Part III
Protocols
Nov 3

Protocol

- Human protocols —the rules followed in human interactions
  - Example: Asking a question in class
- Networking protocols —rules followed in networked communication systems
  - Examples: HTTP, FTP, etc.
- Security protocol —the (communication) rules followed in a security application
  - Examples: SSL, IPSec, Kerberos, etc.

Protocols

- Protocol flaws can be very subtle
- Several well-known security protocols have serious flaws
  - Including IPSec, GSM and WEP
- Common to find implementation errors
  - Such as IE implementation of SSL
- Difficult to get protocols right…

Ideal Security Protocol

??? properties ???

Ideal Security Protocol

- Satisfies security requirements
  - Requirements must be precise
- Efficient
  - Minimize computational requirement —in particular, costly public key operations
  - Minimize delays/bandwidth
- Not fragile
  - Must work when attacker tries to break it
  - Works even if environment changes
- Easy to use and implement, flexible, etc.
- *Very difficult to satisfy all of these!*

Chapter 9
Simple Security Protocols

Secure Entry to NSA

1. Insert badge into reader
2. Enter PIN
3. Correct PIN?

   Yes? Enter

   No? Get shot by security guard

ATM Machine Protocol

1. Insert ATM card
2. Enter PIN
3. Correct PIN?

   Yes? Conduct your transaction(s)

   No? Machine eats card

Identify Friend or Foe (IFF)

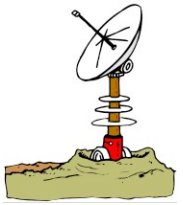
Russian
MIG

Angola

SAAF
Impala

2. E(N,K)

1. N



Namibia

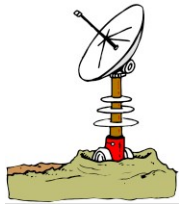
MIG in the Middle

SAAF
Impala

3. N

4. E(N,K)

2. N



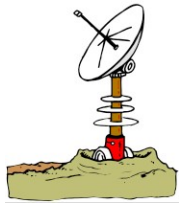
Angola

5. E(N,K)

Russian
MiG

6. E(N,K)

1. N



Namibia

Authentication Protocols

Authentication

- Alice must prove her identity to Bob
  - Alice and Bob can be humans or computers
- May also require Bob to prove he's Bob (mutual authentication)
- May also need to establish a session key
- May have other requirements, such as
  - Use only public keys
  - Use only symmetric keys
  - Use only a hash function
  - Anonymity, plausible deniability, etc., etc.

Authentication

- Authentication on a stand-alone computer is relatively simple
  - ???

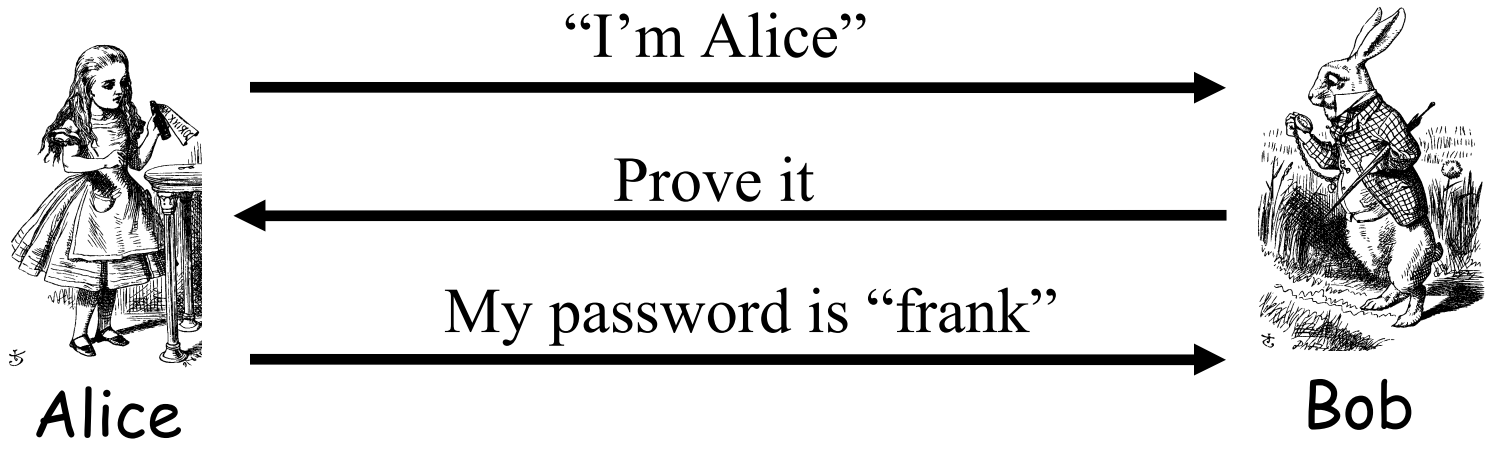
Authentication

- Authentication on a stand-alone computer is relatively simple (hashing, salting, ...)
  - "Secure path" is the primary issue
  - Main concern is an attack on authentication software (we discuss software attacks later)
- Authentication over a network is much more complex
  - ???

Authentication

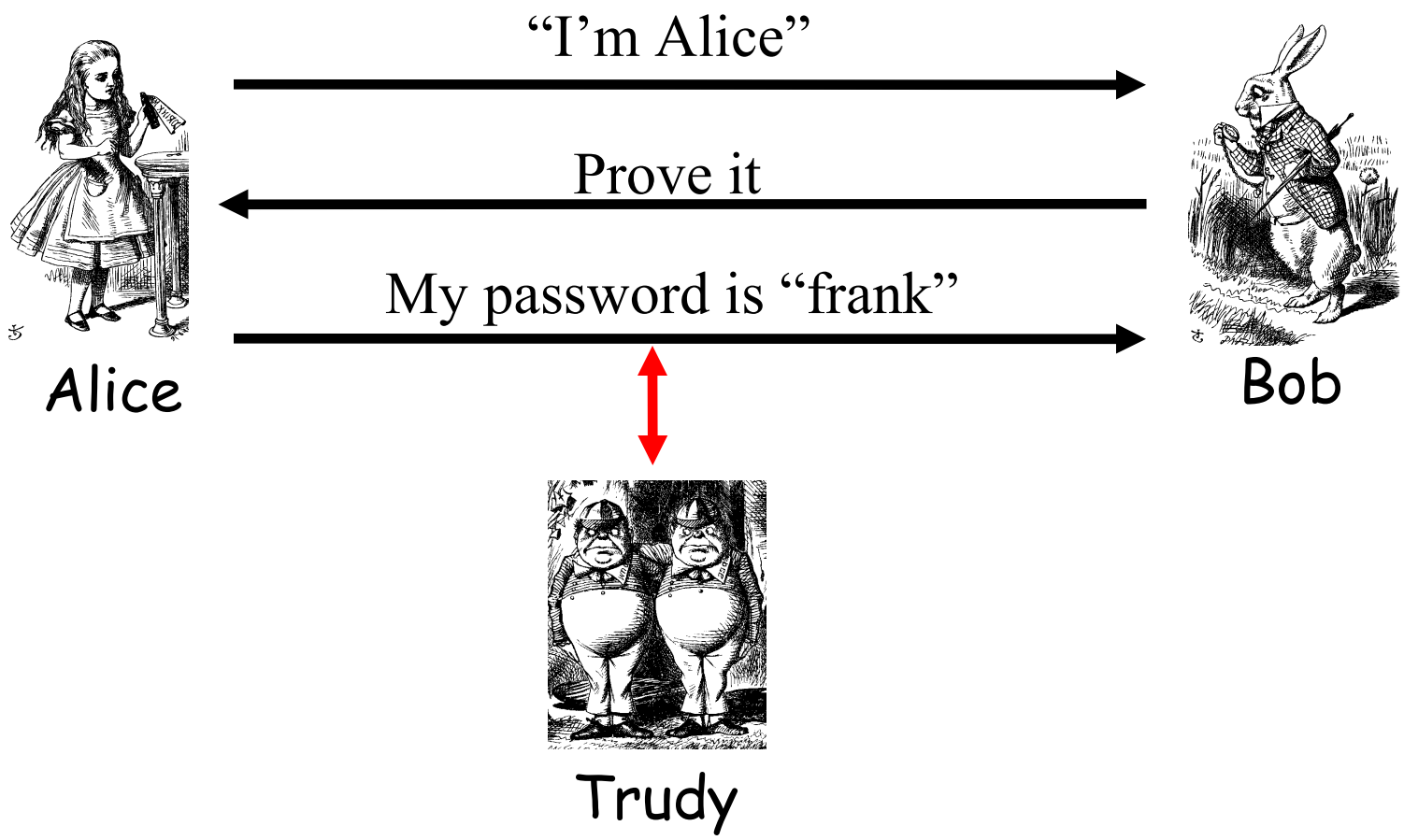
- Authentication on a stand-alone computer is relatively simple (hashing, salting, ...)
  - "Secure path" is the primary issue
  - Main concern is an attack on authentication software (we discuss software attacks later)
- Authentication over a network is much more complex
  - Attacker can passively observe messages
  - Attacker can replay messages
  - Active attacks may be possible (insert, delete, change messages)

Simple Authentication

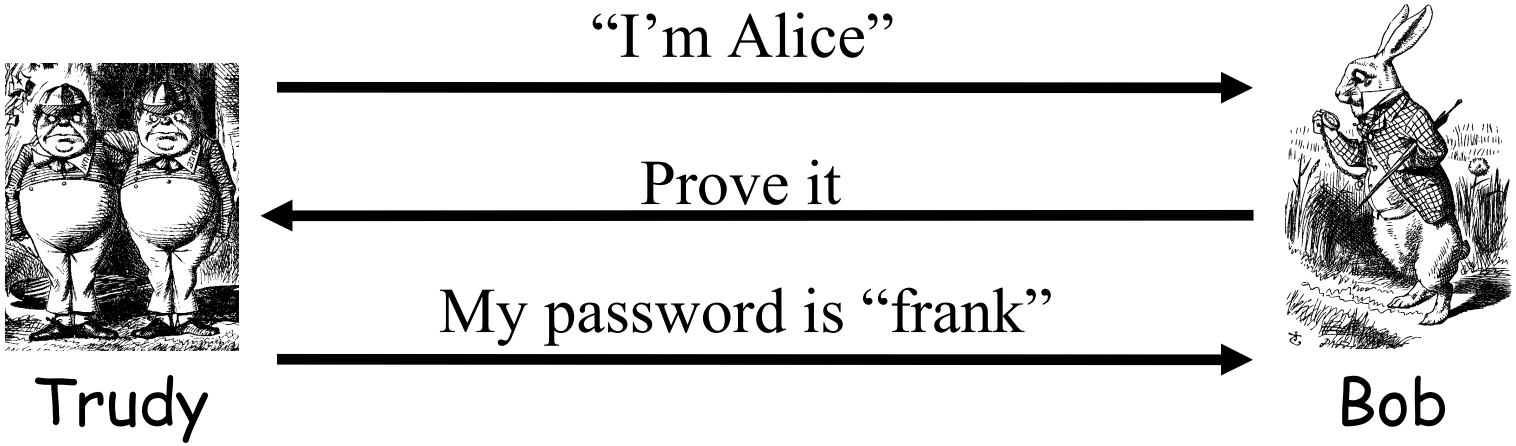


- Simple and may be OK for standalone system
- But insecure for networked system
  - Subject to a replay attack (next 2 slides)
  - Bob must know Alice's password

Authentication Attack



Authentication Attack



- This is a **replay** attack
- How can we prevent a replay?

Simple Authentication

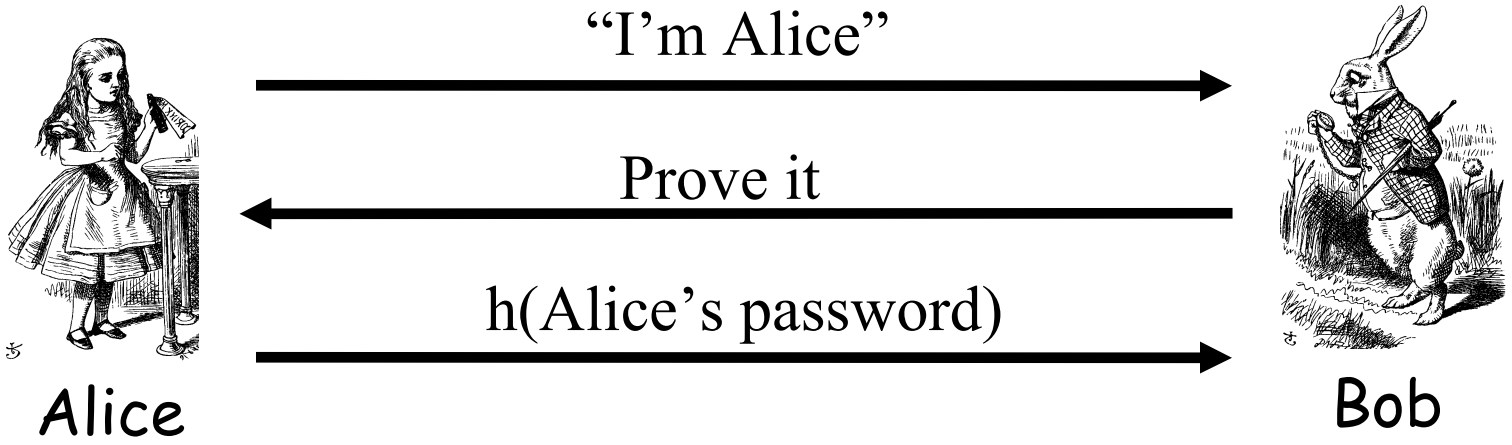
Alice

I'm Alice, My password is "frank"

Bob

- More efficient…
- But same problem as previous version

Better Authentication

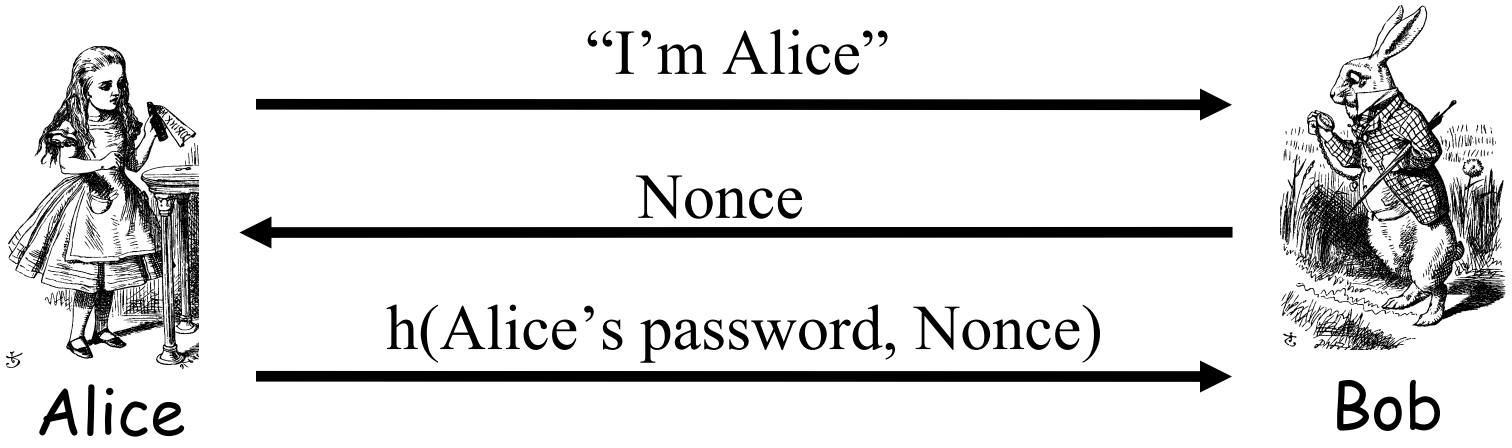


- Better since it hides Alice's password
  - From both Bob and attackers
- But still subject to replay

Challenge-Response

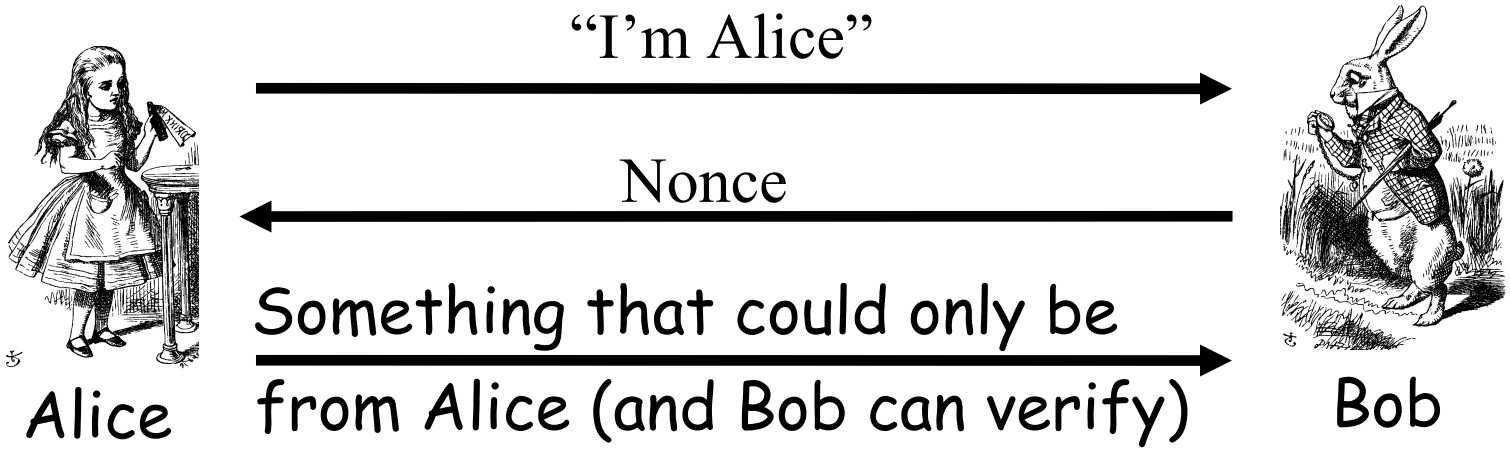
- To prevent replay, challenge-response used
- Suppose Bob wants to authenticate Alice
  - Challenge sent from Bob to Alice
  - Only Alice can provide the correct response
  - Challenge chosen so that replay is not possible
- How to accomplish this?
  - Password is something only Alice should know…
  - For freshness, a "number used once" or **nonce**

Challenge-Response



- Nonce is the **challenge**
- The hash is the **response**
- Nonce prevents replay, insures freshness
- Password is something Alice knows (Trudy does not)
- Note that Bob must know Alice's password

Challenge-Response



- What can we use to achieve this?
- Hashed pwd works, crypto might be better

Symmetric Key Notation

- Encrypt plaintext P with key K

  $C = E(P,K)$

- Decrypt ciphertext C with key K

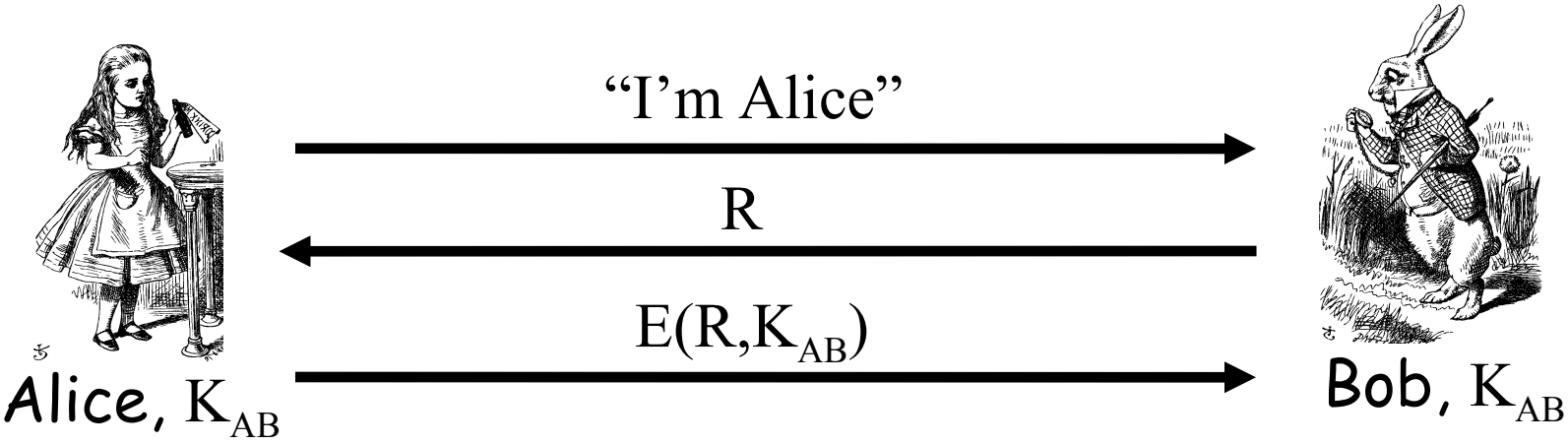
  $P = D(C,K)$

- Here, we are concerned with attacks on **protocols**, not directly on the crypto
- We assume that crypto algorithm is secure

Symmetric Key Authentication

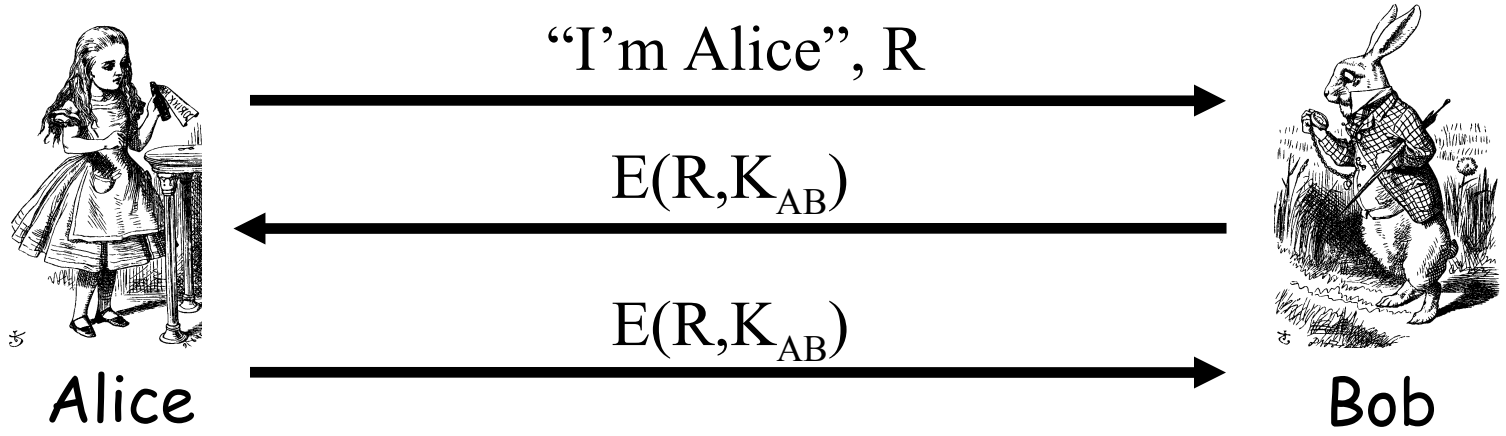
- Alice and Bob share symmetric key K_{AB}
- Key K_{AB} known only to Alice and Bob
- Authenticate by proving knowledge of shared symmetric key
- How to accomplish this?
  - Must not reveal key
  - Must not allow replay attack

Authentication with Symmetric Key



- Secure method for Bob to authenticate Alice
- Alice does not authenticate Bob
- Can we achieve mutual authentication?

Mutual Authentication?

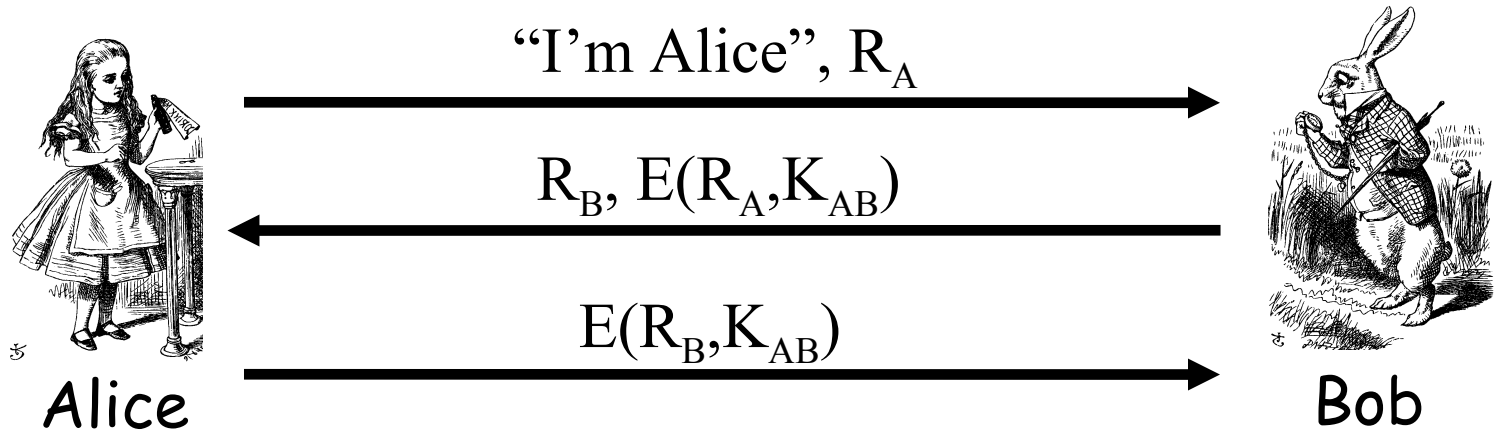


- What's wrong with this picture?
- "Alice" could be Trudy (or anybody else)!

Mutual Authentication

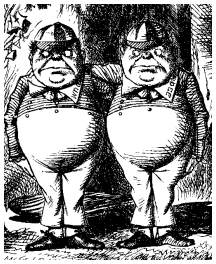
- Since we have a secure one-way authentication protocol…
- The obvious thing to do is to use the protocol twice
  - Once for Bob to authenticate Alice
  - Once for Alice to authenticate Bob
- This has to work…

Mutual Authentication



- This provides mutual authentication…
- …or does it? See the next slide

Mutual Authentication Attack



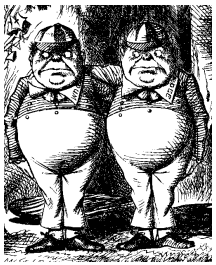
Trudy

1. "I'm Alice", R_A

2. R_B, $E(R_A,K_{AB})$

5. $E(R_B,K_{AB})$

Bob



Trudy

3. "I'm Alice", R_B

4. R_C, $E(R_B,K_{AB})$

Bob

Mutual Authentication

- Our one-way authentication protocol **not** secure for mutual authentication
- Protocols are subtle!
- The "obvious" thing may not be secure
- Also, if assumptions or environment changes, protocol may not work
  - This is a common source of security failure
  - For example, Internet protocols