Authentication

- **Authentication**: deals wih the problem of determining whether a user or (other entity) should be allowed access to a partcular system or resource.
- **Authentication Methods**: the human can be authenticated to a machine based on any combination of the following:
  - Something you know: passwords
  - Something you have: ATM Card or a smartcard
  - Something you are: biometrics
- **Mutual authentication**: when two sides of a communications channel verify each other's identity, instead of only one side verifying the other.
  - Also known as "two-way authentication".

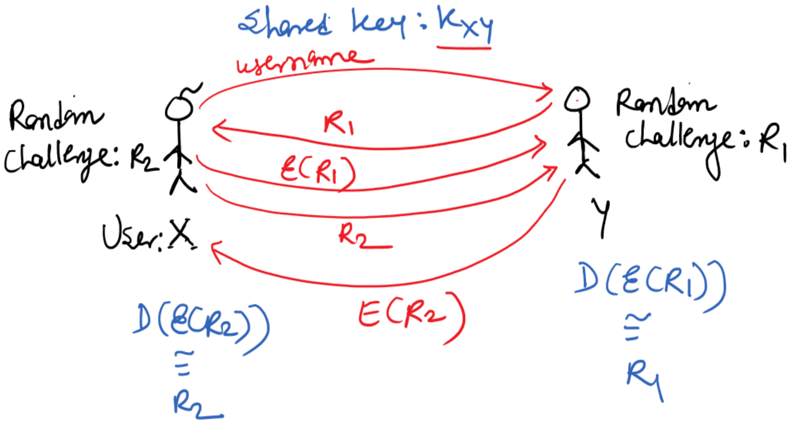
Mutual Authentication using Shared Key

Suppose User X and User Y wants to authenticate using a shared key.

The protocol works as follows:

1. Assume that User X and User Y both have shared key K_{XY}.
2. User X sends his user name to user Y.
3. After receiving the user name from X, user Y sends random challenge R_1 to X.
4. Once X received random challenge R_1, he encrypts R_1 using the shared key K_{XY}.
5. X sends an encrypted random challenge to Y.
6. Again User X sends random challenge R_2 to user Y.
7. Once Y received random challenge R_2, he encrypts R_2 using shared key K_{XY}.
8. Y sends an encrypted random challenge to X.

Mutual Authentication using Shared Key (cont.)



Use cases for one-way authentication:

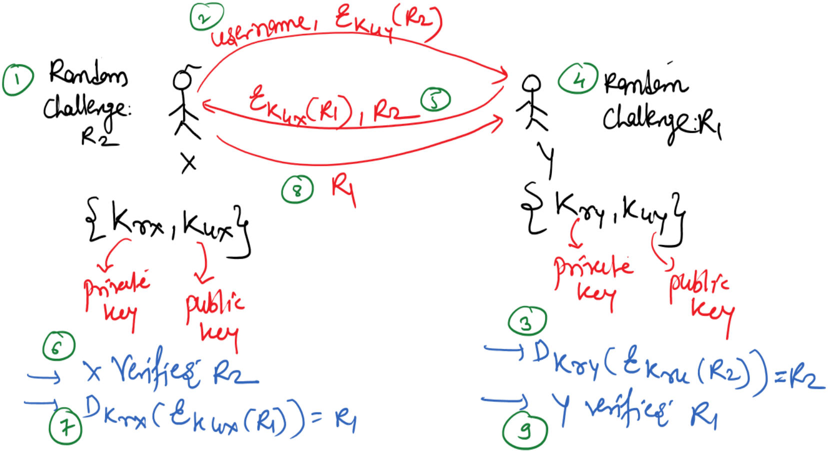
- Netbanking login using One Time Password (OTP).
- Email login using OTP.

Mutual Authentication using Public Keys

Suppose both user X and user Y know's each other's public keys.
The protocol works as follows:

1. User X encrypts random challenge R_2 using the public key of Y and sends it to User Y with his user name.
2. User Y decrypts the random challenge R_2 with his private key. User Y creates its own random challenge R_1 and encrypt it using public key X and send both (encrypted R_1 and decrypted R_2) to X.
3. User X decrypt random challenge R_1 with his private key and send it to Y. User Y verifies R_1.

Mutual Authentication using Public Keys (cont.)



1 Random Challenge: R_2
2 username, $E_{K_{uy}}(R_2)$
X
4 Random Challenge: R_1
Y
5 $E_{K_{ux}}(R_1), R_2$
8 R_1
$\{K_{rx}, K_{ux}\}$
private key
public key
$\{K_{ry}, K_{uy}\}$
private key
public key
3 $D_{K_{ry}}(E_{K_{ru}}(R_2)) = R_2$
6 X verifies R_2
7 $D_{K_{rx}}(E_{K_{ux}}(R_1)) = R_1$
9 Y verifies R_1

Use cases for public key mutual authentication:

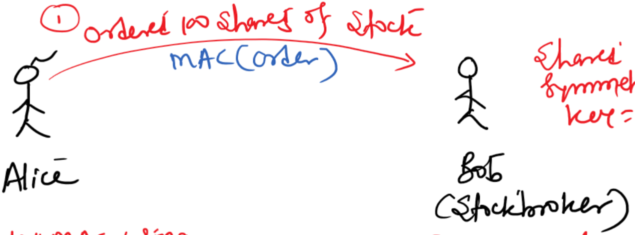
- Secure Shell Protocol (SSH)

Non-repudiation

- **Non-repudiation**: assurance that the sender of information is provided with proof of delivery and the recipient is provided with proof of the sender's identity, so neither can later deny having processed the information.

Non-repudiation (cont.)

- **Scenario**:



Shared symmetric key = KAB

② Computed MAC using KAB to ensure integrity of her order.

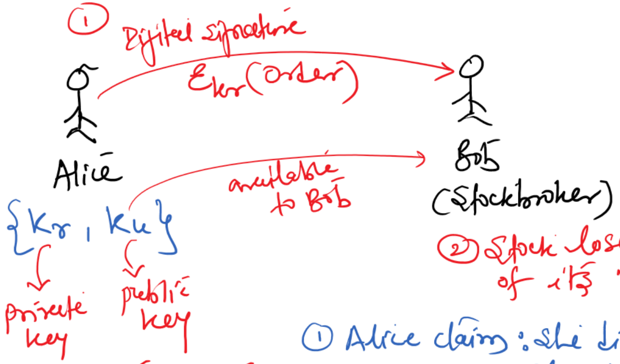
③ Stock loses 80 of its value

④ Alice suppose to pay money to Bob.

① Alice claims: she did not place the order.

② Can Bob prove that Alice placed the order?

Digital Signature



Alice wants to order 100 shares of stock

② Stock loses 80% of its value

① Alice claims: she did not place the order ✗

② Can Bob prove that Alice placed the order? ✓