time): *password*, *default*, *admin*, *guest* etc.
- **Dictionary words:** *chameleon*, *RedSox*, *sandbags*, *bunnyhop!*, *IntenseCrabtree*, etc.
- **Words with numbers appended:** *password1*, *deer2000*, *john1234*, etc.,
- **Words with simple obfuscation:** *p@ssw0rd*, *g0ldf1sh*, etc.
- **Doubled words:** *crabcrab*, *stopstop*, *treetree* *passpass* etc, can be easily tested automatically.

Good Passwords

- Allow long passphrases (FSa7Yago, 0nceuP0nAt1m8)
- Randomly generate passwords, though probably inappropriate for most scenarios
- Check the quality of user-selected passwords
  - use a number of rules of thumb
  - run dictionary attack tools

Bad passwords

- **Anything personally related to an individual:** license plate number, Social Security number, current or past telephone number, student ID, address, birthday, sports team, relative's or pet's names/nicknames/birthdays, etc.,: frank, Fido, AustinStamp(names), (10021995birthdates)
- **Common sequences from a keyboard row:** *qwerty*, *12345*, *asdfgh*, *fred*, etc.
- **Numeric sequences based on well known numbers** such as 911, 314159, or 27182, etc.
- **Identifiers:** *jsmith123*, *1/1/1970*, *555–1234*, "your username", etc

Good Passwords

- Give user suggestions/guidelines in choosing passwords
- e.g., think of a sentence and select letters from it, "It"s 12 noon and I am hungry" => "I"S12&IAH"
- Using both letter, numbers, and special characters

Passord Strength

- Three groups of users – each group advised to select passwords as follows
  - **Group A:** At least 6 chars, 1 non-letter
  - **Group B:** Password based on passphrase
  - **Group C:** 8 random characters
- Results
  1. **Group A:** About 30% of pwds easy to crack
  2. **Group B:** About 10% cracked
     Passwords easy to remember
  3. **Group C:** About 10% cracked
     Passwords hard to remember
- Inferences
  - **Assigned passwords sometimes best**
  - **If passwords not assigned, best advice is…**
    - Choose passwords based on passphrase
    - Use pwd cracking tool to test for weak pwds

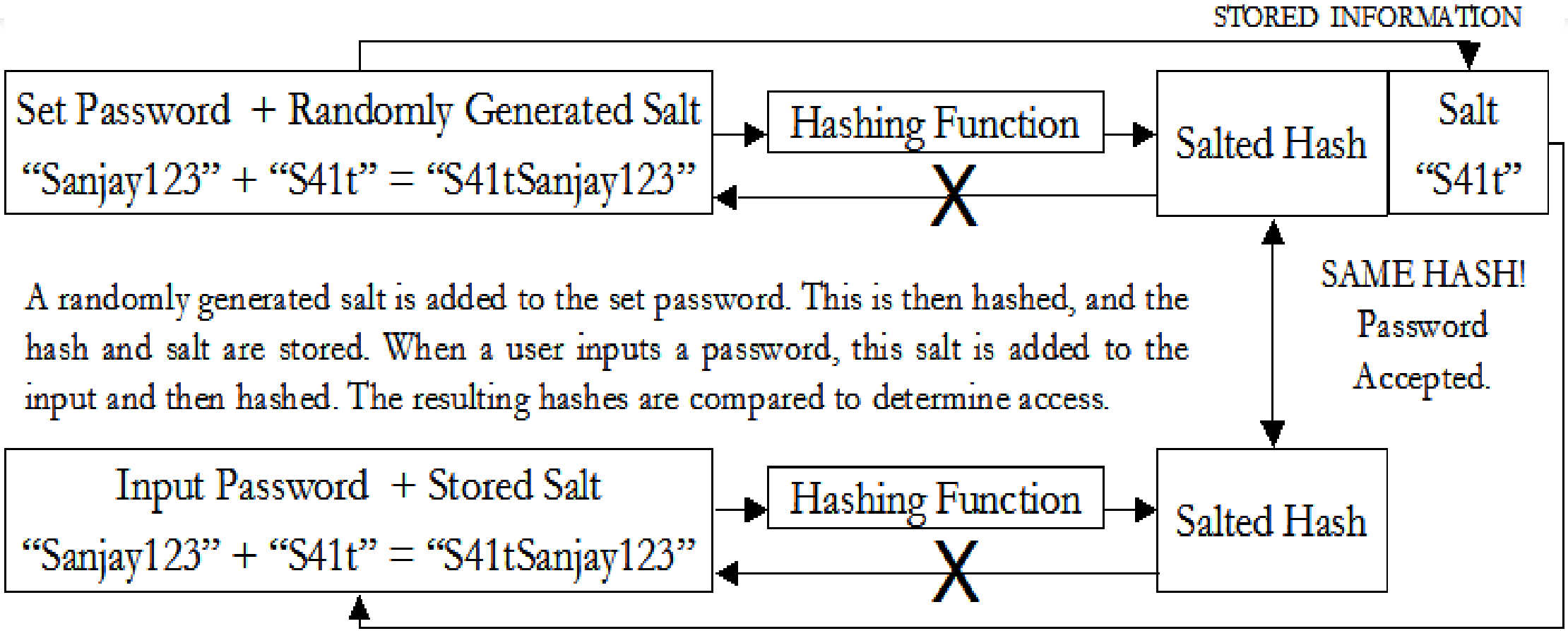
Password Verification

- To determine the validity of entered password machine must have access to the correct password. The passwords can be stored as:
  - **File System:** Clear text
  - **Dedicated Authentication Server:** Clear text
  - **Encrypted Password:** Password + Encryption = bf4ee8HjaQkbw
  - **Hashed Password:** Plain-text is converted into a message digest through the use of a hashing algorithm (i.e. MD5, SHA)
  - Password + Hash function = aad3b435b51404eeaad3b435b51404ee
  - **Salted Hash**(Username + Salt + Password) + Hash function =ad3b435b51404eeaad3b435b51404ee
  - **Create Fake Password files and store it all over the system**

SALT

- Hash password with **salt. Salting is adding a random piece of data to the password before hashing it.**
  - This means that the same string will hash to different values at different times.
  - Users with same password have different entries in the password file.
  - Salt is stored with the other data as a complete hash
- Choose random salt **s** and compute **y = h(password, s)** and store **(s,y)** in the password file.
- Note that the **salt s is not secret**. Still easy to verify salted password?
  - Hacker has to get the salt add it to each possible word and then rehash the data prior to comparing with the stored password.
  - With salt, attacker must compute hashes of all dictionary words once for each password entry.
  - With 12-bit random salt, same password can hash to 2^{12} different hash values.
  - Attacker must try all dictionary words for each salt value in the password file
- Without salt, attacker can pre-compute hashes of all dictionary words once for all password entries.
- Identical passwords hash to identical values; one table of hash values can be used for all password files.

SALT


STORED INFORMATION
Set Password + Randomly Generated Salt
"Sanjay123" + "S41t" = "S41tSanjay123"
Hashing Function
Salted Hash
Salt
"S41t"
A randomly generated salt is added to the set password. This is then hashed, and the hash and salt are stored. When a user inputs a password, this salt is added to the input and then hashed. The resulting hashes are compared to determine access.
SAME HASH!
Password
Accepted.
Input Password + Stored Salt
"Sanjay123" + "S41t" = "S41tSanjay123"
Hashing Function
Salted Hash

Password Security Issues

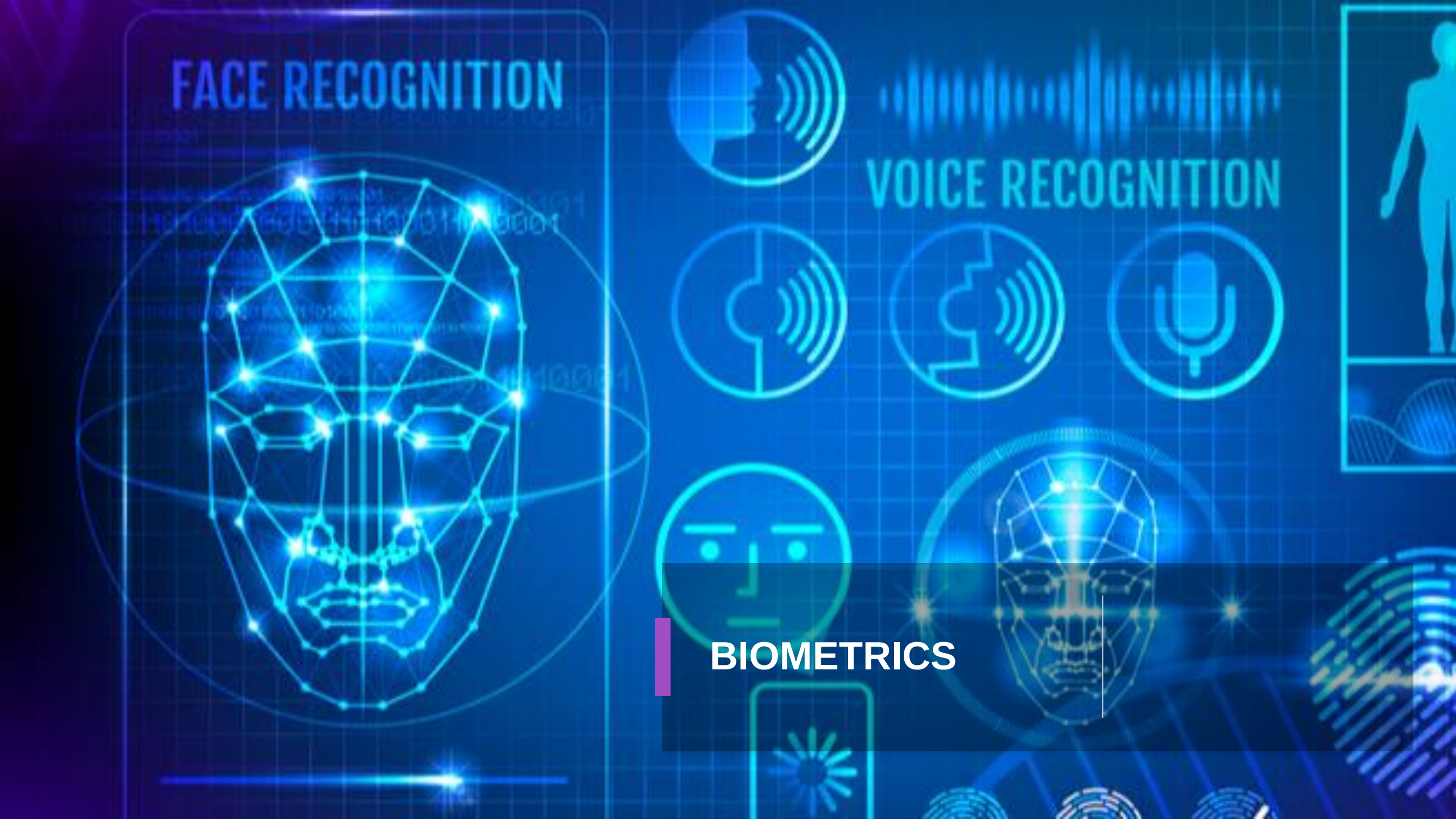
- **Remembering Multiple passwords:** Results in password reuse
- **Social Engineering**
- **Disclosure:** Voluntary disclosure of information and Inadequate guarding of system passwords
- **Inference:** Known pattern to creation of passwords and Use of generated passwords with predictable algorithm
- **Exposure:** Accidental release of password
- **Loss:** Forgetting passwords. Can lead to creation of easy passwords
- **Snooping/Eavesdropping:** Keyloggers and Network sniffing (intercepting of network communication where a password is submitted)
- **Guessing:** Limited amount of choices which can be figured out through process of elimination. Use of blank/common passwords, passwords which can be figured out by knowing name of relatives, pets, etc.
- **Cracking**: Automated "guessing"

Password Cracking Tools

- Popular password cracking tools
  - Password Crackers
  - Password Portal
  - L0phtCrack and LC4 (Windows)
  - John the Ripper (Unix)
- Admins should use these tools to test for weak passwords since attackers will|

Password Security Mechanisms

- Protect stored passwords (use both cryptography & access control)
- Disable accounts with multiple failed attempts
- Require extra authentication (2 or 3 factor) mechanism (e.g., phone, other email account, OTPS etc.)
- Ensure periodic password changes
- The same password can be rehashed many times over to make it more difficult for the hacker to crack the passw ord.
- Enforce strong passwords!
- Set BIOS to boot first from the hard drive.
- Password-protect the BIOS.
- Audit access to important files.



BIOMETRICS

BIOMETRICS: "something you are"

Fingerprint

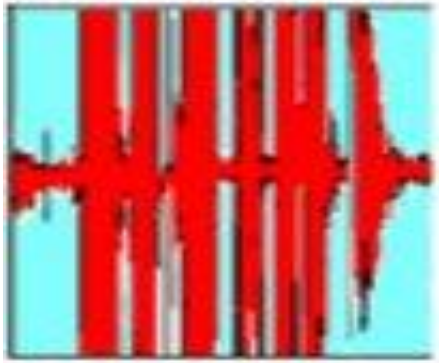
Face

Iris

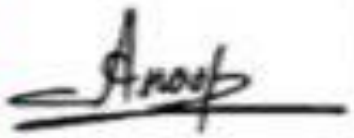
Palm print

Hand vein

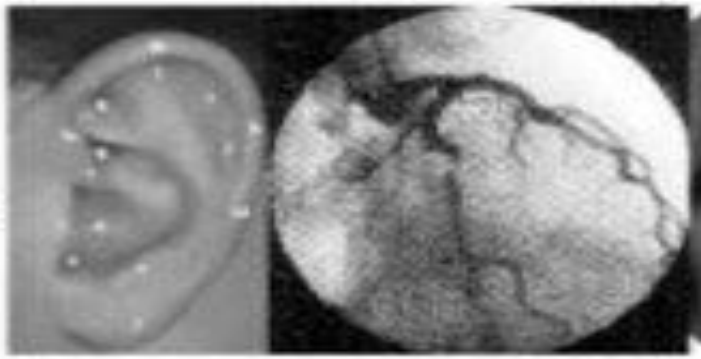
Finger geometry



Voice



Signature



Ear

Retina

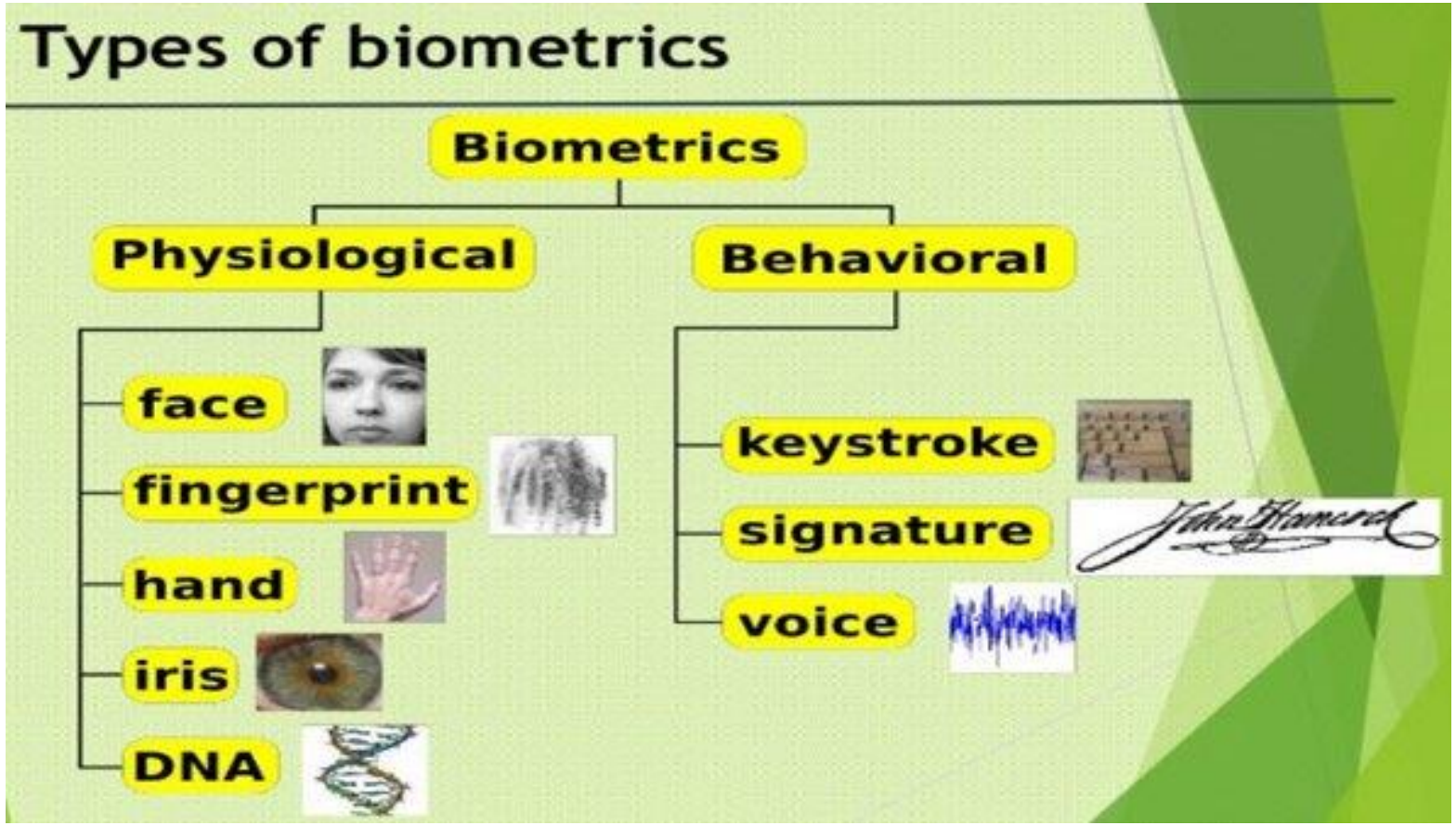
Tooth-shape

Walking gait

BIOMETRICS “You are your key” :Schneier

- Biometrics are automated methods of recognizing a person based on a **physiological or behavioral characteristics.**
- The word biometric is derived from the Greek words bio and metric. Where bio means life and metric means to measure.
- Uses certain biological or behavioral characteristics for authentication
- **Biological/physical Examples**
  - Fingerprint, Iris, Retina, Facial Recognition, & Hand Recognition
- **Behavioral Examples**
  - Handwriting(Signature), Walk/Step(Gait recognition), Typing Rhythm, Mouse Gesture Recognition, digital doggie (Odor recognition), Speech recognition

TYPES OF BIOMETRICS



Why Biometrics?

- Provides **better security and accuracy***. In contrast to passwords, badges, or documents, biometric data cannot be forgotten, exchanged, or stolen, and cannot be forged.
  - Biometrics systems are less prone to attacks
  - Need sophisticated techniques for attacks
  - Cannot steal facial features and fingerprints
  - Need sophisticated image processing techniques for modifying facial features
- Biometrics systems are **more convenient**
  - Need not have multiple passwords or difficult passwords
  - E.g., characters, numbers and special symbols, Need not remember passwords
  - Need not carry any cards or tokens
- **Better accountability**
  - Can determine who accessed the system with less complexity
- **Cheap and reliable**

IDEAL BIOMETRIC CHARACTERISTICS

1	**Universality**	How commonly biometric is found applies to (almost) everyone. In reality, no biometric applies to everyone
2	**Uniqueness/ Distinguishing**	How well biometric distinguishes between others with virtual certainty
3	**Permanence**	Physical characteristics measured should never change. How well biometric resists aging
4	**Collectability**	The physical characteristics should be easy to collect without causing potential harm to subject. How easy biometric is to acquire? Depends on whether subjects are cooperative
5	**Performance**	Reliance, Robust Accuracy, speed, and Userfriendliness of system capturing biometric.
6	**Acceptability**	Degree of approval by the public for use
7	**Circumvention**	How hard it is to fool authentication system

Phases of a biometric system

1. **Enrollment phase**(**Identification**): A raw biometric is captured by a sensing device such as fingerprint scanner or video camera and entered into database. The distinguishing characteristics are extracted from the raw biometrics sample and converted into a processed biometric identifier record called **biometric sample or template or helper data. Its slow phase.** Example: FBI fingerprint database
2. **Recognition phase**(**Authentication**)**:** Biometric authentication works by comparing two sets of data: the first one is preset by the owner of the device, while the second one belongs to a device visitor. If the two data are nearly identical, the device knows that "visitor" and "owner" are one and the same, and gives access to the person. Its fast phase. Biometric sensor devices example: **Fingerprint scanners**: **optical, capacitive** and **ultrasound**

How are your fingerprints stored?

- Google and Apple store your fingerprint on the device itself and do not make a copy of it on their own servers.
- Apple's TouchID won't store the actual image of the fingerprint, but a **mathematical representation** of it. So even if a malicious hacker reaches this mathematical representation, he cannot reverse engineer it to reveal an actual image of your fingerprint. Not only that, but **the fingerprint data itself is encrypted**.

Biometric Errors

- ***Fraud rate***- The rate at which mis-authentication occurs
- For e. g. Bob poses as Alice and the system mistakenly authenticates Bob as Alice
- Occurs when two people have high degree of similarity
  - Facial features, shape of face etc.
  - Template match gives a score that is higher than the threshold
  - If threshold is increased then false match rate is reduced, but False no match rate is increased
- ***Insult rate***- The rate at which the system fails to authenticate the subject.
- For e.g. Alice tries to authenticate as herself, but the system fails to authenticate her
- occurs for the following reasons
  - Changes in user's biometric data
  - Changes in how a user presents biometric data
  - Changes in environment in which data is presented
- ***Equal error rate-*** rate for which the fraud and insult rates are the same. A way to ***compare*** different biometrics
- For any biometric, can decrease fraud or insult, but other one will increase. For example
  - 99% voiceprint match $\Rightarrow$ low fraud, high insult
  - 30% voiceprint match $\Rightarrow$ high fraud, low insult

Fingerprint

- Fingerprints do not change over time unlike other biometrics.
- Fingerprint image of a person is captured and features are enhanced using various image processing techniques. It records its features like **arches, whorls** and **loops** along with the outlines of **edges**, **minutiae** and **furrows**. This is known as fingerprint recognition.

Loop (double)

Whorl

Arch

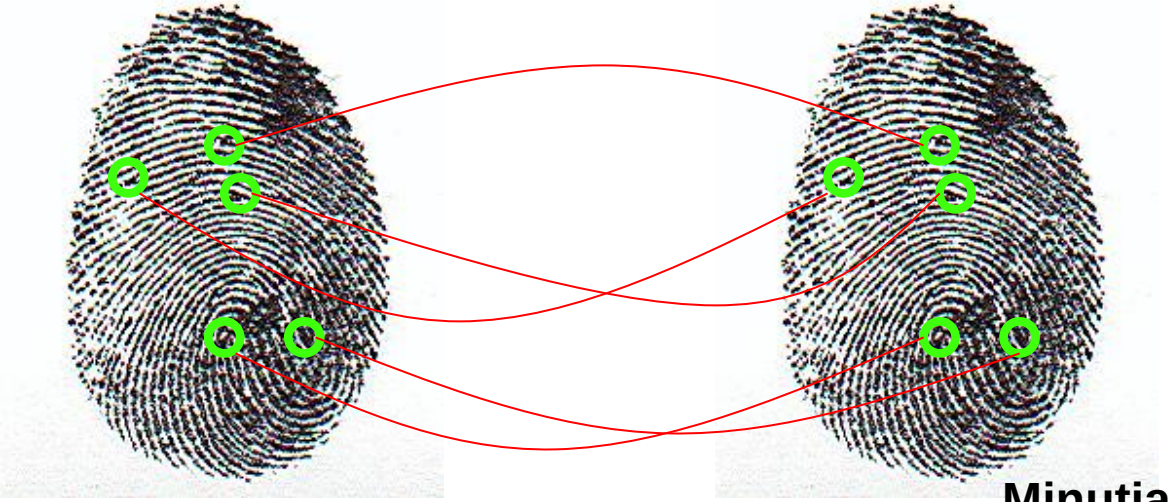
Fingerprint: Enrollment

- Capture image of fingerprint
- Enhance image
- Identify “points”
- Lines that create a finger print is called **ridges** and spaces in between are called **valleys**.

Fingerprint: Comparison

- Fingerprint comparison can be attained in three ways, such as **minutiae**(specific points), **correlation and ridge.**

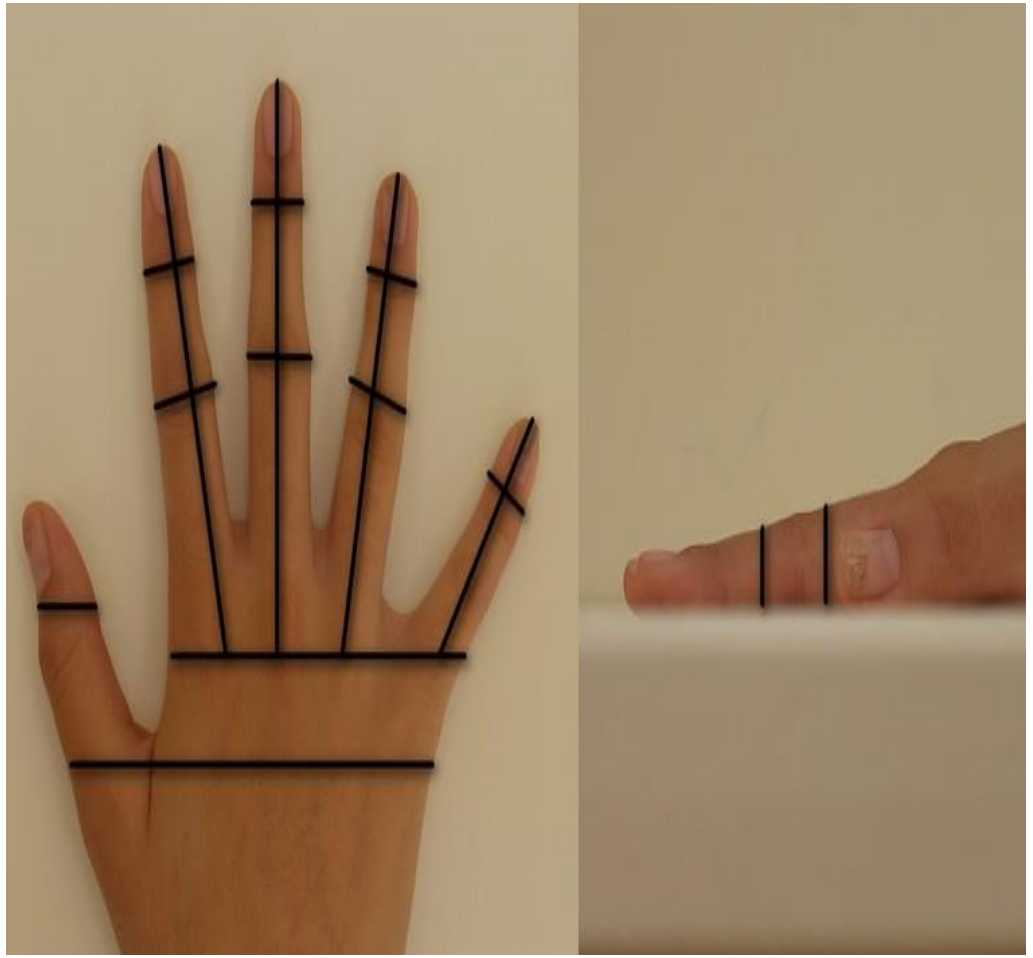
1. **Minutiae based fingerprint** matching stores a plane includes a set of points and the set of points are corresponding in the template and the i/p minutiae.
2. **Correlation based fingerprint** matching overlays two fingerprint images and association between equivalent pixels is calculated.
3. **Ridge feature based fingerprint** matching is an innovative method that captures ridges, as minutiae based fingerprint capturing of the fingerprint images is difficult in low quality.



Minutia comparison

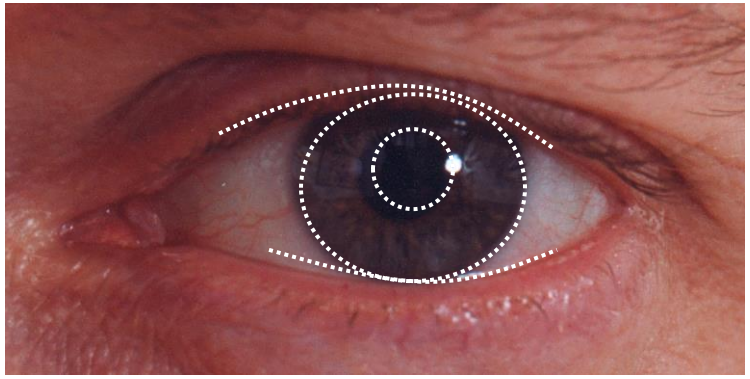
Hand Geometry

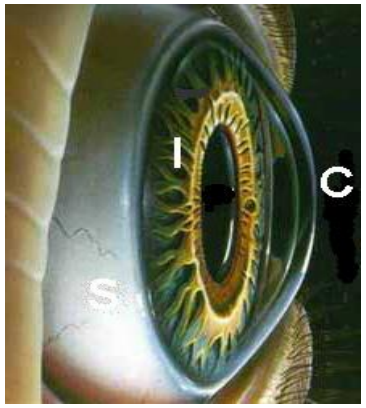
- A popular biometric that measures shape of hand(16 measures)
  - Width of hand, fingers
  - Length of fingers, etc.
- Though human hands not so unique, Hand geometry sufficient for many situations which is quick and robust.
- Not useful for ID problem
- **Advantages**
  - Quick: 1 minute for enrollment, 5 seconds for recognition
  - Hands are symmetric so recognition is fast.
- **Disadvantages**
  - Cannot use on very young or very old
  - Relatively high equal error rate



Iris Patterns

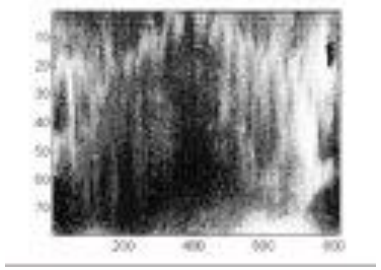
- **Iris recognition** is a one type of bio-metric method used to identify the people based on single patterns in the region of ring shaped pupil of the eye. Generally, the iris has a blue, brown, gray or green color with difficult patterns which are noticeable upon close inspection.
- It is often suggested as ultimate biometric for authentication.
- **Attack possible:** High clarity photographic images of user. To prevent this replay attack, iris scan systems can shine a light on the eye to verify whether pupil is contracting before confirmation.

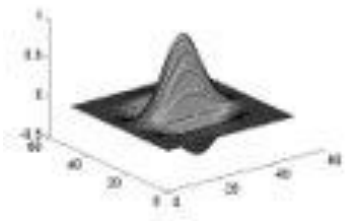


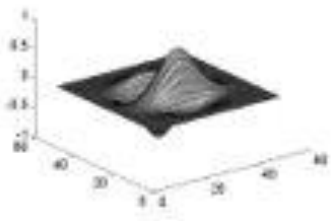


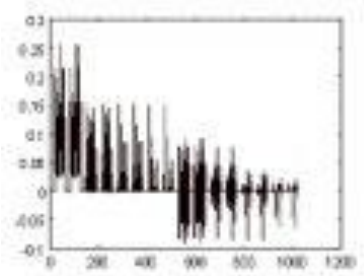
Iris Scan-Recognition Process

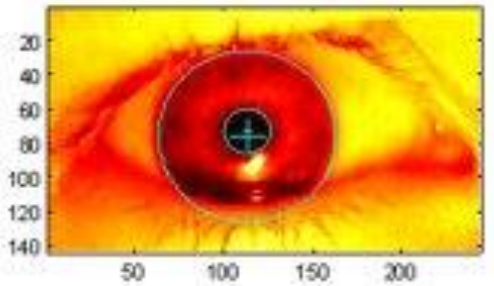
1. An iris scanner first locates the eyes and takes black and white photo of eyes.
2. The resulting image is processed using a 2D-wavelet transform, which results in a 256 byte **"iris code"**.
3. Iris codes are compared based on **hamming distance** between Iris codes.Eg: y-Iris scan of Alice stored in Database and x-iris code computed. Then **d(x,y)=no. Of non-match bits/no. Of bits compared** d(0010,0101)=¾ d(101111,101001)=2/6. A perfect match corresponds d(x,y) to 0. Usually a threshold of acceptance os kept as 0.32 otherwise non-match.





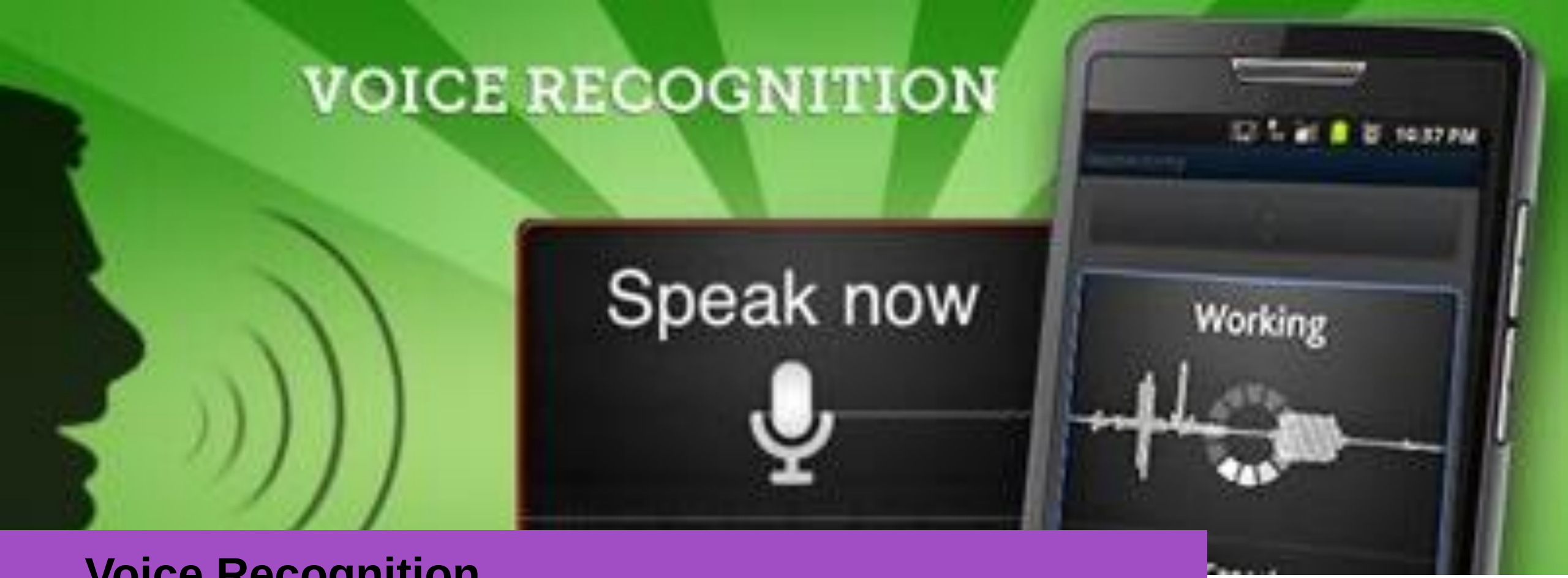






FACE RECOGNITION

- Present facial recognition systems work with face prints and these systems can recognize **80 nodal points** on a human face.
- Nodal points are nothing but end points used to measure variables on a person's face, which includes the length and width of the nose, cheekbone shape and the eye socket depth.



Voice Recognition

- Voice recognition technology is used to produce speech patterns by combining behavioral and physiological factors that can be captured by processing the speech technology. The most important properties used for speech authentication are **nasal tone, fundamental frequency, inflection, cadence.**
- Voice recognition can be separated into different categories based on the kind of authentication domain, such as a **fixed text method**, in the text dependent method, the **text independent method** and **conversational technique**.

Equal Error Rate Comparison

- **Equal error rate (EER):** The point at which fraud rate == insult rate
- **Fingerprint** biometrics used in practice have EER ranging from about 10^{-3} to as high as 5%
- **Hand geometry** has EER of about 10^{-3}
- In theory, **iris scan** has EER of about 10^{-6}
  - Enrollment phase may be critical to accuracy
- Most biometrics much worse than fingerprint!
- Biometrics useful for authentication…
  - …but for identification, not so impressive today

Biometrics: The Bottom Line

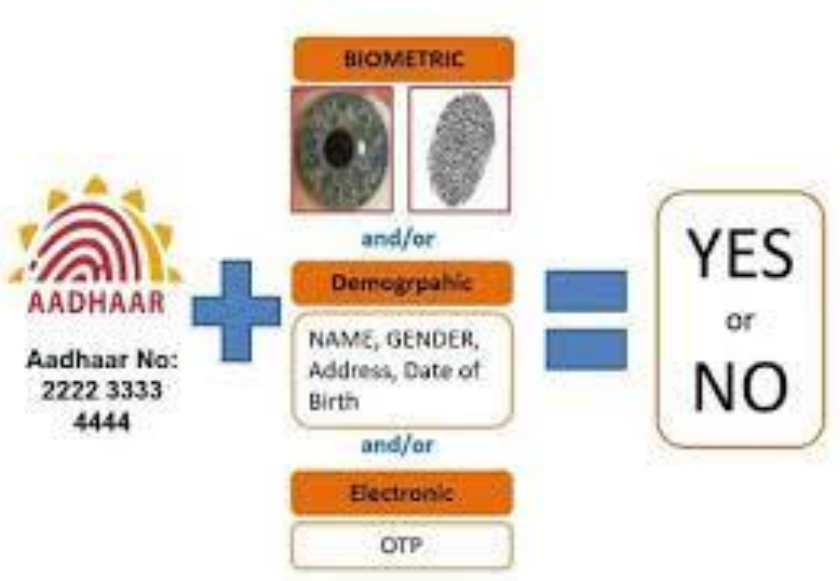
- Biometrics are hard to forge
- But attacker could
  - Steal Alice's thumb
  - Photocopy Bob's fingerprint, eye, etc.
  - Subvert software, database, "trusted path" …
- And how to revoke a "broken" biometric?
- **Biometrics are not foolproof**
- Biometric use is relatively limited today
- That should change in the (near?) future

Something You Have: SMART CARDS

- A smart card looks like credit card but includes small amount of memory and computing resources so that it can store cryptographic keys or other secrets and able to do some computations.
- Other Examples of "something you have " authentication: ATM Card, Password generator, ADHAR

AADHAR

- India's **Aadhaar project** is emblematic of biometric registration
- Aadhaar number is a 12-digit unique identity number issued to all Indian residents. This number is based on their biographic and biometric data (a photograph, ten fingerprints two iris scans).
- Initially the project has been linked to **public subsidy** and **unemployment benefit schemes** but it now includes a **payment scheme**.
- Financial inclusion is expected to be a key application of Aadhaar authentication with **Aadhaar Pay**, a new payment scheme announced in 2017.



TOKEN BASED AUTHENTICATION

- A security token is a peripheral device used to gain access to an electronically restricted resource.
- An authentication token is a small device that generates a new random value every time it is used, which is an alternative to password that can be used for authentication.
- Each authentication token is pre-programmed with a unique number called random seed. It ensures the uniqueness of the output produce by the token.

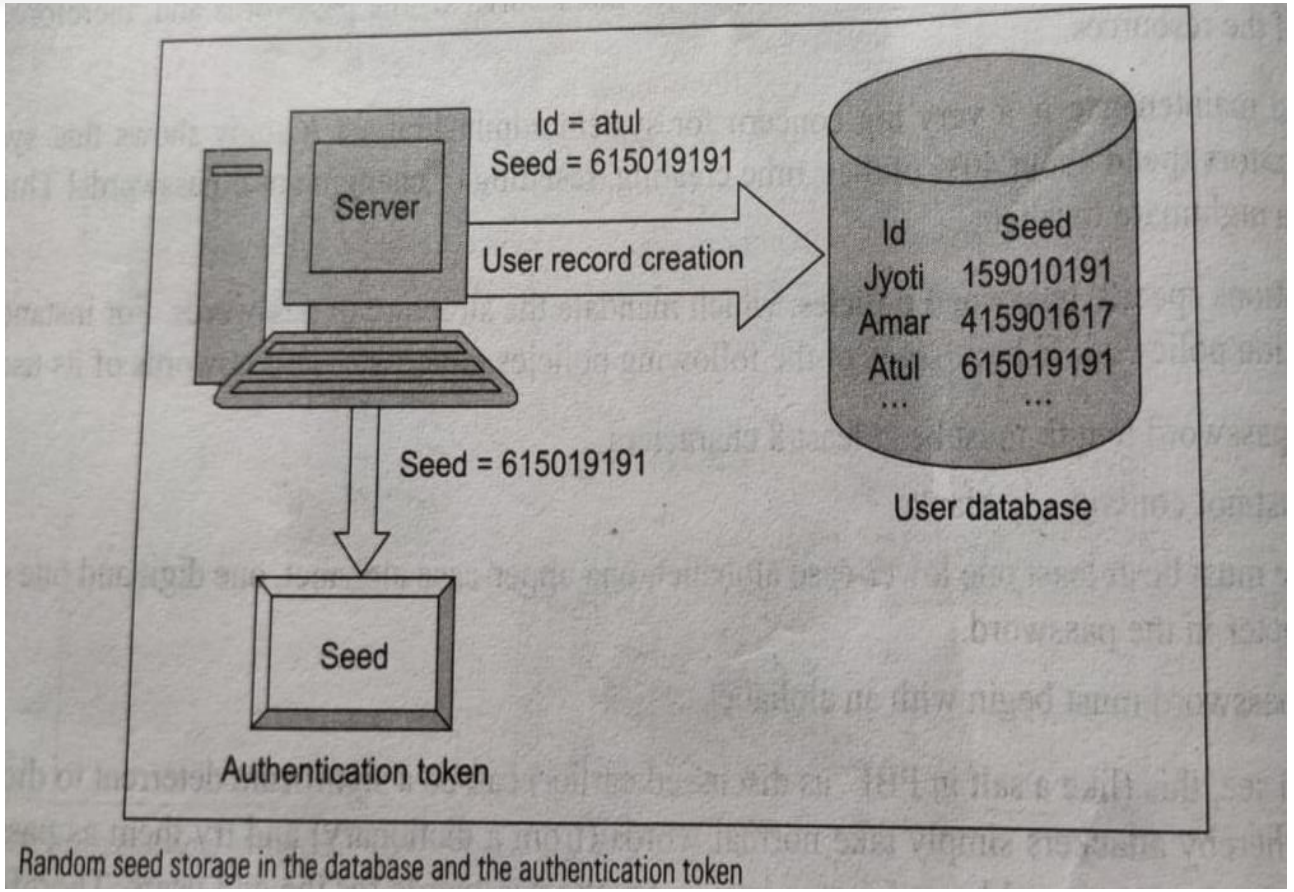
Authentication Token Device

- Authentication Device has the following features:
  - Processor
  - LCD for displaying outputs
  - Battery
  - Optionally a small keypad for entering information
  - Optionally a real-time clock

Working of Authentication Token

1. Creation of a Token

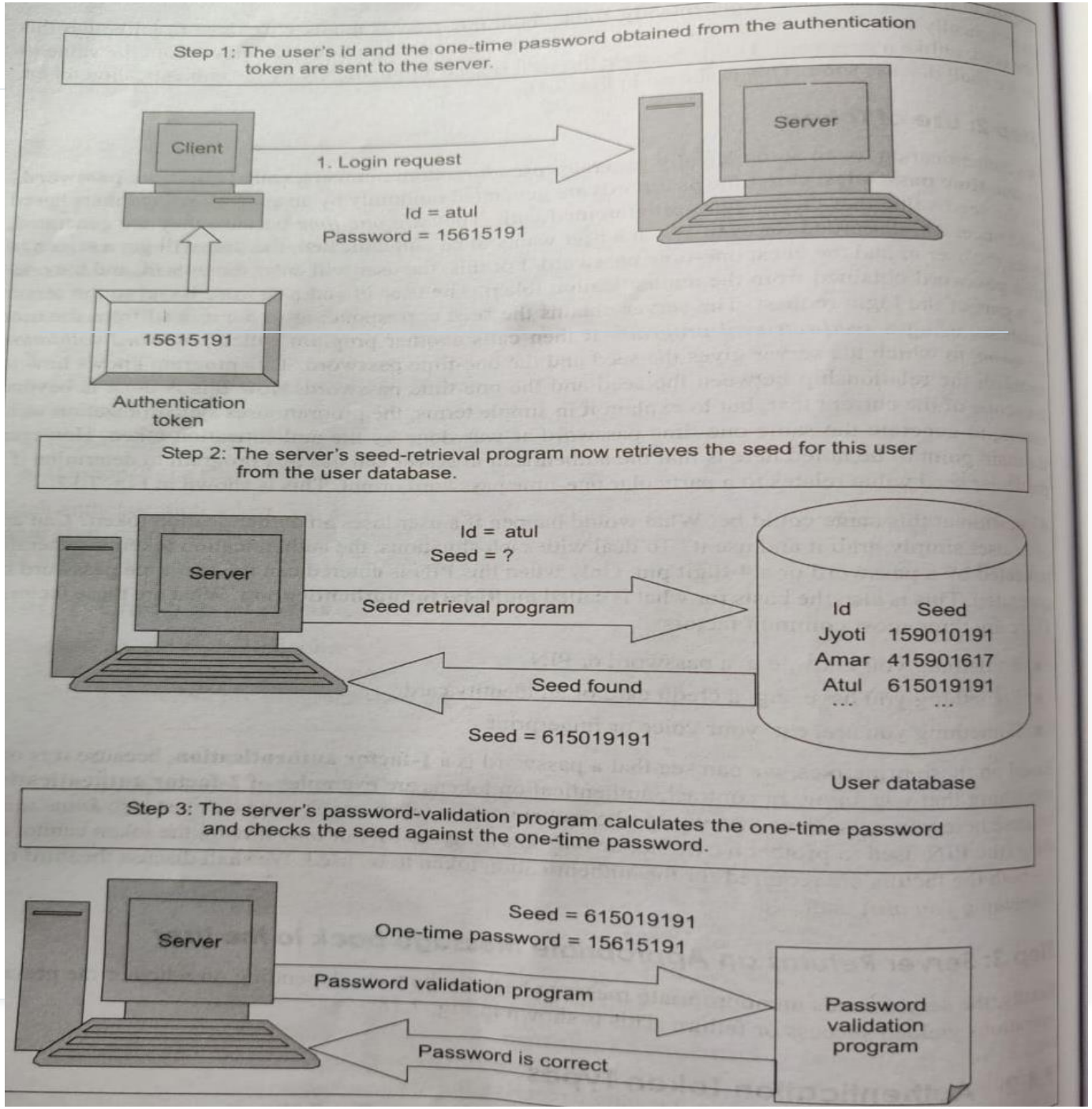
- Authentication tokens are created by Authentication servers along with random seed for token. It is automatically placed or pre-programmed inside each token by the server.
- Server also keeps a copy of the seed against the user ID in the user database.
- Seed can be conceptually considered as a user password. Difference is that the user password is known to the user, seed value remains unknown to the user



Random seed storage in the database and the authentication token

2. Use of the Token

- An Authentication Token automatically generates pseudorandom numbers called one-time passcodes or one time passwords.
- One time passwords are generated randomly by authentication tokens using the seed value.
- When a user wants to be authenticated by any server, the user will get a screen to enter user ID and the latest onetime password. The users enters its ID and gets is latest one-time password from the authentication token. The user ID and password travels to the server as a part of the login request.
- Server obatins the seed corresponding to the userid from database and verifies using some mechanism that this one-time password is created using the valid seed value.



Security

- **What happens if user loses an authentication token? Can another user grab token and use it?**
- Authentication token is generally protected by a password or a 4-digit PIN. Only when PIN is entered one-time password will be generated.
- Hence authentication tokens are 2-factor authentication techniques.

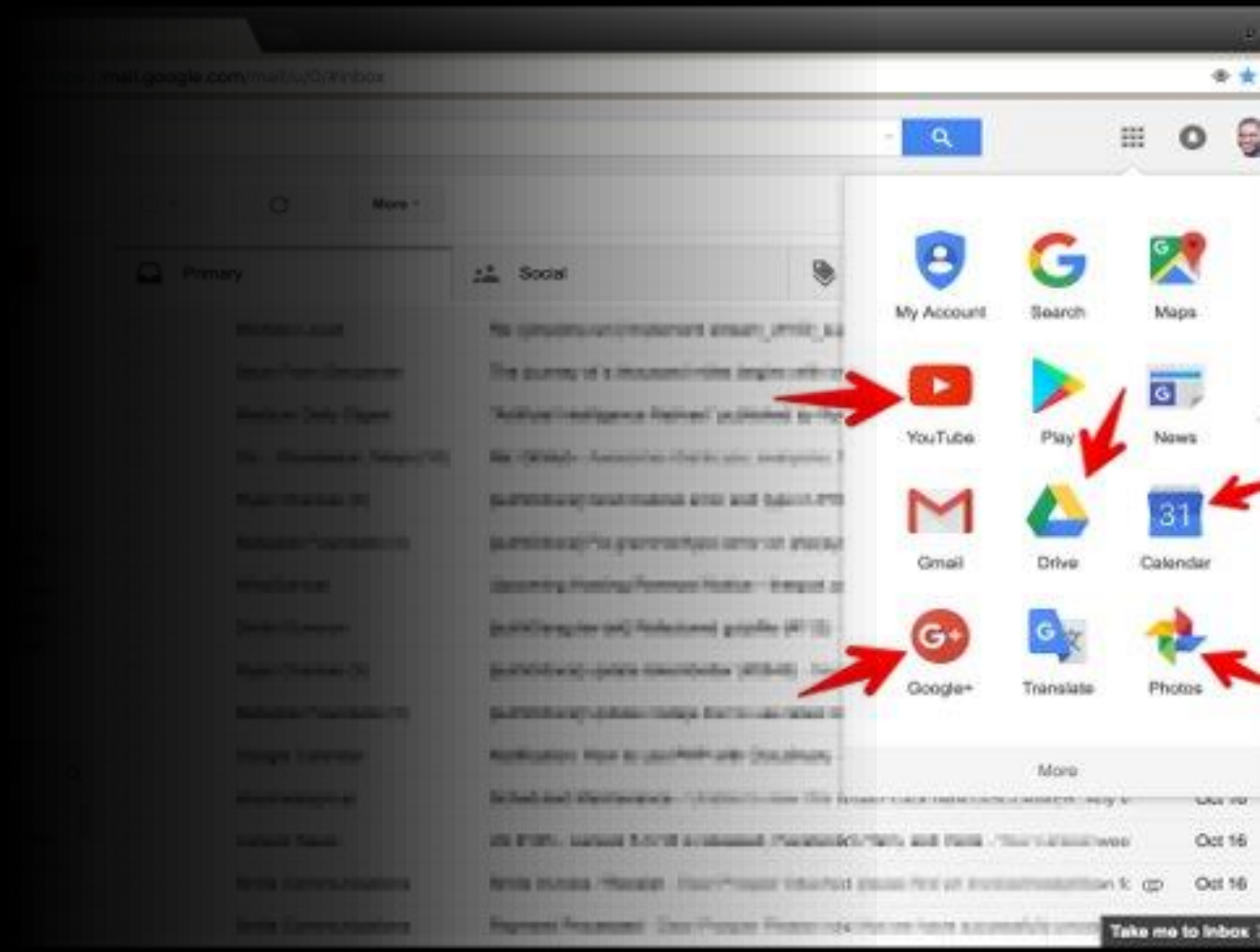
Types of Authentication Tokens

1. Challenge Response Tokens
2. Time based Tokens

2-factor Authentication

- Requires any 2 out of 3 of
  - Ø Something you **know**
  - Ø Something you **have**
  - Ø Something you **are**
- Ø Examples
  - Ø ATM: Card and PIN
  - Ø Credit card: Card and signature
  - Ø Password generator: Device and PIN
  - Ø Smartcard with password/PIN

Single Sign On



Single Sign-On

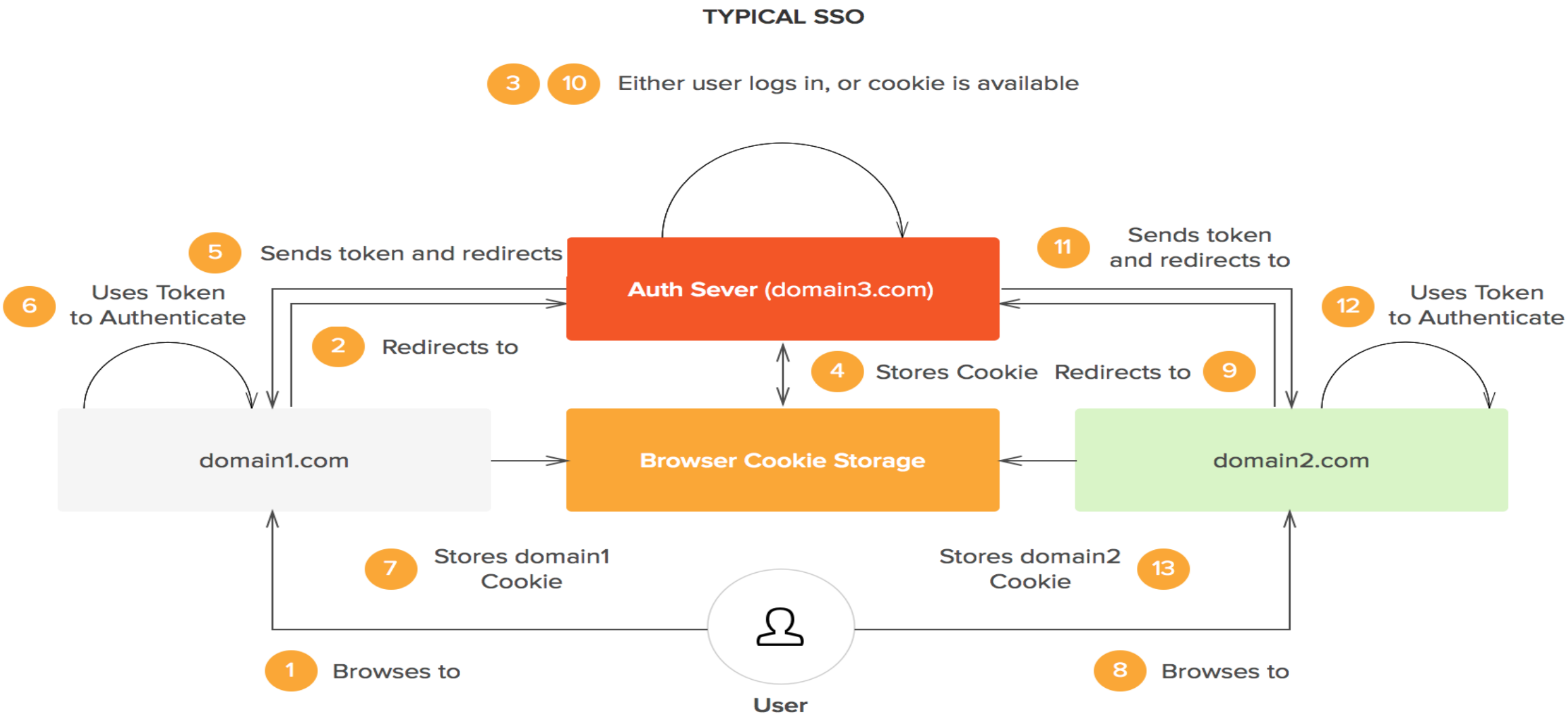
- **Single Sign On (SSO) login** refers to when a user logs in to an application with a single set of credentials and is then automatically signed into multiple applications where the "credentials" stay with them wherver they go on Internet. With SSO login, a user gains access to multiple software systems without maintaining different login credentials such as usernames and passwords.
- Single sign-on (SSO) is a property of access control of multiple, related, but independent software systems. Single sign-off is the reverse property whereby a single action of signing out terminates access to multiple software systems.
- As different applications and resources support different authentication mechanisms, single sign-on has to internally translate to and store different credentials compared to what is used for initial authentication.
- Initial authentication requires person's participation, but subsequent authentications would happen behind the scenes.
- **EXAMPLE:** Google accounts, auth0

Key Benefits of SSO Login

- **Eliminate the time spent** re-entering user credentials, thus improving productivity for users and increasing conversion rates for product owners
- **Eliminate password fatigue** from having to store or remember different usernames and passwords.
- **Reduce complaints about password problems**, thus reducing the costs associated with setting up several helpdesk systems for password-reset issues, invalid credentials, etc.
- **Minimize phishing**, thus improving security.
- Streamlines the local, desktop, and remote application workflows, thus improving users' productive capacity.

How SSO Login Works

- **Session** is shared with other domains in some secure way e.g., a signed JSON Web Token (JWT).
- **GIVEN SCENARIO:** Three apps have been developed separately: **App FOO**, **App BAR**, and **App BAZ**. They are also hosted on three different domains: **foo.com**, **bar.com** and **baz.com** respectively.
- With SSO login, a *central authentication server exists which is* **foobarbaz.com**.
    1. The user accesses **foo.com**.
    2. The user is redirected to **foobarbaz.com**, where an authentication-related cookie is generated.
    3. The user navigates to **bar.com**.
    4. The user is redirected to **foobarbaz.com**.
    5. **foobarbaz.com** checks whether the user already has an authentication-related cookie and redirects the user back to **bar.com**, providing access to its features and content.
    6. The same process applies to **baz.com**.


TYPICAL SSO
3
10
Either user logs in, or cookie is available
5
Sends token and redirects
Auth Sever (domain3.com)
11
Sends token and redirects to
6
Uses Token to Authenticate
2
Redirects to
4
Stores Cookie
Redirects to
9
12
Uses Token to Authenticate
domain1.com
Browser Cookie Storage
domain2.com
7
Stores domain1 Cookie
Stores domain2 Cookie
13
1
Browses to
8
Browses to
User

Web Cookies

- An **HTTP cookie / web cookie/ Internet cookie/ browser cookie/ cookie** is a small piece of data (in the form of an HTTP header that consists of a text-only string.)sent from a website (Web Server)and stored on the user's computer by the user's web browser while the user is browsing.
- Browser stores each data in a small file, called cookie.txt. Data is stored as name-value pairs. a Web site might generate a unique ID number for each visitor and store the ID number on each user's machine using a cookie file.
- browser accesses the cookie file again when you visit the website that created the cookie file. The browser uses the information stored in the cookie file to help ease your navigation of the website by letting you log in automatically or remembering settings you selected during your earlier visits to the website, among many other functions.
- **Who creates cookie?**
- cookie file is generated by the site you're browsing and is accepted and processed by your computer's browser software.
- **Is it a program?** No. They cannot gather any information on their own.
- **NOTE:** Only website that created cookie can read the cookie file it has created.

Cookie Parameters

- **Cookies have six parameters that can be passed to them:**

1. The **name** of the cookie.
2. The **value** of the cookie.
3. The **expiration** date of the cookie - this determines how long the cookie will remain active in your browser.
4. The path the cookie is valid for - this sets the **URL** path the cookie us valid in. Web pages outside of that path cannot use the cookie.
5. The **domain** the cookie is valid for - this takes the path parameter one step further. This makes the cookie accessible to pages on any of the servers when a site uses multiple servers in a domain.
6. The **need for a secure connection** - this indicates that the cookie can only be used under a secure server condition, such as a site using SSL.

Necessity of cookies

- **Tracking**
  - **To facilitate** hassle-free automatic logins and authentication
  - **To monitor advertisements ie** third party ad serving, ad management,
  - **To collect demographic information about who is visiting the Web site:** a web server can gather information about which web pages are used the most, and which pages are gathering the most repeat hits.
  - **a website can infer that the user has already visited.**
- **Personalization/Better User Experience(user preferences):** Servers can use cookies to provide personalized web pages. When you select preferences at a site that uses this option, the server places the information in a cookie. When you return, the server uses the information in the cookie to create a customized page for you.
- **Session management :**Logins, shopping carts, game scores, or anything else the server should remember Cookies are also used for online shopping. Online stores often use cookies that record any personal information you enter, as well as any items in your electronic shopping cart, so that you don't need to re-enter this information each time you visit the site. e-commerce sites, the unique ID can be used to set up the shopping cart for a user. A user can add items to a shopping cart and this preference is linked to the site's database along with the unique id. Later, when the same user visits the site, this unique id from cookie can be used to retrieve the shopping preference of that particular user.

Who earns with COOKIE?

- **Cookie-based ad tracking :** helps in user profiling/website preference tracking.
- Many largest websites online use large-scale third-party ad serving networks which cover many sites.
- Googles Adsense/Adwords ad serving network. Literally, millions of pages run Adsense ads. For every click a valid user makes on a Google-served ad on their site, site owners make money ranging from pennies to dollars.
- Webmasters have always been able to track access to their sites, but cookies make it easier to do so. In some cases, cookies come not from the site you're visiting, but from advertising companies that manage the banner ads for a set of sites (such as DoubleClick.com). These advertising companies can develop detailed profiles of the people who select ads across their customers' sites.
- Accepting a cookie does not give a server access to your computer or any of your personal information (except for any information that you may have purposely given, as with online shopping). Also, it is not possible to execute code from a cookie, and not possible to use a cookie to deliver a virus.

Types of Cookies

1. **Transient Cookies**
   - This type of cookie is stored in the computer's memory, i.e. RAM, and contains a session ID that stores information on the user's browsing session. These cookies are also known as **session cookies** and are deleted from the computer as soon as the browser is shut down. The session id in a session cookie acts as the user id that allows navigating from one page to another without logging in repeatedly. These session cookies will become inaccessible if the session becomes inactive for a specific period of time.
2. **Persistent Cookies**
   - These types of cookies are stored on the computer's hard drive and contain information on user preferences for a website. These preferences can be used for further browsing sessions and are retained as long as the user allows them. Unlike transient cookies, these cookies are permanently stored and are not deleted when a browser is shutdown.
3. **Flash Cookies**
   - These cookies are small Flash files such as video clips or gifs stored on a user's computer by websites. They contain the same data as normal cookies and they function in the same way. Unlike other cookies, Flash cookies cannot be deleted by any mechanism from the browser level. They can only be deleted manually or by using some special add-ons. So, if a Flash cookie is retained on a computer's hard disk, it will be used as a backup for the normal cookie and hence can be used to recall the preferences of a particular user to a website.

Risk Associated with Cookies

- Common risks of cookies are:
  1. **Cross Site Request Forgery Attack (XSRF):** When a website receives a request, it cannot distinguish whether the action is initiated by the user or not. user visits a legitimate site and receives a legitimate cookie. The user then visits a malicious site that instructs the user's browser to perform some action targeting the legitimate site. The legitimate site receives the request along with the legitimate cookie and performs the action since it appears to be initiated by a legitimate user.
  2. **Session Fixation:** an attacker impels the user to use the attacker's or another's session ID. This can be done by using the cookie's browser directive path, hence the user pretends to be someone else. Using this method, an attacker can urge the user to log in as the attacker on various application levels. a user receives a malicious cookie that contains the cookie issuer's session ID. When the user attempts to log into a targeted domain, the issuer's session ID is logged in instead of the user's session ID. In this way, it looks to the targeted domain like the issuer is performing actions that the user is actually performing.
  3. **Cross-Site Scripting:** a user visits a malicious website and receives a cookie that contains a script payload targeting a different website. The malicious cookie is disguised to look like it originated from the targeted website. When the user visits the targeted site, the malicious cookie, including the script payload, is sent to the server hosting the targeted site. This occurs when an attacker takes advantage of a website that allows its users to post unfiltered HTML and JavaScript content. By posting malicious HTML and JavaScript code, the attacker can cause the victim's web browser to send the victim's cookies to a website the attacker controls.
  4. **Cookie Tossing Attack:** a user visits a malicious site that provides a cookie designed to look like it originated from a subdomain of a targeted site, such as *http://subdomain.example.com*. When the user visits the targeted site, *http://example.com* in this case, the subdomain cookie is sent along with any legitimate cookies. If the subdomain cookie is interpreted first, the data in that cookie will overrule the data contained in any subsequent legitimate cookies.**Cookie Overflow Attack**
  5. **Session Hijacking/Cookie Hijacking**
  6. some viruses and malware may be disguised as cookies: "supercookies","zombie cookies"(a cookie that recreates itself after being deleted**)**

COOKIE LOCATION

- **Where is the location of the Cookies folder?**
- **Firefox cookies are stored in "cookies.txt"**
  at C:\Users\Default\AppData\Roaming\Microsoft\Windows
- **IE**
- C:\Documents and Settings\<User name>\Local Settings
- **Chrome**
- C:\Documents and Settings\<user name>\Local Settings\AppData\Google\Chrome\User Data\Default\Cookies
- C:\Users\sdk\AppData\Local\Google\Chrome\User Data\Default

Attack Prevention

- **Browser setting controls:** Every browser gives you a range of options for handling cookies. It lets you delete existing cookies in a single click and choose how future cookies are collected or stored.
  - **Banning all cookies**? **NO**. makes some websites difficult or impossible to navigate. However, a **setting that controls** or limits third-party and tracking cookies can help protect your privacy while still making it possible to shop online and carry out similar activities.
- **Use https method of authentication** wherever possible: Network traffic analysis can be done by intruder, which includes cookies sent on ordinary unencrypted HTTP sessions. MITM attack, Evesdropping
- Even if the cookies are stolen, developers should come up with codes that **check the source address and referrer of the authenticated user periodically**. If there is a **mismatch, the session need to be closed suddenly.**
- The **expiry time of the cookie** that is allocated from the web server needs to be **shortened.**

Thank you

anooja@somaiya.edu