Key Exchange Protocols

(exercise)

Levente Buttyán

CrySyS Lab, BME

buttyan@crysys.hu

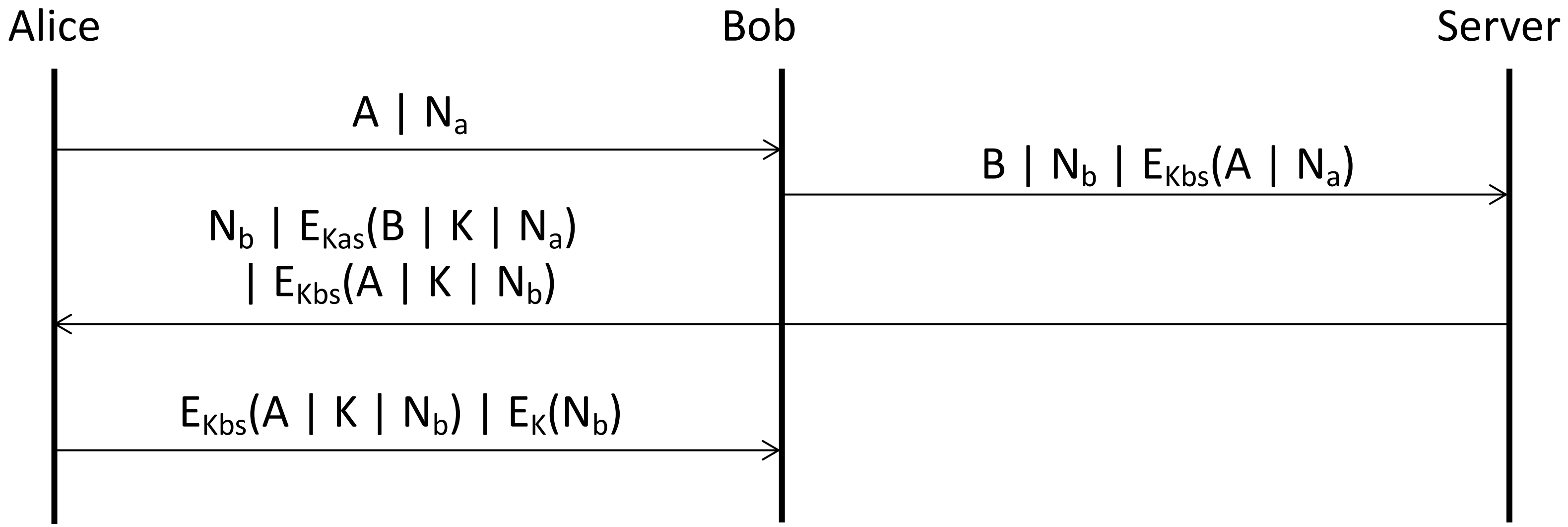
Reminder on main design objectives

1. **Secrecy of the key:** When the protocol is executed by Alice and Bob, no other parties (with the possible exception of Trent) should learn the value of the established key.

2. **Key authentication:** If Alice believes that she successfully executed the protocol and establsihed a new key K with Bob, then Bob was indeed present and he should believe that he executed the protocol and established the same key K with Alice.

3. **Key freshness:** Both parties should believe that the established key is fresh (new, not used before).

Exercise



Alice

Bob

Server

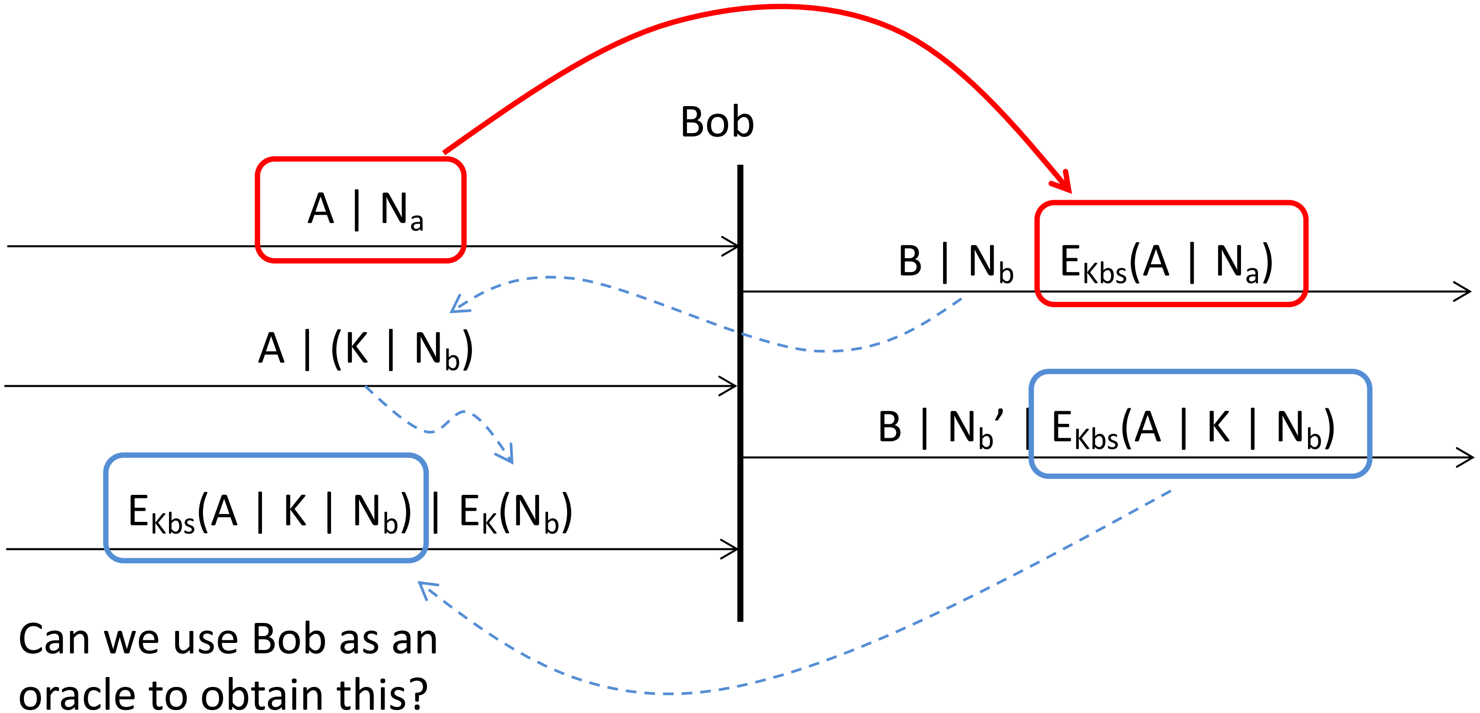
$A \mid N_a$

$B \mid N_b \mid E_{Kbs}(A \mid N_a)$

$N_b \mid E_{Kas}(B \mid K \mid N_a)$
$\mid E_{Kbs}(A \mid K \mid N_b)$

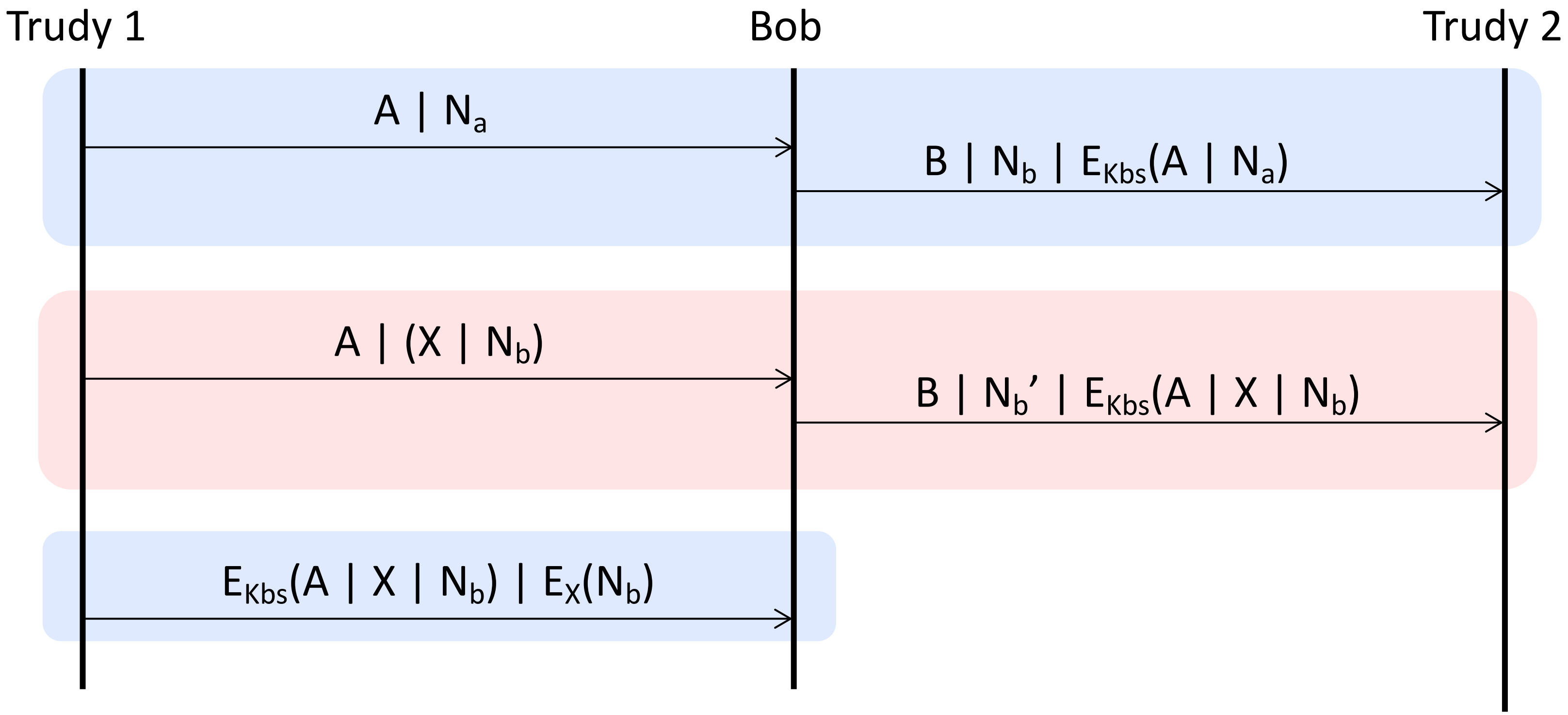
$E_{Kbs}(A \mid K \mid N_b) \mid E_K(N_b)$

Exercise: An idea...


Bob
$A \mid N_a$
$B \mid N_b$
$E_{Kbs}(A \mid N_a)$
$A \mid (K \mid N_b)$
$B \mid N_b' \mid E_{Kbs}(A \mid K \mid N_b)$
$E_{Kbs}(A \mid K \mid N_b) \mid E_K(N_b)$

Can we use Bob as an oracle to obtain this?

Exercise: The resulting attack



Trudy 1

Bob

Trudy 2

$A \mid N_a$

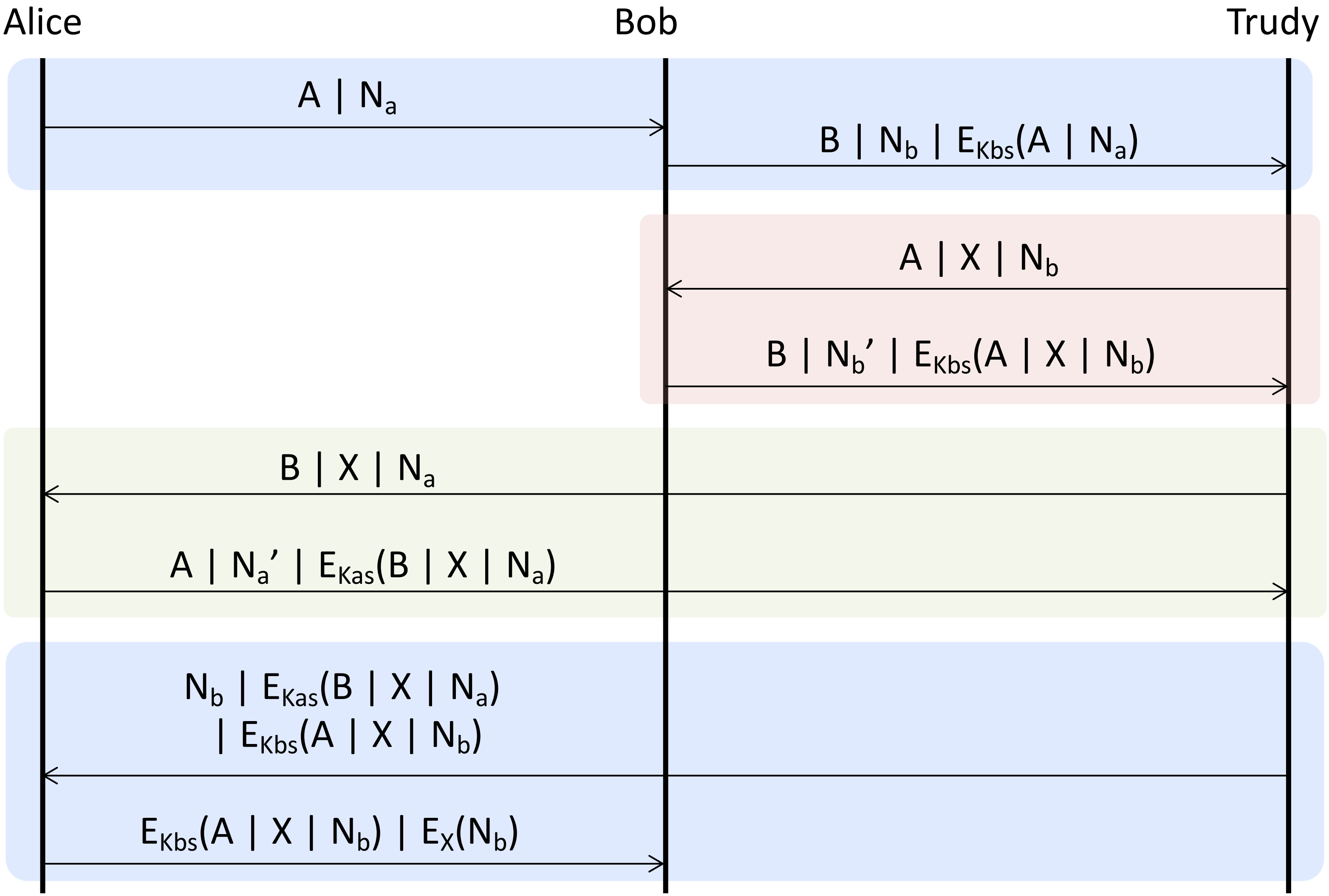
$B \mid N_b \mid E_{Kbs}(A \mid N_a)$

$A \mid (X \mid N_b)$

$B \mid N_b' \mid E_{Kbs}(A \mid X \mid N_b)$

$E_{Kbs}(A \mid X \mid N_b) \mid E_X(N_b)$

Exercise: A variant



Alice

Bob

Trudy

$A \mid N_a$

$B \mid N_b \mid E_{Kbs}(A \mid N_a)$

$A \mid X \mid N_b$

$B \mid N_b' \mid E_{Kbs}(A \mid X \mid N_b)$

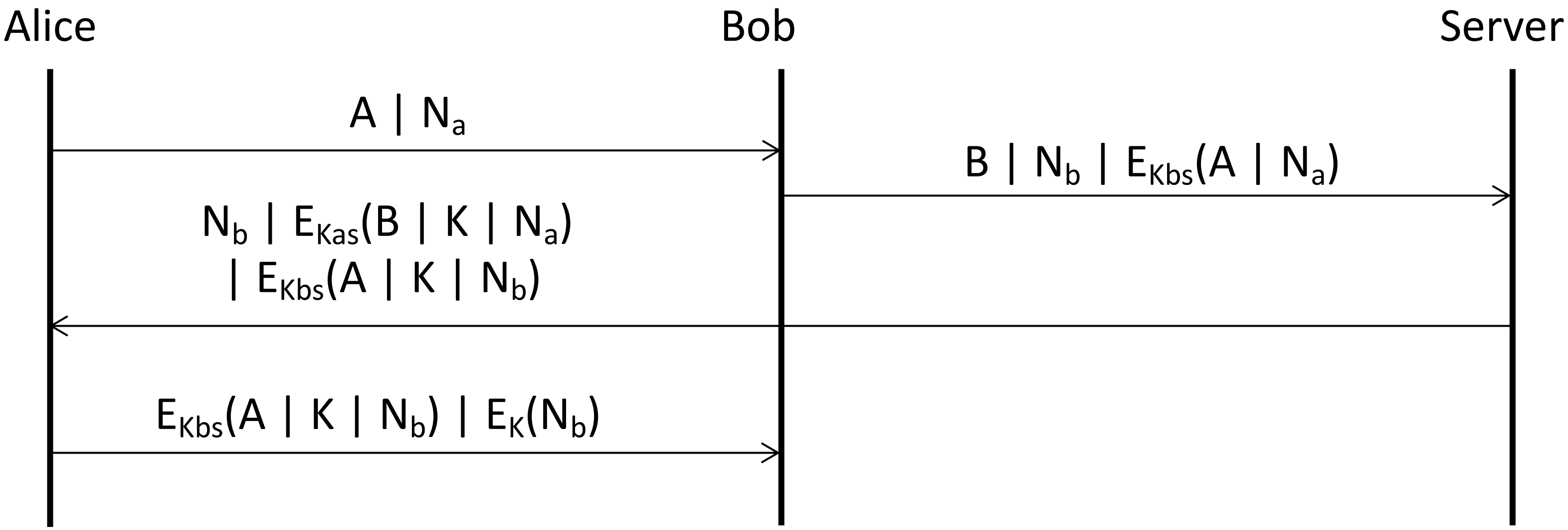
$B \mid X \mid N_a$

$A \mid N_a' \mid E_{Kas}(B \mid X \mid N_a)$

$N_b \mid E_{Kas}(B \mid X \mid N_a)$
$\mid E_{Kbs}(A \mid X \mid N_b)$

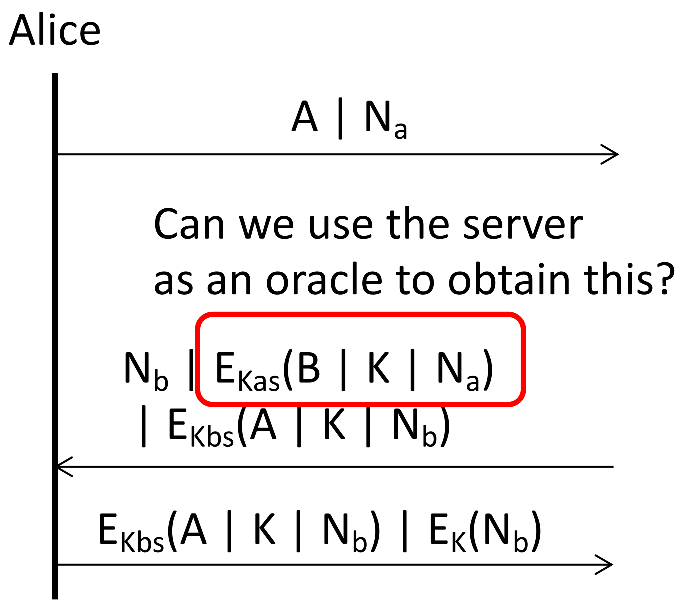
$E_{Kbs}(A \mid X \mid N_b) \mid E_X(N_b)$

Exercise: Another idea...



Alice
Bob
Server
$A \mid N_a$
$B \mid N_b \mid E_{Kbs}(A \mid N_a)$
$N_b \mid E_{Kas}(B \mid K \mid N_a)$
$\mid E_{Kbs}(A \mid K \mid N_b)$
$E_{Kbs}(A \mid K \mid N_b) \mid E_K(N_b)$

Exercise: Another idea...



Alice

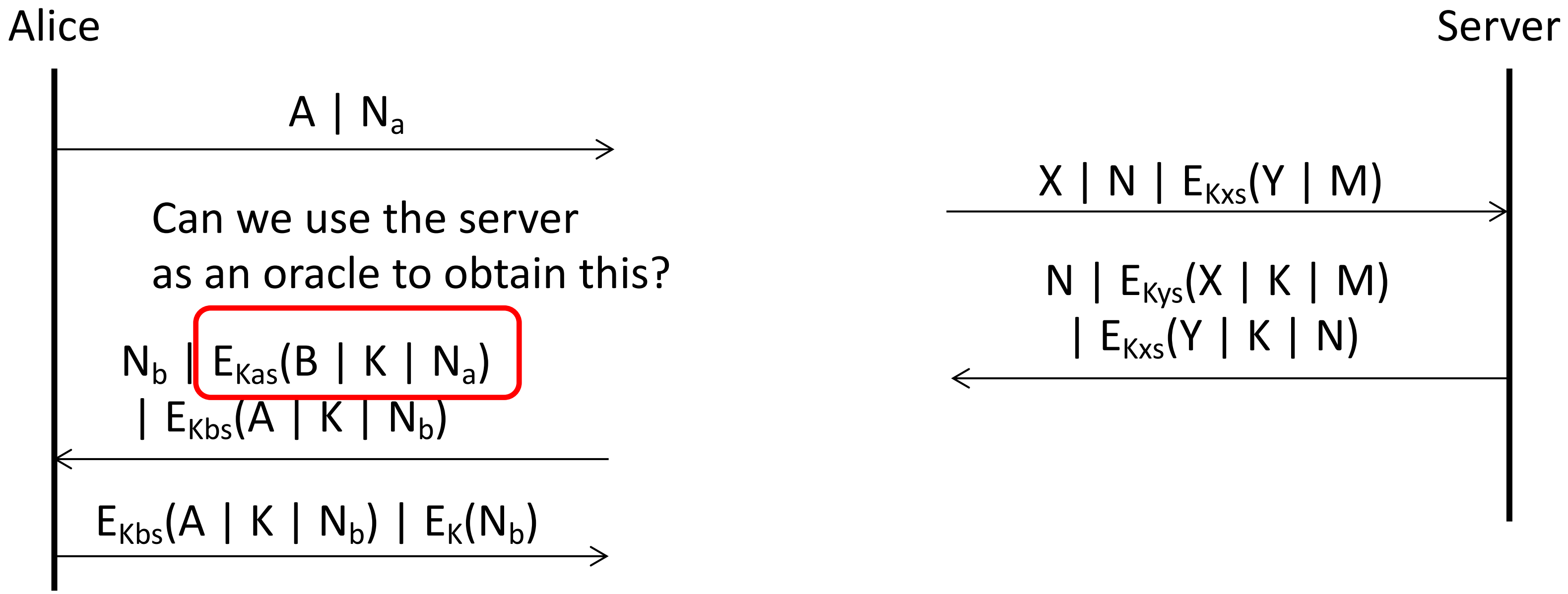
$A \mid N_a$

Can we use the server
as an oracle to obtain this?

$N_b \mid \boxed{E_{Kas}(B \mid K \mid N_a)}$
$\mid E_{Kbs}(A \mid K \mid N_b)$

$E_{Kbs}(A \mid K \mid N_b) \mid E_K(N_b)$

Exercise: Another idea...



Alice

$A \mid N_a$

Can we use the server
as an oracle to obtain this?

$N_b \mid \boxed{E_{Kas}(B \mid K \mid N_a)} \mid E_{Kbs}(A \mid K \mid N_b)$

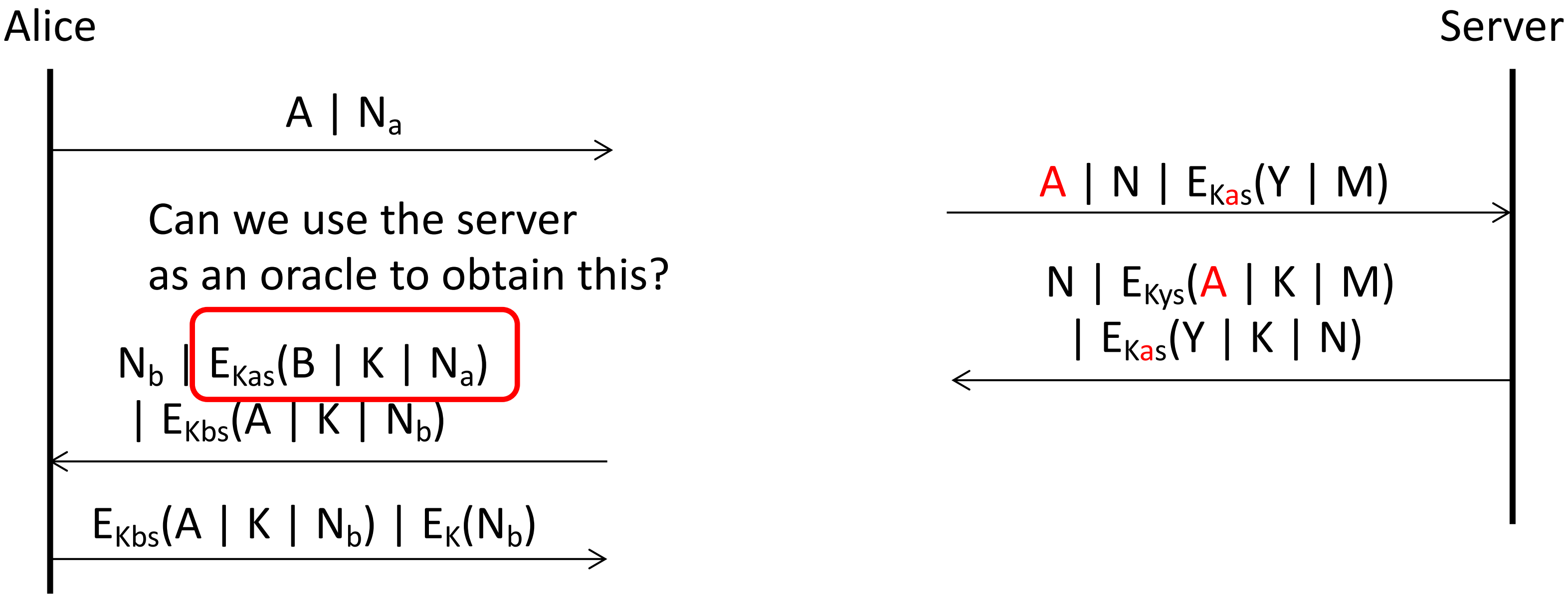
$E_{Kbs}(A \mid K \mid N_b) \mid E_K(N_b)$

Server

$X \mid N \mid E_{Kxs}(Y \mid M)$

$N \mid E_{Kys}(X \mid K \mid M) \mid E_{Kxs}(Y \mid K \mid N)$

Exercise: Another idea...



Alice

Server

$A \mid N_a$

Can we use the server
as an oracle to obtain this?

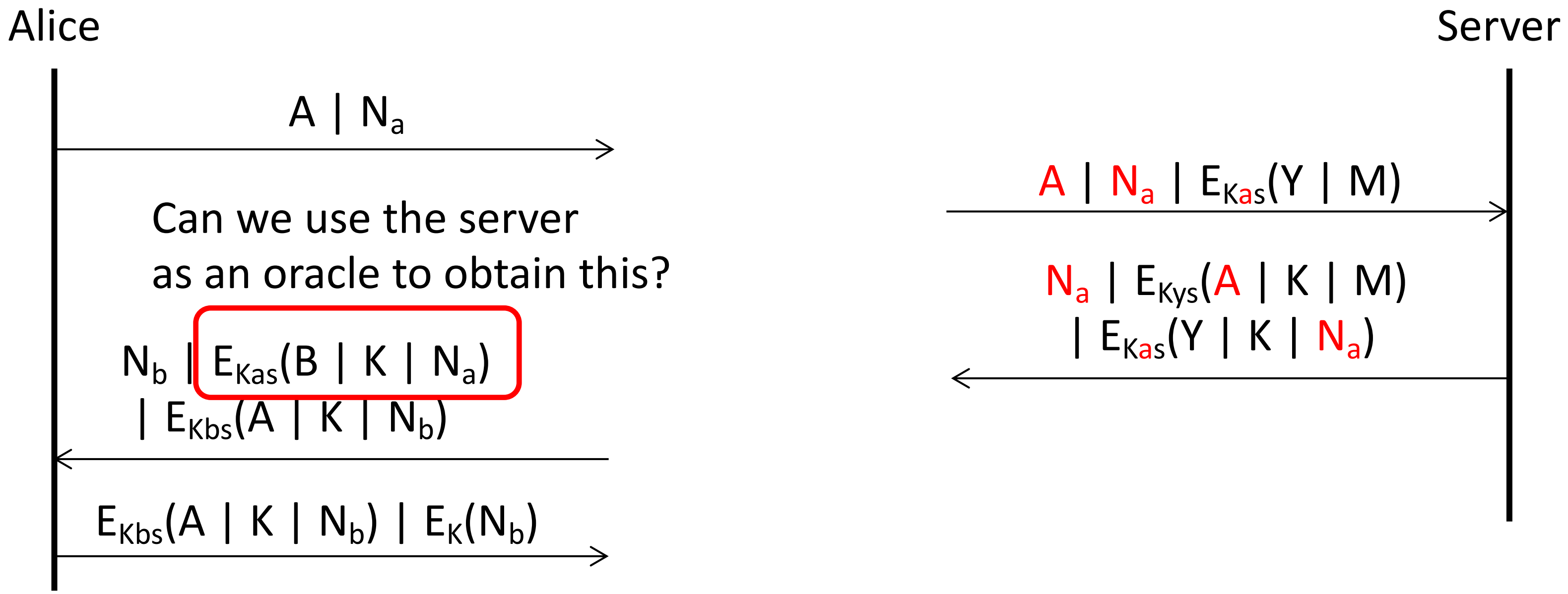
$N_b \mid \boxed{E_{Kas}(B \mid K \mid N_a)}$
$\mid E_{Kbs}(A \mid K \mid N_b)$

$E_{Kbs}(A \mid K \mid N_b) \mid E_K(N_b)$

$A \mid N \mid E_{Kas}(Y \mid M)$

$N \mid E_{Kys}(A \mid K \mid M)$
$\mid E_{Kas}(Y \mid K \mid N)$

Exercise: Another idea...



Alice

$A \mid N_a$

Can we use the server as an oracle to obtain this?

$N_b \mid \boxed{E_{Kas}(B \mid K \mid N_a)} \mid E_{Kbs}(A \mid K \mid N_b)$

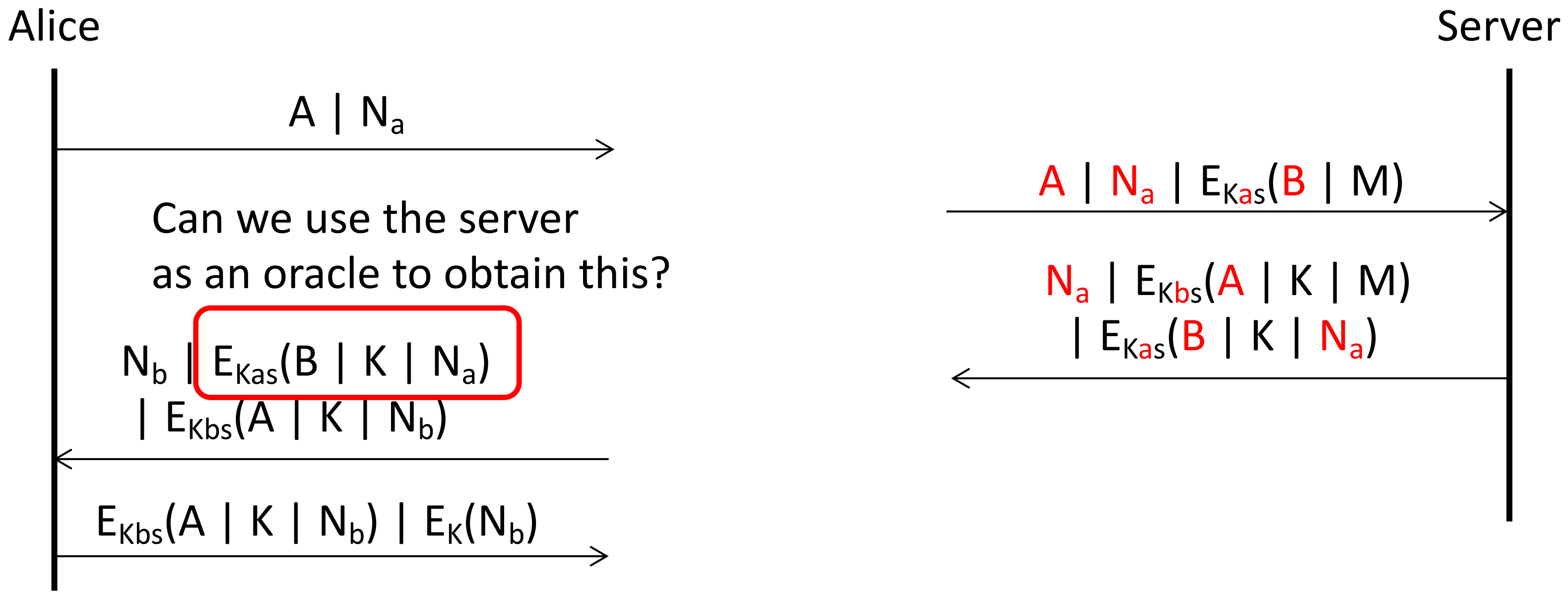
$E_{Kbs}(A \mid K \mid N_b) \mid E_K(N_b)$

Server

$A \mid N_a \mid E_{Kas}(Y \mid M)$

$N_a \mid E_{Kys}(A \mid K \mid M) \mid E_{Kas}(Y \mid K \mid N_a)$

Exercise: Another idea...



Alice

Server

$A \mid N_a$

Can we use the server as an oracle to obtain this?

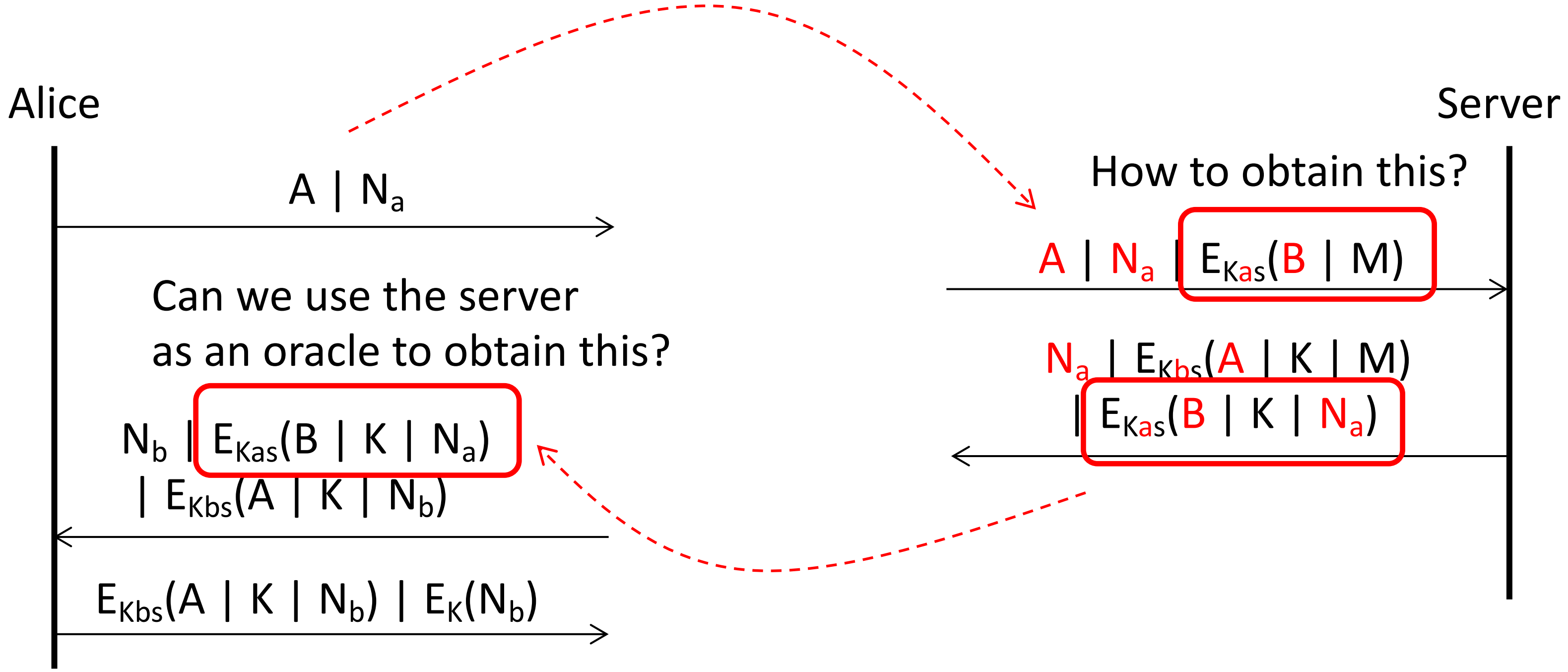
$N_b \mid \boxed{E_{Kas}(B \mid K \mid N_a)} \mid E_{Kbs}(A \mid K \mid N_b)$

$E_{Kbs}(A \mid K \mid N_b) \mid E_K(N_b)$

$A \mid N_a \mid E_{Kas}(B \mid M)$

$N_a \mid E_{Kbs}(A \mid K \mid M) \mid E_{Kas}(B \mid K \mid N_a)$

Exercise: Another idea...



Alice

$A \mid N_a$

Can we use the server
as an oracle to obtain this?

$N_b \mid E_{Kas}(B \mid K \mid N_a)$
$\mid E_{Kbs}(A \mid K \mid N_b)$

$E_{Kbs}(A \mid K \mid N_b) \mid E_K(N_b)$

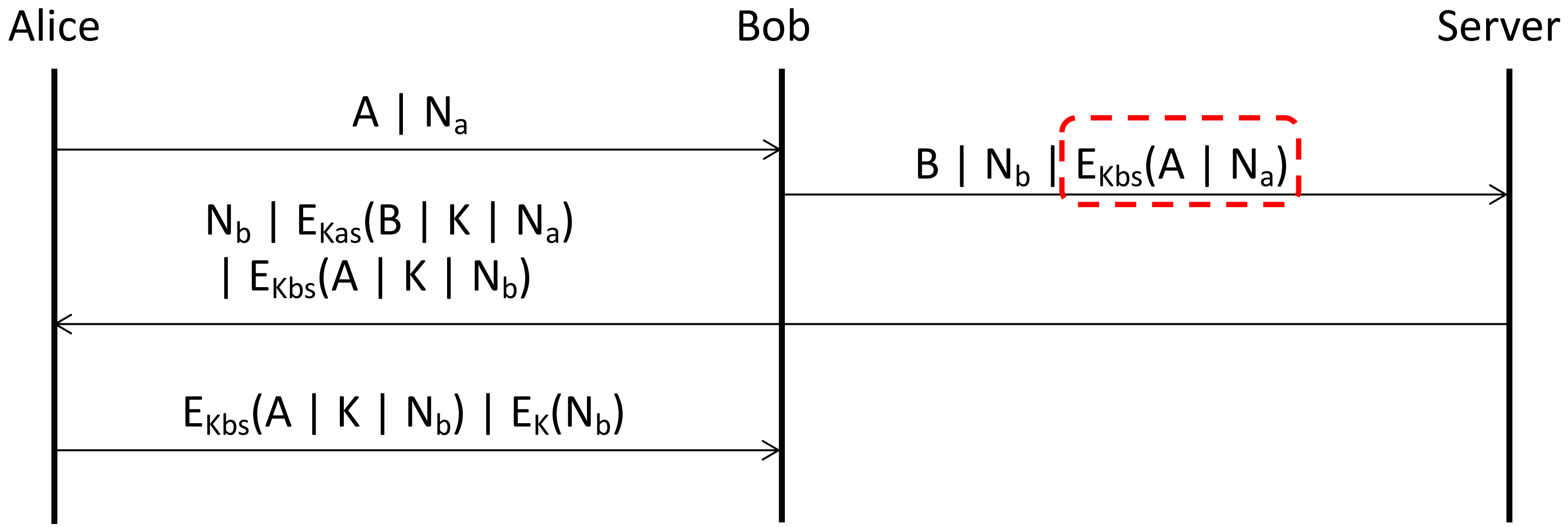
Server

How to obtain this?

$A \mid N_a \mid E_{Kas}(B \mid M)$

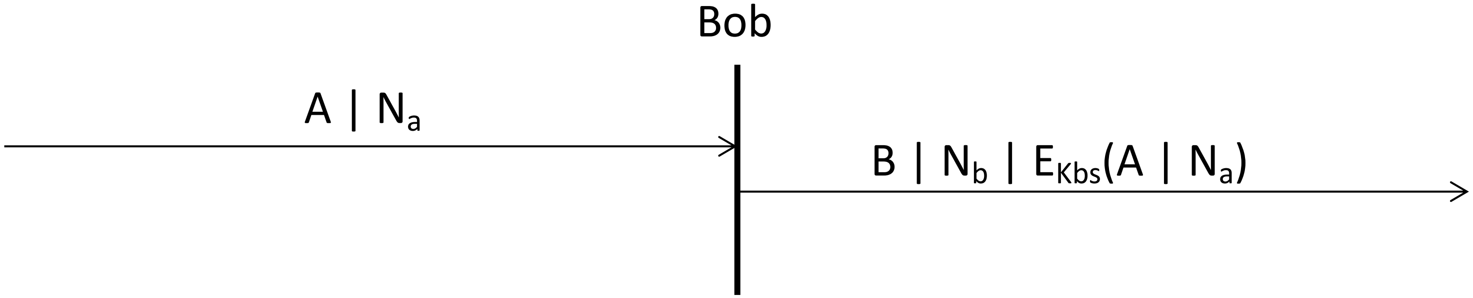
$N_a \mid E_{Kbs}(A \mid K \mid M)$
$\mid E_{Kas}(B \mid K \mid N_a)$

Exercise: Another idea...


Alice
Bob
Server
$A \mid N_a$
$B \mid N_b \mid E_{Kbs}(A \mid N_a)$
$N_b \mid E_{Kas}(B \mid K \mid N_a)$
$\mid E_{Kbs}(A \mid K \mid N_b)$
$E_{Kbs}(A \mid K \mid N_b) \mid E_K(N_b)$

$E_{Kas}(B \mid M)$

Exercise: Another idea...



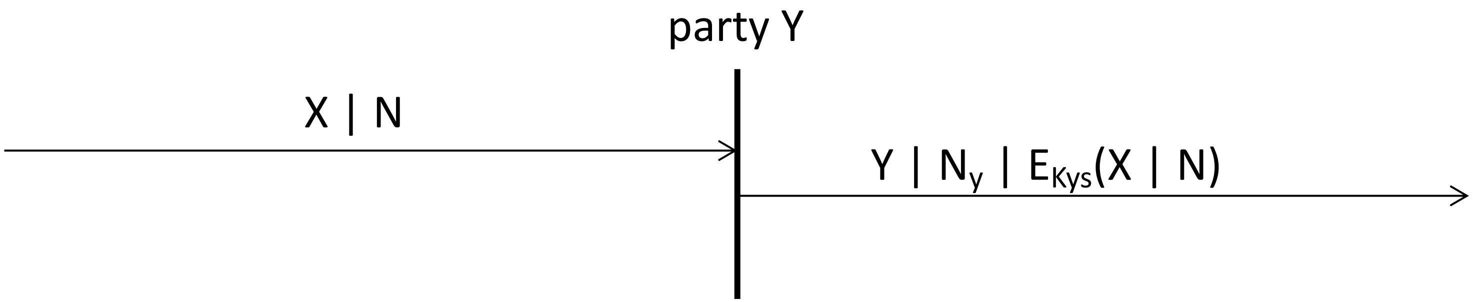
Bob

$A \mid N_a$

$B \mid N_b \mid E_{Kbs}(A \mid N_a)$

$\boxed{E_{Kas}(B \mid M)}$

Exercise: Another idea...



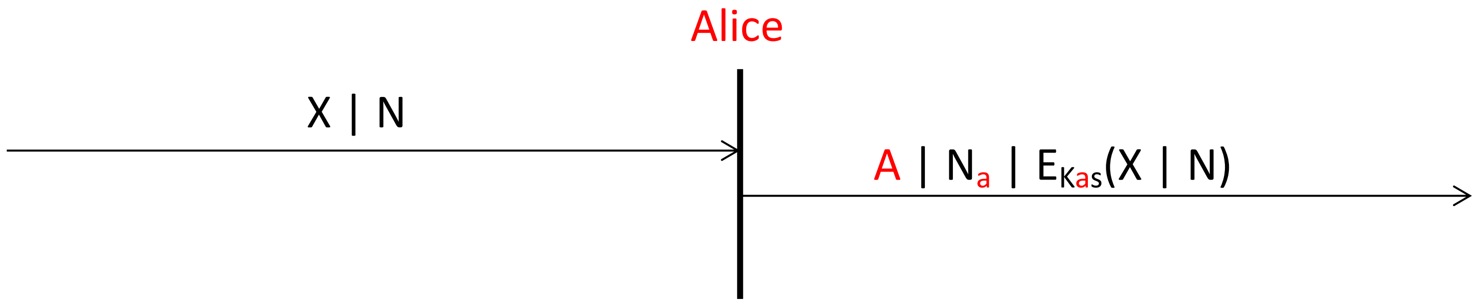
party Y

X | N

Y | N_y | E_{Kys}(X | N)

E_{Kas}(B | M)

Exercise: Another idea...



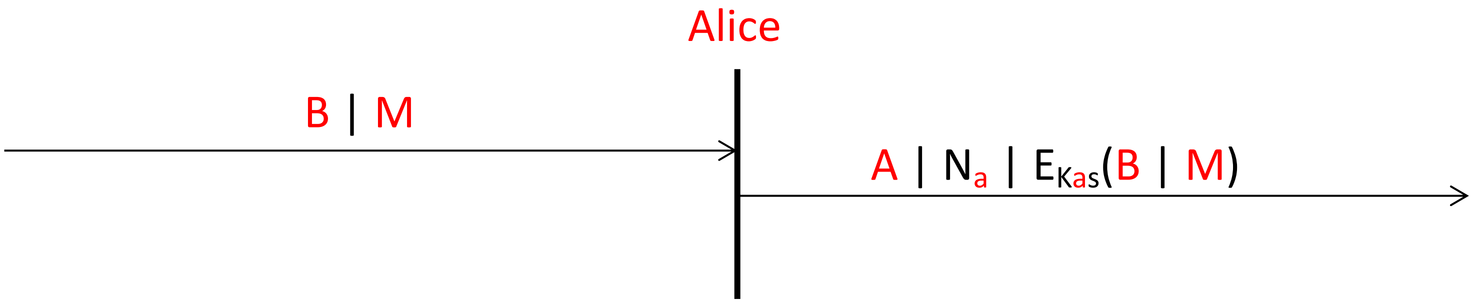
Alice

$X \mid N$

$A \mid N_a \mid E_{Kas}(X \mid N)$

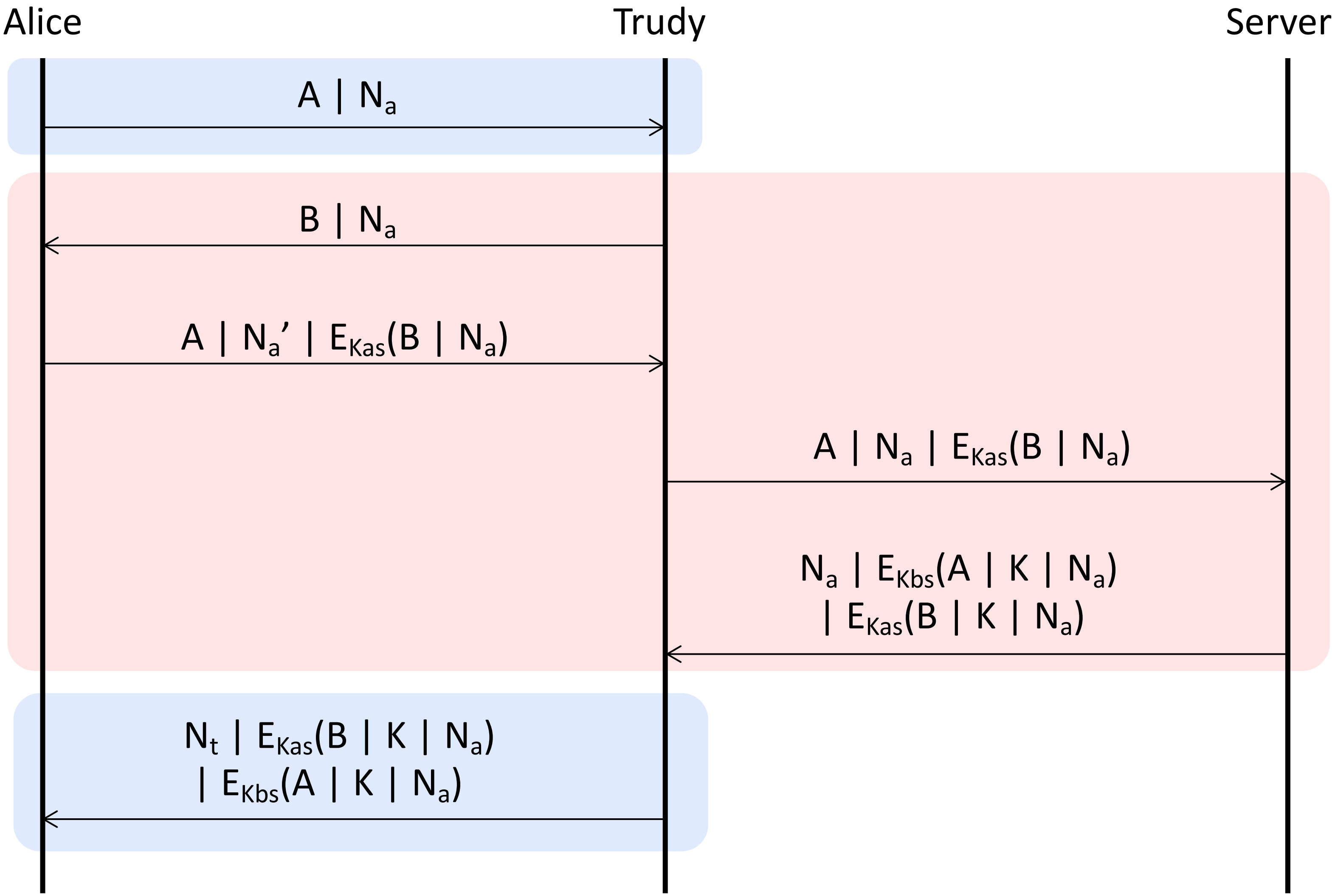
$\boxed{E_{Kas}(B \mid M)}$

Exercise: Another idea...



$\boxed{E_{Kas}(B \mid M)}$

Exercise: The resulting attack



Alice

Trudy

Server

$A \mid N_a$

$B \mid N_a$

$A \mid N_a' \mid E_{Kas}(B \mid N_a)$

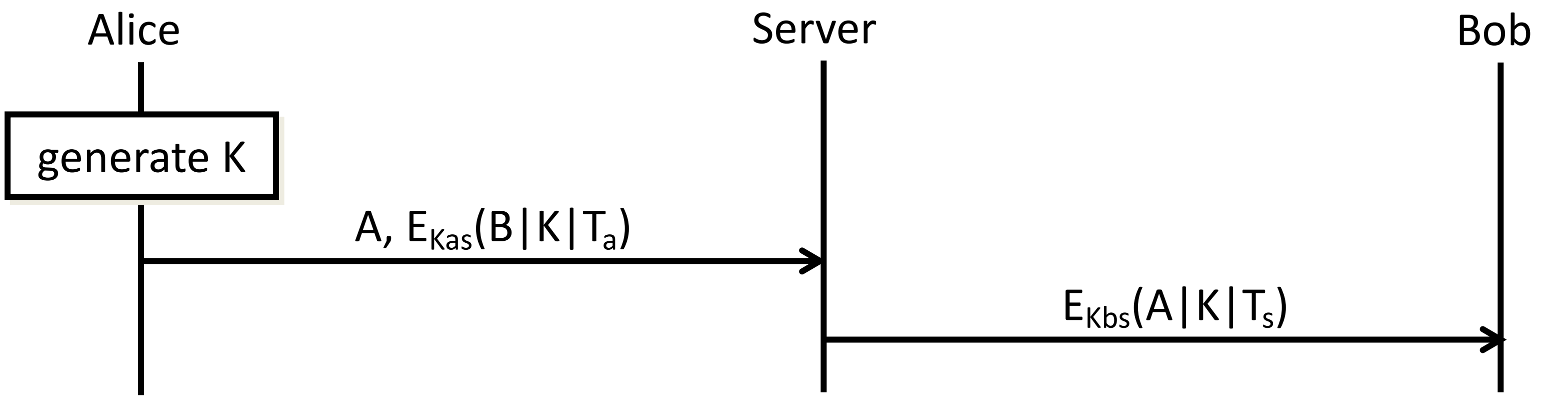
$A \mid N_a \mid E_{Kas}(B \mid N_a)$

$N_a \mid E_{Kbs}(A \mid K \mid N_a)$
$\mid E_{Kas}(B \mid K \mid N_a)$

$N_t \mid E_{Kas}(B \mid K \mid N_a)$
$\mid E_{Kbs}(A \mid K \mid N_a)$

Challenges

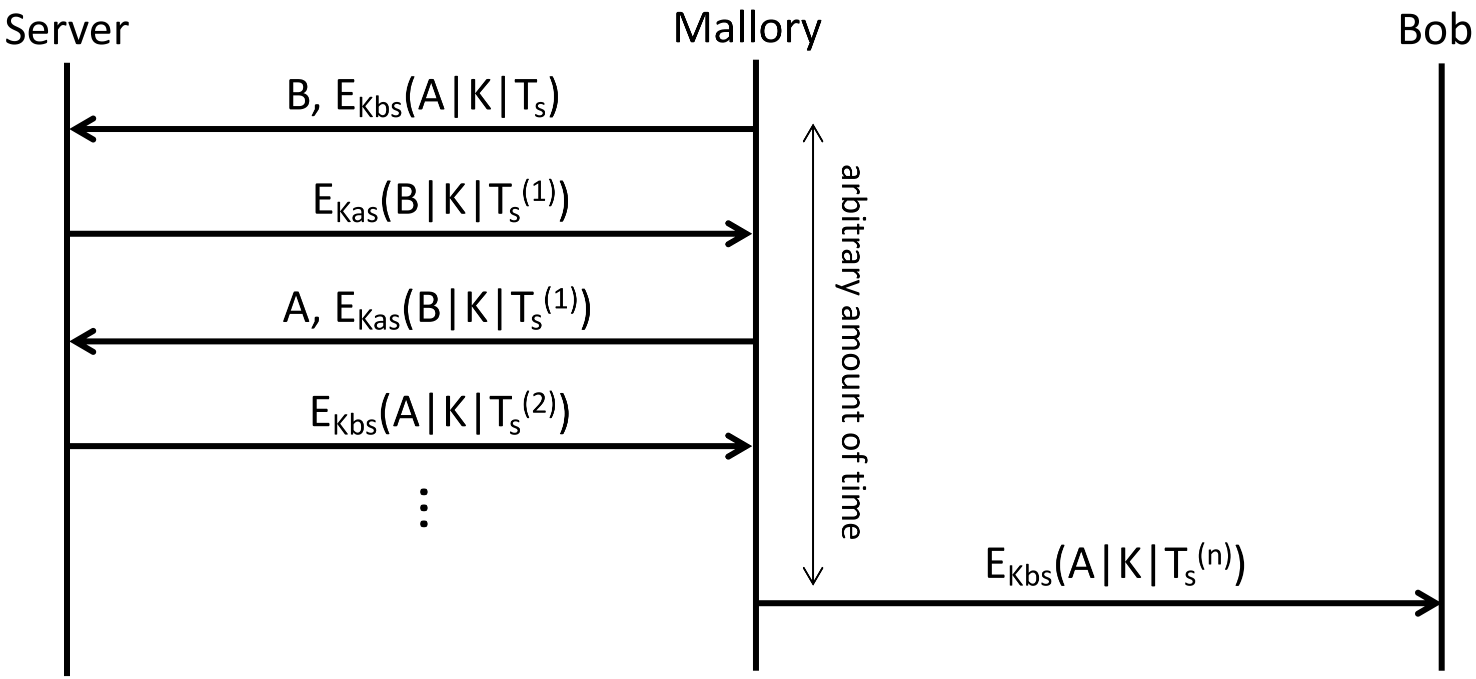
The Wide-Mouth-Frog protocol



notes:

- Alice is trusted to generate good quality keys
- Server is trusted for verification of timestamp and secure relaying of the key to the indicated other party
- key freshness for Bob is meant to be provided by the server’s timestamp, but ...

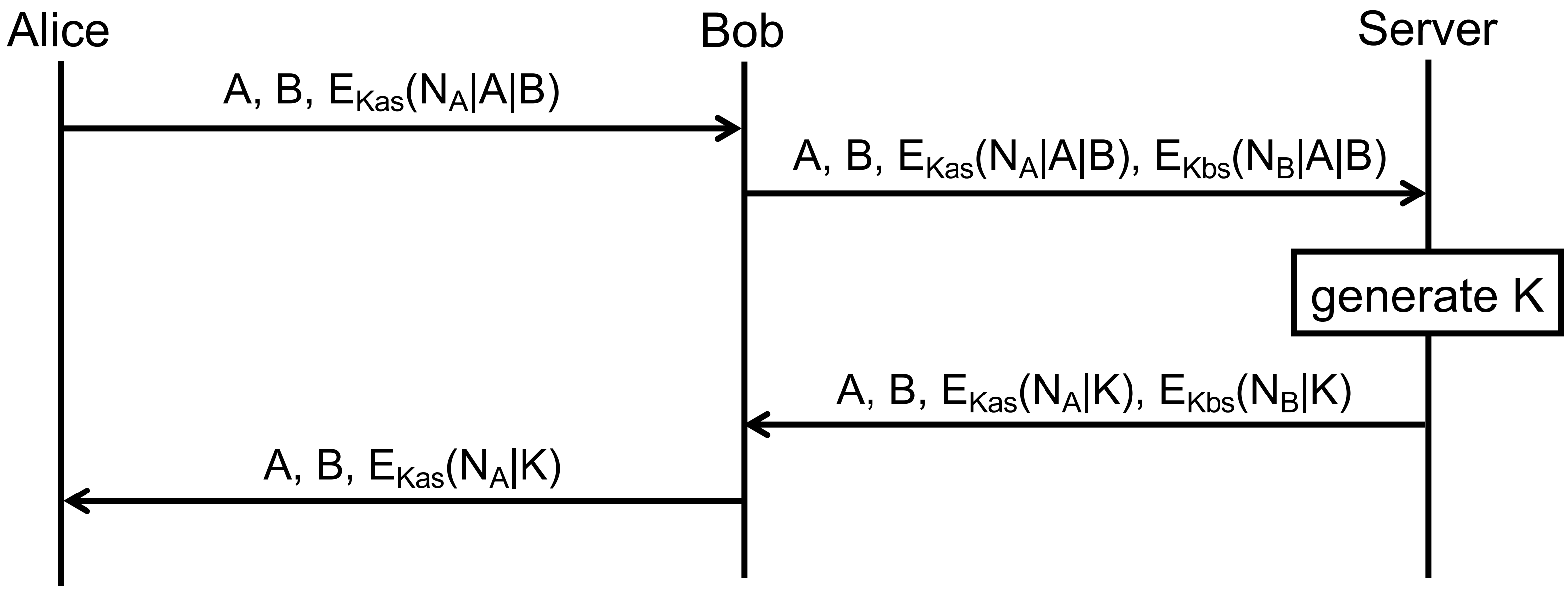
A reflection attack on the WMF protocol



notes:

- the problem is that the first and the second messages of the WMF protocol have the same structure → easy to replace one for the other
- also, messages encoded with symmetric keys can be replayed back to their source, and will be decoded correctly
- **use direction bits or different keys in different directions**

Otway-Rees protocol

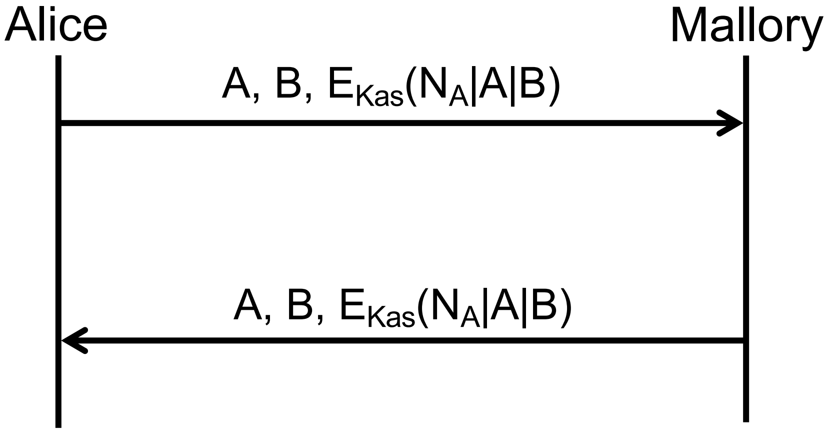


Alice
Bob
Server
A, B, $E_{Kas}(N_A|A|B)$
A, B, $E_{Kas}(N_A|A|B)$, $E_{Kbs}(N_B|A|B)$
generate K
A, B, $E_{Kas}(N_A|K)$, $E_{Kbs}(N_B|K)$
A, B, $E_{Kas}(N_A|K)$

notes:

- names are omitted in the server's response, because A and B have already been bound to N_A and N_B by the encryption in the first two messages (not a recommendable practice, though)

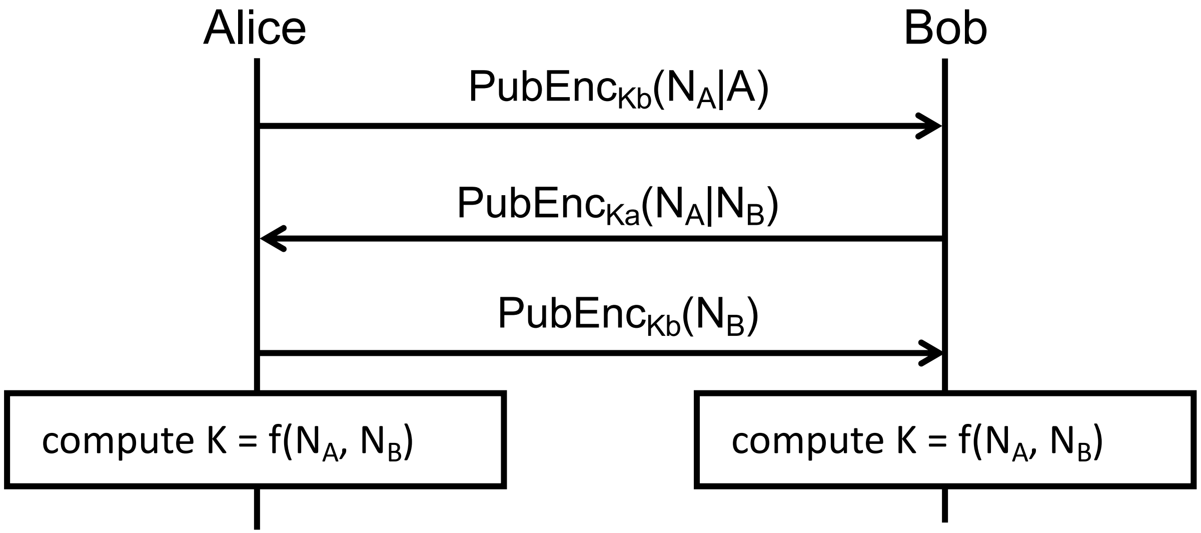
A typing attack on Otway-Rees



notes:

- the bit string A|B (known to Mallory) may be interpreted as a new key K
- reflection attacks can be avoided by using direction bits in messages
- **even better if the protocol is designed in such a way that it is possible to tell about any message which protocol's which message it is**
- **type identifiers in messages can also be useful**, in order to be sure that no typing attack is possible

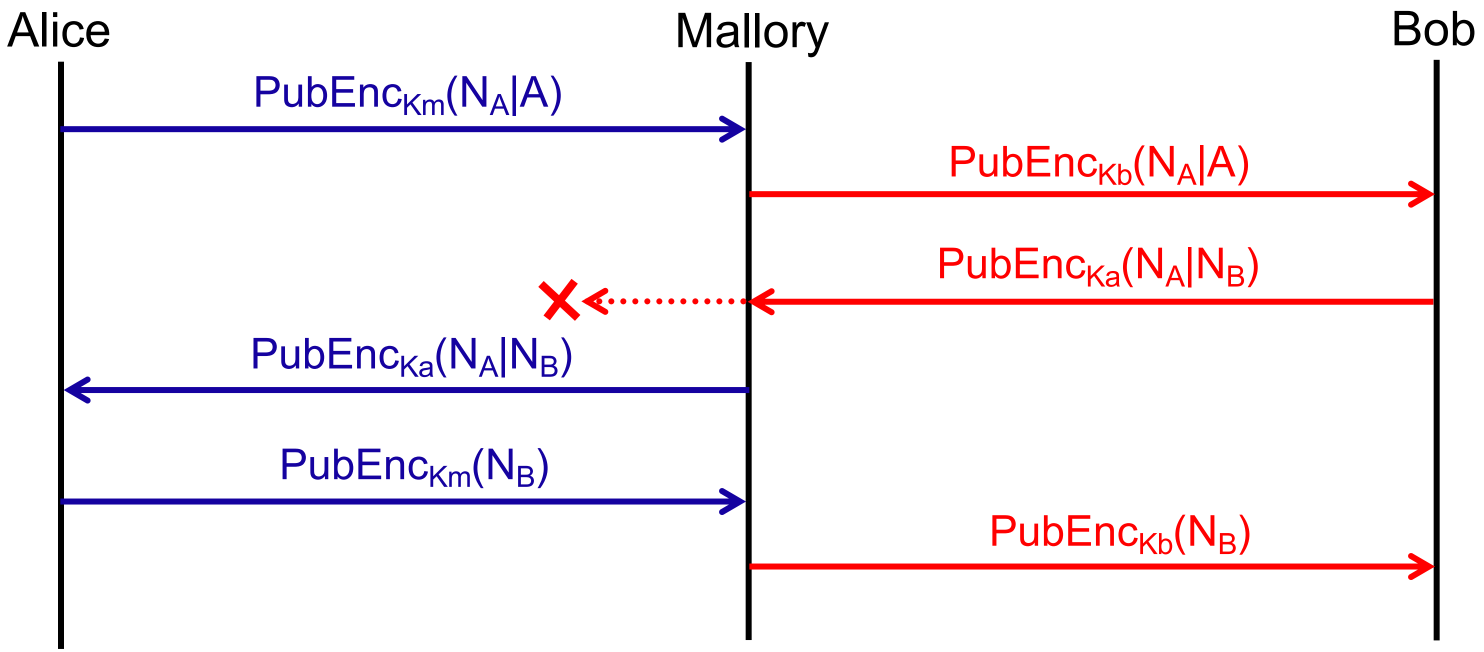
The public-key Needheam-Schröder protocol


Alice
Bob
PubEnc$_{Kb}$(N_A|A)
PubEnc$_{Ka}$(N_A|N_B)
PubEnc$_{Kb}$(N_B)
compute K = f(N_A, N_B)
compute K = f(N_A, N_B)

notes:

- originally proposed for partner authentication, and it solves that problem well
- the nonces never appear in clear on the channel, hence the idea to derive a session key from them
- this proved to be a bad idea...

An interleaving attack on the NS protocol



notes:

- one problem is that the message $PubEnc_{Ka}(N_A|N_B)$ can be copied and pasted from one instance of the protocol to another
- **if the message had included names explicitly** (e.g., $PubEnc_{Ka}(B|N_A|N_B)$)**, then this would not be possible!**