Authentication: Part 2

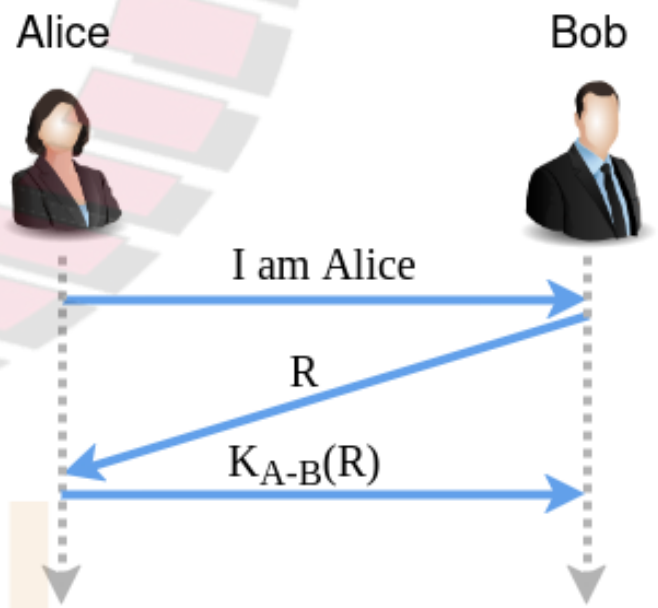
Gaurav S. Kasbekar
Dept. of Electrical Engineering
IIT Bombay

References

- J. Kurose, K. Ross, "*Computer Networking: A Top Down Approach*", Sixth Edition, Pearson Education, 2013
- C. Kaufman, R. Perlman, M. Speciner, "*Network Security: Private Communication in a Public World*", Pearson Education, 2nd edition, 2002

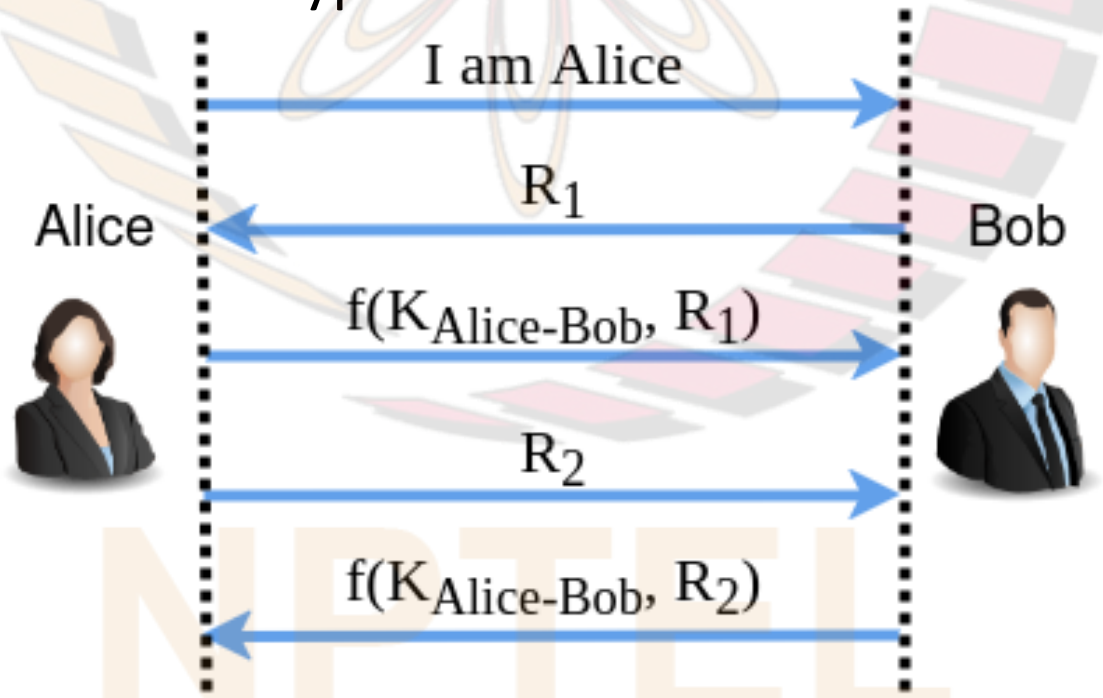
Another Weakness of Protocol ap4.0

- Recall: protocol ap4.0 vulnerable to server database reading attack
- Another weakness that it has:
  - ❑ Bob authenticates Alice, but Alice does not authenticate Bob
  - ❑ If an intruder on path between Alice and Bob intercepts the "I am Alice" message, he/ she can send any number R to Alice and ignore her response $K_{A-B}(R)$
  - ❑ Alice thinks that she is communicating with Bob
- Suppose Alice and Bob have a shared symmetric key K_{A-B}
- How can we modify protocol ap4.0 to achieve *mutual authentication,* i.e., Bob authenticates Alice and vice versa?



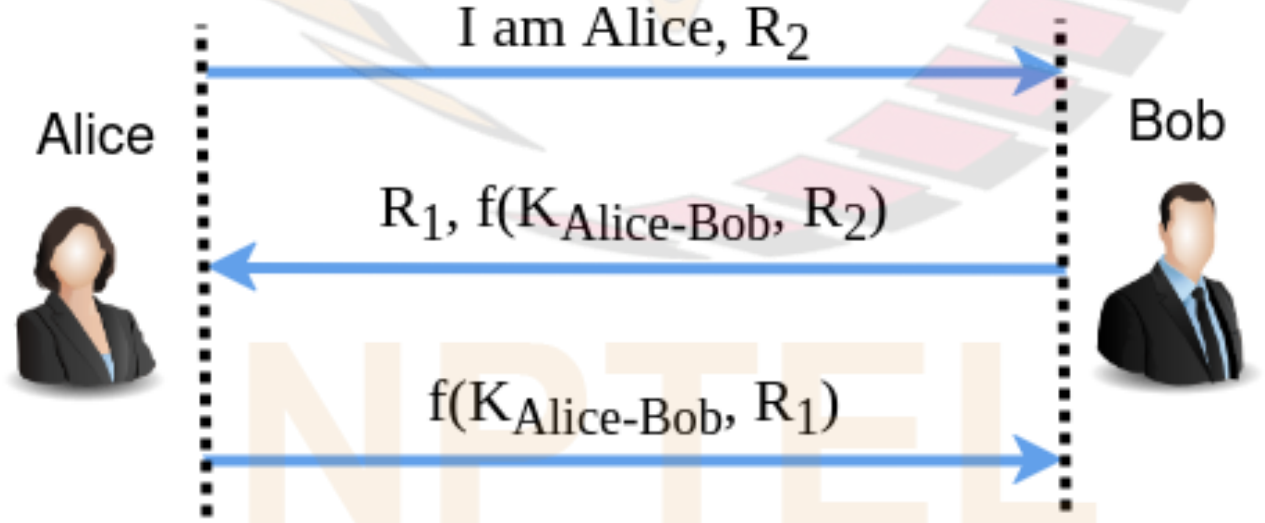
Mutual Authentication

- After Alice sends the message "I am Alice" to Bob, we do an authentication exchange in each direction
- $f(K_{A-B}, R)$ may be:
  1) $K_{A-B}(R)$ (R encrypted using the key K_{A-B}) or
  2) $H(R, K_{A-B})$ (hash of R concatenated with K_{A-B})
- The protocols with both 1) and 2) achieve mutual authentication; 2) has the advantage that it does not require encryption and decryption



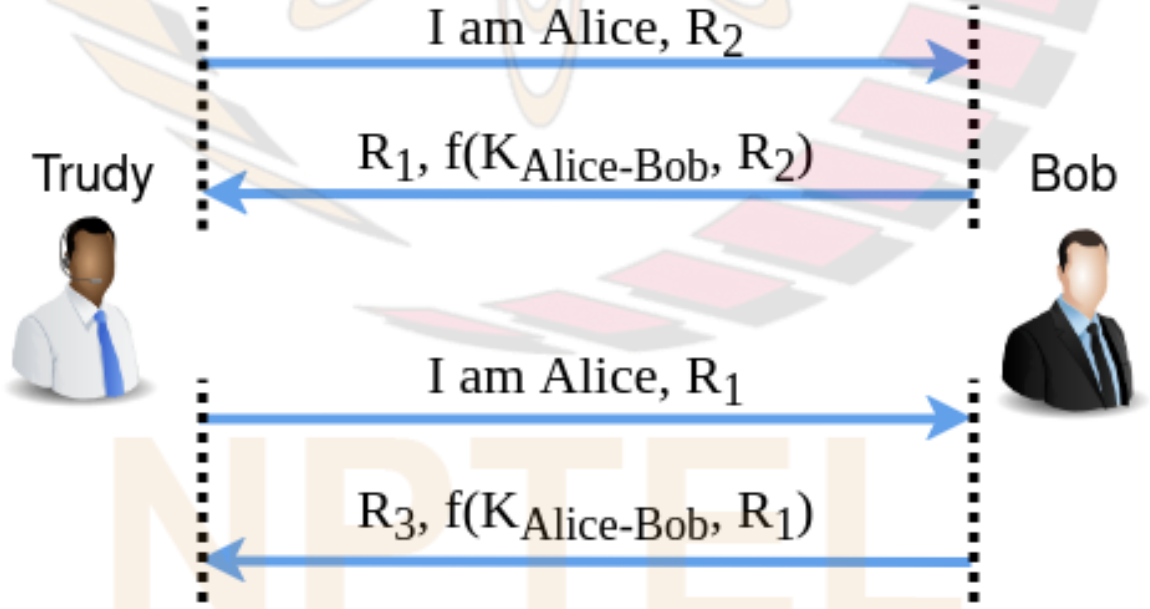
Mutual Authentication (contd.)

- Above protocol is inefficient since it uses *five* messages
- A modified version that uses *three* messages shown in fig.
- Is this modified version secure?
  - ❑No; vulnerable to "*reflection attack*"



Reflection Attack

- Suppose it is possible for a client to open multiple sessions to server
- Then an intruder, Trudy, can fraudulently authenticate herself as Alice to Bob as follows
- First, initiates a mutual authentication as in above three message protocol
  - she receives $R_1, f(K_{Alice-Bob}, R_2)$ from Bob
  - initiates a new session by sending "I'm Alice, R_1" to Bob; receives $f(K_{Alice-Bob}, R_1)$ from Bob; uses it to complete the first session

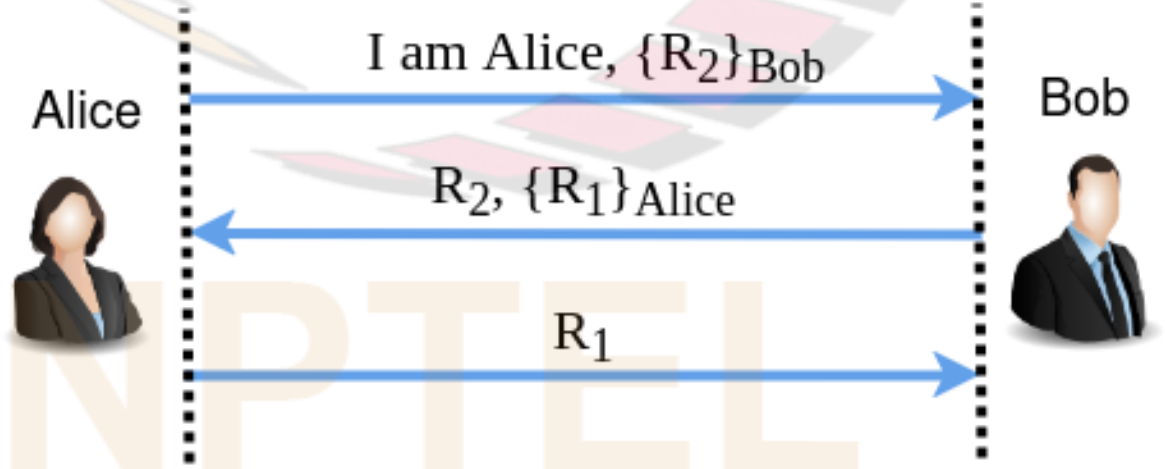


Defences Against Reflection Attack

1) Different keys used to authenticate Alice to Bob and to authenticate Bob to Alice
2) When Alice sends "I'm Alice, R_2" to Bob, Bob responds with $R_1, f(K_{Alice-Bob}, R_2, \text{Bob})$; then Alice responds with $f(K_{Alice-Bob}, R_1, \text{Alice})$

Mutual Authentication Using Public Keys

- Suppose Alice and Bob know each other's public keys
- How can we perform mutual authentication?
  - ❑as in fig. below
- Alternatively, Alice sends "I'm Alice,R_2" to Bob; then Bob responds with R_2 signed with his key and R_1; then Alice responds with R_1 signed with her key
- Are these protocols vulnerable to reflection attack?
  - ❑No

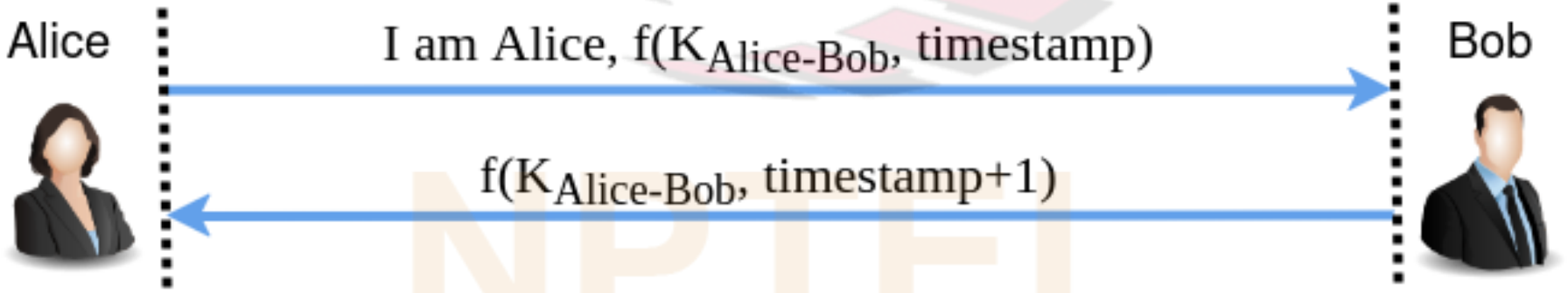


Mutual Authentication Using Synchronized Clocks

- Recall: above protocols use *three* messages
- Suppose Alice and Bob have a shared symmetric key K_{A-B} and approximately synchronized clocks
- Can we modify above protocol (based on shared symmetric key) so that it uses only *two* messages instead of three?

Mutual Authentication Using Synchronized Clocks (contd.)

- Mutual auth. can be performed as in fig. below
- Note that Bob sends $f(K_{Alice-Bob}, \text{timestamp} + 1)$ instead of $f(K_{Alice-Bob}, \text{timestamp})$
- Alternative schemes:
  - Alice sends $f(K_{Alice-Bob}, \text{timestamp}, \text{Alice})$ and Bob sends $f(K_{Alice-Bob}, \text{timestamp}, \text{Bob})$
  - Alice and Bob use same timestamp, but different keys
- Actions similar to the ones we discussed for one-way auth. using timestamps must be performed (e.g., Bob must check that the timestamp used by Alice within clock skew, Bob must remember timestamps used by Alice in past until they expire)



Mutual Authentication Using Synchronized Clocks (contd.)

- Suppose mutual auth. is performed as in fig. below
- An intruder, Trudy, can later use $f(K_{Alice-Bob}, \text{timestamp} + 1)$ to authenticate herself as Alice to Bob or to another server that shares the same key, $K_{Alice-Bob}$, with Alice
- Defence against above attack:
  - Alice sends $f(K_{Alice-Bob}, \text{timestamp}, \text{Bob})$ and Bob sends $f(K_{Alice-Bob}, \text{timestamp}, \text{Alice})$; also, Alice does not use "timestamp" again to authenticate to any server