CSE 3400 - Introduction to Computer & Network Security (aka: Introduction to Cybersecurity)

Lecture 9
Shared Key Protocols – Part II

Ghada Almashaqbeh
UConn

From Textbook Slides by Prof. Amir Herzberg

UConn

Outline

❑ Handshake protocol extensions.

❑ Key distribution centers.

❑ Improving resilience to key exposure.

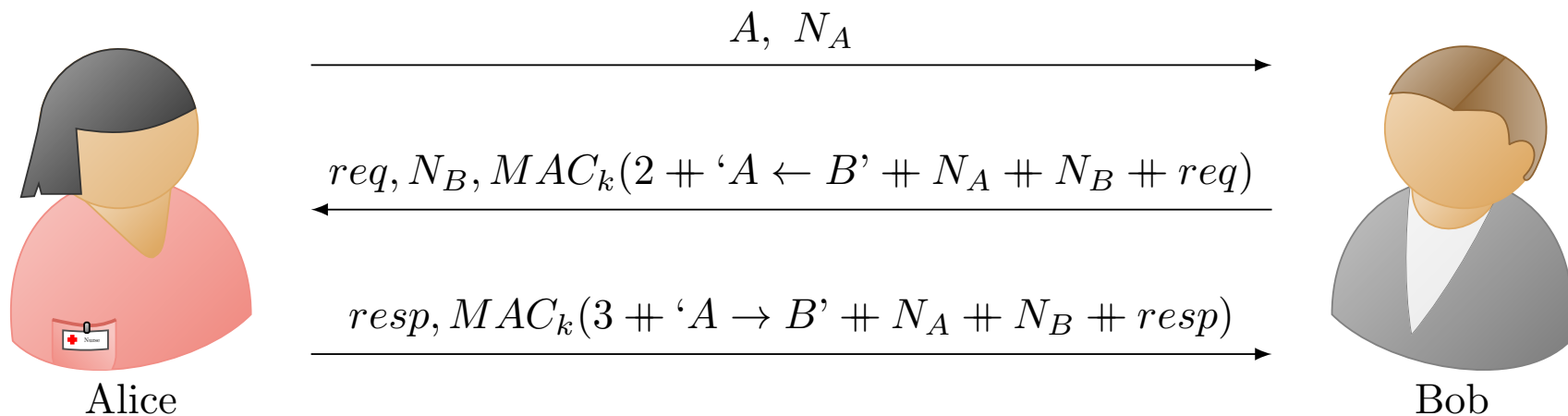# Handshake Protocols Extensions

Authenticated Request-Response Protocols

❑ Beside authenticating entities, these protocols authenticate the exchange of a request and a response between the entities.

❑ Required properties:

  ❑ **Request authentication.**

    ❑ The request was indeed sent by the peer.

  ❑ **Response authentication**

    ❑ The response was indeed sent by the peer.

  ❑ **No replay.**

    ❑ Every request/response was received at most the number of times it was sent by the peer.

Authenticated Request-Response Protocols

- ❑ Five variants:
    - ❑ 2PP-RR
    - ❑ 2RT-2PP
    - ❑ Counter-based-RR
    - ❑ Time-based-RR.
    - ❑ Key-exchange.
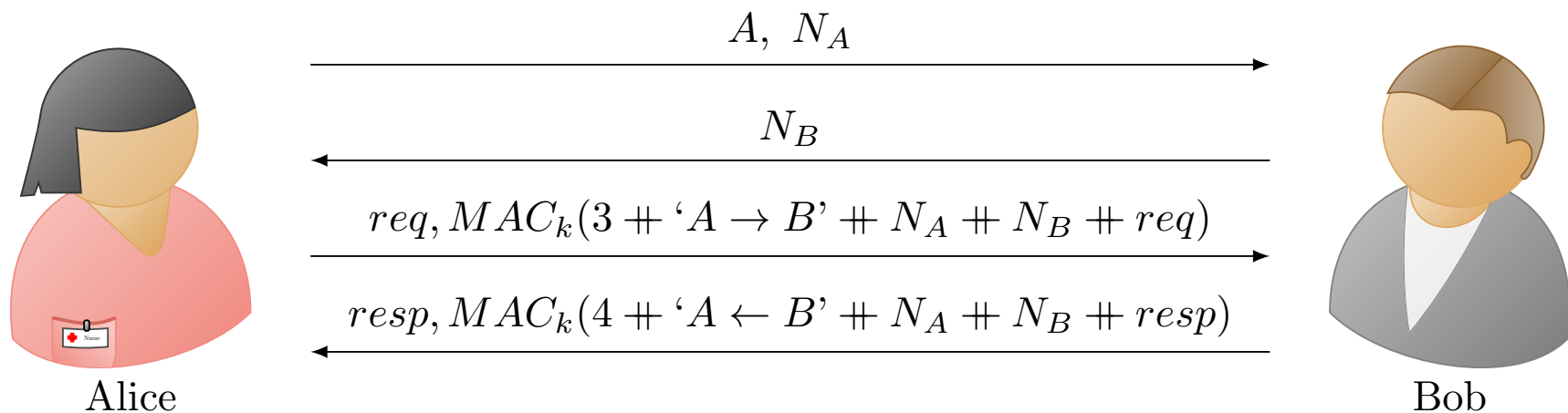
2PP-RR

- A three-flow nonce-based protocol.
- Significant drawback:
  - The request is sent by the responder and the initiator sends the response.
  - So initiator must wait for a request rather than sending it!!

$$A, \ N_A$$

$$req, N_B, MAC_k(2 + \text{`}A \leftarrow B\text{'} + N_A + N_B + req)$$

$$resp, MAC_k(3 + \text{`}A \rightarrow B\text{'} + N_A + N_B + resp)$$

Alice

Bob

2RT-2PP

- A four-flow nonce-based protocol.
- Mainly fixes the drawback of 2PP-RR (see previous slide).

Alice $\xrightarrow{\quad A,\ N_A \quad}$ Bob

$\xleftarrow{\quad N_B \quad}$

$\xrightarrow{\quad req, MAC_k(3 + `A \rightarrow B' + N_A + N_B + req) \quad}$

$\xleftarrow{\quad resp, MAC_k(4 + `A \leftarrow B' + N_A + N_B + resp) \quad}$

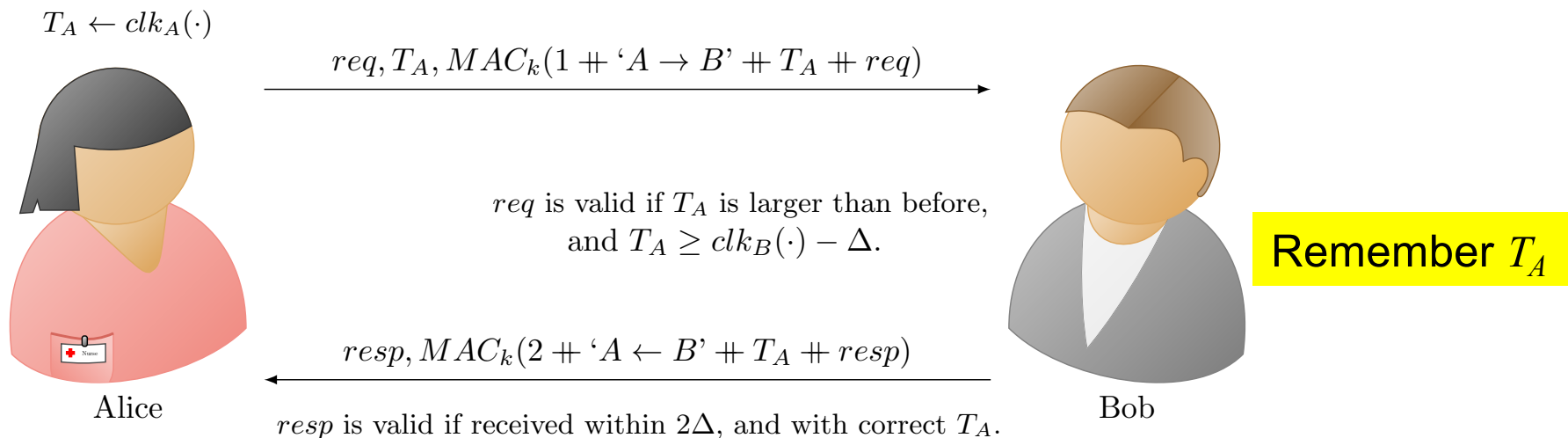Alice   Bob

Counter-Based Authenticated RR

- **Simple stateful (counter) solution, requiring only one round:**
    - Unidirectional (run once for each direction if both are needed).
    - Parties maintain synchronized counter i of requests (and responses) to avoid replay attacks.
    - Recipient (e.g., Bob) validates counter received is $i + 1$
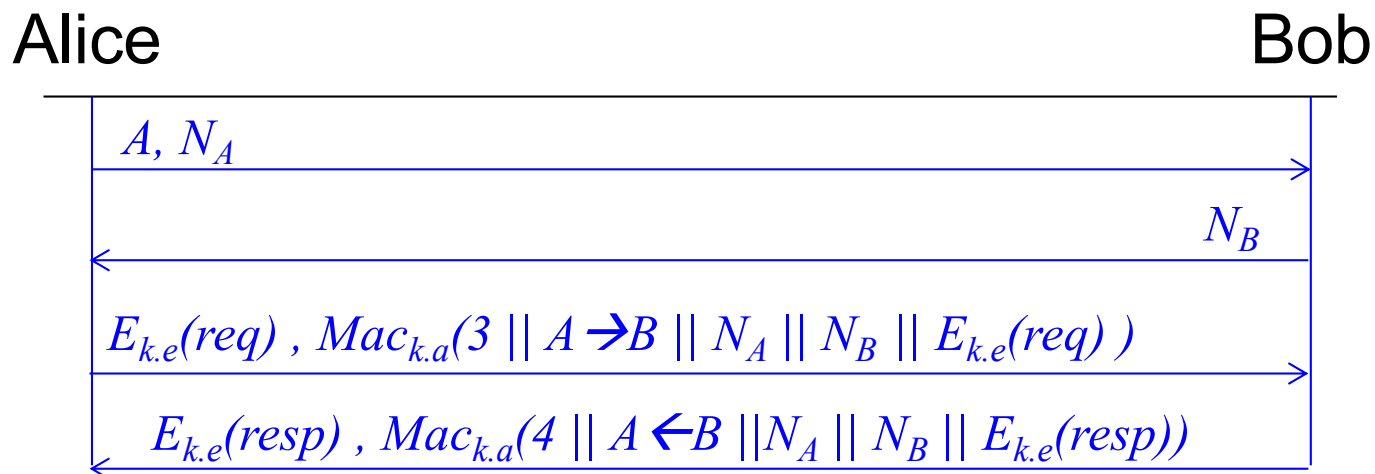   - Both parties must remember counter

$$req, i_A, MAC_k(1 + `A \rightarrow B' + i_A + req)$$

If $i_A \neq i_B + 1$: ignore
Else; $i_B \leftarrow i_B + 1$

$$resp, i_B, MAC_k(2 + `A \leftarrow B' + i_B + resp)$$

Alice

Accept if $i_A = i_B$

Bob
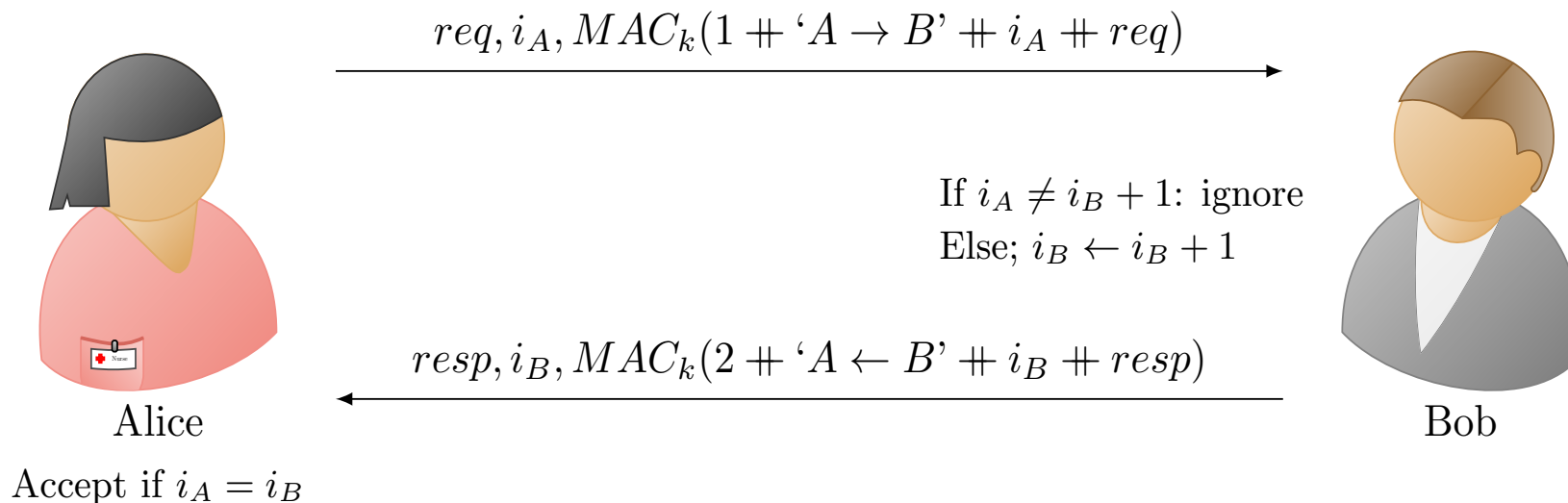
Time-Based Authenticated RR

- **Simple stateful (time) solution, requiring only one round:**
  - Use local clocks T_A, T_B instead of counters with two assumptions: bounded delays and bounded clock skews.
  - Responder (Bob):
    - Rejects request if: $T_B > T_A + \Delta$ where $\Delta \equiv \Delta_{skew} + \Delta_{delay}$
    - Or if he received larger T_A already
    - Maintains last T_A received, until $T_A + \Delta$
  - Initiator (Alice) does not need **any** state, when can Bob discard his?

$T_A \leftarrow clk_A(\cdot)$

$req, T_A, MAC_k(1 + \text{'}A \rightarrow B\text{'} + T_A + req)$

req is valid if T_A is larger than before, and $T_A \geq clk_B(\cdot) - \Delta$.

Remember T_A

$resp, MAC_k(2 + \text{'}A \leftarrow B\text{'} + T_A + resp)$

Alice

$resp$ is valid if received within 2Δ, and with correct T_A.
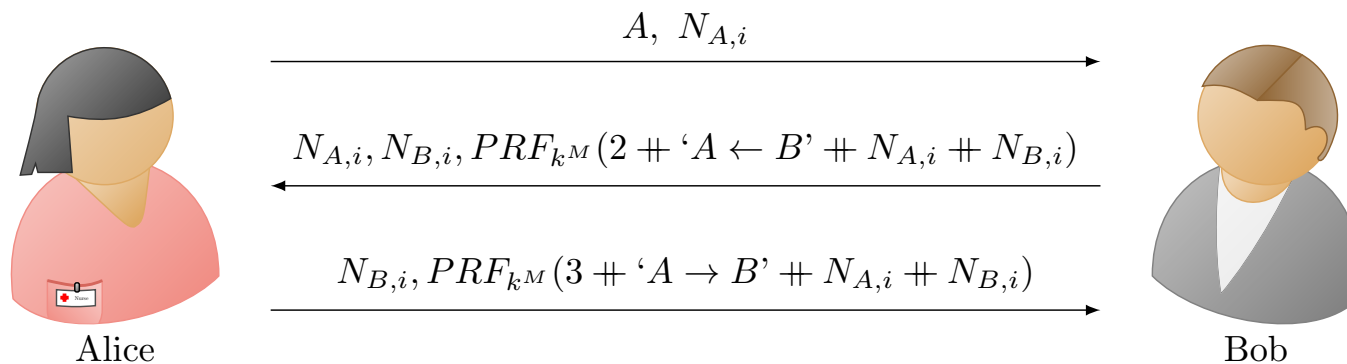
Bob

2RT-2PP with Confidentiality

- **Secure connection: authentication, freshness, secrecy**
  - Independent keys: for encryption $k.e$, for authentication: $k.a$
  - How can we derive them both from a single key k ?
  - $k.e=PRP_k(\text{"Encrypt"}), k.a=PRP_k(\text{"MAC"})$
  - Hmm… same key encrypts all messages, in all sessions ☹
- Can we improve security, by changing keys, e.g., btw sessions ?

Alice   Bob

A, N_A

N_B

$E_{k.e}(req) , Mac_{k.a}(3 \parallel A \rightarrow B \parallel N_A \parallel N_B \parallel E_{k.e}(req))$

$E_{k.e}(resp) , Mac_{k.a}(4 \parallel A \leftarrow B \parallel N_A \parallel N_B \parallel E_{k.e}(resp))$

2PP Key Exchange Protocol

- Independent session keys, e.g. $k=PRF_{MK}(N_A,N_B)$

- Or, `directly' for authentication and for encryption:
  $k.e=PRF_{MK}(\text{``Encrypt''}, N_A,N_B), k.a=PRF_{MK}(\text{``MAC''}, N_A,N_B)$

- Improves security:

    - Exposure of session key does not expose (long-term) `master key' MK

    - And does not expose keys of other sessions

    - Limited amount of ciphertext exposed with each session key k

- Later: reduce risk also from exposure of Master Key MK

Why a PRF is used instead of the MAC as before?

$$A,\ N_{A,i}$$

$$N_{A,i}, N_{B,i}, PRF_{k^M}(2 + \text{`} A \leftarrow B\text{'} + N_{A,i} + N_{B,i})$$

$$N_{B,i}, PRF_{k^M}(3 + \text{`} A \rightarrow B\text{'} + N_{A,i} + N_{B,i})$$

Alice

Bob

$$k_i^S = PRF_{k^M}(N_{A,i} + N_{B,i})$$

$$k_i^S = PRF_{k^M}(N_{A,i} + N_{B,i})$$

11

Key Distribution Centers (KDCs)

Establish a shared key between two or more entities, usually with the help of a trusted third party referred to as KDC

Key Distribution Center (KDC)

- Will focus on three party protocols; Alice, Bob, and KDC.

- KDC: shares keys with all parties (k_A, k_B…)

- Goal: help parties (A, B) <u>establish</u> k_{AB}

- We will study two protocols; simplified versions of:

  - The Kerberos protocol (secure) widely used in computer networks.

  - The GSM protocol (insecure) used by cellular networks.

The Kerberos KDC Protocol

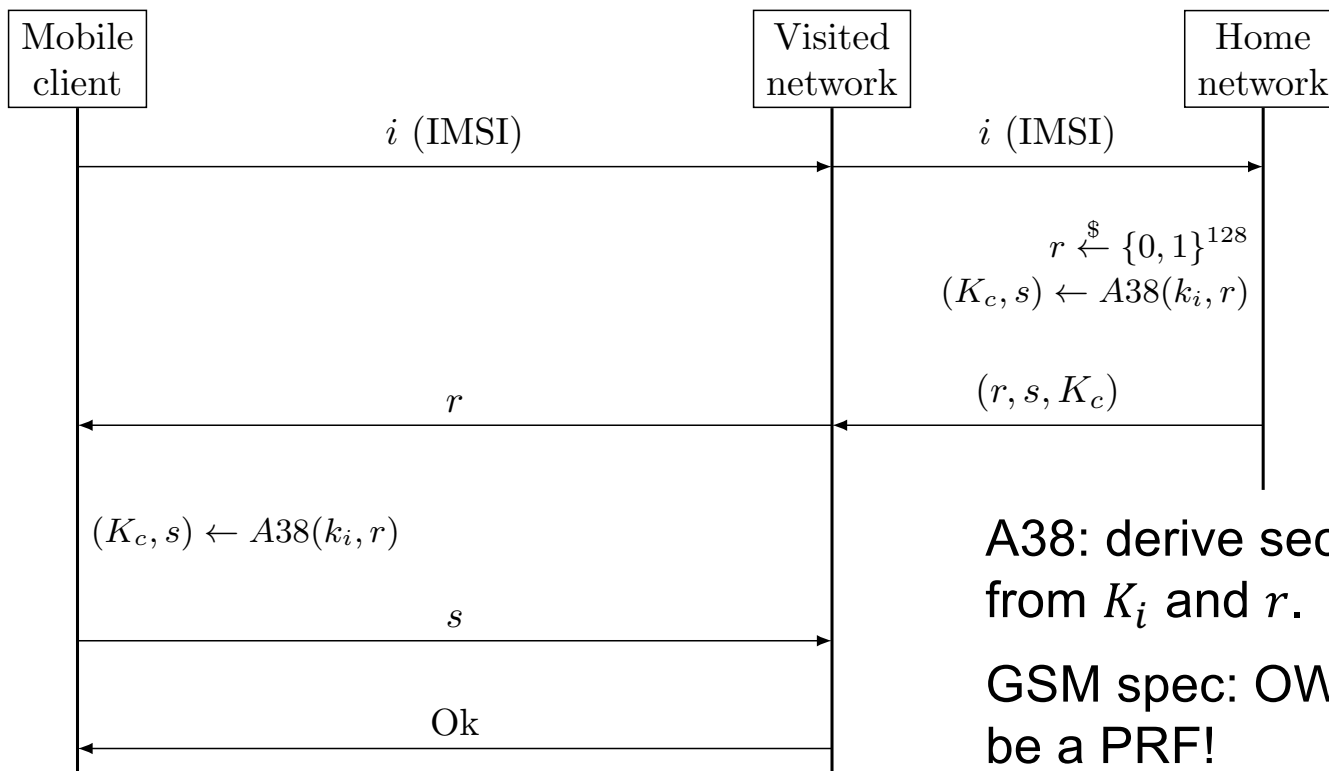❑ KDC shares keys k_A^E (enc.), k_A^M (MAC) with Alice and k_B^E, k_B^M with Bob

❑ Goal: Alice and Bob share k_{AB}, then derive: k_{AB}^E, k_{AB}^M

❑ KDC performs access control as well; controlling whom Alice can contact.

| Alice | | KDC | Bob |

'Bob', $time$, $MAC_{k_A^M}(time + \text{'Bob'})$

$c_A = E_{k_A^E}(k_{AB})$, $m_A = MAC_{k_A^M}(time + \text{'Bob'} + c_A + c_B + m_B)$
$c_B = E_{k_B^E}(k_{AB})$, $m_B = MAC_{k_B^M}(time + \text{'Alice'} + c_B)$

Use m_A to validate c_A, then extract k_{AB};
$k_{AB}^M \leftarrow PRF_{k_{AB}}(\text{'MAC'})$, $k_{AB}^E \leftarrow PRF_{k_{AB}}(\text{'Enc'})$

c_B, m_B, $c_{Req} = E_{k_{AB}^E}(\text{Request})$, $m_{Req} = MAC_{k_{AB}^M}(1 + A \rightarrow B + time + c_{Req})$

Validate and decrypt c_B,
and derive k_{AB}^E, k_{AB}^M

$c_{Resp} = E_{k_{AB}^E}(\text{Response})$, $m_{Resp} = MAC_{k_{AB}^M}(2 + A \leftarrow B + time + c_{Resp})$
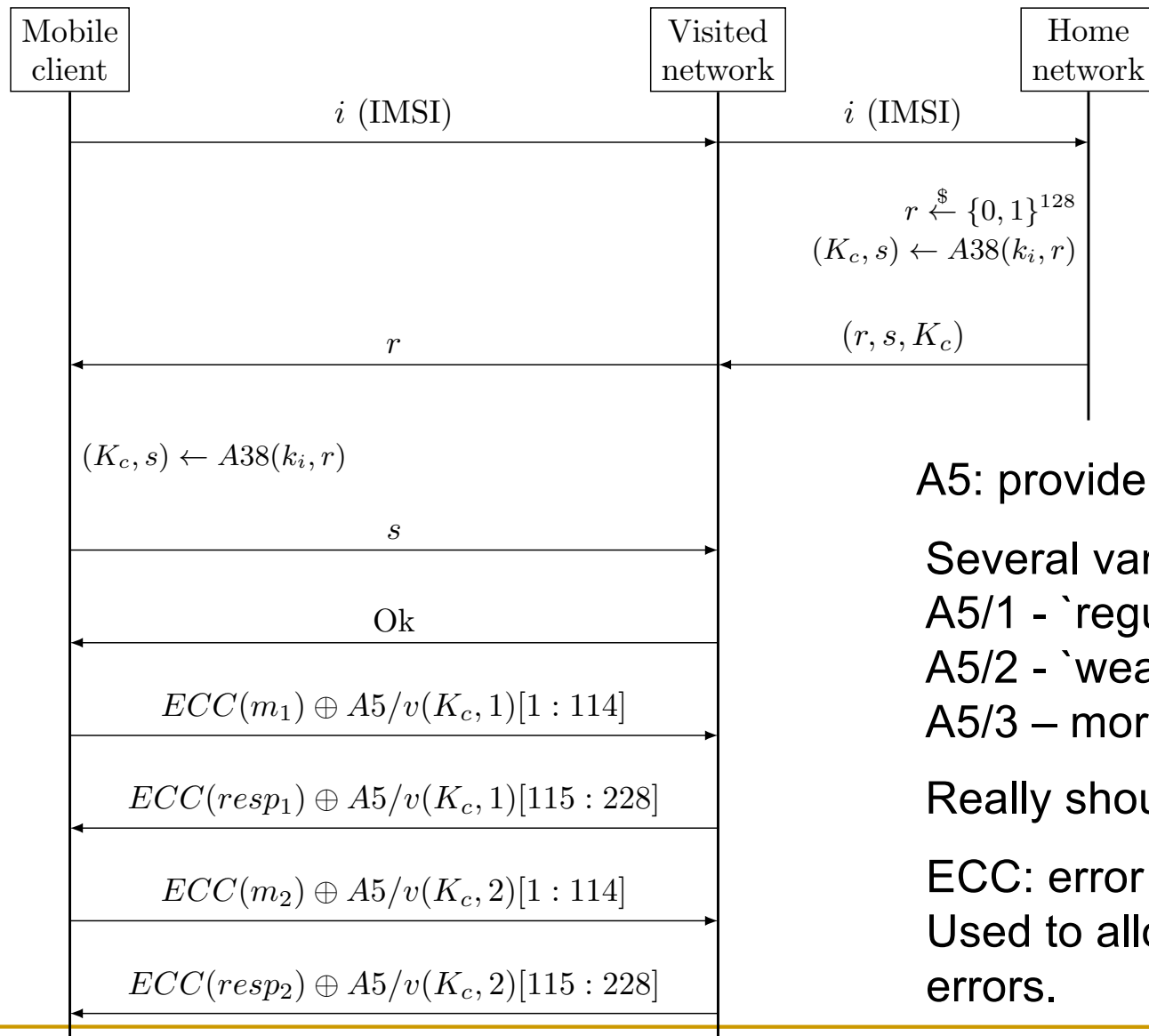
14
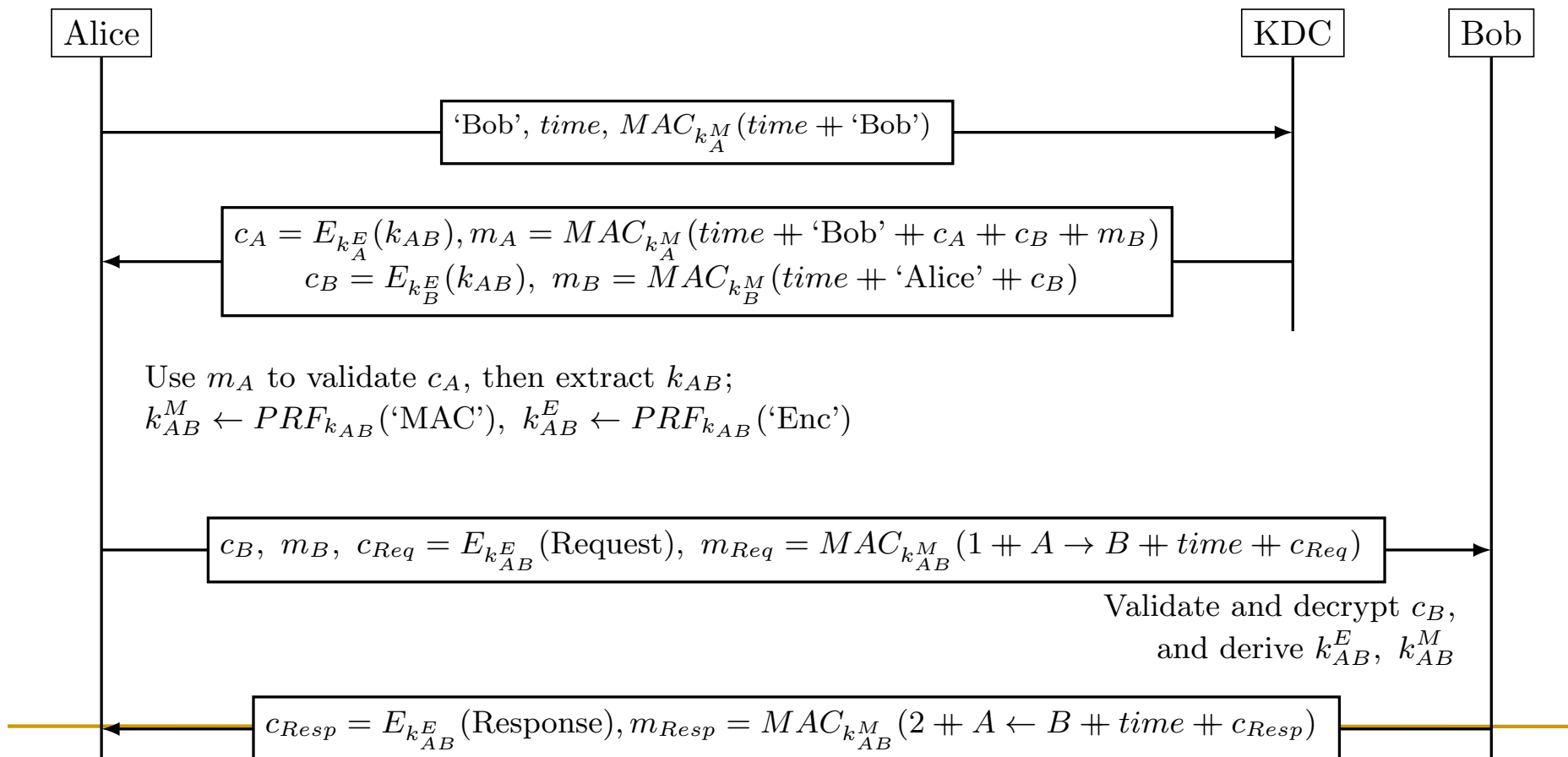
The GSM Handshake Protocol

- ❑ **Mobile client**
  - ❑ Identified by i (IMSI: International Mobile Subscriber Identifier)
- ❑ **Visited network (aka Base station); not fully trusted !**
- ❑ **Home network; trusted, shares key k_i with client i**

```
Mobile                          Visited                  Home
client                          network                  network
```

Mobile client → Visited network: i (IMSI)

Visited network → Home network: i (IMSI)

Home network:
$$r \xleftarrow{\$} \{0,1\}^{128}$$
$$(K_c, s) \leftarrow A38(k_i, r)$$

Home network → Visited network: (r, s, K_c)

Visited network → Mobile client: r

Mobile client:
$$(K_c, s) \leftarrow A38(k_i, r)$$

Mobile client → Visited network: s

Visited network → Mobile client: Ok

A38: derive secret, random K_c, s , from K_i and r.

GSM spec: OWF, but really should be a PRF!

Example – Sending two messages

Mobile client	Visited network	Home network

Mobile client → Visited network: i (IMSI)

Visited network → Home network: i (IMSI)

Home network:
$$r \xleftarrow{\$} \{0,1\}^{128}$$
$$(K_c, s) \leftarrow A38(k_i, r)$$

Home network → Visited network: (r, s, K_c)

Visited network → Mobile client: r

Mobile client:
$$(K_c, s) \leftarrow A38(k_i, r)$$

Mobile client → Visited network: s

Visited network → Mobile client: Ok

Mobile client → Visited network: $ECC(m_1) \oplus A5/v(K_c, 1)[1 : 114]$

Visited network → Mobile client: $ECC(resp_1) \oplus A5/v(K_c, 1)[115 : 228]$

Mobile client → Visited network: $ECC(m_2) \oplus A5/v(K_c, 2)[1 : 114]$

Visited network → Mobile client: $ECC(resp_2) \oplus A5/v(K_c, 2)[115 : 228]$

Kc is the session key
s is called a secret authenticator

A5: provide 'pad' for encryption

Several variants:
A5/1 - `regular'
A5/2 - `weak'
A5/3 – more secure

Really should be a PRF!

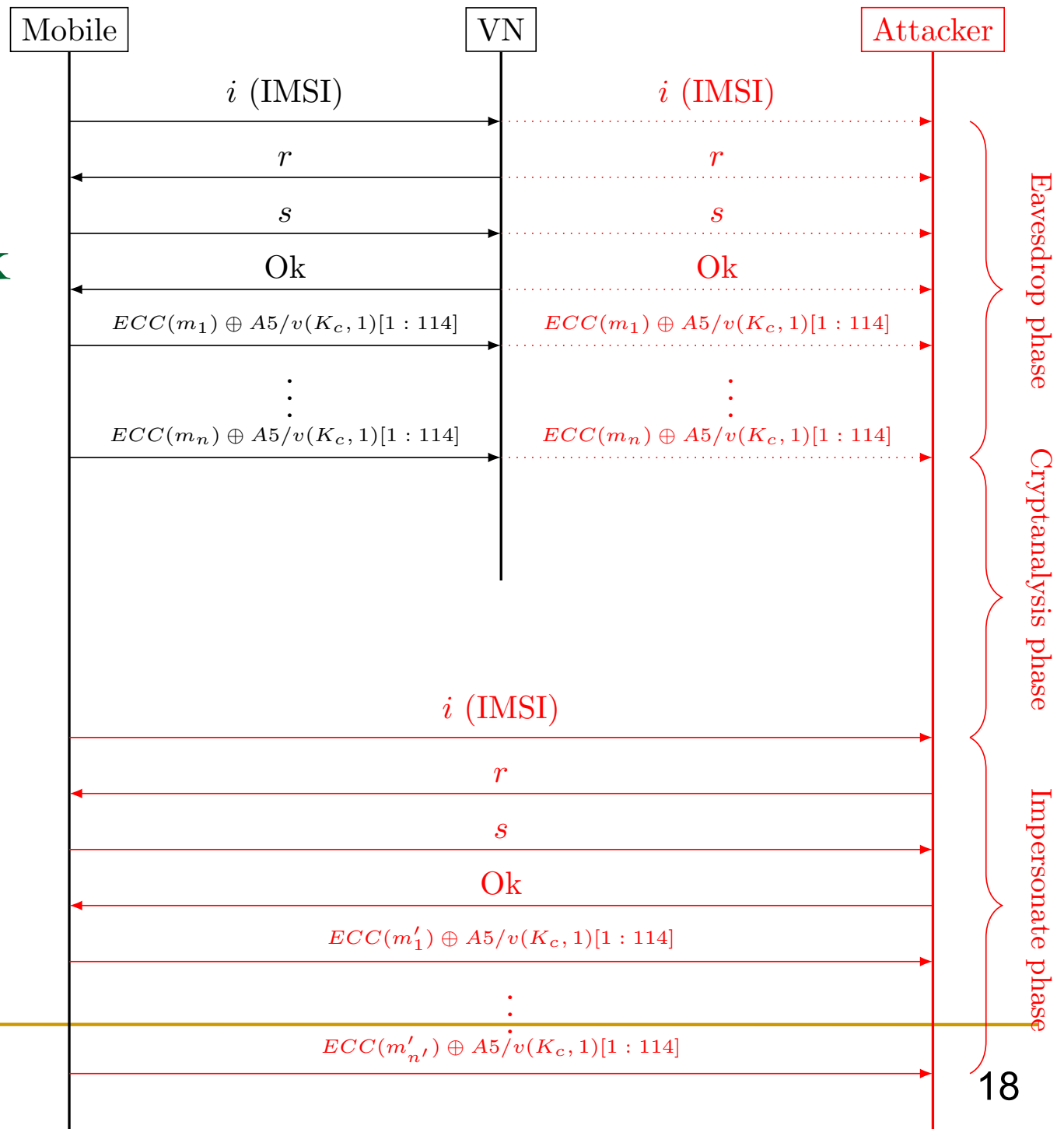ECC: error correcting code. Used to allow recovery from errors.

Attacks on GSM

- We will explore two such attacks:
    - Visited network impersonation replay attack.
    - Downgrade attack.

Visited-network Impersonation Attack

Note: does NOT Impersonate **mobile,** only Visited network.

In the cryptanalysis phase, the attacker will try to obtain Kc based on the cyphertexts it collected in the eavesdropping phase (recall A5/1 and A5/2 are not secure)

Mobile — VN — Attacker

i (IMSI) i (IMSI)

r r

s s

Ok   Ok

$ECC(m_1) \oplus A5/v(K_c, 1)[1:114]$ $ECC(m_1) \oplus A5/v(K_c, 1)[1:114]$

$\vdots$ $\vdots$

$ECC(m_n) \oplus A5/v(K_c, 1)[1:114]$ $ECC(m_n) \oplus A5/v(K_c, 1)[1:114]$

i (IMSI)

r

s

Ok

$ECC(m'_1) \oplus A5/v(K_c, 1)[1:114]$

$\vdots$

$ECC(m'_{n'}) \oplus A5/v(K_c, 1)[1:114]$

18

GSM Ciphersuites Downgrade Attack

- A ciphersuite is the set of cryptographic schemes used in a protocol execution.

- Ciphersuite negotiation:
    - Mobile sends list of cipher-suites it supports
    - Visited-network selects best one that it also supports

- GSM encryption algorithms E_k:
    - A5/0: none, A5/1: broken, A5/2: useless (break with only 1sec), A5/3: 'other'

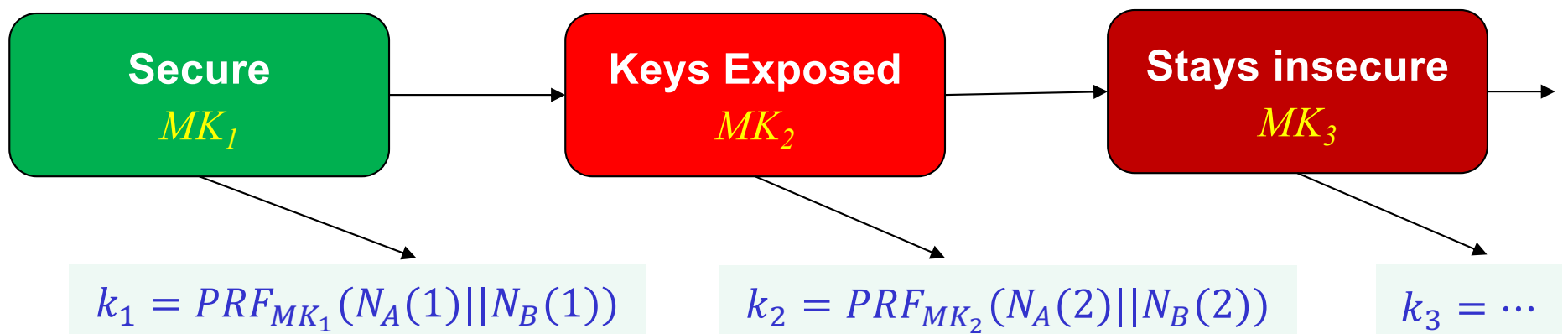- A MitM attacker may trick these parties to use a weak suite although the parties can support a stronger one.

Cipher mode messages, negotiation

- ❑ Mobile sends list of supported ciphers
- ❑ VN sends choice in: CIPHMODCMD
  - ❑ **Cipher Mode Command**
- ❑ Mobile confirms by sending <u>encrypted:</u> CIPHMOD**COM: cipher mode complete**
  - ❑ If not received (in few msecs), VN disconnects
- ❑ VN Acks: CIPHMOD**OK: cipher mode Ok**
  - ❑ If not received, mobile resends CIPHMOD**COM**
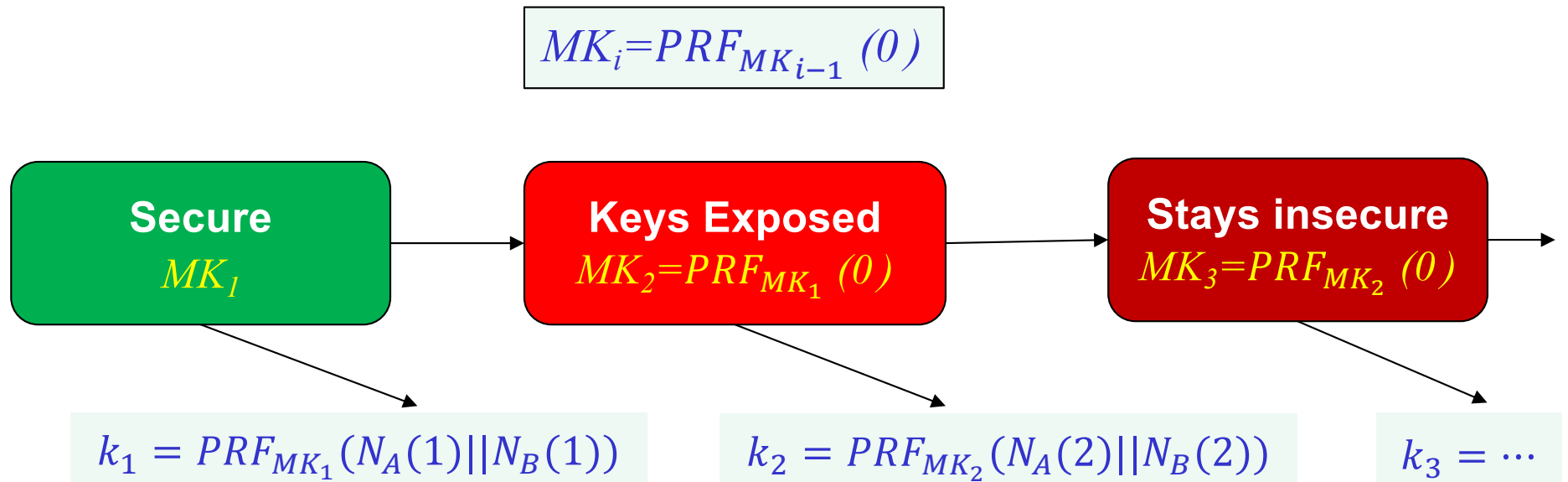- ❑ *Details in the textbook*

Improving Resiliency to Key Exposure

Forward Secrecy I

- **So far:** session key $k_i \not\Rightarrow k_j$ (expose no other keys)
  - And master key was fixed for all sessions
- Idea: we can do better!
  - Change the master key each session: MK_1, MK_2,...
- Forward Secrecy (FS): <u>master</u> key $MK_i \not\Rightarrow k_j (j < i)$
  - I.e., MK_i (and k_i) don't expose keys, communication of <u>previous</u> sessions $(j < i)$

Secure MK_1	Keys Exposed MK_2	Stays insecure MK_3

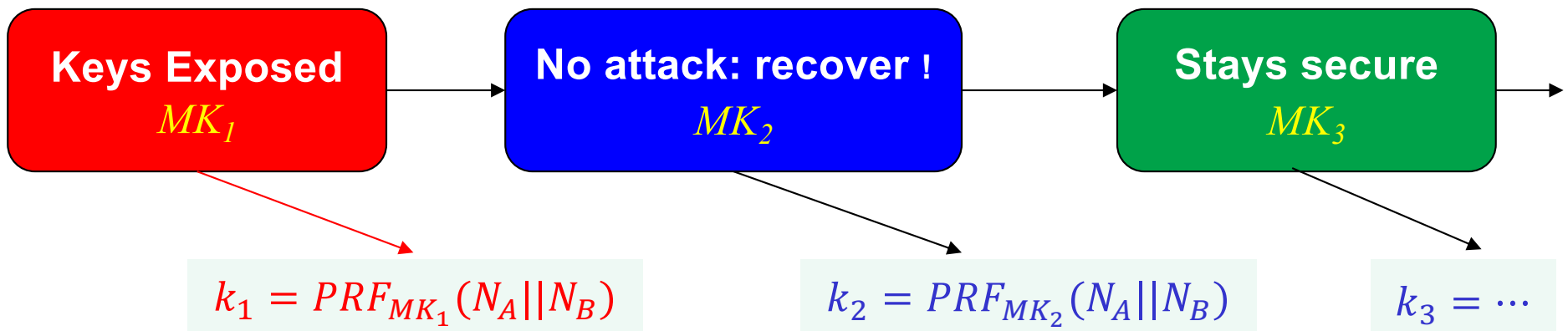$k_1 = PRF_{MK_1}(N_A(1)||N_B(1))$ $k_2 = PRF_{MK_2}(N_A(2)||N_B(2))$ $k_3 = \cdots$

Forward Secrecy II

- **Forward Secrecy (FS)**: <u>master</u> key $MK_j \nRightarrow k_i (j > i)$
  - Session i is secret even if any state of later sessions is exposed.
  - Uni-directional: $MK_i \rightarrow MK_{i+1}$, but $MK_{i+1} \nrightarrow MK_i$
  - **How?** Solution: PRF!

$$MK_i = PRF_{MK_{i-1}}(0)$$
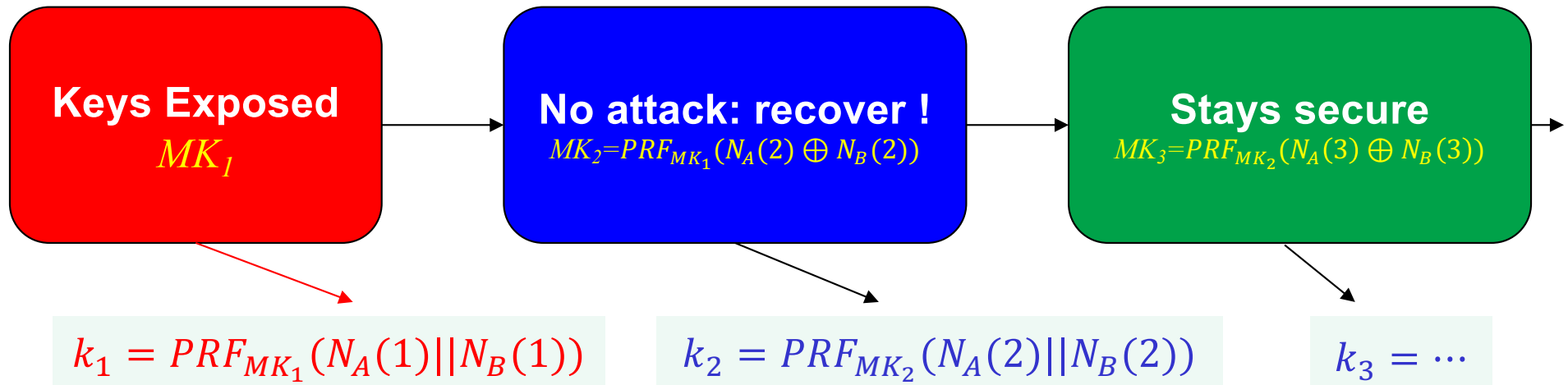
Secure	**Keys Exposed**	**Stays insecure**
MK_1	$MK_2 = PRF_{MK_1}(0)$	$MK_3 = PRF_{MK_2}(0)$

$k_1 = PRF_{MK_1}(N_A(1) \| N_B(1))$ $k_2 = PRF_{MK_2}(N_A(2) \| N_B(2))$ $k_3 = \cdots$

Recover Security

- Can we also **recover** security?
  - MK_{i_R-1} exposed, yet MK_{i_R}, MK_{i_R+1} ... secure ?
  - Idea: assume **no attack** during 'recovery session' i_R

| Keys Exposed MK_1 | → | No attack: recover ! MK_2 | → | Stays secure MK_3 | → |

$$k_1 = PRF_{MK_1}(N_A || N_B)$$

$$k_2 = PRF_{MK_2}(N_A || N_B)$$

$$k_3 = \cdots$$

Recover Security (RS)

- **Recover security:** key setup protocols where a single session without eavesdropping or other attacks, suffices to recover security from previous key exposures.

- That is, session i is secure if it's keys are not given to attacker, and either session $i - 1$ is secure, or there is no attack during session i

- How? The RS-Ratchet Protocol:

  - Let $N_A(i), N_B(i)$ denote session's i nonces
  - Then: $MK_i = PRF_{MK_{i-1}}(N_A(i) \oplus N_B(i))$

Keys Exposed	No attack: recover !	Stays secure
MK_1	$MK_2 = PRF_{MK_1}(N_A(2) \oplus N_B(2))$	$MK_3 = PRF_{MK_2}(N_A(3) \oplus N_B(3))$

$$k_1 = PRF_{MK_1}(N_A(1) || N_B(1)) \qquad k_2 = PRF_{MK_2}(N_A(2) || N_B(2)) \qquad k_3 = \cdots$$

Covered Material From the Textbook

- Chapter 5
  - Sections 5.3 – 5.7 (except 5.7.3)

Thank You!