SECURE COMMUNICATIONS

CRYPTOGRAPHIC PROTOCOLS (Forward Secrecy)

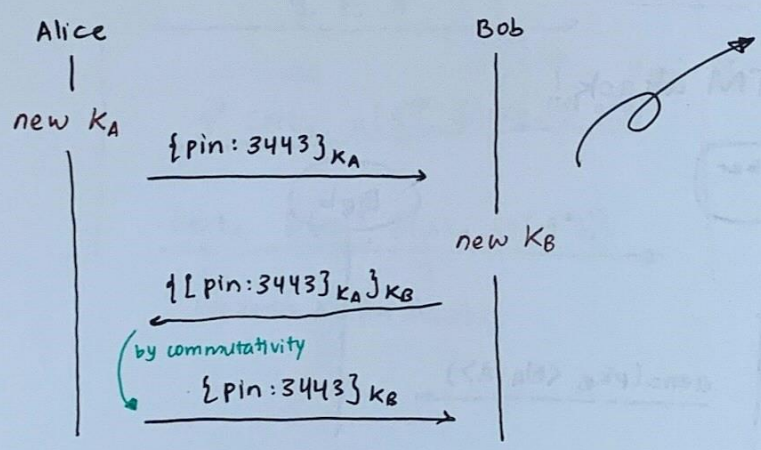
→ are distributed programs relying on cryptographic primitives and whose goal is the establishment of 'secure' communications

→ Many exploitable errors are not due to design errors in the primitives, but to the way they are used (i.e. Bad protocol design)

LOGICAL ATTACK

e.g. Assume a commutative symmetric encryption scheme

$$\{\{m\}_{K_1}\}_{K_2} = \{\{m\}_{K_2}\}_{K_1}$$



Prone to attacks bc. there is no authentication!
↓
An attacker can pretend to be Bob.

AUTHENTICATION & KEY-AGREEMENT PROTOCOLS

1) Long-term keys shld be used as little as possible to reduce 'attack surface'

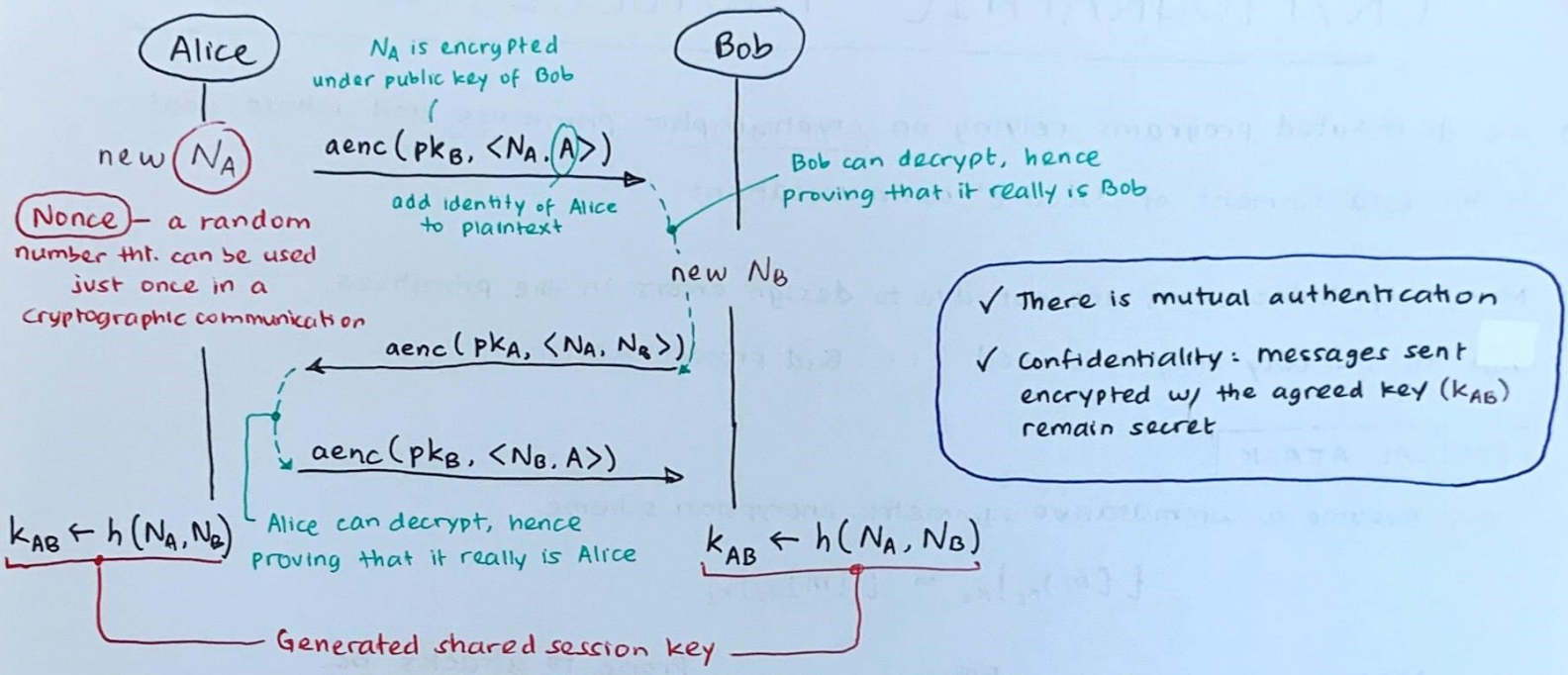
2) The use of a key shld. be restricted to a specific purpose
   e.g. Don't use same RSA key for both encryption & signing

3) Public key algos. tend to be more computationally expensive than symmetric key algos.

→ The result of this is a short-term session key.

NSPK PROTOCOL

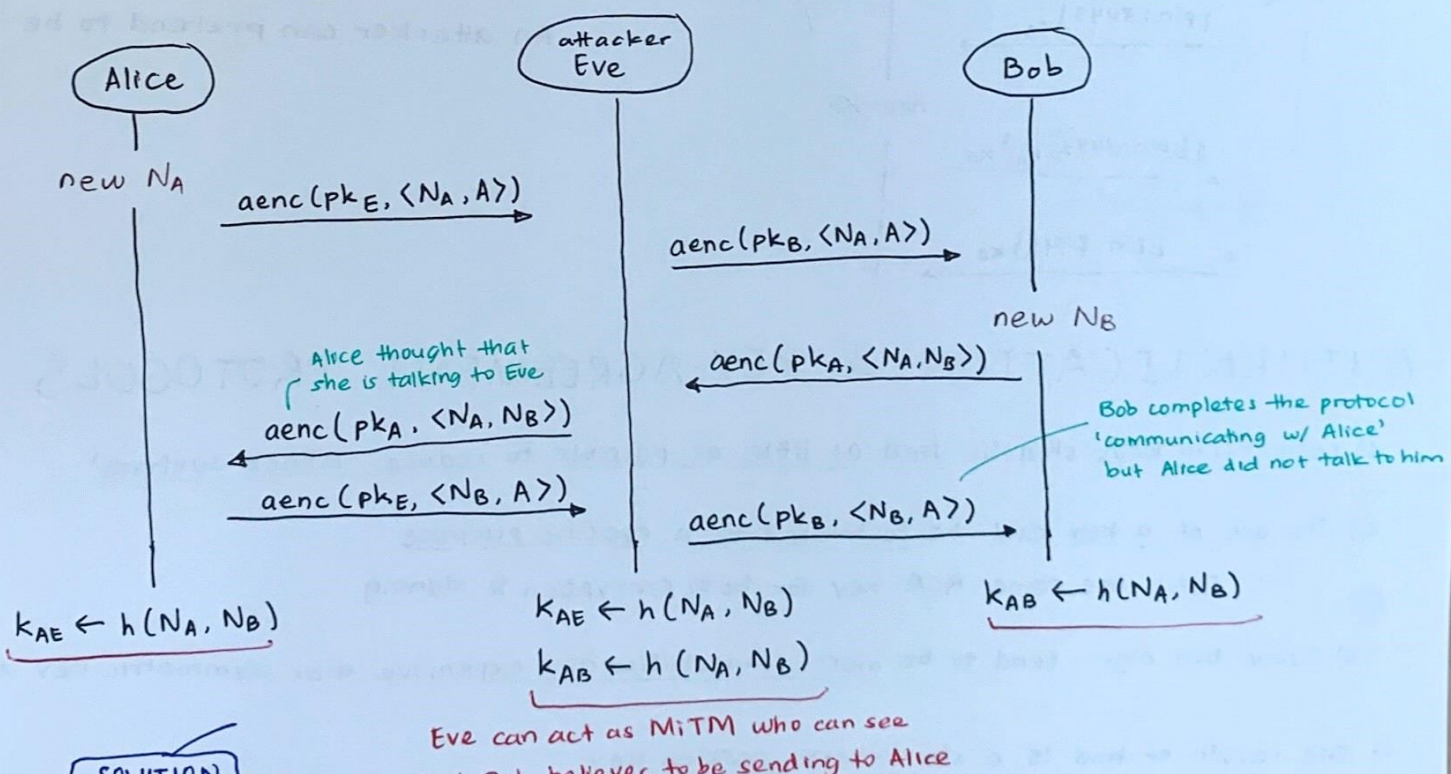
Needham-Schroeder Public-Key



Nonce – a random number that can be used just once in a cryptographic communication

- ✓ There is mutual authentication
- ✓ Confidentiality: messages sent encrypted w/ the agreed key (k_{AB}) remain secret

HOWEVER... vulnerable to MiTM attack!



Eve can act as MiTM who can see what Bob believes to be sending to Alice

SOLUTION

NSL PROTOCOL

→ Add Bob's identity: 2nd STEP: $aenc(pk_A, \langle N_A, \langle N_B, B \rangle \rangle)$

→ The NSL protocol is secure against an attacker that controls the network

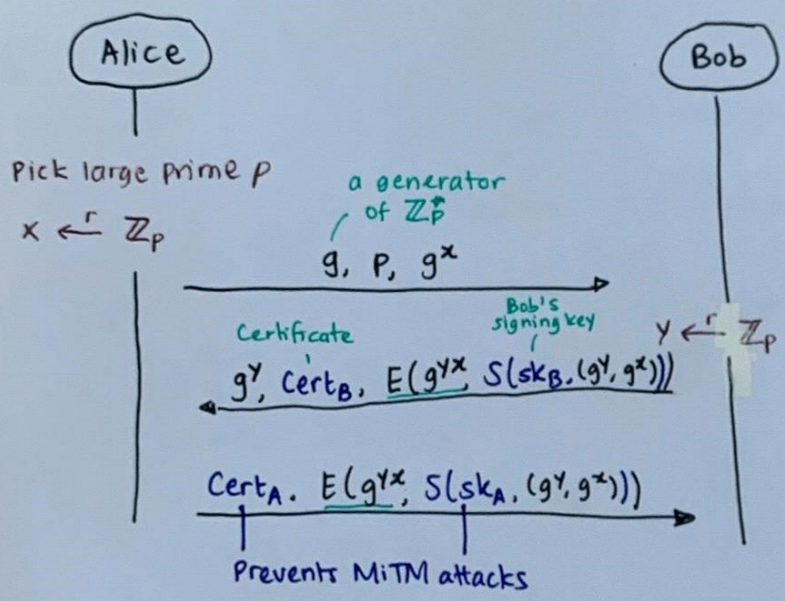
BUT... what if Alice's and Bob's priv. keys get compromised?

Can we still protect confidentiality?

FORWARD SECRECY

→ A protocol ensures **forward secrecy** if even if long-term keys are compromised, past sessions of the protocol are still kept confidential even if an attacker actively interferred

STATION-TO-STATION PROTOCOL



The StS ensures:
- ✓ mutual authentication
- ✓ key agreement
- ✓ forward secrecy

SSL/TLS RENEGOTIATION

→ Renegotiation is making a new handshake while in the middle of a SSL/TLS connection

→ The incorrect implicit assumption is tht. the client doesn't change through renegotiation

↳ CAN BE EXPLOITED!

→ Renegotiation has priority over application data

→ Renegotiation can take place in the middle of an application layer transaction.