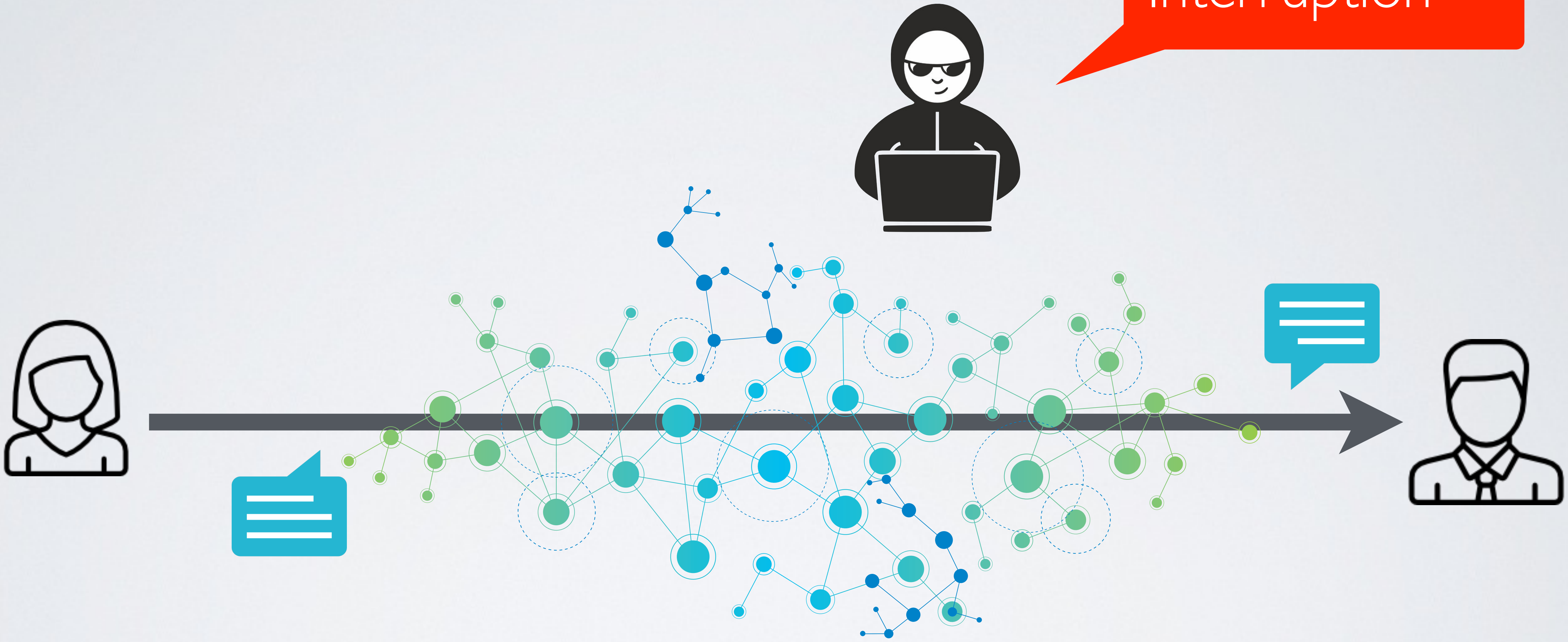
Symmetric Cryptography Protocols

Thierry Sans

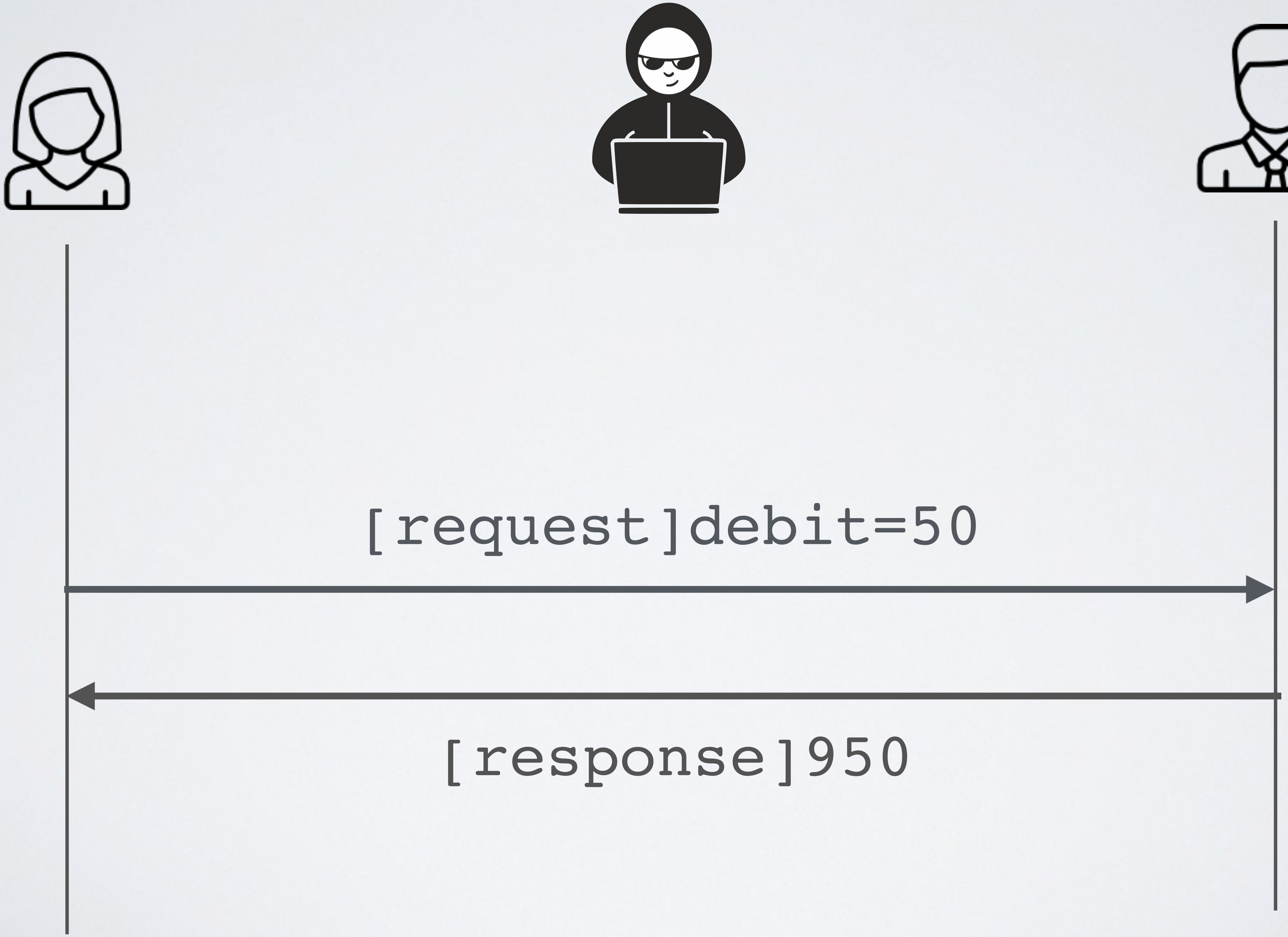
Security goals vs attacker's model

Interception
Modification
Fabrication
Interruption



Let us consider **confidentiality, integrity** ~~**and availability**~~

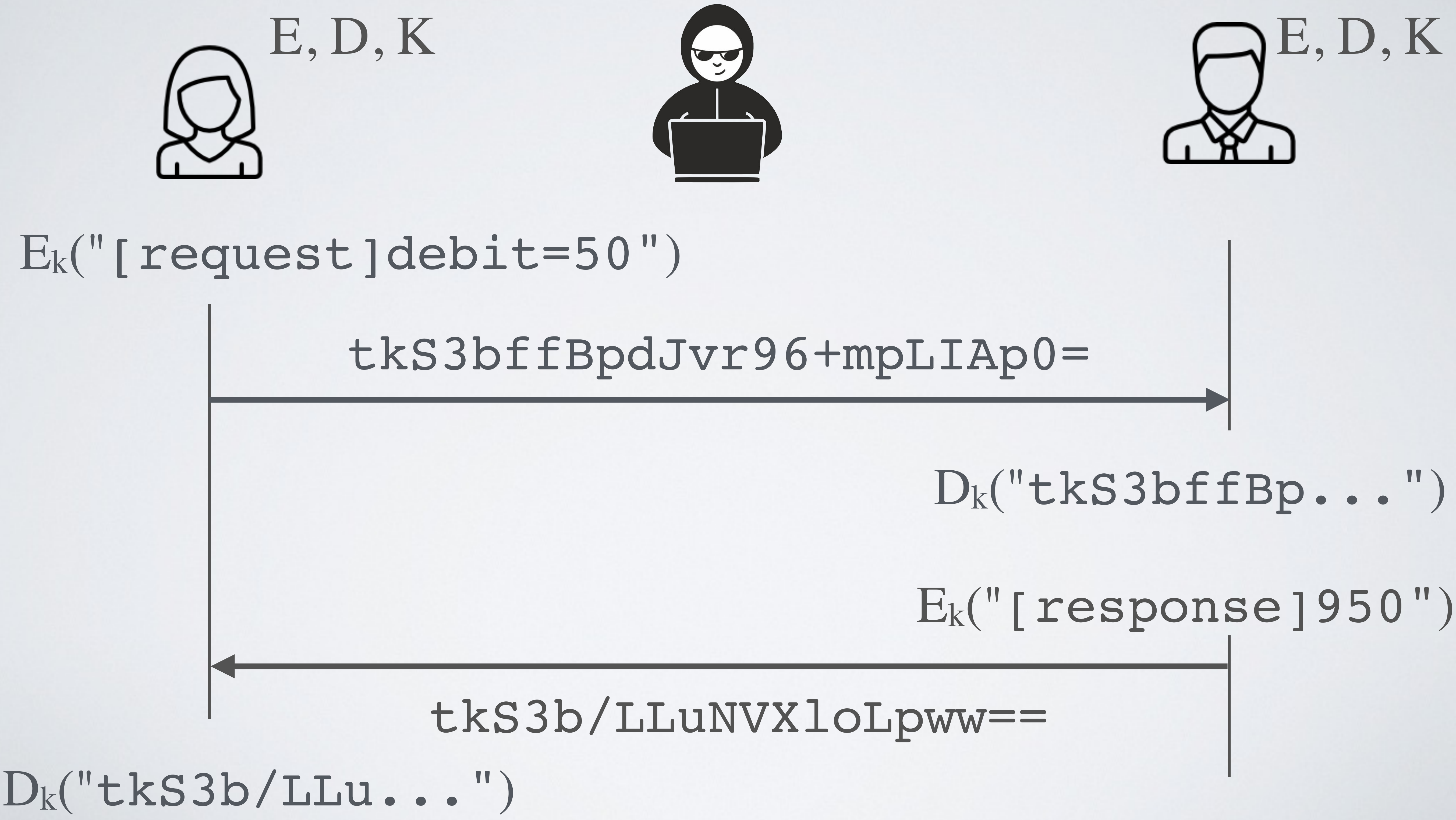
Example



[request]debit=50

[response]950

Ensuring confidentiality with encryption



E, D, K

E, D, K

E_k("[request]debit=50")

tkS3bffBpdJvr96+mpLIAp0=

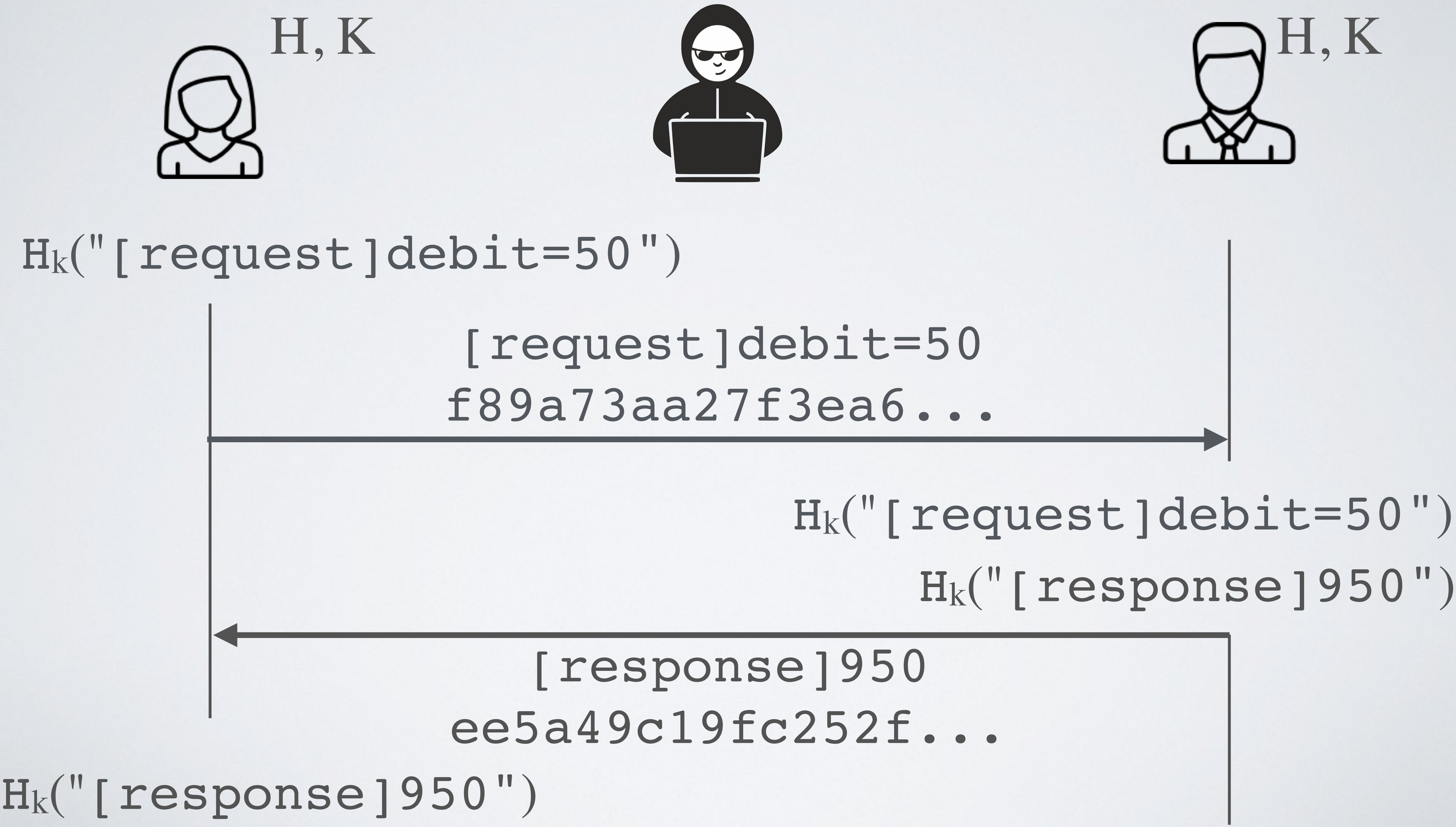
D_k("tkS3bffBp...")

E_k("[response]950")

tkS3b/LLuNVXloLpww==

D_k("tkS3b/LLu...")

Ensuring integrity with an HMAC



H, K

H, K

H_k("[request]debit=50")

[request]debit=50
f89a73aa27f3ea6...

H_k("[request]debit=50")

H_k("[response]950")

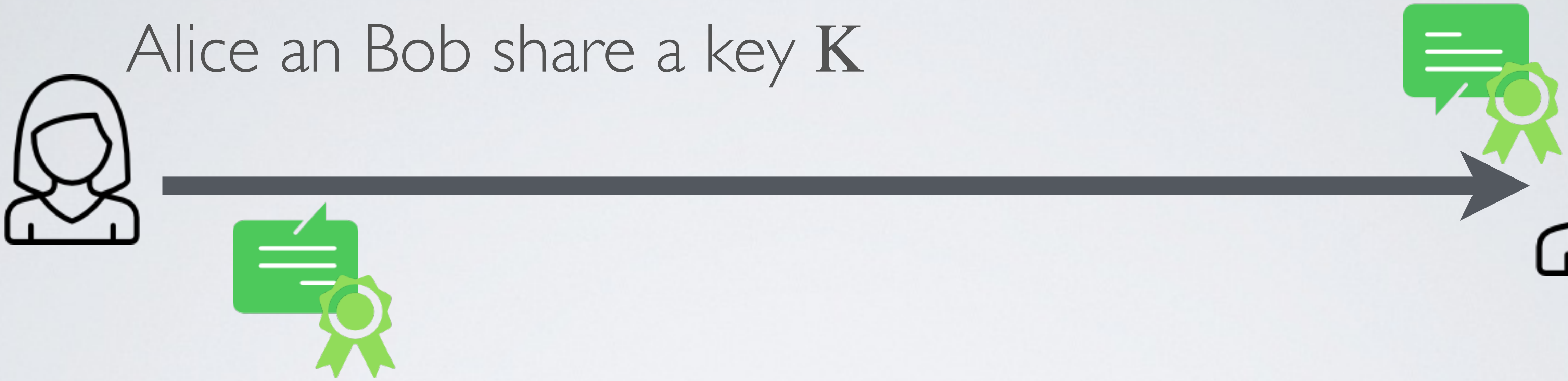
[response]950
ee5a49c19fc252f...

H_k("[response]950")

Security mechanisms

	Encryption	MAC	Authenticated Encryption
Confidentiality	✓	✗	✓
Integrity	✗	✓	✓

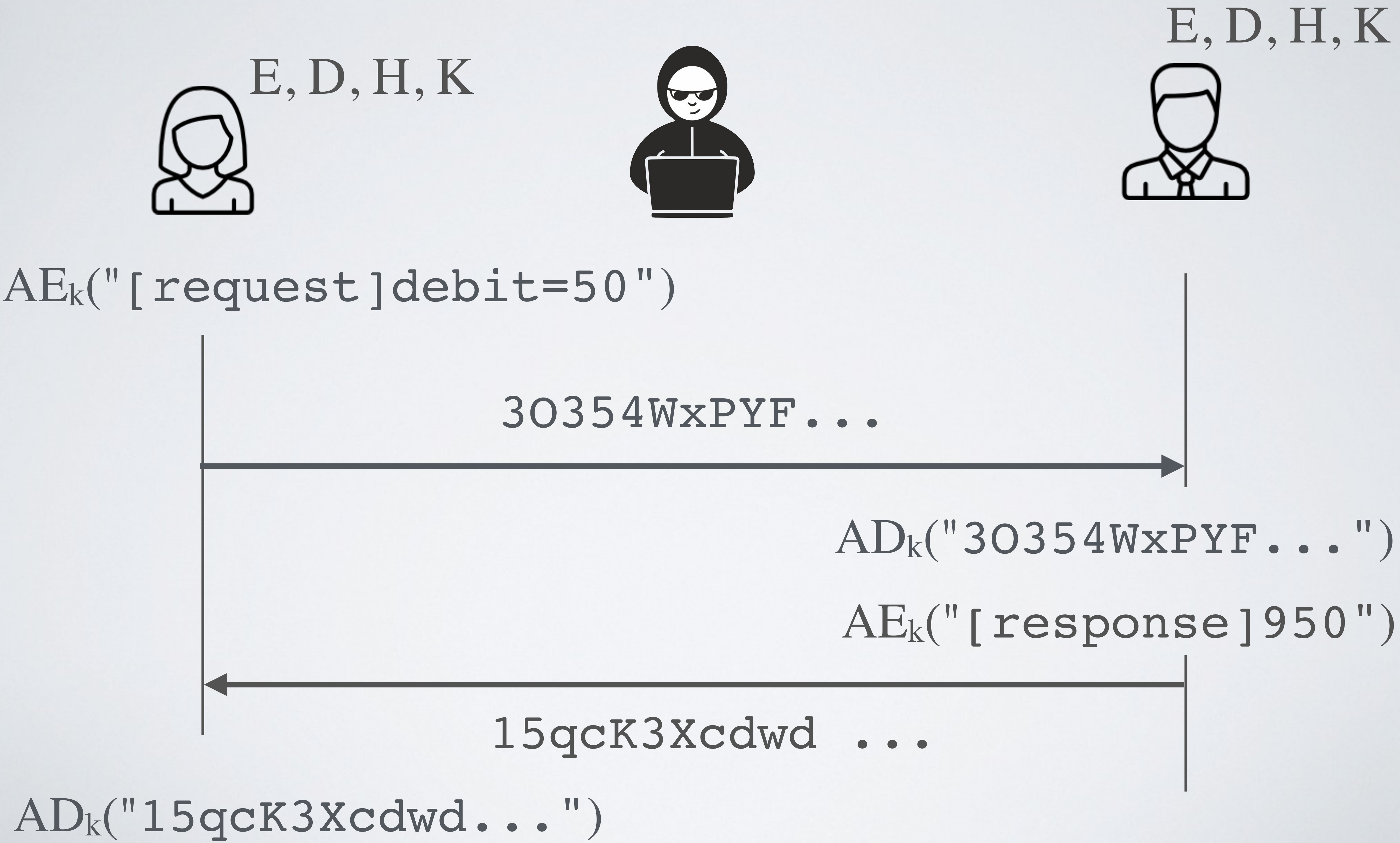
Authenticated Encryption (2013)



Alice an Bob share a key **K**

Encrypt-and-MAC (E&M)	$AE_k(m) = E_K(m) \parallel H_K(m)$	e.g *SSH*
MAC-then-Encrypt (MtE)	$AE_k(m) = E_K(m \parallel H_K(m))$	e.g *SSL*
Encrypt-then-MAC (EtM)	$AE_k(m) = E_K(m) \parallel H_K(E_K(m))$	e.g *IPsec*

Ensuring confidentiality and integrity with Authenticated Encryption



E, D, H, K

E, D, H, K

AE_k("[request]debit=50")

30354WxPYF...

AD_k("30354WxPYF...")

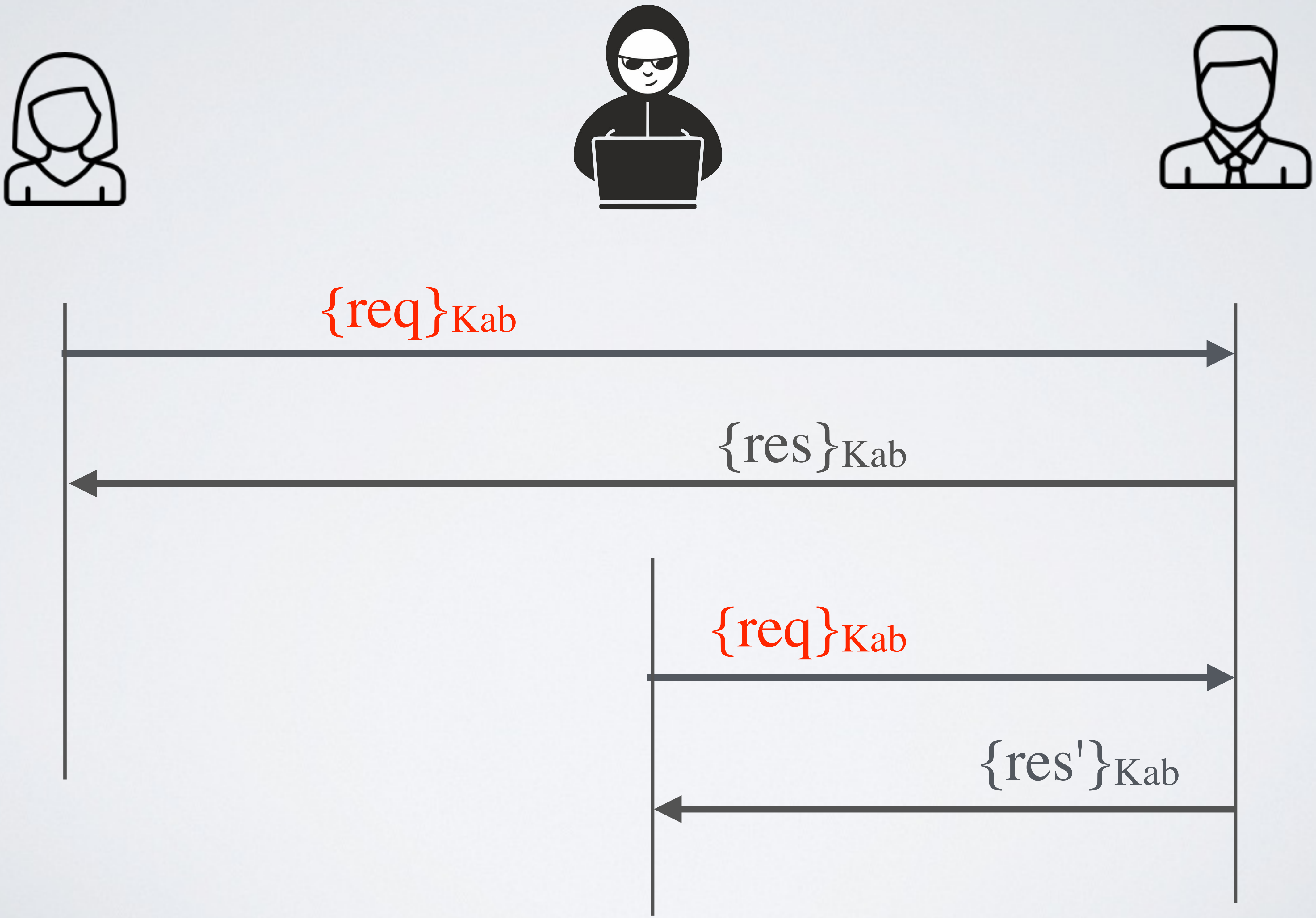
AE_k("[response]950")

15qcK3Xcdwd ...

AD_k("15qcK3Xcdwd...")

Replay attacks

Replay attack



$\{req\}_{Kab}$

$\{res\}_{Kab}$

$\{req\}_{Kab}$

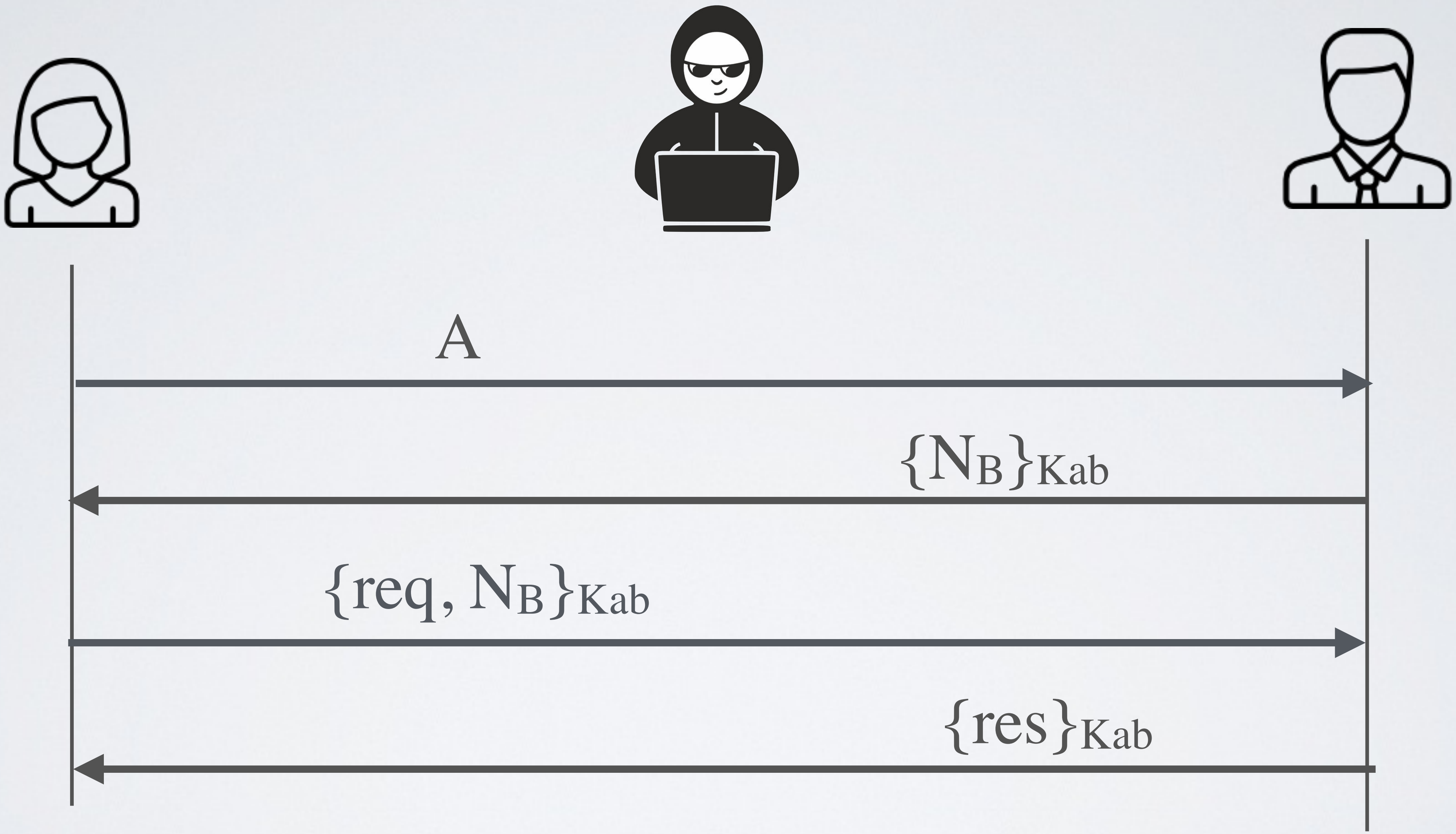
$\{res'\}_{Kab}$

Counter replay attacks

Several solutions:

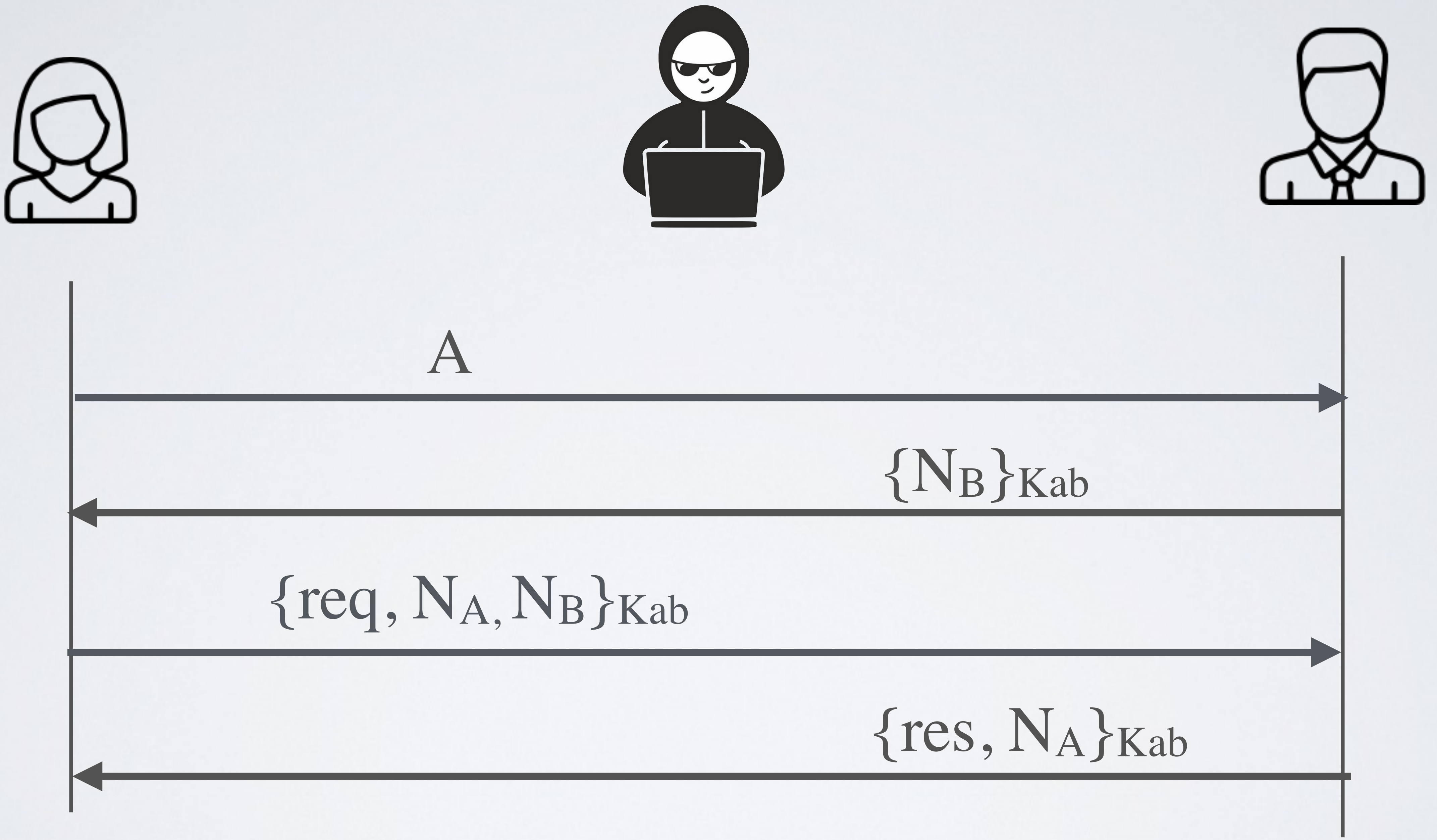
- use a nonce (random number)
- use sequence numbers
- use timestamps
- have fresh key for every transaction
  (key distribution problem)

Defeat replay attack with a nonce (not fully secured)



Replay attack on the response!

Defeat replay attack with a double nonce



A

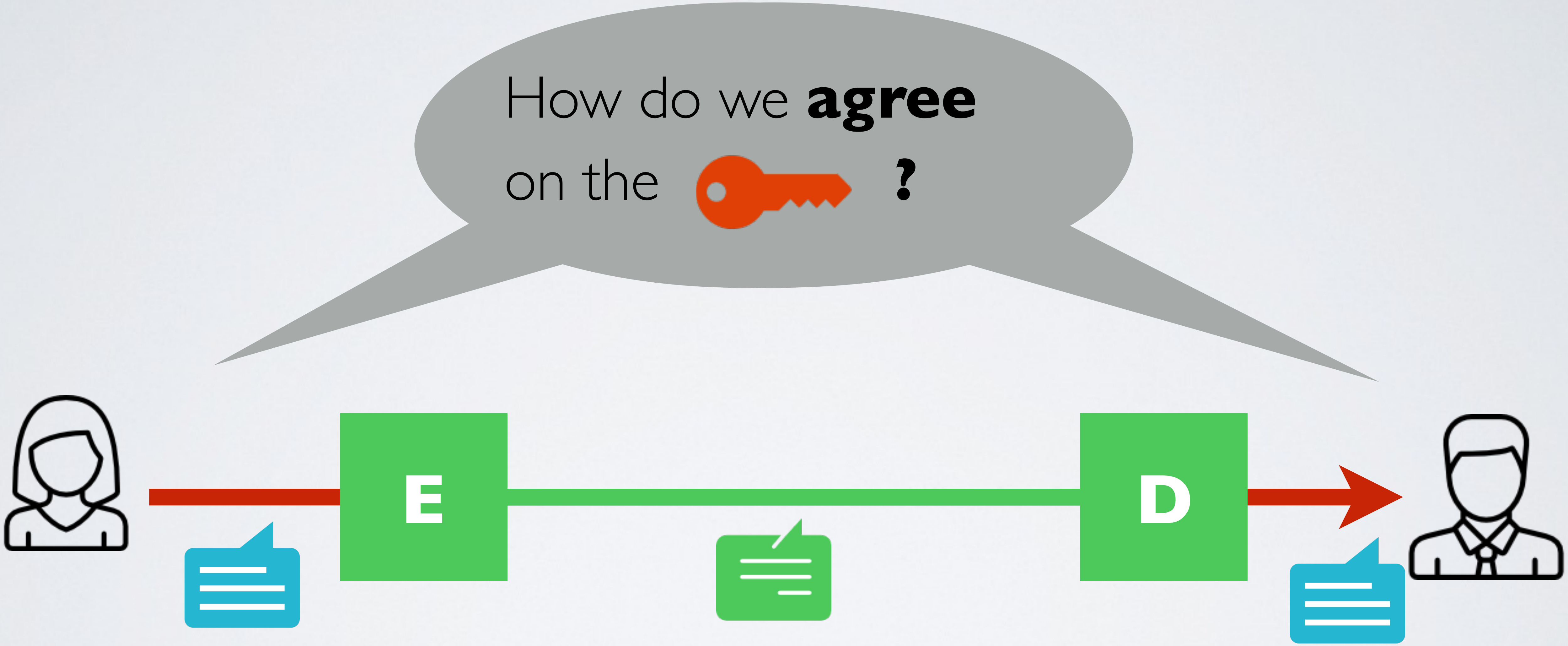
$\{N_B\}_{Kab}$

$\{req, N_A, N_B\}_{Kab}$

$\{res, N_A\}_{Kab}$

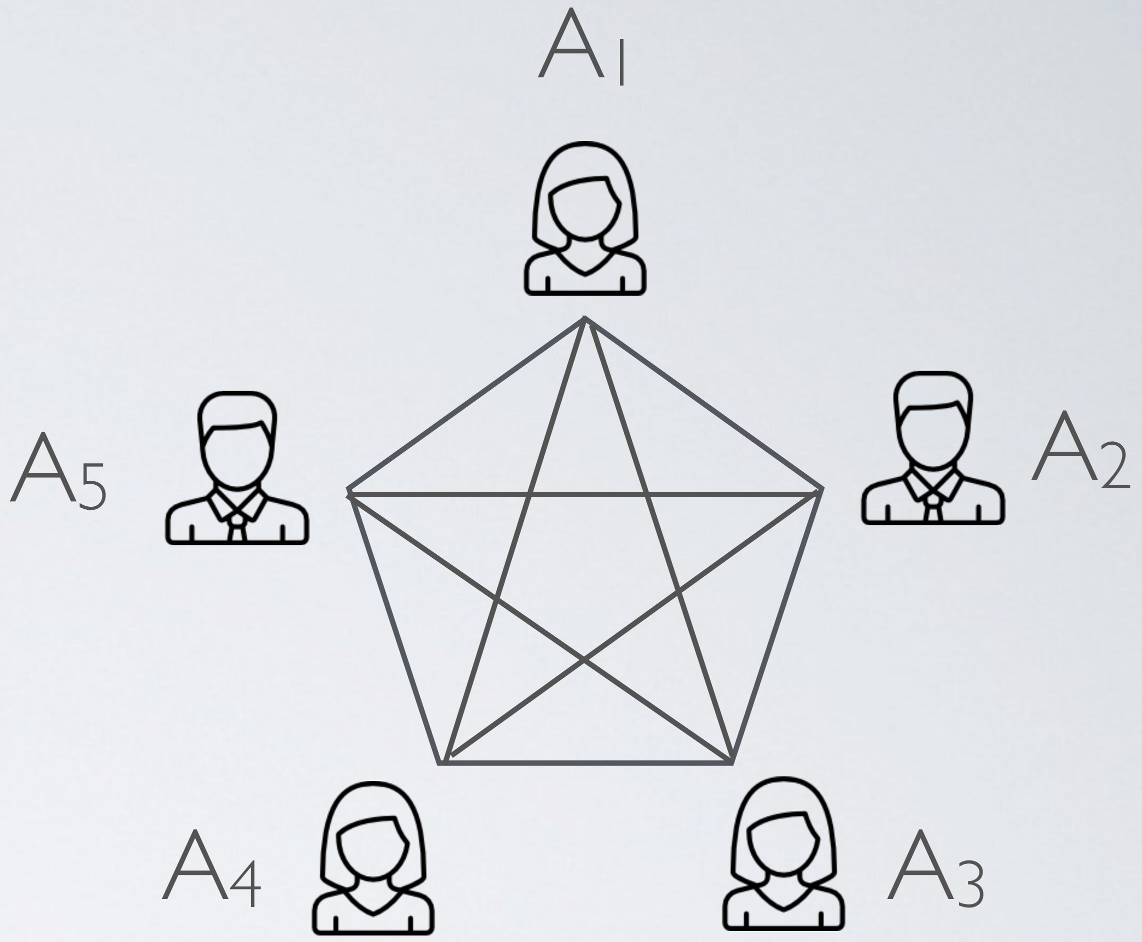
The challenge of key exchange

The big challenge with symmetric cryptosystems?



How do we **agree** on the 🔑 **?**

Naive Key Management

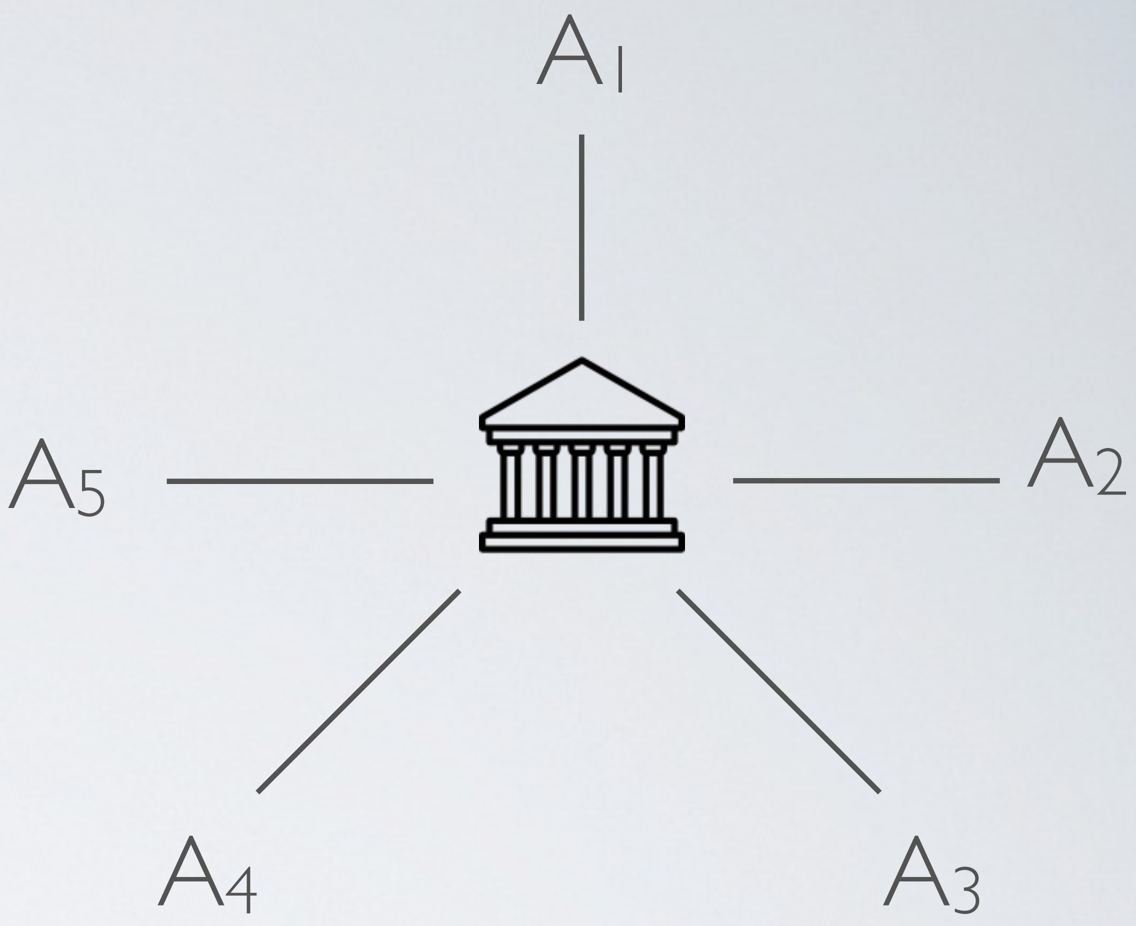


A_1, A_2 … A_5 want to talk

- ➡ Each pair needs a key : **n (n-1) / 2** keys

- ◉ Keys must be exchanged physically using a secure channel

(Better) centralized solution



A_1, A_2 … A_5 can talk to the KDC (Key Distribution Center)

- ➡ When A_i and A_j want to talk, the KDC can generate a new key and distribute it to them

- ◉ We still have n keys to distribute somehow using a secure channel
- ◉ The KDC must be trusted
- ◉ The KDC is a single point of failure

- ➡ The is how *Kerberos* works

The Needham-Shroeder Symmetric Key Protocol

Assumptions

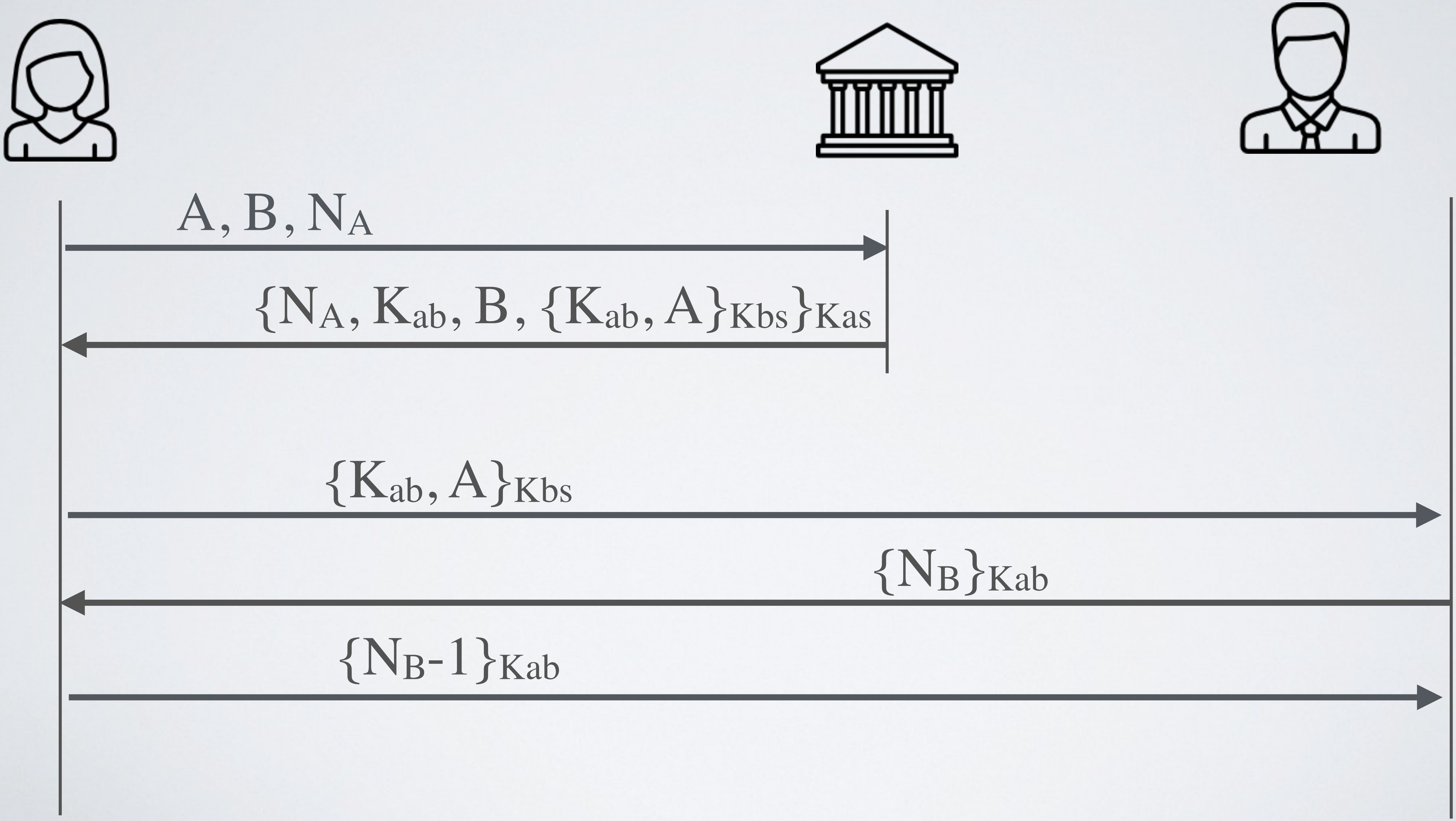
- 4 principals : **A**lice, **B**ob, **M**allory, Key Distribution **S**erver
- **S** shares a key with **A**, **B** and M respectively $\mathbf{K}_{as}$, $\mathbf{K}_{bs}$, $\mathbf{K}_{ms}$
- **A**, **B**, **M** and **S** talk to each other using the same protocol

Goals

When two parties want to engage in the communication, they want to

1. make sure that they talk to the right person (authentication)
2. establish a session key

The vulnerable Needham-Shroeder Symmetric Key Protocol (1978)



A, B, N_A

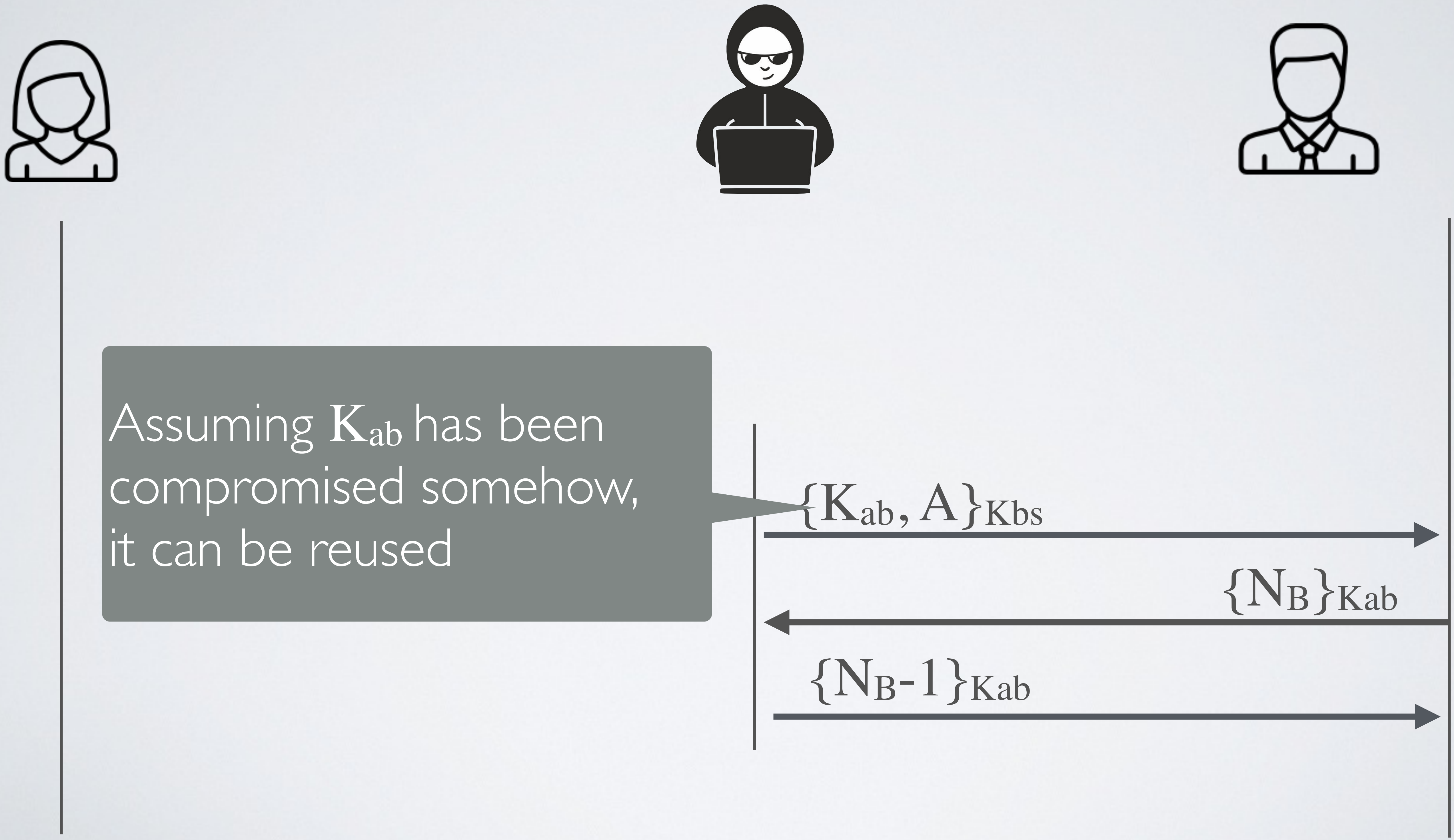
$\{N_A, K_{ab}, B, \{K_{ab}, A\}_{Kbs}\}_{Kas}$

$\{K_{ab}, A\}_{Kbs}$

$\{N_B\}_{Kab}$

$\{N_B\text{-}1\}_{Kab}$

Breaking the Needham-Shroeder Symmetric Key Protocol (1981)



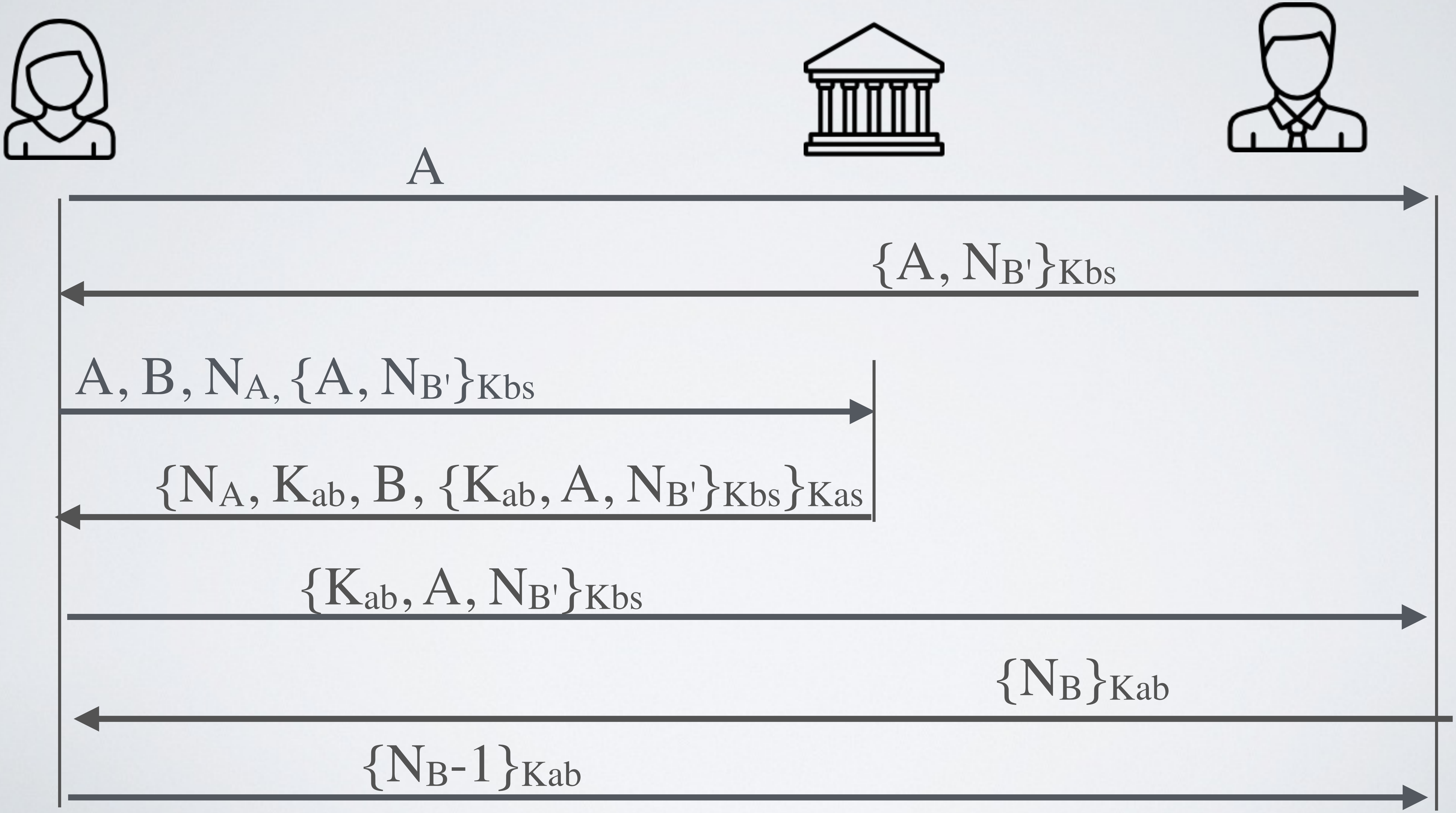
Assuming $\mathbf{K}_{ab}$ has been compromised somehow, it can be reused

$\{K_{ab}, A\}_{K_{bs}}$

$\{N_B\}_{K_{ab}}$

$\{N_B-1\}_{K_{ab}}$

Fixing the Needham-Shroeder Symmetric Key Protocol (1987)



A

$\{A, N_{B'}\}_{Kbs}$

$A, B, N_{A,} \{A, N_{B'}\}_{Kbs}$

$\{N_A, K_{ab}, B, \{K_{ab}, A, N_{B'}\}_{Kbs}\}_{Kas}$

$\{K_{ab}, A, N_{B'}\}_{Kbs}$

$\{N_B\}_{Kab}$

$\{N_B-1\}_{Kab}$