SECURITY HANDSHAKE PITFALLS

Dr. Nguyen Tuan Nam
nam.nguyen@brumob.com

SECURITY HANDSHAKE PITFALLS

Security in communications always includes

- An initial authentication handshake
- Sometimes, in addition, integrity protection and/or encryption of data

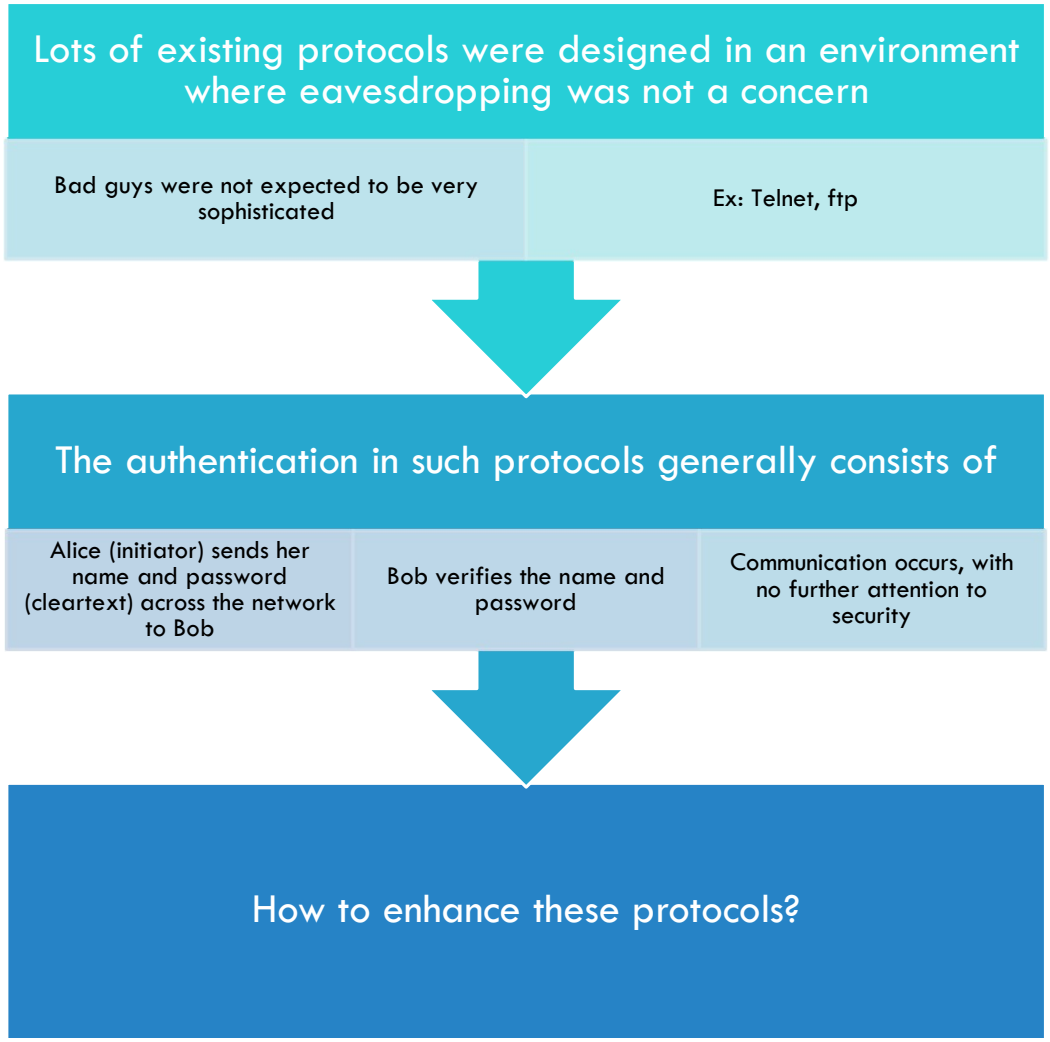
Minor variants of secure protocols can have security holes

As a matter of fact, many deployed protocols have been designed with security flaws

There is no one "best" protocol

- Some threats are more likely in some situations
- Different resources are available in terms of computational power, specialized hardware, money to pay off patent holders

Slightest alteration can introduce security flaws


Lots of existing protocols were designed in an environment where eavesdropping was not a concern
Bad guys were not expected to be very sophisticated
Ex: Telnet, ftp
The authentication in such protocols generally consists of
Alice (initiator) sends her name and password (cleartext) across the network to Bob
Bob verifies the name and password
Communication occurs, with no further attention to security
How to enhance these protocols?

LOGIN ONLY

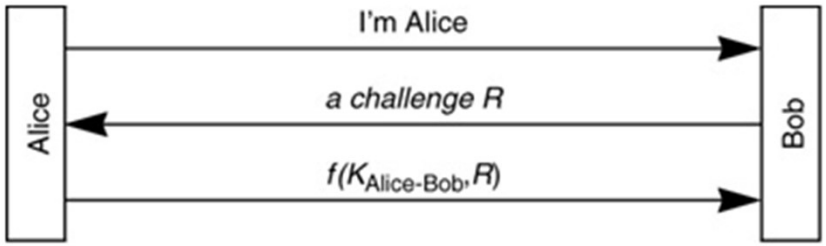
SHARED SECRET

Authentication is not mutual

Trudy can hijack the conversation after the initial exchange

Eavesdropper could mount off-line password guessing attack

Someone who reads the DB at Bob can later impersonate Alice



MINOR VARIANT

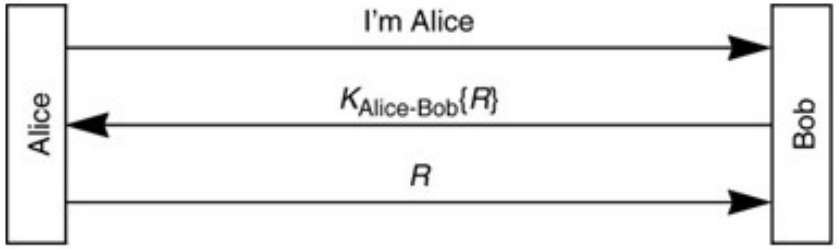
This protocol requires reversible cryptography

- Hash function is faster but cannot be used

Dictionary attack if R is a recognizable quantity

R must be limited lifetime to foil the replaying attack

- Random number concatenated with a timestamp



ANOTHER VARIANT

Requires Bob and Alice have reasonably synchronized clocks

Can be added very easily to a protocol designed for sending cleartext passwords

- Adds no additional messages

More efficient

- Saving messages
- No need to keep any volatile state

Eavesdropper can use Alice's transmitted $K_{Alice\text{-}Bob}\{timestamp\}$ to impersonate Alice, if done within the acceptable clock skew

Multiple servers for which Alice uses the same secret → impersonate Alice to a different server. How to fix it?

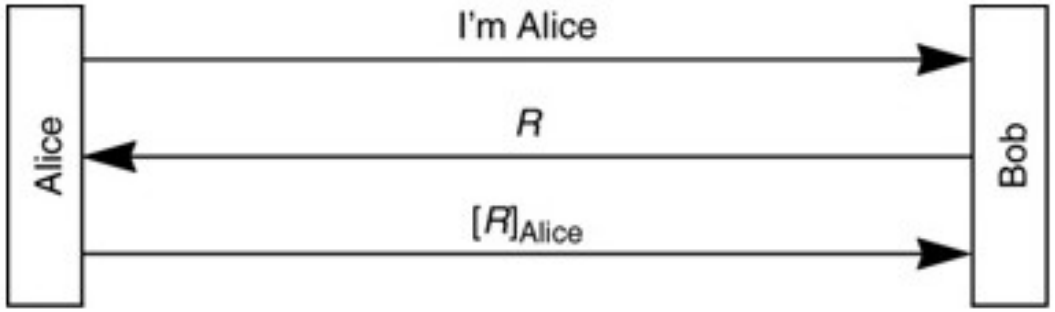
Trudy convinces Bob to set his clock back

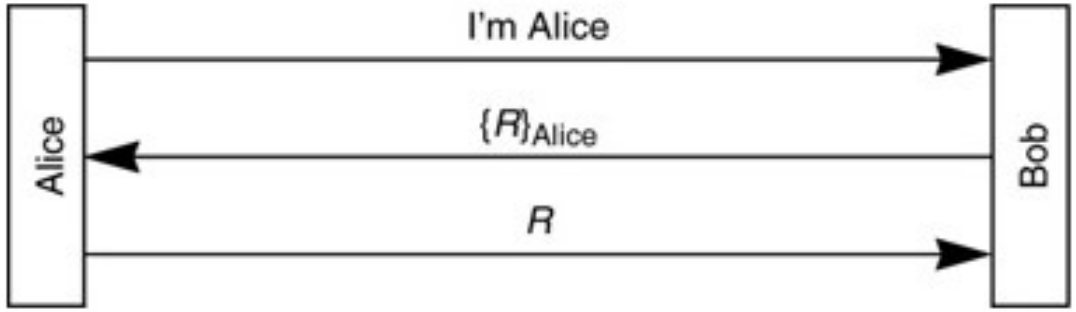
Requires security handshake for managing clock setting

Reversible encryption?

ANOTHER VARIANT USING HASH

ONE-WAY PUBLIC KEY





Any problem?

PROBLEM

Problem

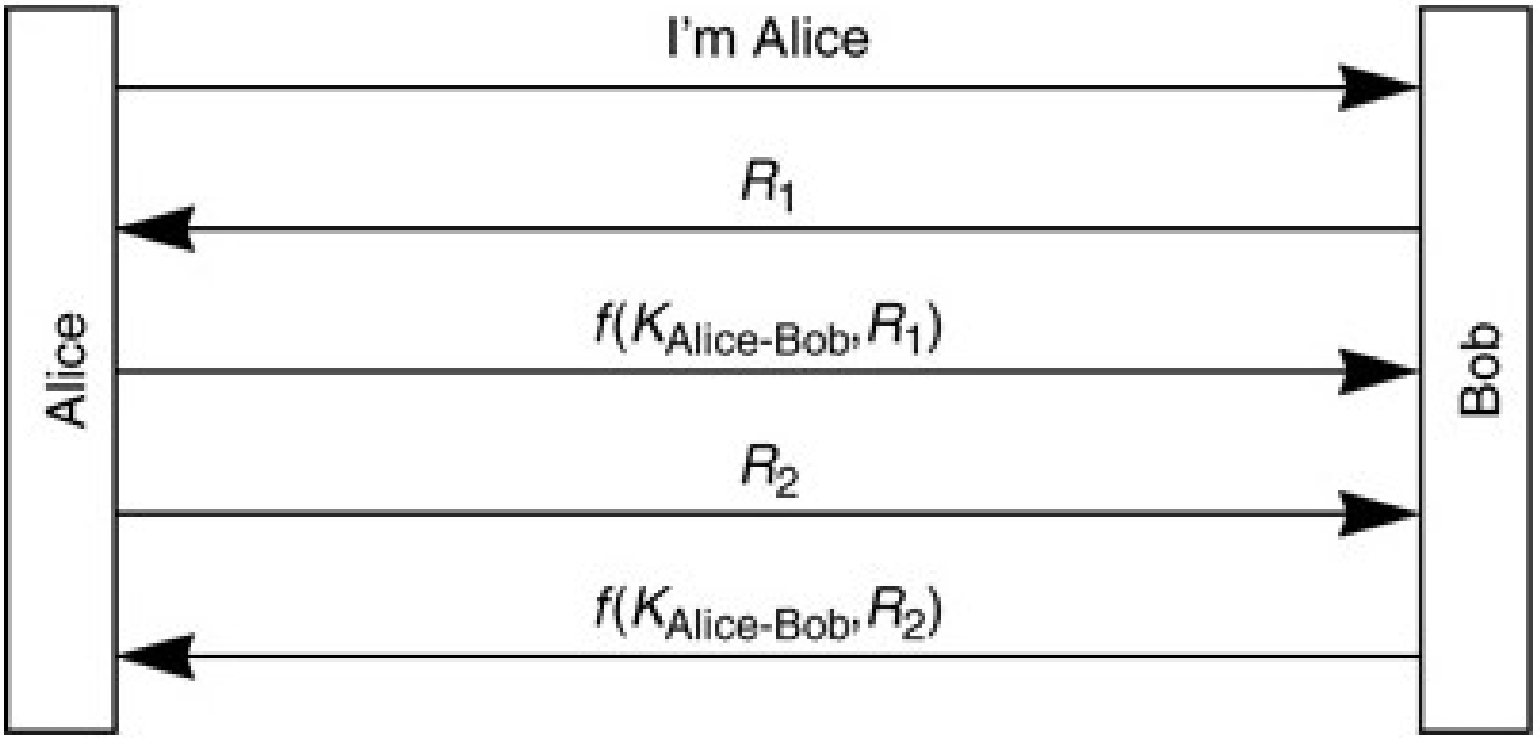
- Trick someone into signing something
- Trick someone to decrypt a ciphertext

How to solve it

- Should NOT use the same key for 2 different purposes unless the designs are coordinated so that attacker cannot use 1 protocol to break another
- Adding type field → PKCS standards

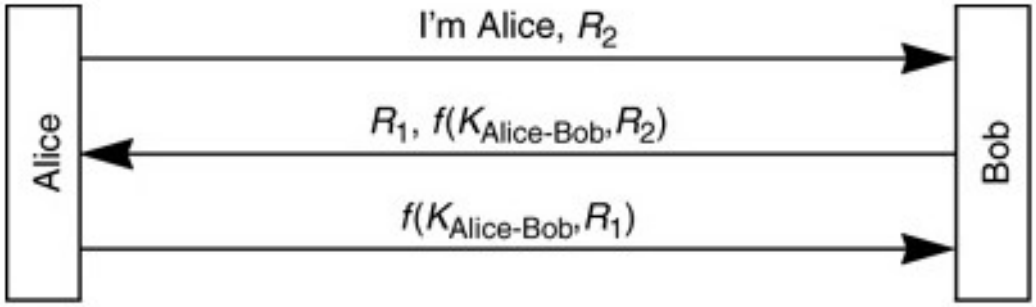
Chilling implication

- Design several schemes where each is independently secure, but when you use more than one → may have a problem
- New protocol whose deployment would compromise the security of existing schemes



MUTUAL AUTHENTICATION

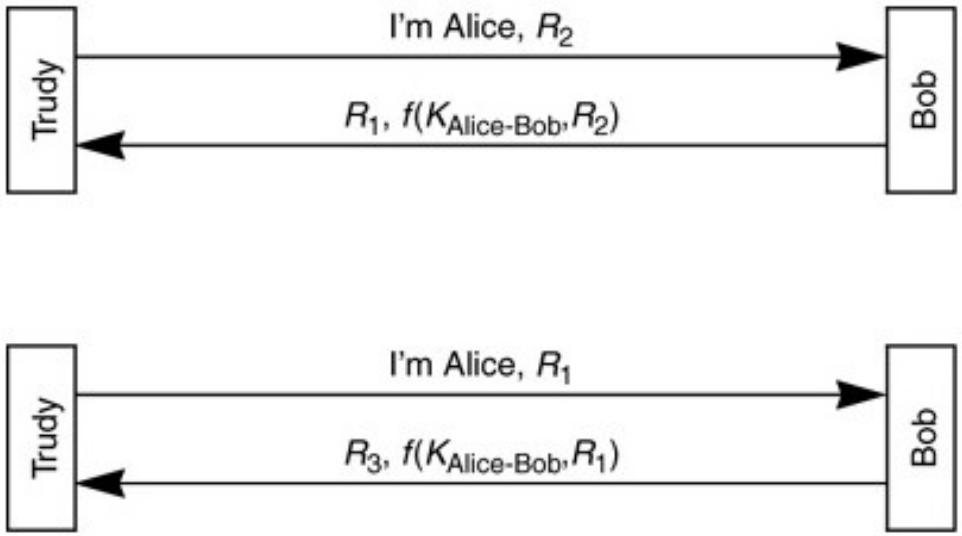
OPTIMIZED VERSION



Any problem?

- **Reflection attack**
- **Password guessing**

REFLECTION ATTACK



Open multiple simultaneous connection to the same server

Multiple servers with the same secret for Alice

HOW TO FIX IT?

Different keys to authenticate Alice and Bob

Different challenges

- Odd vs. even challenge

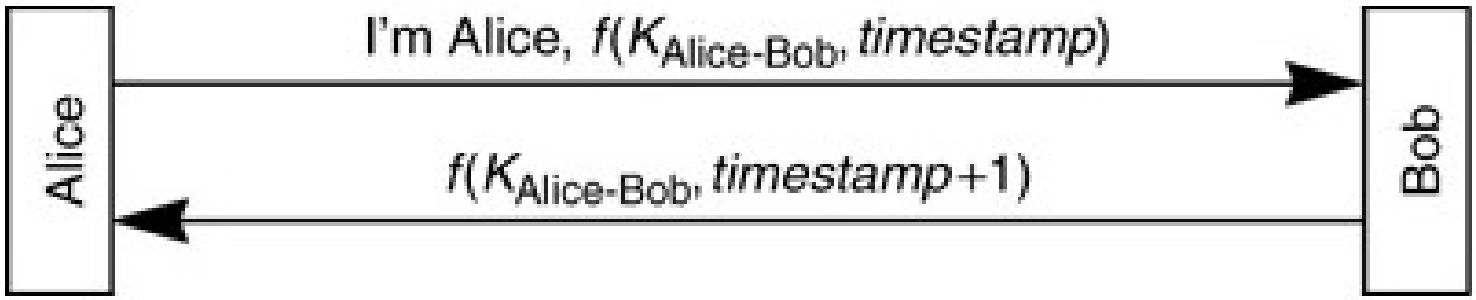
PASSWORD GUESSING

Trudy can have a pair of plaintext and ciphertext

TIMESTAMPS

Timestamp + 1 is not the best choice

- Trudy using $K_{Alice\text{-}Bob}\{timestamp + 1\}$ to impersonate Alice



INTEGRITY/ENCRYPTION FOR DATA

- It is desirable for Alice and Bob to establish a shared secret per-conversation key, known as session key to be used for integrity protection and encryption

SHARED SECRET

- Take shared secret $K_{Alice-Bob}$, modify it
- Encrypt challenge R using the modified $K_{Alice-Bob}$
- Use the result as the session key

TWO-WAY PUBLIC KEY BASED AUTHENTICATION

Option 1

- Alice chooses a random number R
- Encrypt it with Bob's public key
- Send $\{R\}_{Bob}$ to Bob
- Security flaw?

Option 2

- Alice, in addition to encrypting R with Bob's public key, sign the result

PRIVACY AND INTEGRITY

Currently no standard algorithm for providing both privacy and integrity with a single key and a single cryptographic pass over the data

Plausible solutions

- Develop 2 keys in the authentication exchange and do the 2 operations independently
- Make a second key by modifying the 1st
- Use different cryptographic algorithms so a common key is irrelevant