SAS-based Authenticated Key Agreement

Sylvain Pasini and Serge Vaudenay

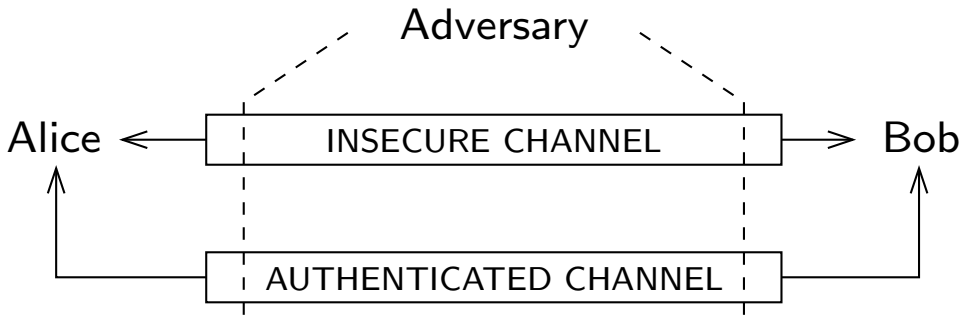
LASEC EPFL

Public Key Cryptography '06

April the 26^{th} 2006

Setting up a Secure Communication

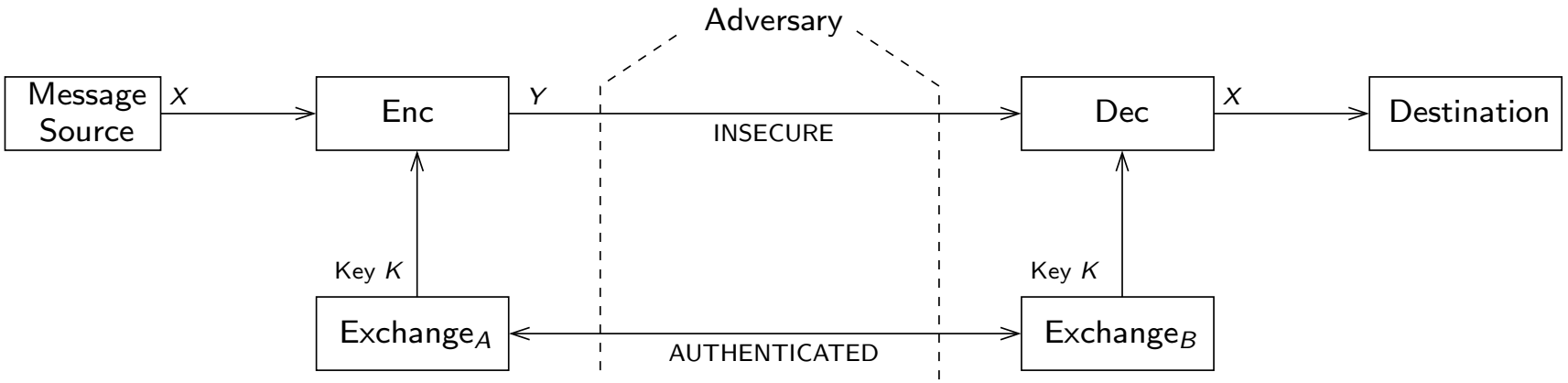
- Suppose Alice and Bob want to communicate securely:



- No prior exchanged key
- Insecure channel:
  - Adversaries have full control.
- Authenticated channel:
  - Authenticity and integrity are protected.
  - Anyone can read, delay, remove, ...

Public Key Cryptography

The Merkle-Diffie-Hellman model:



- E.g. using the Diffie-Hellman protocol.
- We no longer need confidentiality.
- An authenticated (extra) channel is enough.

Authenticated Key Agreement

In a nutshell:

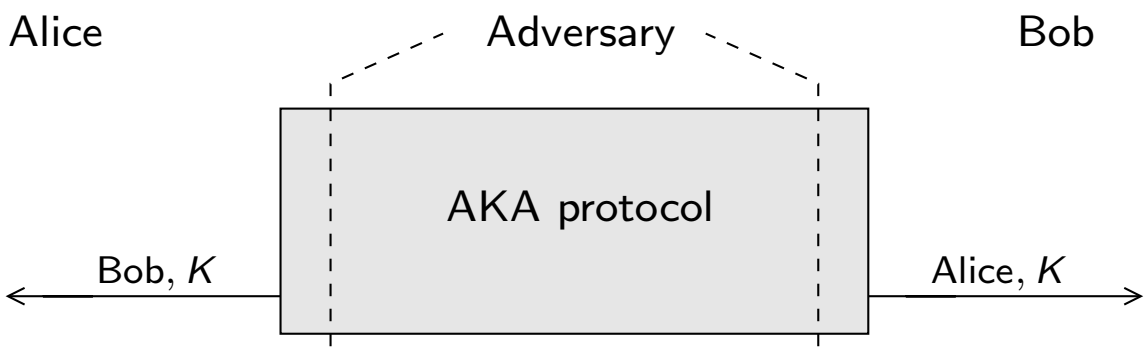
- A secure communication can be setup
  with an Authenticated Key Agreement (AKA).

Objective

Reduce the amount of authenticated data.

AKA as a building block

- Alice and Bob agree on a same key (i.e. K).




Early Proposals

PGPfone Key Agreement Protocol (1995)

A 3-move AKA:

Alice		**Bob**
pick x_A, $y_A \leftarrow g^{x_A}$		pick x_B, $y_B \leftarrow g^{x_B}$
	$\xrightarrow{\text{commit}(y_A)}$	
	$\xleftarrow{y_B}$	
$z_A \leftarrow \hat{y}_B^{x_A}$	$\xrightarrow{\text{open}(y_A)}$	$z_B \leftarrow \hat{y}_A^{x_B}$
$\text{SAS} \leftarrow \text{truncH}(y_A \Vert \hat{y}_B)$	$\xrightarrow{\text{authenticate}_{\text{Alice}}(\text{SAS})}$	$\text{SAS} \stackrel{?}{=} \text{truncH}(\hat{y}_A \Vert y_B)$
check SAS is the same	$\xleftarrow{\text{authenticate}_{\text{Bob}}(\text{SAS})}$	
final state: Bob, z_A		**final state**: Alice, z_B

Problem: no security proof

Hoepman Key Agreement Protocol (2004)

A 4-move AKA:

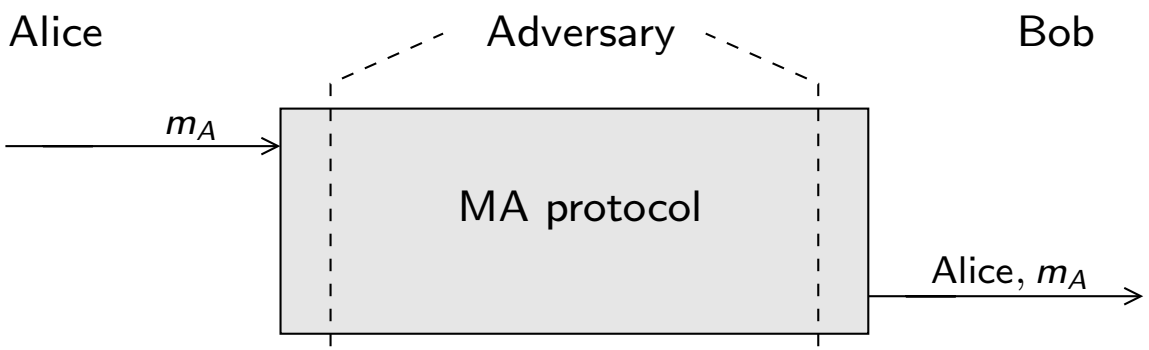
Alice		**Bob**
pick x_A, $y_A \leftarrow g^{x_A}$	$\xrightarrow{\text{commit}(y_A)}$	pick x_B, $y_B \leftarrow g^{x_B}$
	$\xleftarrow{\text{commit}(y_B)}$	
$\text{SAS}_A \leftarrow \text{pieceof}(y_A)$	$\xrightarrow{\text{authenticate}_{\text{Alice}}(\text{SAS}_A)}$	
	$\xleftarrow{\text{authenticate}_{\text{Bob}}(\text{SAS}_B)}$	$\text{SAS}_B \leftarrow \text{pieceof}(y_B)$
	$\xrightarrow{\text{open}(y_A)}$	check commit, SAS
check commit, SAS	$\xleftarrow{\text{open}(y_B)}$	
$z_A \leftarrow (\hat{y}_B)^{x_A}$		$z_B \leftarrow (\hat{y}_A)^{x_B}$
final state: Bob, z_A		**final state**: Alice, z_B

Problem: no security proof


Recalls on Message Authentication

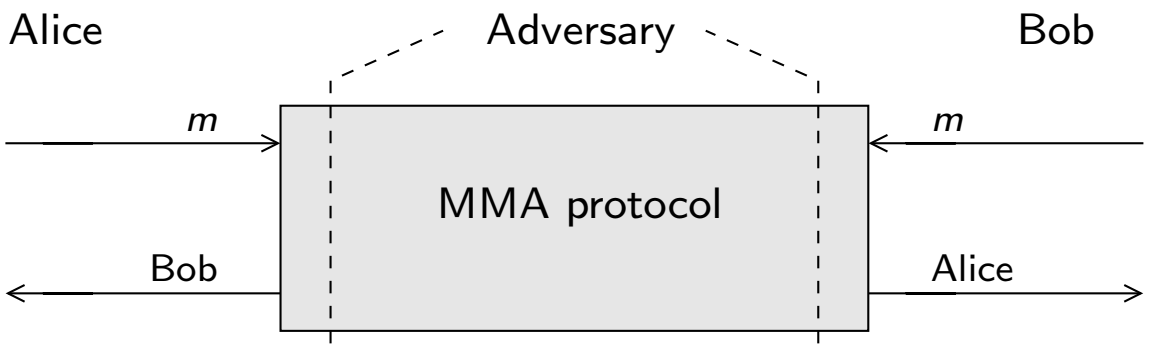
Message Authentication Protocol

- Alice authenticates her message to Bob (i.e. m_A).



Message Mutual-Authentication (MMA) Protocol

- Alice and Bob agree on the same message (i.e. $m_A = m_B$).

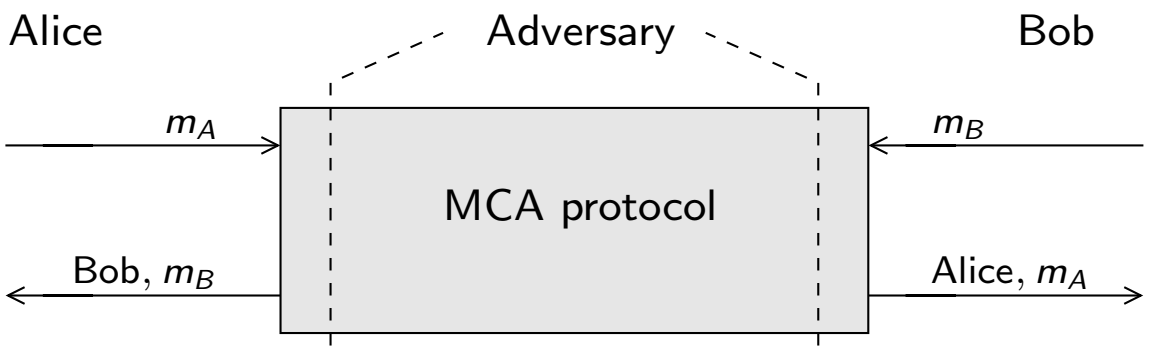


- A trivial 0-move MMA:

Alice		**Bob**
input: m_A		**input**: m_B
$\mathsf{SAS} \leftarrow H(m_A)$	$\xrightarrow{\text{authenticate}_{\text{Alice}}(\mathsf{SAS})}$	$\mathsf{SAS} \stackrel{?}{=} H(m_B)$
check SAS is the same	$\xleftarrow{\text{authenticate}_{\text{Bob}}(\mathsf{SAS})}$	
output: Bob		**output**: Alice

Message Cross-Authentication (MCA) Protocol

- Alice and Bob authenticate their message (i.e. m_A, m_B).



MCA from MMA

Suppose an n-move MMA with $n \geq 1$.

We can build a $(n+1)$-move MCA as follows:

1. Bob first sends m_B to Alice (extra move).
2. Alice starts the MMA with input $m_A||\hat{m}_B$
   - Alice sends m_A with the first message.
   - Bob follows the MMA protocol with input $\hat{m}_A||m_B$.

Vaudenay's SAS-based MCA protocol (2005)

A 4-move MCA:

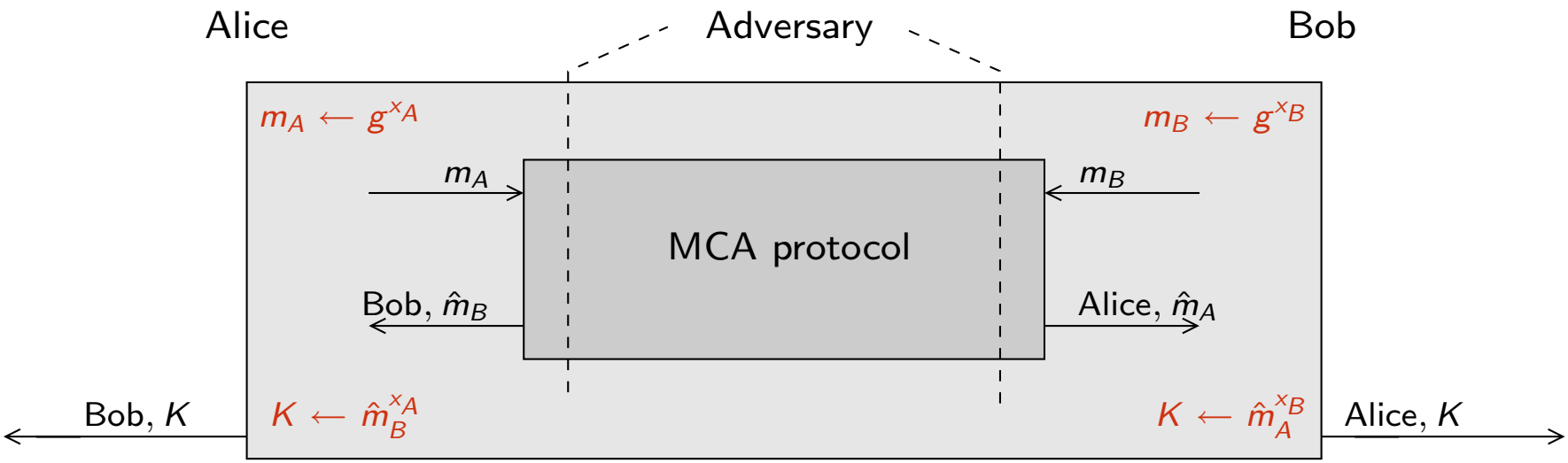
Alice		**Bob**
input: m_A		**input**: m_B
pick $R_A \in_U \{0,1\}^k$		pick $R_B \in_U \{0,1\}^k$
$(c_A, d_A) \leftarrow \text{commit}(0\|\|m_A, R_A)$	$\xrightarrow{m_A, c_A}$	
	$\xleftarrow{m_B, c_B}$	$(c_B, d_B) \leftarrow \text{commit}(1\|\|m_B, R_B)$
	$\xrightarrow{d_A}$	$\hat{R}_A \leftarrow \text{open}(0\|\|\hat{m}_A, \hat{c}_A, \hat{d}_A)$
$\hat{R}_B \leftarrow \text{open}(1\|\|\hat{m}_B, \hat{c}_B, \hat{d}_B)$	$\xleftarrow{d_B}$	
$\text{SAS} \leftarrow R_A \oplus \hat{R}_B$	$\xrightarrow{\text{authenticate}_{\text{Alice}}(\text{SAS})}$	$\text{SAS} \stackrel{?}{=} \hat{R}_A \oplus R_B$
check SAS is the same	$\xleftarrow{\text{authenticate}_{\text{Bob}}(\text{SAS})}$	
output: Bob, $\hat{m}_B$		**output**: Alice, $\hat{m}_A$


Reducing Key Agreement to
Message Authentication

An Example of Construction

We can build an AKA in two steps

1. Both users run a MCA protocol.
2. Input messages are DH values g^{x_A}, g^{x_B}.



A Generic Construction

1. Run an AKA_0 (over the insecure channel).
2. Run a MCA,
   - input messages are the view of the protocol transcript
   - AKA_0's last message is send with the MCA's first (each user saves a move)

Theorem 1

Consider:

- an n_k-move (T, ε_k)-AKA_0 protocol
- an n_a-move (T, ε_a)-MCA protocol

There exists a μ s.t. the generic construction is an

$$\max(n_k, n_k + n_a - 2)\text{-move } (T\mu, \varepsilon_k + \varepsilon_a)\text{-AKA protocol.}$$

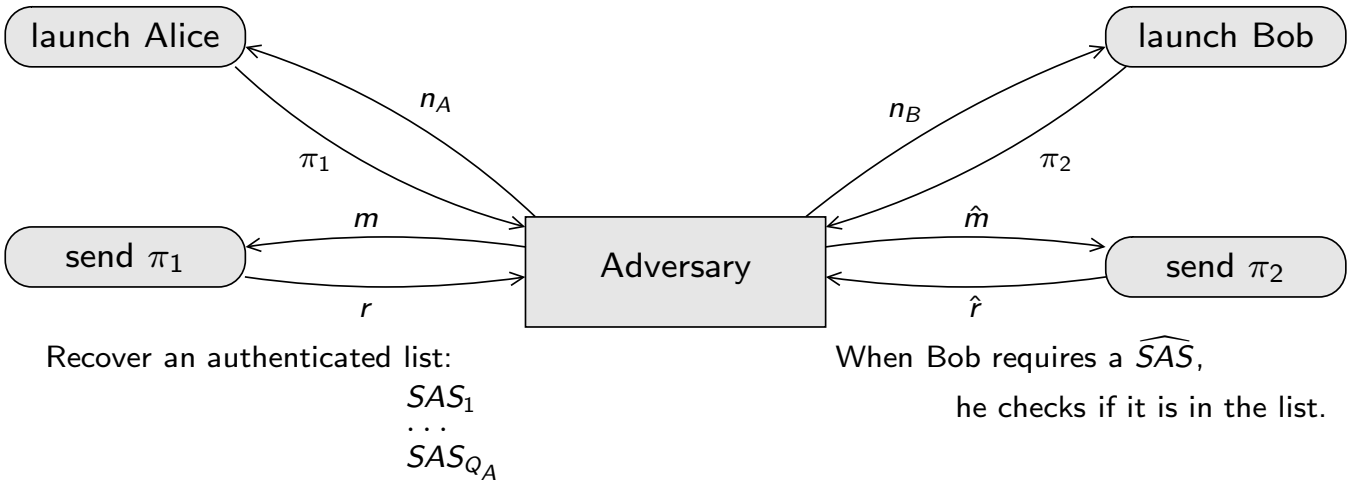
Adversarial model against MCA protocols

Adversaries are generally bounded by:

- complexity T,
- Q_A (Q_B) instances of Alice (Bob),

Successful if some (incorrupted) instance

- outputs (m, ID)
- no instance with identity ID was lauched with input m




A New MMA Protocol

Our New SAS-Based MMA Protocol

Our new 3-move MMA:

Alice		**Bob**
input: m_A		**input**: m_B
pick $R_A \in_U \{0,1\}^k$		pick $R_B \in_U \{0,1\}^k$
$(c,d) \leftarrow \text{commit}(m_A, R_A)$	$\xrightarrow{c}$	
	$\xleftarrow{R_B}$	
	$\xrightarrow{d}$	$\hat{R}_A \leftarrow \text{open}(m_B, \hat{c}, \hat{d})$
$\text{SAS} \leftarrow R_A \oplus \hat{R}_B$	$\xrightarrow{\text{authenticate}_{\text{Alice}}(\text{SAS})}$	$\text{SAS} \stackrel{?}{=} \hat{R}_A \oplus R_B$
check SAS is the same	$\xleftarrow{\text{authenticate}_{\text{Bob}}(\text{SAS})}$	
output: Bob		**output**: Alice

Our New SAS-Based MMA Protocol

Theorem 2

Consider any adversary $\mathcal{A}$ bounded by

- complexity T
- Q_A instances of Alice
- Q_B instances of Bob.

Assume we have

- (T_C, ε)-secure equivocable commitment scheme.

There exists a (small) constant μ such that $\mathcal{A}$ wins

- either with $\Pr[\text{success}] \leq Q_A \cdot (Q_A + Q_B) \cdot (2^{-k} + \varepsilon)$
- or with complexity $T \geq T_C - \mu$.

Our New SAS-Based MMA Protocol (proof)

Any adversary $\mathcal{A}$ has to replace (c, d) or R_B by $(\hat{c}, \hat{d})$ or $\hat{R}_B$.

pick $R_A \in_U \{0,1\}^k$		pick $R_B \in_U \{0,1\}^k$
$(c, d) \leftarrow \text{commit}(m_A, R_A)$	$\xrightarrow{c}$	
	$\xleftarrow{R_B}$	
	$\xrightarrow{d}$	$\hat{R}_A \leftarrow \text{open}(m_B, \hat{c}, \hat{d})$
$\text{SAS} \leftarrow R_A \oplus \hat{R}_B$	$\xrightarrow{\text{authenticate}_{\text{Alice}}(\text{SAS})}$	$\text{SAS} \stackrel{?}{=} \hat{R}_A \oplus R_B$
check SAS is the same	$\xleftarrow{\text{authenticate}_{\text{Bob}}(\text{SAS})}$	

Whatever the target, $\mathcal{A}$ needs an authenticated SAS.
He uses an Alice to get it $(Q_A(Q_A + Q_B - 1))$.

Target Alice:

- He has to realease an $\hat{R}_B$ by guessing R_A (hiding game).

Target Bob:

- He has to realease a $\hat{d}$ s.t. $\hat{R}_A = R_A \oplus R_B \oplus \hat{R}_B$ (binding game).

$$\Pr[\text{success}] \leq Q_A(Q_A + Q_B)\ \left(2^{-k} + \varepsilon\right)$$


A New MCA Protocol

Our New SAS-Based MCA Protocol

Our new 3-move MCA:

Alice		**Bob**
input: m_A		**input**: m_B
pick $K \in_U \{0,1\}^\kappa$		pick $R \in_U \{0,1\}^\rho$
$(c,d) \leftarrow \mathsf{commit}(m_A, K)$	$\xrightarrow{m_A, c}$	
	$\xleftarrow{m_B, R}$	
	$\xrightarrow{d}$	$\hat{K} \leftarrow \mathsf{open}(\hat{m}_A, \hat{c}, \hat{d})$
$\mathsf{SAS} \leftarrow \hat{R} \oplus h_K(\hat{m}_B)$	$\xrightarrow{\mathsf{authenticate}_{\mathsf{Alice}}(\mathsf{SAS})}$	$\mathsf{SAS} \stackrel{?}{=} R \oplus h_{\hat{K}}(m_B)$
check SAS is the same	$\xleftarrow{\mathsf{authenticate}_{\mathsf{Bob}}(\mathsf{SAS})}$	
check Alice $\neq$ Bob		
output: Bob, $\hat{m}_B$		**output**: Alice, $\hat{m}_A$

We have a 3-move AKA (e.g. using DH).

Our New SAS-Based MCA Protocol

Theorem 3

Consider:

- an ε-random oracle commitment scheme
- an ε_h-universal hash function family h

Any adversary bounded by Q instances of Alice or Bob cannot succeed with probability higher than

$$Q^2(2^{-\rho} + \varepsilon + \varepsilon_h).$$

Applications

Secure VoIP:

- **Start** in an **insecure** mode.
  - The channel is authenticated (not confidential).
- Run a DH through our SAS-based MCA.
  Now,
  - both share a secret key K (thanks to DH)
  - K is cross-authenticated (thanks to our MCA)
- **Switch** to a **secure** conversation.
  - Use K with standard cryptography, e.g. AES.
  - The channel is now confidential and authenticated!

Summary of our results

We proposed a **new MCA** protocol:

- Based on SAS
- 3 moves only
- Same SAS in both directions
- Requires random oracle commitment
- Optimal

Now, we have

a **3-move SAS-based AKA**.

Future work:

- Try to avoid the use of the random oracle...