Seguridad en Sistemas de Información

Curso en Tamaulipas [Q2 2014]

Luis J. Dominguez Perez

Cinvestav, Junio 11 de 2014 - L7

Contenido, sección 1

Group

A group $< G, \circ >$ is a non-empty set G together with a binary operation $\circ$ such that:

- *It is closed*
- *it is associative*
- *has an identity and inverse element*

A group G is **Abelian** (or **commutative**) if $a \circ b = b \circ a$, $\forall a, b \in G$.

A group is **finite** if G has a finite number of elements. This is called the **order** of G and denoted as $|G|$.

Finite field

A field F is a group with $+$, $\times$ operations as $(F, +)$ and $(F \backslash \{0\}, \times)$ which also satisfies:

- *Additive identity and inverse*
- *Multiplicative identity and inverse*
- *Commutative*

i.e. the set of integers modulo p-prime, also denoted as $\mathbb{F}_p$, is a finite field.

Elliptic curves over finite fields

Let p-prime > 3. The elliptic curve

$$y^2 = x^3 + ax + b, \qquad \text{over } \mathbb{F}_p$$

denoted by $E(\mathbb{F}_p)$, is the set of solutions $x, y \in \mathbb{F}_p$ satisfying

$$y^2 \equiv x^3 + ax + b \bmod p$$

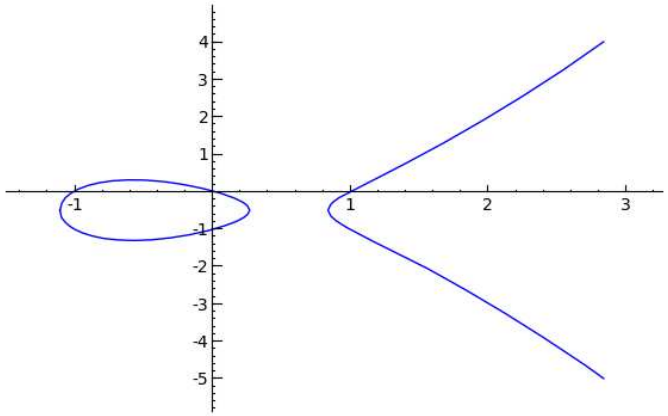
where $a, b \in \mathbb{F}_p$ and

$$4a^3 + 27b^2 \neq 0 \bmod p$$

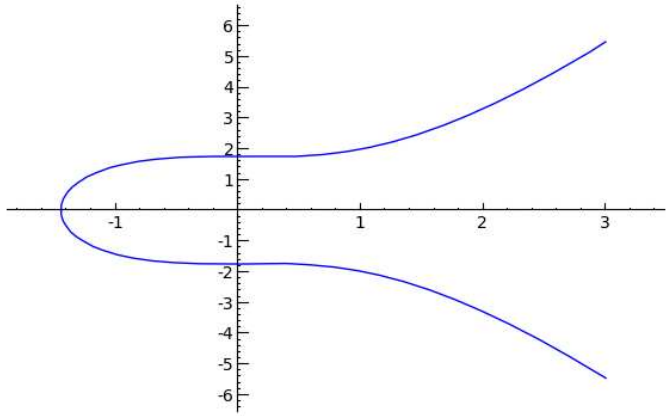
together with the point $\mathcal{O}$

The **order**, or number of points on $E(\mathbb{F}_p)$ is denoted as $\#E(\mathbb{F}_p) = p + 1 \pm t$ and $t \leq 2\sqrt{p}$.

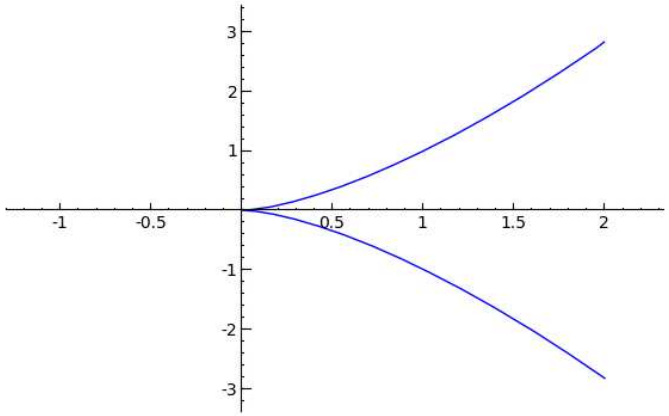
Types of elliptic curves (over $\mathbb{C}$)



With 3 distinct real roots



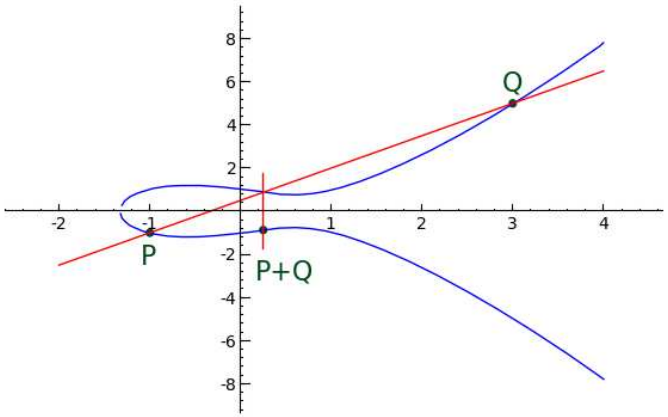
With 1 real and 2 complex roots



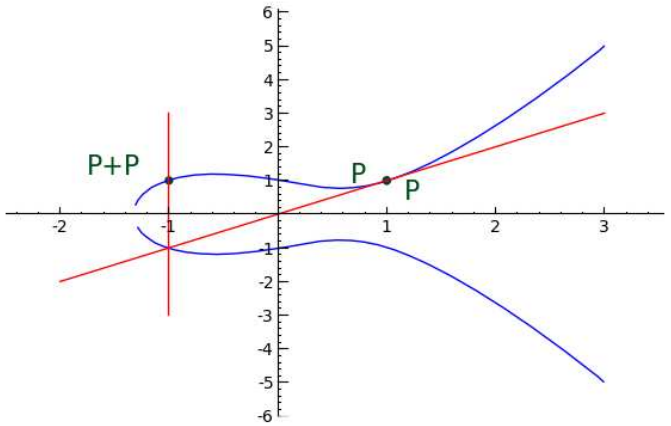
With a triple real root

The group law on EC

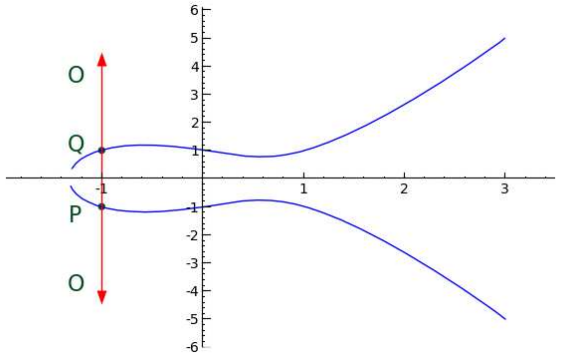
Suppose $P = (x_1, y_1)$ and $Q = (x_2, y_2)$ with $P, Q \in E(\mathbb{F}_p)$ $P + Q = (x_3, y_3)$, where $x_3 = \lambda^2 - x_1 - x_2, y_3 = \lambda(x_1 - x_3) - y_1$



$$\lambda = \frac{y_2 - y_1}{x_2 - x_1}$$



$$\lambda = \frac{3x_1^2 + a}{2y_1}$$



$$P + Q = \mathcal{O} \text{ or } Q = -P$$

Discrete logarithm problem

Let $P = (x, y) \in E(\mathbb{F}_p)$ and $n \in \mathbb{Z}$, $n \geq 0$. Define $[n]P = P + P + \cdots + P$. The **order of the point** P is the smallest n such that $[n]P = \mathcal{O}$.

Denote $< P >$ the group generated by P. In other words,

$$< P >= \{\mathcal{O}, P, P + P, P + P + P, ...\}$$

Let $Q \in < P >$. Given Q, find n such that $Q = [n]P$. This is known as the **Elliptic Curve Discrete Logarithm Problem (ECDLP)**.

Known attacks affect some anomalous curves, P with a small prime order and some weak combinations of parameters.

Contenido, sección 2

DSA vs ECDSA

EC version of Elgamal encryption

Public parameter generation A trusted partge chooses and publishes a prime p an elliptic curve E over $\mathbb{F}_p$, and a point $P \in E(\mathbb{F}_p)$	
Alice	Bob
Key Generation	
Chooses a private key n_A Computes $Q_A = [n_A]P \in E(\mathbb{F}_p)$ Publishes Q_A	
Encryption	
	Chooses plaintext $M \in E(\mathbb{F}_p)$ Chooses an ephemeral key k Uses Alice's public key Q_A to compute $C_1 = [k]P \in E(\mathbb{F}_p)$ and $C_2 = M + [k]Q_A \in E(\mathbb{F}_p)$ Send ciphertext $(C_1, C2)$
Decryption	
Computes $C_2 - [n_A]C_1 \in E(\mathbb{F}_p)$ This value is equal to M	

Contenido, sección 3

Some complains

- what happens if the user is not explicitly in the system?
- or what about giving access to a group of users with the same characteristics?

Some complains

- what happens if the user is not explicitly in the system?
- or what about giving access to a group of users with the same characteristics?

Wait!

Some complains

- what happens if the user is not explicitly in the system?
- or what about giving access to a group of users with the same characteristics?

Wait! Why would someone would like to encrypt something for a non-existant user?... That's non-sense!

or is it?

What's next?

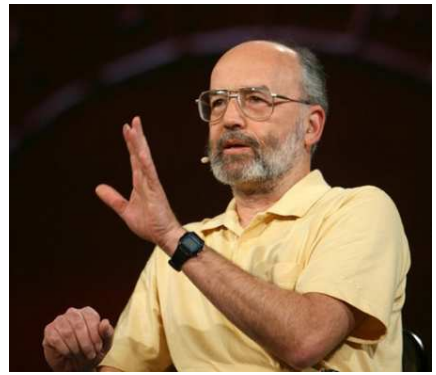
Are there anything we can do to solve all of this?

What's next?

Are there anything we can do to solve all of this?

Yes! We can use the Pairing-Based Cryptography.

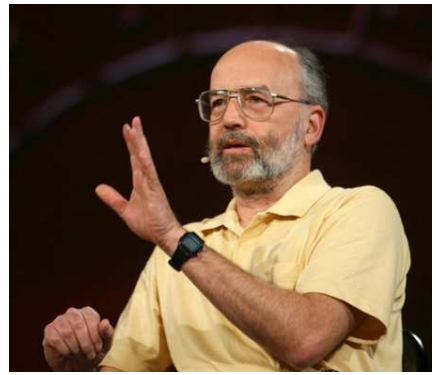
How does it started?



In 1984, Shamir posed a challenge:

"create a cryptographic system that permits any two users to comunicate securely and to verify each other's signatures without exchanging private or public keys, without keeping key directories, and without using the services of a third party."

How does it started?



In 1984, Shamir posed a challenge:

"create a cryptographic system that permits any two users to comunicate securely and to verify each other's signatures without exchanging private or public keys, without keeping key directories, and without using the services of a third party."

How does it started? 2

However, in 2001, Boneh and Franklin, solved this challenge using cryptographic pairings. They presented what it is now called Identity-Based Encryption.

... also, in 2000, Antoine Joux presented a breaking-through paper involving pairings, but we are focusing in this talk on Identity-Based Encryption.

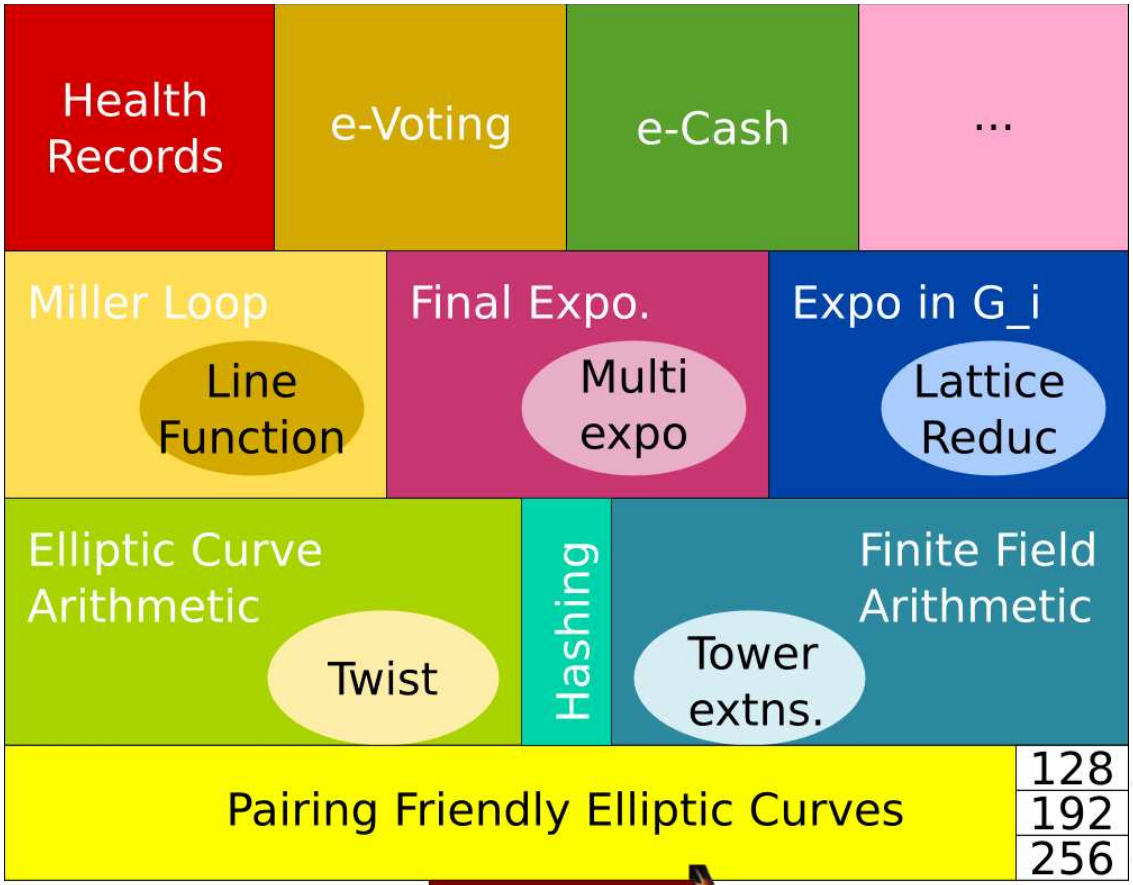
Pairing-Based Cryptography

Identity-Based Encryption is a type of the Pairing-Based Encryption, this is, we use some cryptographic function called the pairing.

In essence, a cryptographic pairing is a particular function of groups over elliptic curves.

$$\langle \cdot, \cdot \rangle : M \times M \longrightarrow R$$

Pairing-Based Cryptography



The bilinear pairing can be used as a primitive to build cryptosystems with certain functionality. Examples of use:

The bilinear pairing can be used as a primitive to build cryptosystems with certain functionality. Examples of use:

- Short signatures schemes,

The bilinear pairing can be used as a primitive to build cryptosystems with certain functionality. Examples of use:

- Short signatures schemes,
- Identity-Based Encryption,

The bilinear pairing can be used as a primitive to build cryptosystems with certain functionality. Examples of use:

- Short signatures schemes,
- Identity-Based Encryption,
- Attribute-Based Encryption,

The bilinear pairing can be used as a primitive to build cryptosystems with certain functionality. Examples of use:

- Short signatures schemes,
- Identity-Based Encryption,
- Attribute-Based Encryption,
- and other protocols already deployed.

Some protocols are impossible with currently deployed technology, in other cases, they are faster.

Example of PBC

Identity-Based Encryption case:

- Enables any pair of users to communicate securely and to verify each others' signatures **without exchanging** private or public keys;
- Needs **no key server repositories**;
- Requires a trusted server for key generation **only**.
- **No certificate required** to bind the public key to the identity.

Implementation issues...

Pairing-Based Cryptography has become relevant in industry.

Although there are plenty of applications, however efficiently implementing the pairings function is often difficult as it requires more knowledge than previous cryptographic primitives.

There are many implementation issues just with the primitive itself!

... implementation issues

- **Non-familiar** technology;
- Lack of **programming framework**;
- More **difficult to understand** compared to the already deployed technology;
- **Unavailability** of implementations with novel (faster) computing methods;
- Complex area.

Depending on the scenario, a developer must choose from a selection of parameters and apply the corresponding optimizations for efficiency...

What to do when... ?

- **bandwidth** use is expensive;
- **low memory** is available;
- a **slow** processor is used (old);
- a **small** processor (in bits) is the only option;
- we have a **Desktop** environment;
- we have a device with **multiprocessors**;
- a **higher security** is required;

Some basic operations that are cheap in some environments are expensive in others!

Protocol primitives

The operations involved in a Pairing-Based protocol are:

- The pairing function
- Elliptic Curve point addition and point doubling
- Scalar-point multiplication
- exponentiation
- hash onto a curve
- hash into a subgroup
- matrix conversion
- boolean function analysis. . .

Many more!

Pairing definition

A **pairing** is a map: $\mathbb{G}_1 \times \mathbb{G}_2 \rightarrow \mathbb{G}_T$.

These groups are finite and cyclic. $\mathbb{G}_1$ and $\mathbb{G}_2$ are additively-written and at least one is of prime order r. $\mathbb{G}_T$, is multiplicatively-written and of order r.

Properties:

- *Bilinearity*
- *Non-degeneracy*
- *Efficiently computable*

Pairing properties

Properties:

- *Bilinearity*

$$e(P + P', Q) = e(P, Q) \times e(P', Q)$$

$$e(P, Q + Q') = e(P, Q) \times e(P, Q')$$

Pairing properties

Properties:

- *Bilinearity*

  $e(P + P', Q) = e(P, Q) \times e(P', Q)$

  $e(P, Q + Q') = e(P, Q) \times e(P, Q')$

- *Non-degeneracy*

  $\forall P \in \mathbb{G}_1, P \neq \mathcal{O}$: $\exists Q \in \mathbb{G}_2$ **s.t.** $e(P, Q) \neq 1$

  $\forall Q \in \mathbb{G}_2, Q \neq \mathcal{O}$: $\exists P \in \mathbb{G}_1$ **s.t.** $e(P, Q) \neq 1$ $\quad$ $e(P, Q) \neq 1$

Pairing properties

Properties:

- *Bilinearity*

  $e(P + P', Q) = e(P, Q) \times e(P', Q)$

  $e(P, Q + Q') = e(P, Q) \times e(P, Q')$

- *Non-degeneracy*

  $\forall P \in \mathbb{G}_1, P \neq \mathcal{O}$: $\exists Q \in \mathbb{G}_2$ **s.t.** $e(P, Q) \neq 1$

  $\forall Q \in \mathbb{G}_2, Q \neq \mathcal{O}$: $\exists P \in \mathbb{G}_1$ **s.t.** $e(P, Q) \neq 1$ $\quad$ $e(P, Q) \neq 1$

- *Efficiently computable*

(Ab)Using the pairing

The most important property of a pairing is:

$$e([a]Q, [b]P) = e([b]Q, [a]P) = e(Q, [ab]P) = e(Q, P)^{ab}$$

where $Q \in \mathbb{G}_2$, $P \in \mathbb{G}_1$, and the result is in $\mathbb{G}_T$.

In our context, the $\mathbb{G}_2$ group is larger than $\mathbb{G}_1$. The group $\mathbb{G}_T$ is also larger and has a different set of operations.

(Ab)Using the pairing II

- Since $\mathbb{G}_2$ is larger than $\mathbb{G}_1$, it is wise to exchange operations from one group to the other.

- $\mathbb{G}_T$ is significantly larger and has a different set of operations, we also try to avoid it, but we keep it handy, because...

(Ab)Using the pairing II

- Since $\mathbb{G}_2$ is larger than $\mathbb{G}_1$, it is wise to exchange operations from one group to the other.
- $\mathbb{G}_T$ is significantly larger and has a different set of operations, we also try to avoid it, but we keep it handy, because...
- An operation in $\mathbb{G}_T$ is cheaper than computing the pairing itself.

In short, we use the groups at will.

Contenido, sección 4

Boneh's short signatures

Boneh's short signatures are based on the mathematical problem:

Given $(P, [a]P, Q, [b]Q)$, it is hard to decide if $a = b$

The computational variant of this hard problem is:

Given $(P, Q, [n]Q)$, compute $[n]P$

Boneh, Lynn and Shacham constructed a short signature scheme based on this problem as follows:

... the steps

Key generation. Choose $n \in_R \mathbb{Z}_r$, set $R \leftarrow [n]Q$. The public key is: Q, R. The secret key is n

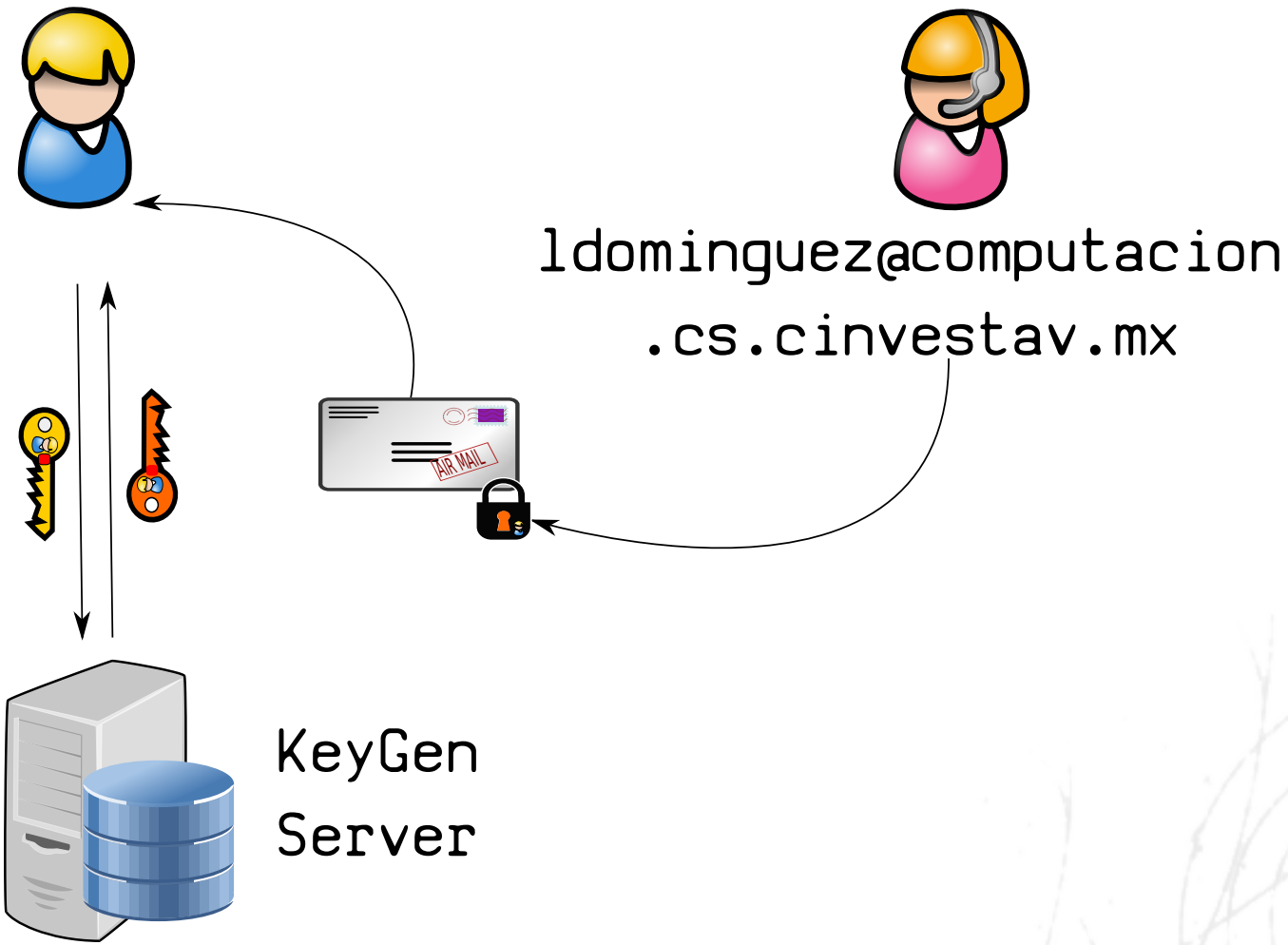
Sign. Map to a point the message to sign as P_M, set $S_M \leftarrow [n]P_M$. The signature is the x-coordinate of S_M.

Verify. Given the x-coordinate of S_M, find $\pm S$. Decide: $e(Q, S) \stackrel{?}{=} e(R, h(M))$

Tripartite Diffie-Hellman key exchange

Public parameter generation		
A trusted party publishes a finite field $\mathbb{F}_p$, an elliptic curve $E(\mathbb{F}_p)$ a point $P \in E(\mathbb{F}_p)$ of prime order ℓ, and an ℓ-distorsion map ϕ for P		
Alice	Bob	Carl
Private Computations		
Chooses a secret n_A $Q_A = [n_A]P \in E(\mathbb{F}_p)$	Chooses a secret n_B $Q_B = [n_B]P \in E(\mathbb{F}_p)$	Chooses a secret n_C $Q_C = [n_C]P \in E(\mathbb{F}_p)$
Publication of values		
Alice, Bob and Carl publish their points Q_A, Q_B, Q_C		
Further Private Communications		
Alice	Bob	Carl
Compute $e(Q_B, Q_C)^{n_A}$	Compute $e(Q_A, Q_C)^{n_B}$	Compute $e(Q_A, Q_B)^{n_c}$
The shared secret value is $e(P, P)^{n_A n_b n_C}$		

Encryption for an identity

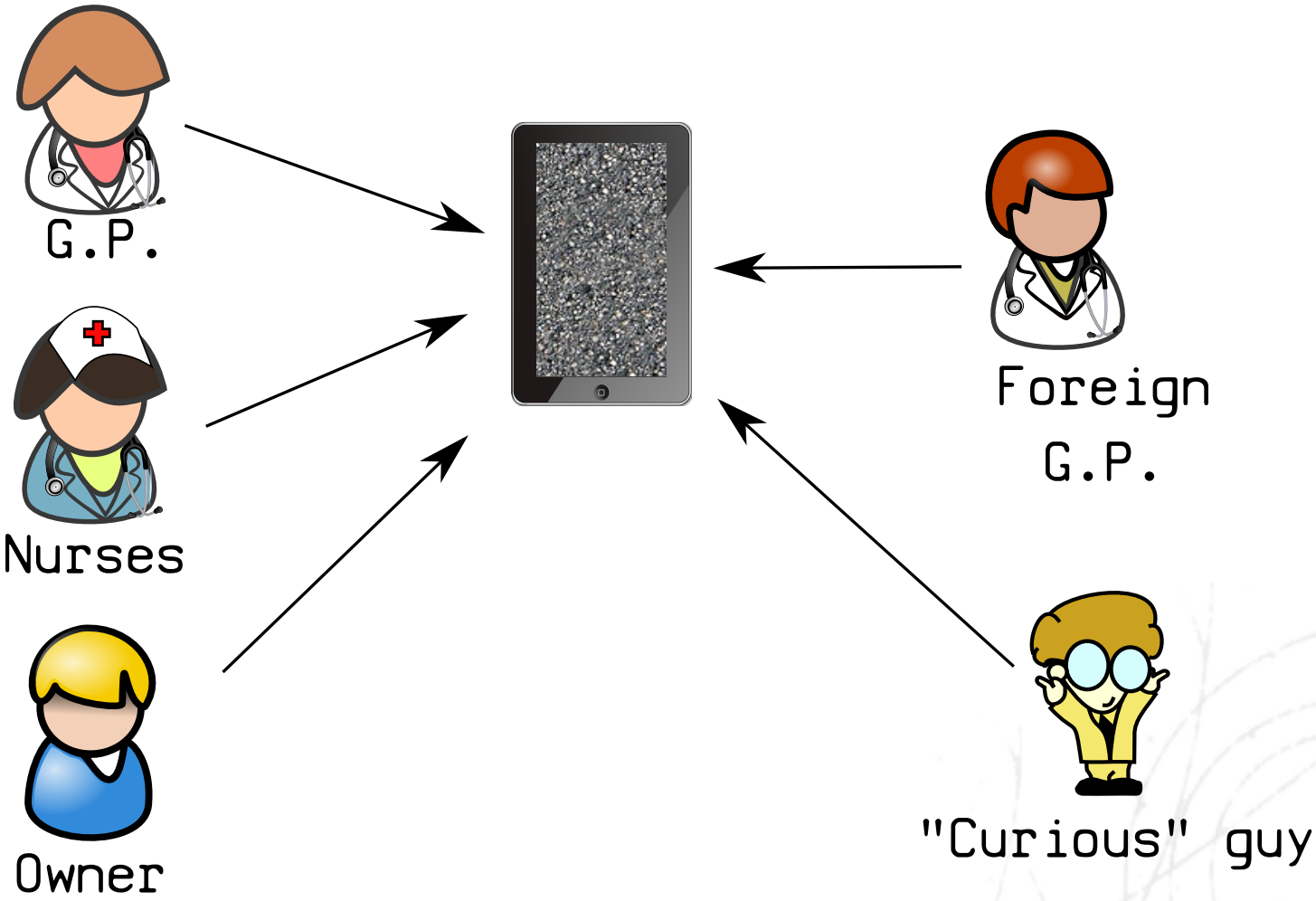


Attribute-Based Encryption

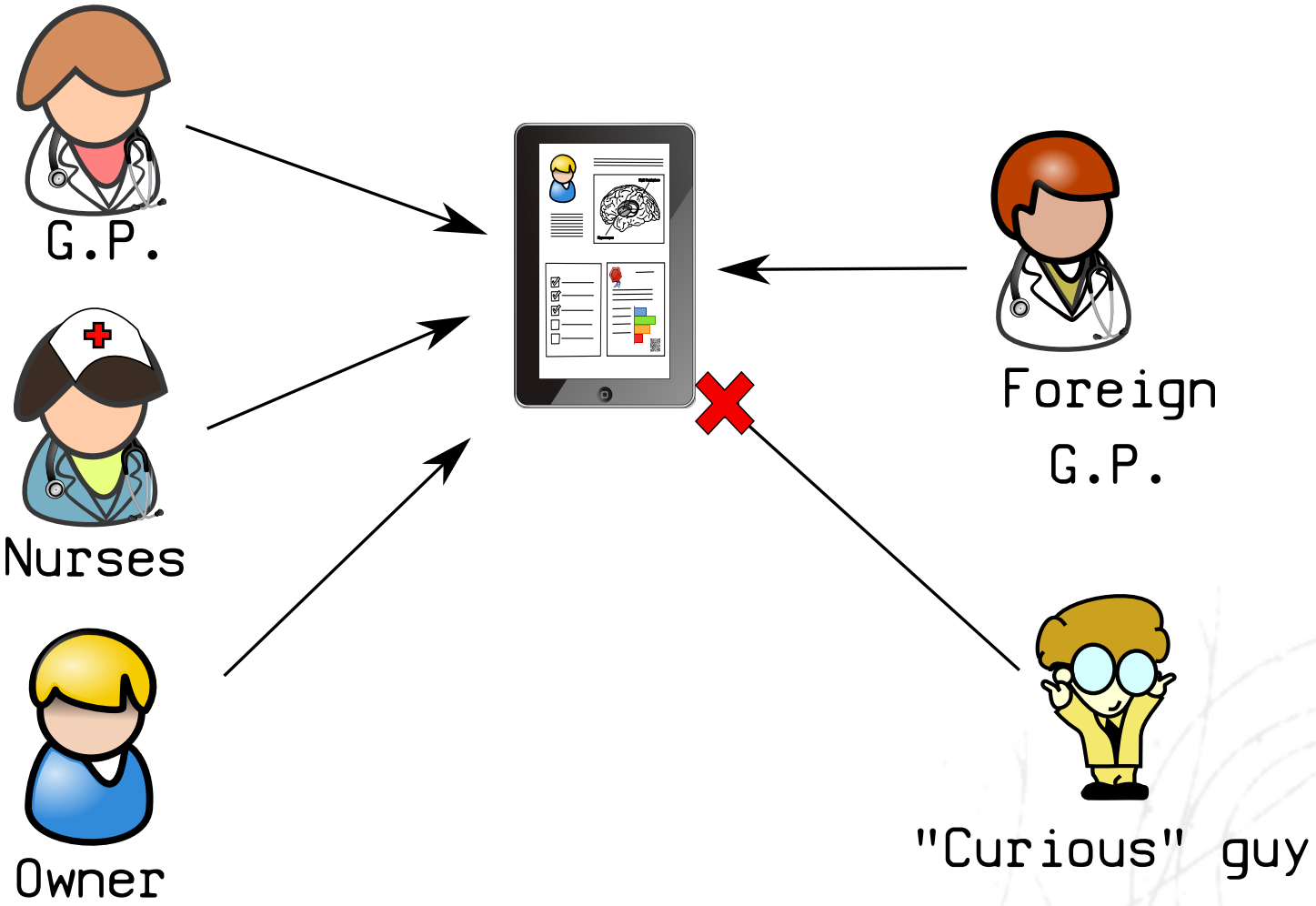
Fuzzy Identity-Based Encryption. Also known as Attribute-based encryption.

- An identity is a set of attributes
- An entity is valid if it presents a minimum number of attributes
- Better for sharing a small secret: a symmetric key.

Attribute examples

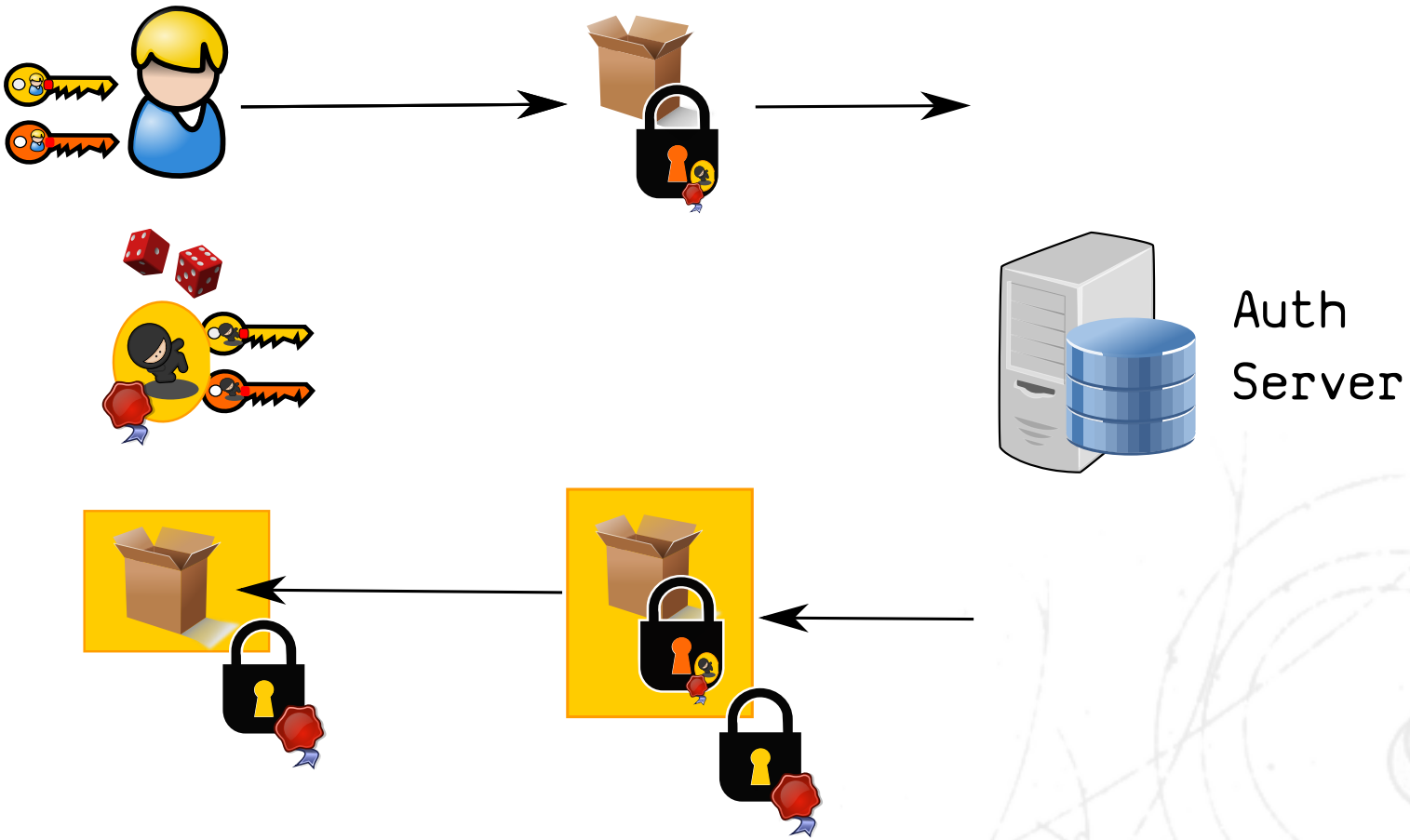


Attribute examples



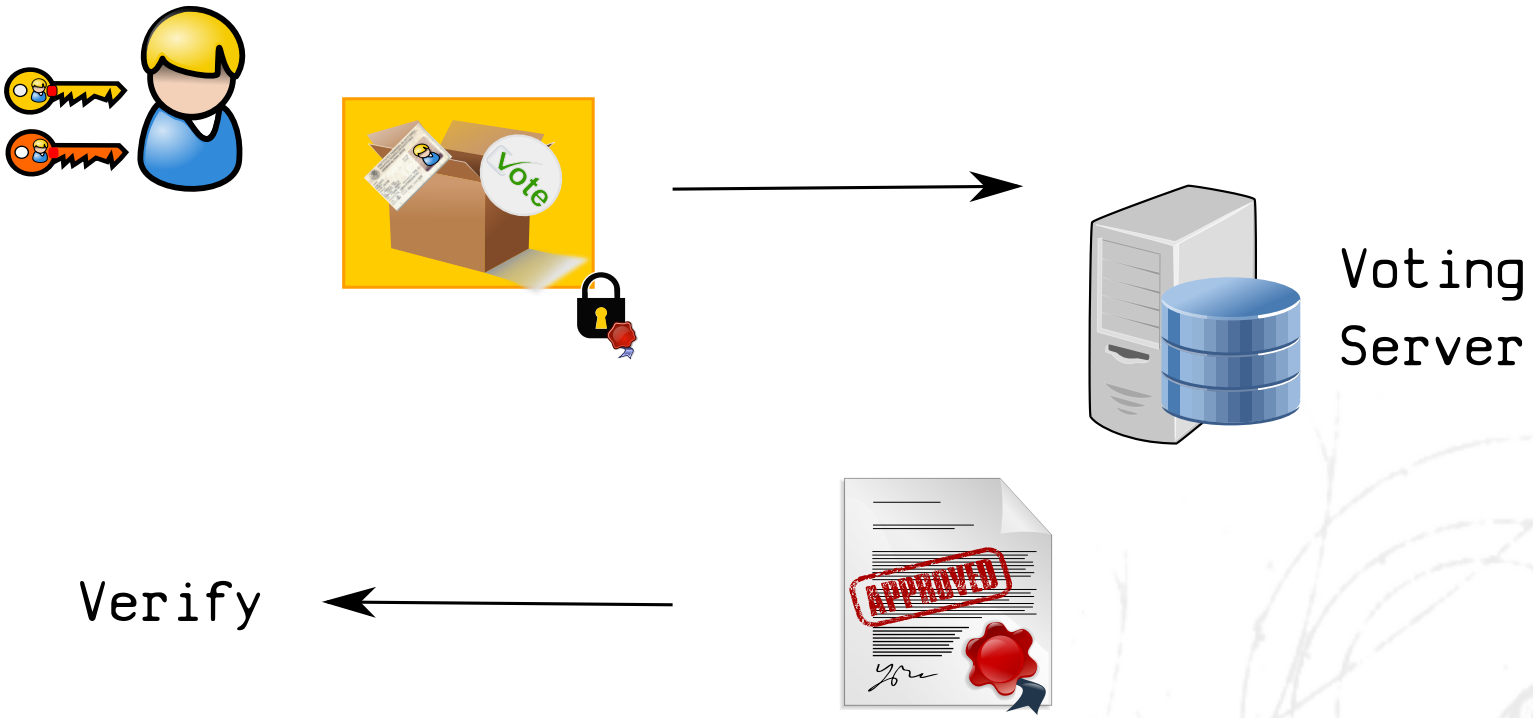
e-voting system based on pairings

An e-voting system based on short and blind signatures by Lopez-Garcia and Rodriguez-Henriquez.



e-voting system based on pairings

With a blind signature, we can cast our vote in the blank ballot.



Detailed e-voting system... 1/2

An e-voting system based on short and blind signatures by Lopez-Garcia and Rodriguez-Henriquez.

Authentication phase

Voter		Authentication Server (AS)
$b, d_t \in \mathbb{Z}_r$		
$V_t = [d_t]Q \in \mathbb{G}_2$		
$m = m2s(V_t) \in \{0,1\}^{1016}$		
$\tilde{M} = [b]H_1(m) \in \mathbb{G}_1$		
$S_{\tilde{M}} = [d_V]\tilde{M} \in \mathbb{G}_1$		
	$\{ID_V, t, \tilde{M}, S_{\tilde{M}}\}$ $\longrightarrow$	
		$e(Q, S_{\tilde{M}}) \stackrel{?}{=} e(V_V, \tilde{M})$
		$\tilde{S} = [d_{AS}]\tilde{M} \in \mathbb{G}_1$
	$\{t, \tilde{S}\}$ $\longleftarrow$	
$S_{V_t} = [b^{-1}]\tilde{S} \in \mathbb{G}_1$		

... detailed e-voting system 2/2

Voting phase

Voter		Voting server (VS)
$S_v = [d_t]H_1(v) \in \mathbb{G}_1$		
$B = \{V_t, S_{V_t}, v, S_v\}$	$\xrightarrow{\{B\}}$	$m = m2s(V_t)$
		$e(Q, S_{V_t}) \stackrel{?}{=} e(V_{AS}, H_1(m))$
		$e(Q, S_v) \stackrel{?}{=} e(V_t, H_1(v))$
		$a \in \mathbb{Z}_r$
		$ACK = H(V_t\|\|S_{V_t}\|\|v\|\|S_v\|\|a)$
		$S_{ACK} = [d_{VS}]H_1(ACK)$
	$\xleftarrow{\{ACK, S_{ACK}\}}$	
$e(Q, S_{ACK}) \stackrel{?}{=} e(V_{VS}, H_1(ACK))$		

Timings of the e-voting protocol

Scheme	# Cryptographic operation	# Cycles
Kharchineh & Ettelace	4 RSA-public	6,053,528
	6 RSA-private	253,251,894
	4 DLP-exponentiations	87,135,920
	Total	**346,441,342**
Li et al.	15 RSA-public	22,700,730
	9 RSA-private	379,877,841
	Total	**402,578,571**
Chung & Wu	5 RSA-public	7,566,910
	4 RSA-private	168,834,596
	Total	**176,401,506**
The proposed scheme	1 scalar multiplication in $\mathbb{G}_2$	380,000
	6 scalar multiplications in $\mathbb{G}_1$	1,800,000
	6 map-to-point functions H_1	1,890,000
	8 bilinear pairings	14,630,000
	Total	**18,700,000**

Contenido, sección 5

MACs

- Un Código de Autenticación de Mensajes (MAC), también conocido como un suma de verificación criptográfica o función hash con llave, es un instrumento muy utilizado.

- En términos de funcionalidad de seguridad, los MACs comparten algunas propiedades con las firmas digitales, ya que también proveen integridad y autenticación de mensajes. Sin embargo, a diferencia de las firmas digitales, los MACs son esquemas de llave simétrica, por lo que no proveen no-repudiación.

Propiedades

Las propiedades de los Códigos de Autenticación de Mensajes son:

- **Suma de verificación criptográfica**. Genera una etiqueta de autenticación criptográficamente segura para un mensaje dado.
- **Simetría**. Están basadas en llaves simétricas. Para la firma y verificación de las partes debe de compartirse una clave secreta.
- **Tamaño de mensaje arbitrario**. Aceptan mensajes de tamaño arbitrario.
- **Tamaño de salida fijo**. Generan etiquetas de tamaño fijo.
- . . .

. . .propiedades

- . . .
- **Integridad de mensaje**. Proveen integridad de mensaje: cualquier manipilación del mensaje durante la transmisión será detectado por el receptor.
- **Autenticación del mensaje**. La parte receptora se asegura del origen del mensaje.
- **No no-repudiación**. Dado que los MACs están basados en principios de criptografía simétrica, no proveen no-repudiación.

HMAC

- Una manera de hacer MACs es utilizando una función picadillo como SHA-1
- Este tipo de construcciones se les conoce como HMAC

Una manera vulnerable de construirlas es la siguiente:

- MAC de prefijo secreto: $m = MAC_k(x) = h(k||x)$
- MAC de sufijo secreto: $m = MAC_k(x) = h(x||k)$.

HMAC

- Una manera de hacer MACs es utilizando una función picadillo como SHA-1
- Este tipo de construcciones se les conoce como HMAC

Una manera vulnerable de construirlas es la siguiente:

- MAC de prefijo secreto: $m = MAC_k(x) = h(k||x)$
- MAC de sufijo secreto: $m = MAC_k(x) = h(x||k)$.

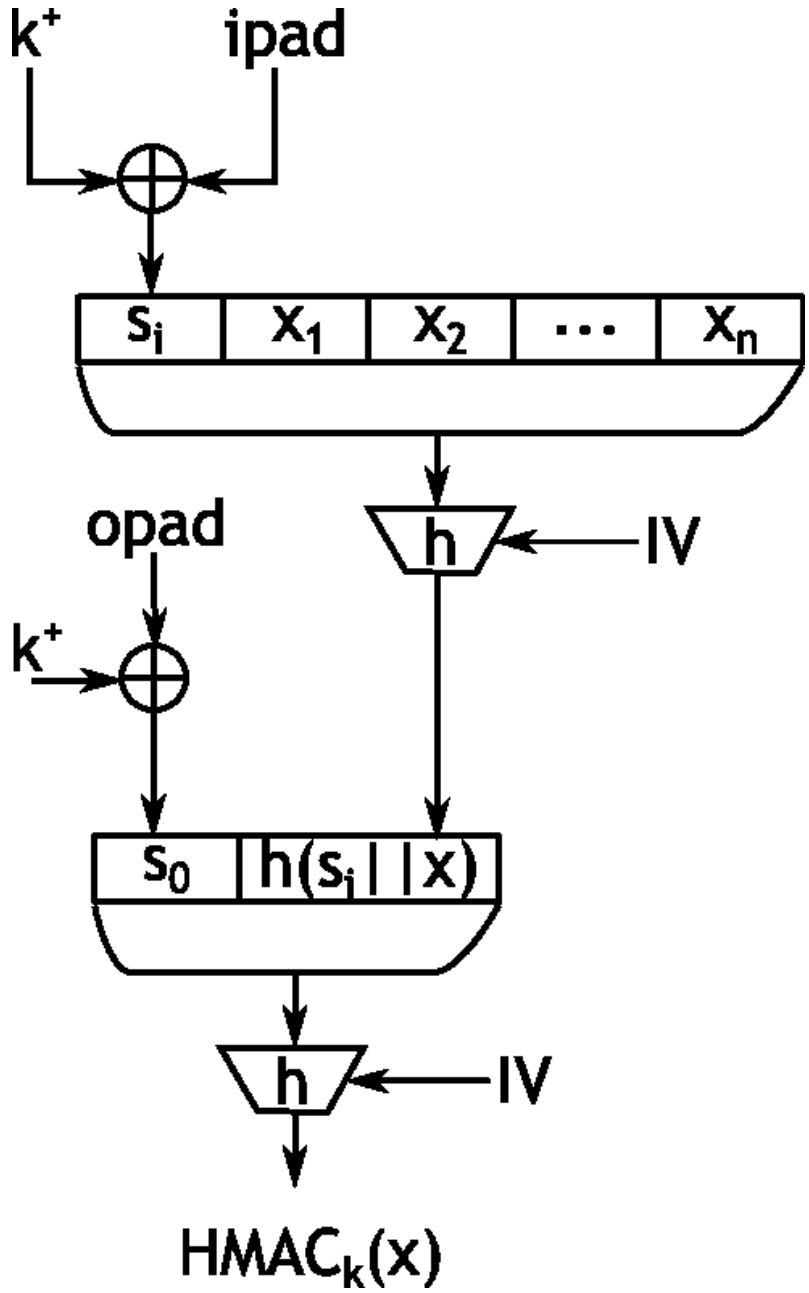
Ambas construcciones son vulnerables.

Ataques

- Prefijo secreto: dada la iteratividad del MAC, se pueden agregar mensajes al final de la secuencia

- Sufijo secreto: dada una coalisión en la función hash, es posible producir otro mensaje diferente que sea válido.

Diagrama HMAC



Contenido, sección 6

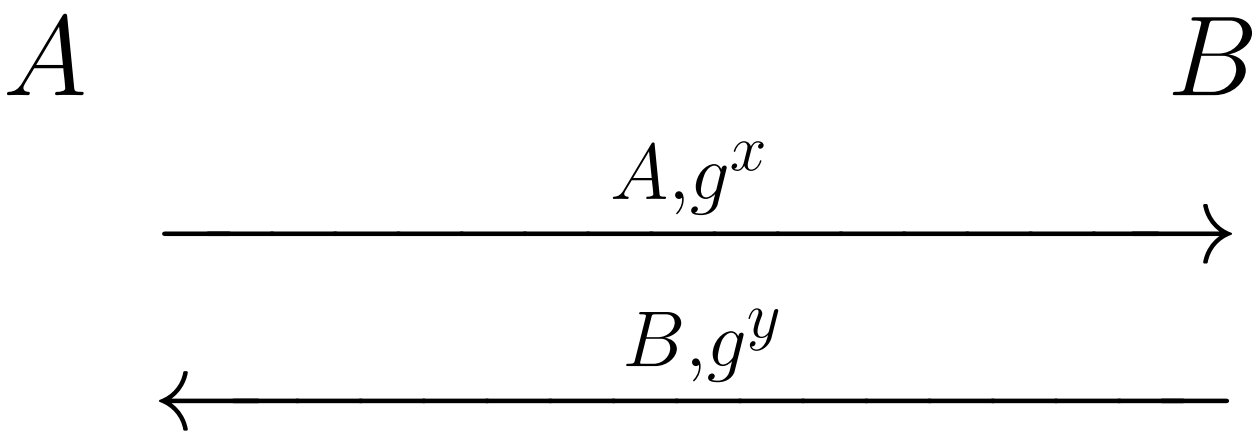
IPSec

- Seguridad de transporte en la capa IP
- Provee tráfico seguro entre dos sistemas IP
- Ofrece servicios de seguridad para los paquetes IP
- Generación y mantenimiento de la Asociación de Seguridad
- Independiente de la aplicación (software)

Para establecer un canal, primero hay que intercambiar claves (simétricas). . .

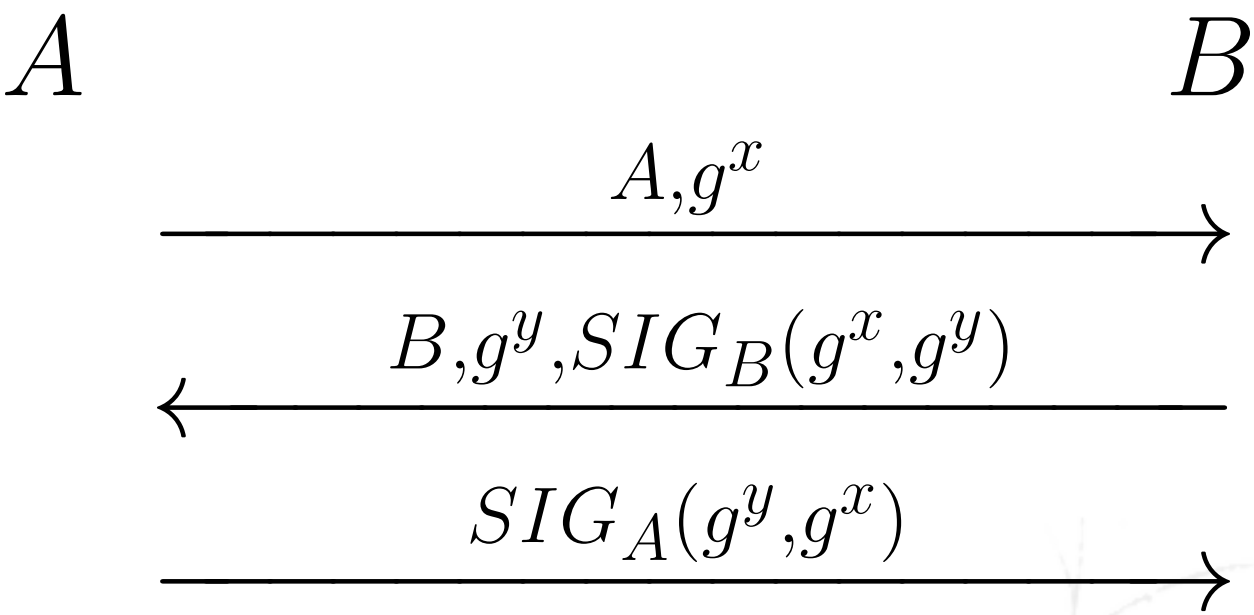
Diffie-Hellman

DH'76 - Diffie-Hellman Exchange

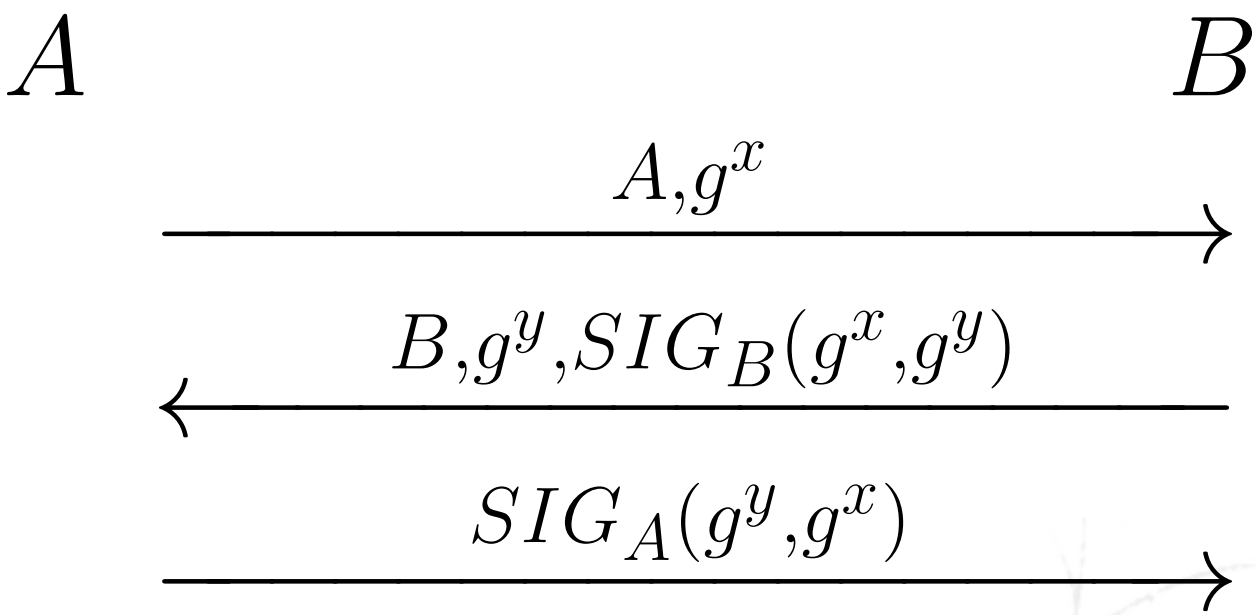


- Ambas partes pueden calcular la llave secreta $K = g^{xy}$
- Dados g^x, y g^y, g^{xy} parece un elemento aleatorio
- Abierto a un ataque M-I-T-M en un ambiente sin autenticación

DH Autenticado básico (BADH)

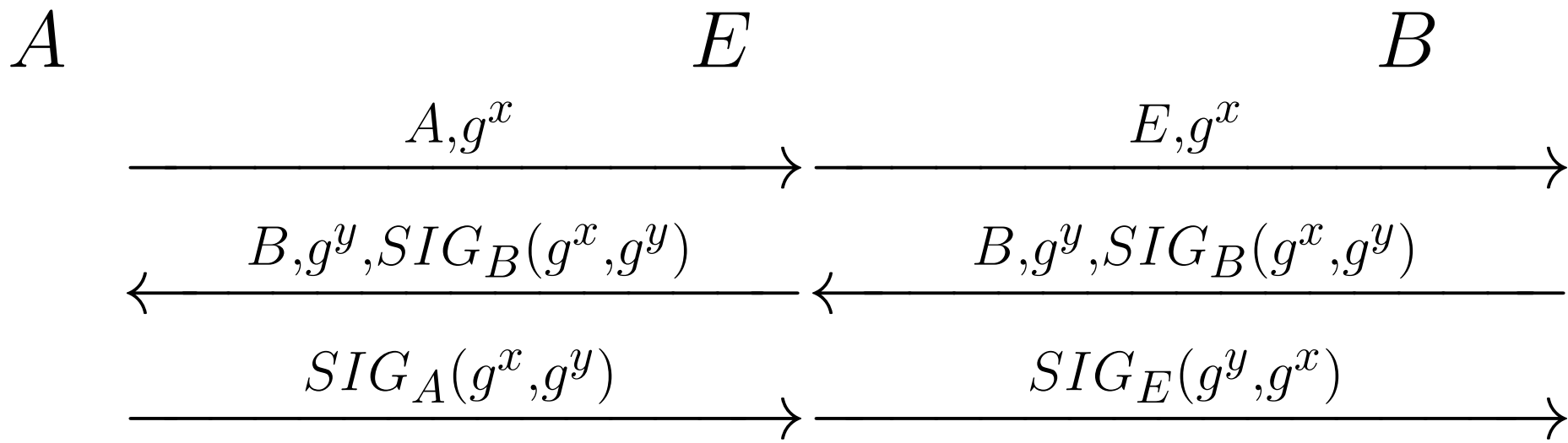


DH Autenticado básico (BADH)

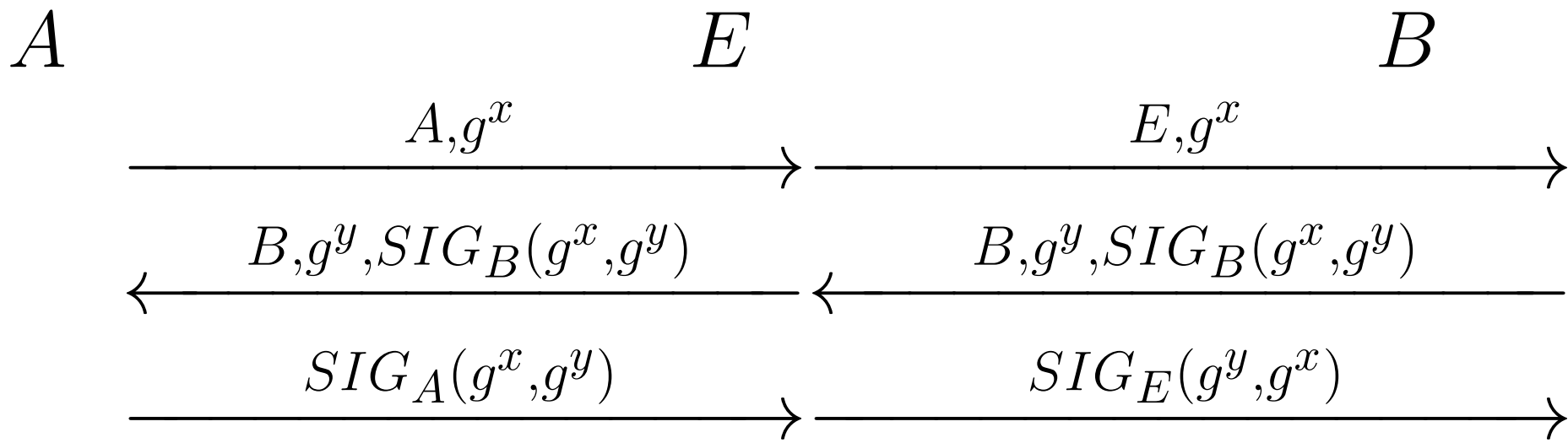


sin embargo...

Ataque a BADH



Ataque a BADH

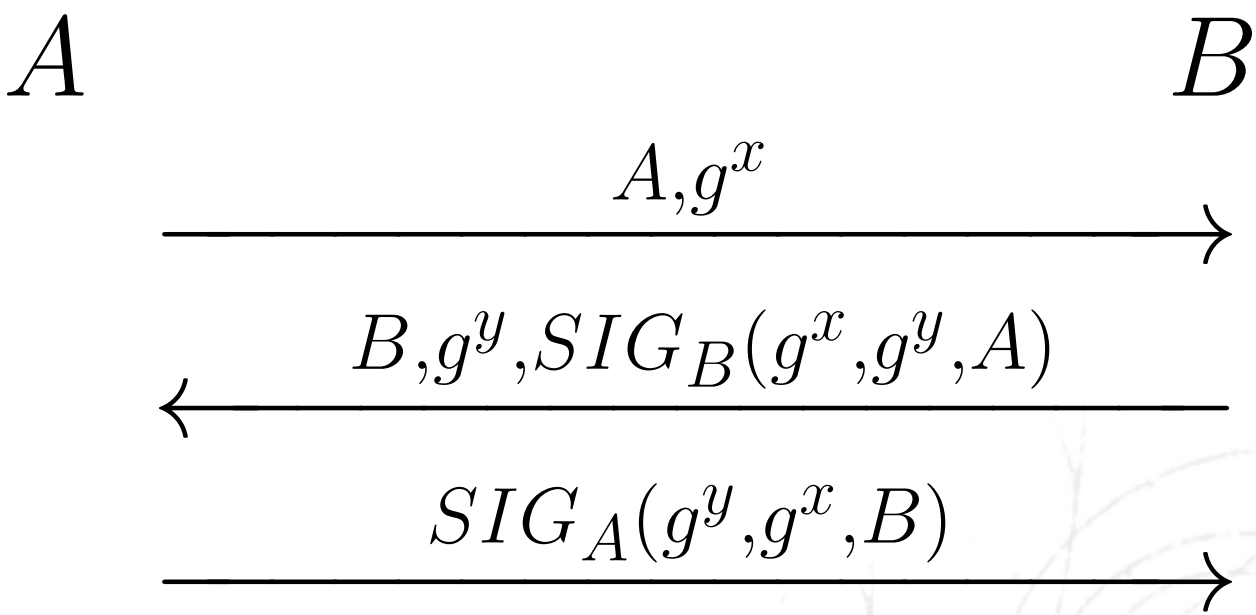


Aunque E no conoce $K = g^{xy}$, B recibe lo que le envíe E pensando que es A.

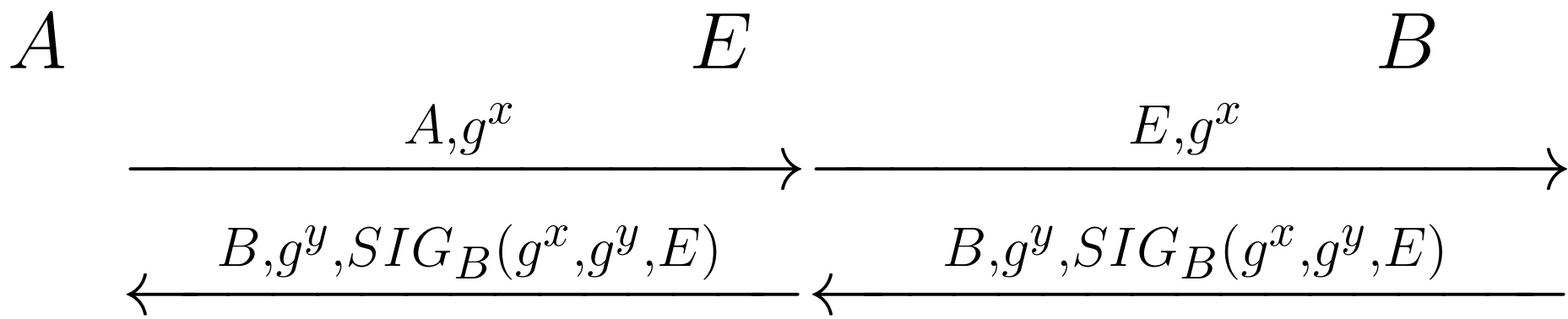
- A recibe mensajes de B correctos, E no puede hacer nada.
- B recibe mensajes de A con identificador de E.

Solución ISO-9796. . .

Incluir la identidad del receptor dentro de la firma:



Ataque. . .



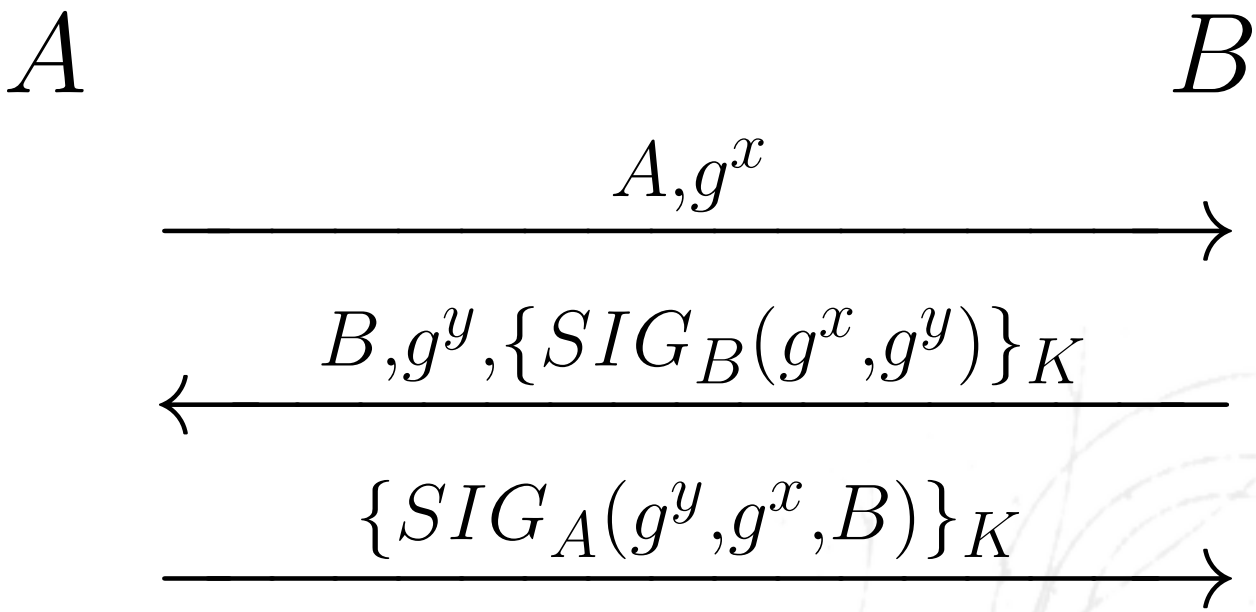
- A: ¡Ajá! B se está comunicando con E, no conmigo
- E no puede producir $SIG_B(g^x, g^y, A)$

¿Es seguro?

- Es técnicamente seguro
- No sirve para protección de identidades:
  - B necesita saber la identidad de A antes de autenticarse, lo mismo para A
  - Privacidad: hay evidencia firmada de la comunicación
  - Hacer que cada parte firme su identidad resuelve el problema de la privacidad, pero volvería el protocolo inseguro (ataque M-I-T-M).

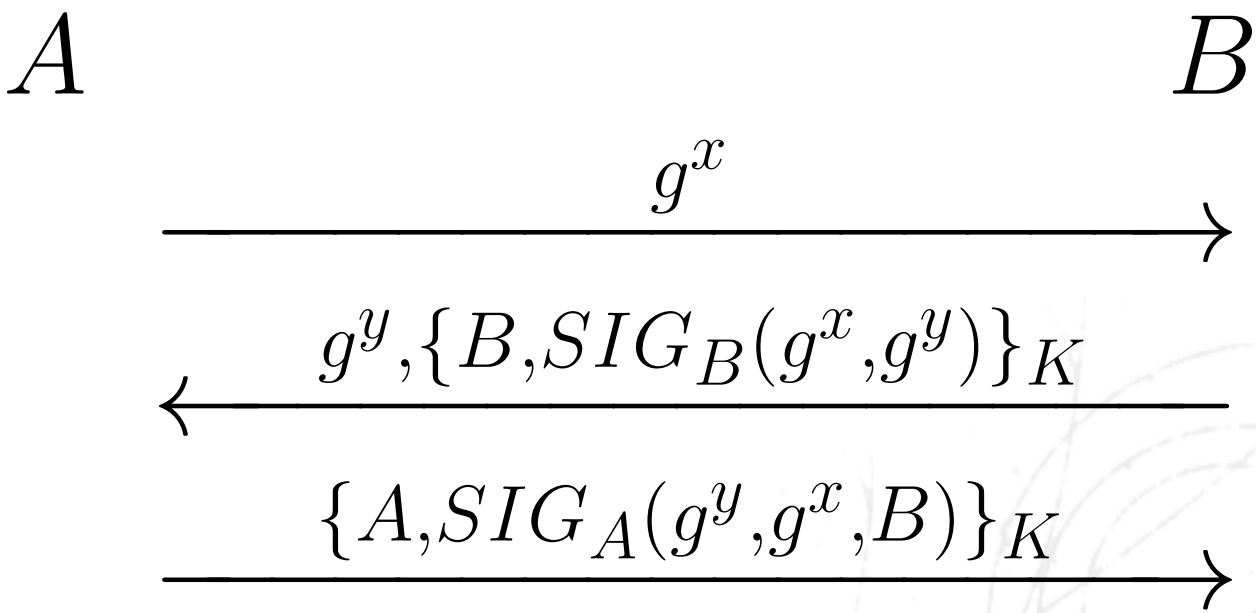
Otra solución: STS

Probar conocimiento de la $K = g^{xy}$ utilizando un cifrado simétrico:



¿Es seguro?

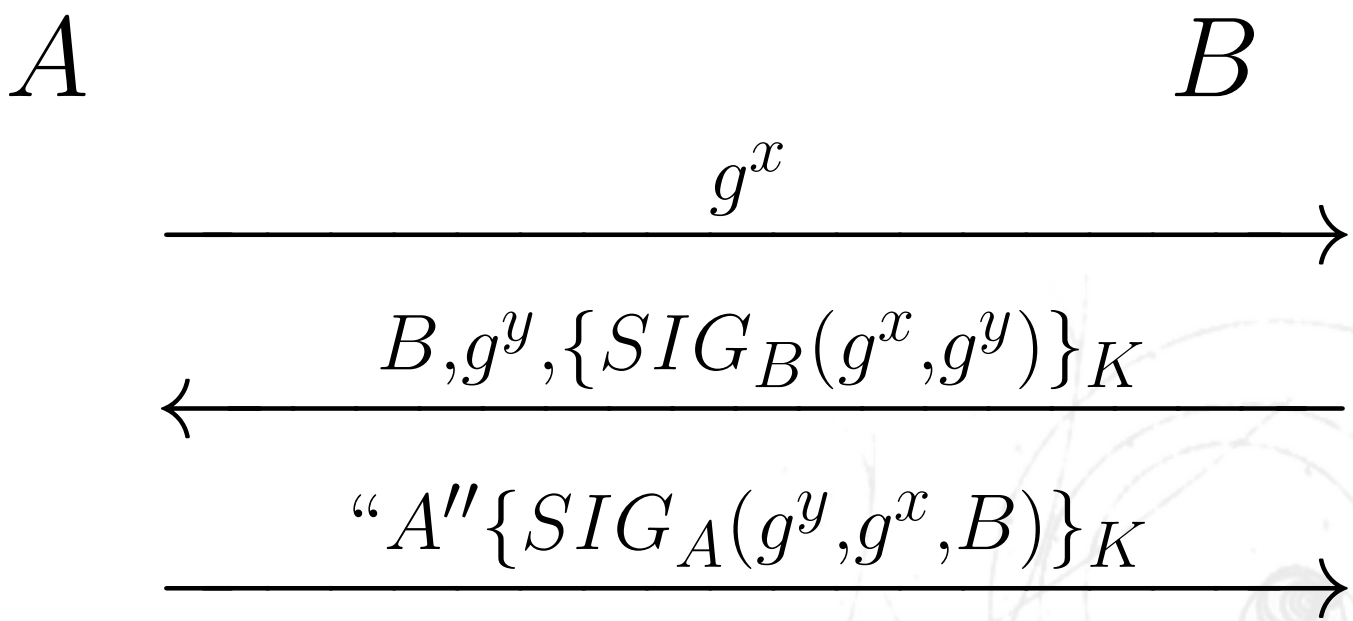
- Es seguro
- se puede extender para proteger a las identidades de la transacción



. . .¿es seguro?

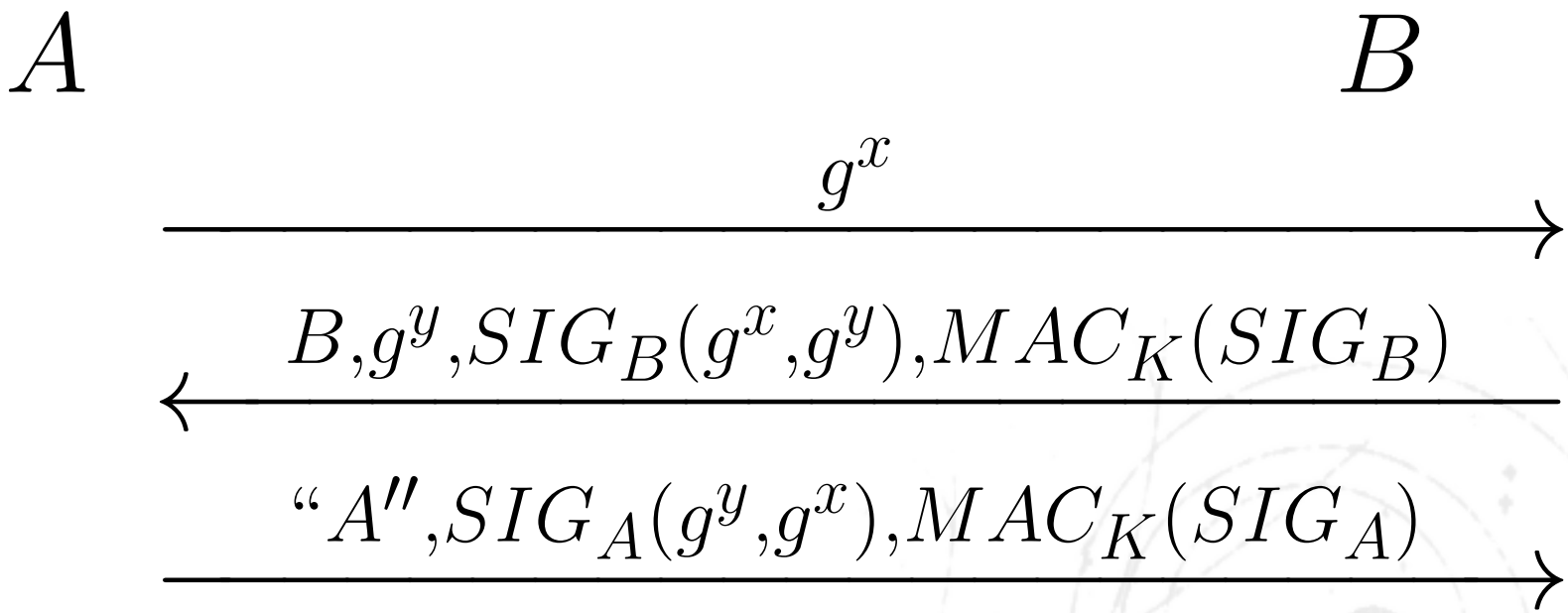
El cifrado, no es la función correcta para provar conocimiento de una llave

- alguien podría registrar como suya la llave pública de otra persona, y montar un ataque I-M (y sin conocer $K = g^{xy}$)



STS con MAC

Algunos esquemas verifican la identididad, pero no la posesión de la llave privada (PoP)

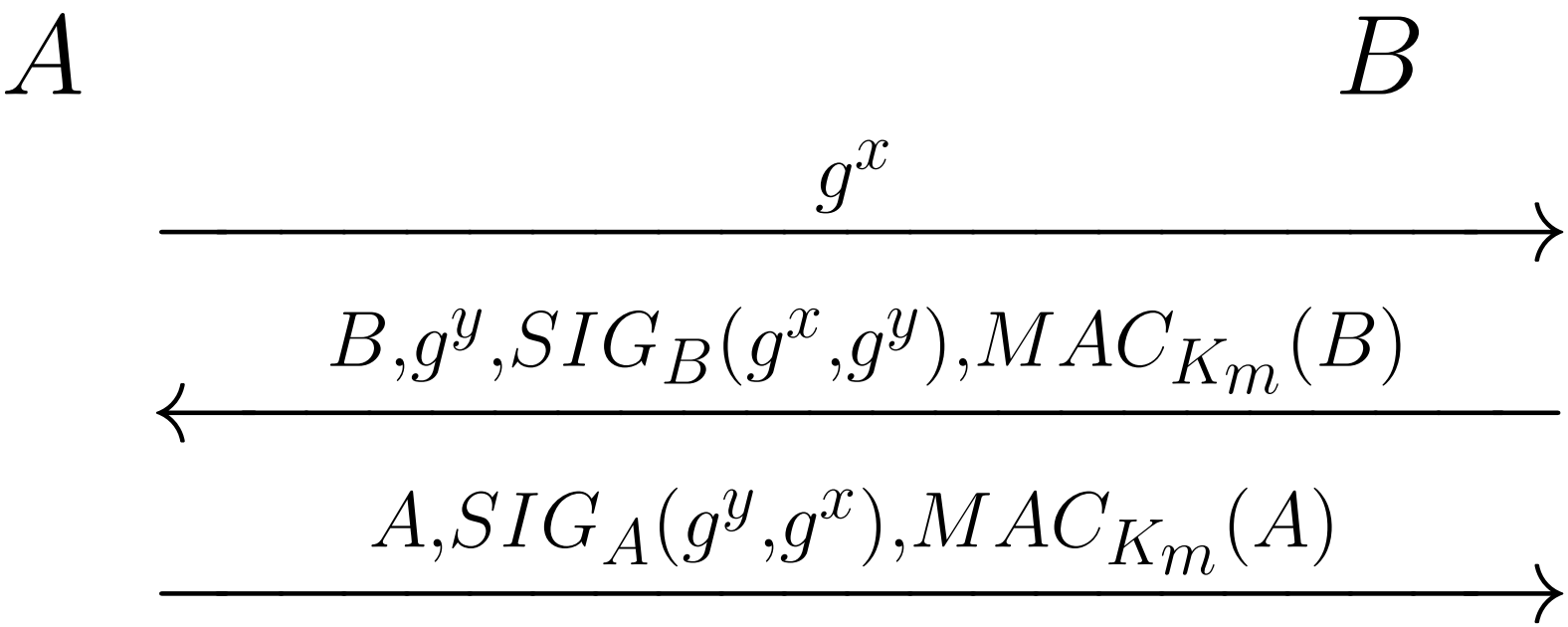


¿Qué pasa?

- El punto es que probar que se conozca la $K = g^{xy}$ no es el problema.
- Lo que se requiere es:
  - asociar la llave K con las identidades

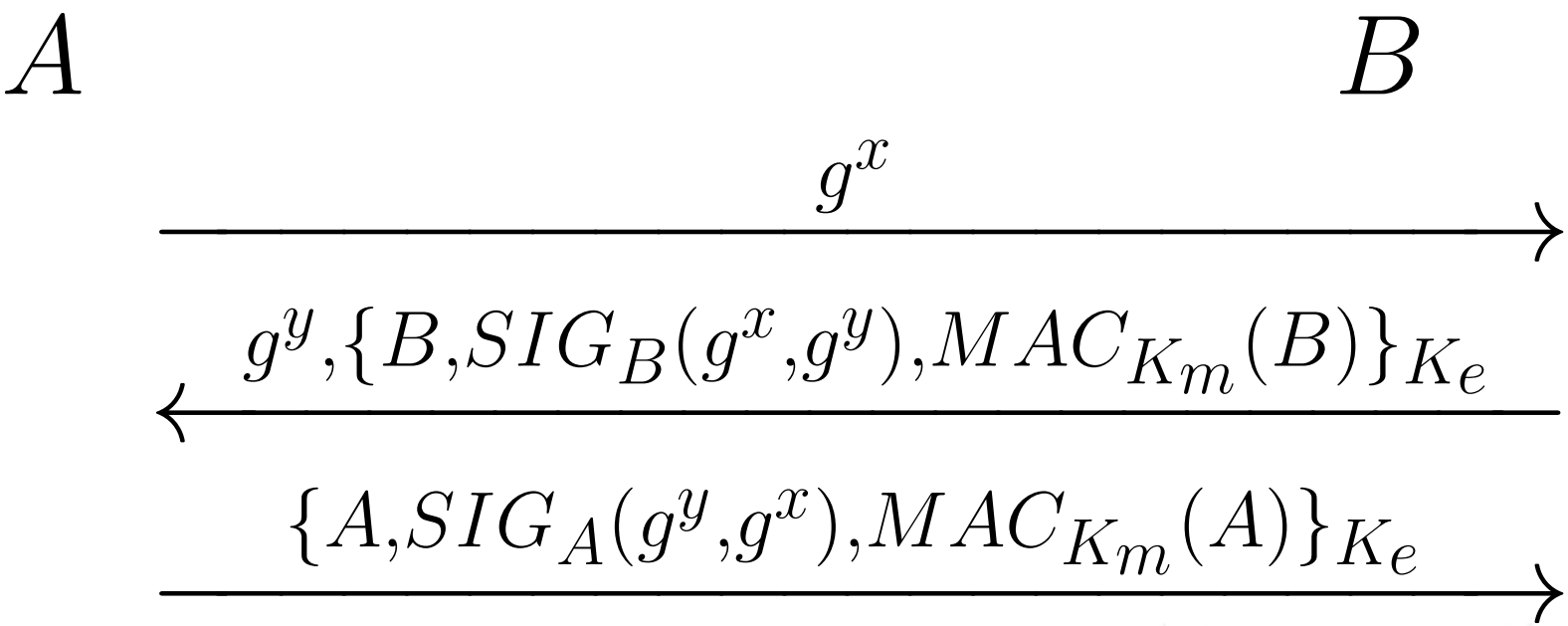
- La solución sería:
  - SIGn-and-MAC su identidad

SIGMA básico



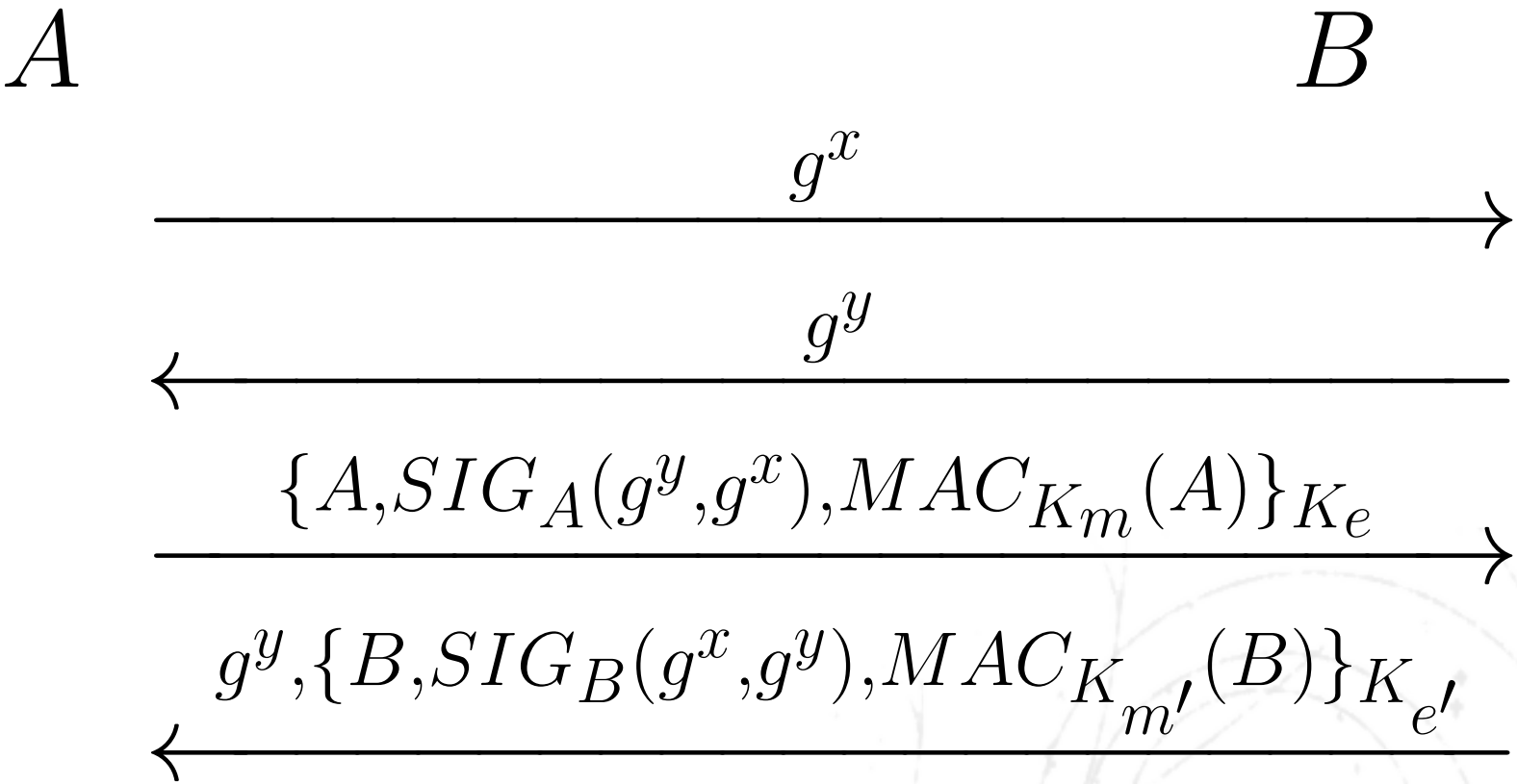
- Se genera una llave K_m a partir de la llave K
- SIG y MAC tienen roles complementarios un ataque M-I-T-M y para asociar la identidad
- No requiere conocer el ID de las partes para su identificación.

SIGMA-I: protección activa del ID del Iniciador



- Se genera una llave K_m, y una K_e a partir de la llave K
- B es el primero que se identifica
- La identidad del Iniciador (A) está protegida.

SIGMA-R: protección activa del ID del Destinatario (R)



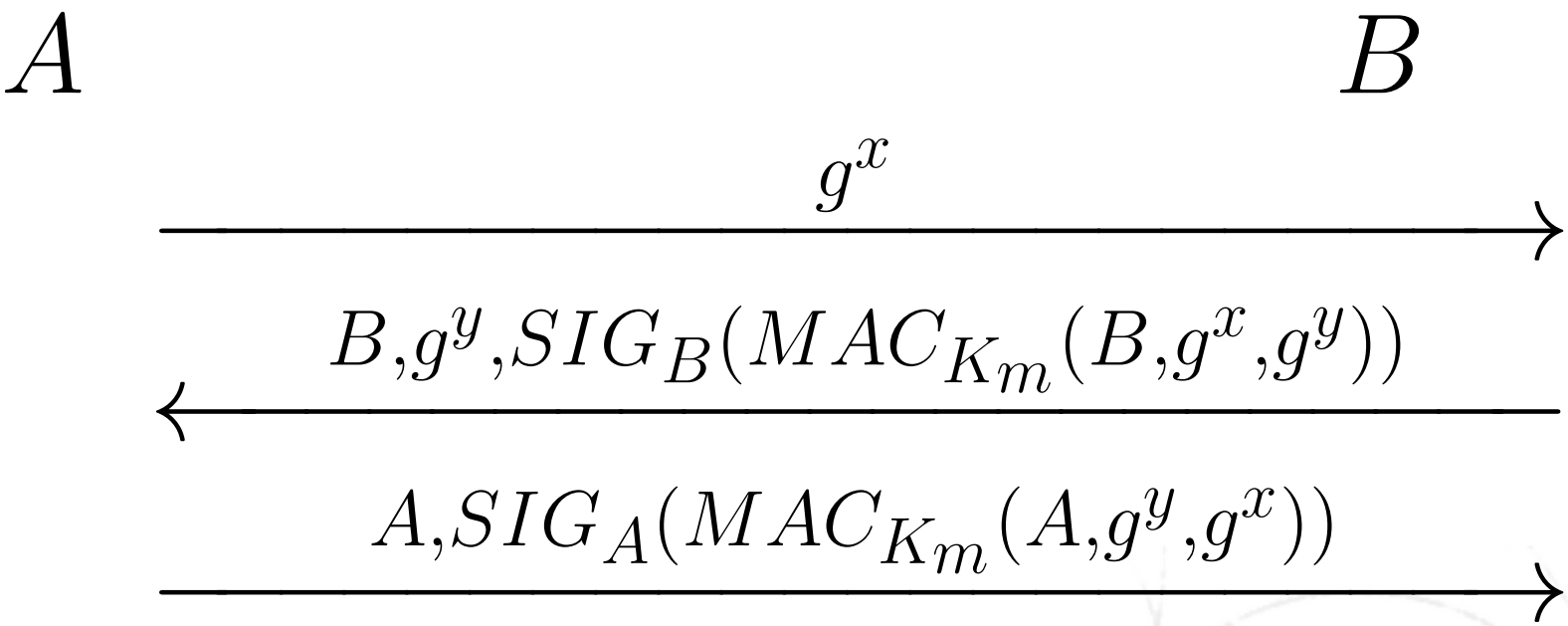
- Se generan llaves K_m, $K_{m'}$, K_e, y K'_e a partir de la llave K

IKE: Internet Key Exchange

- Crea SAs (Asociaciones de Seguridad) para utilizarse en un enlace IPSec
- Administra las SAs

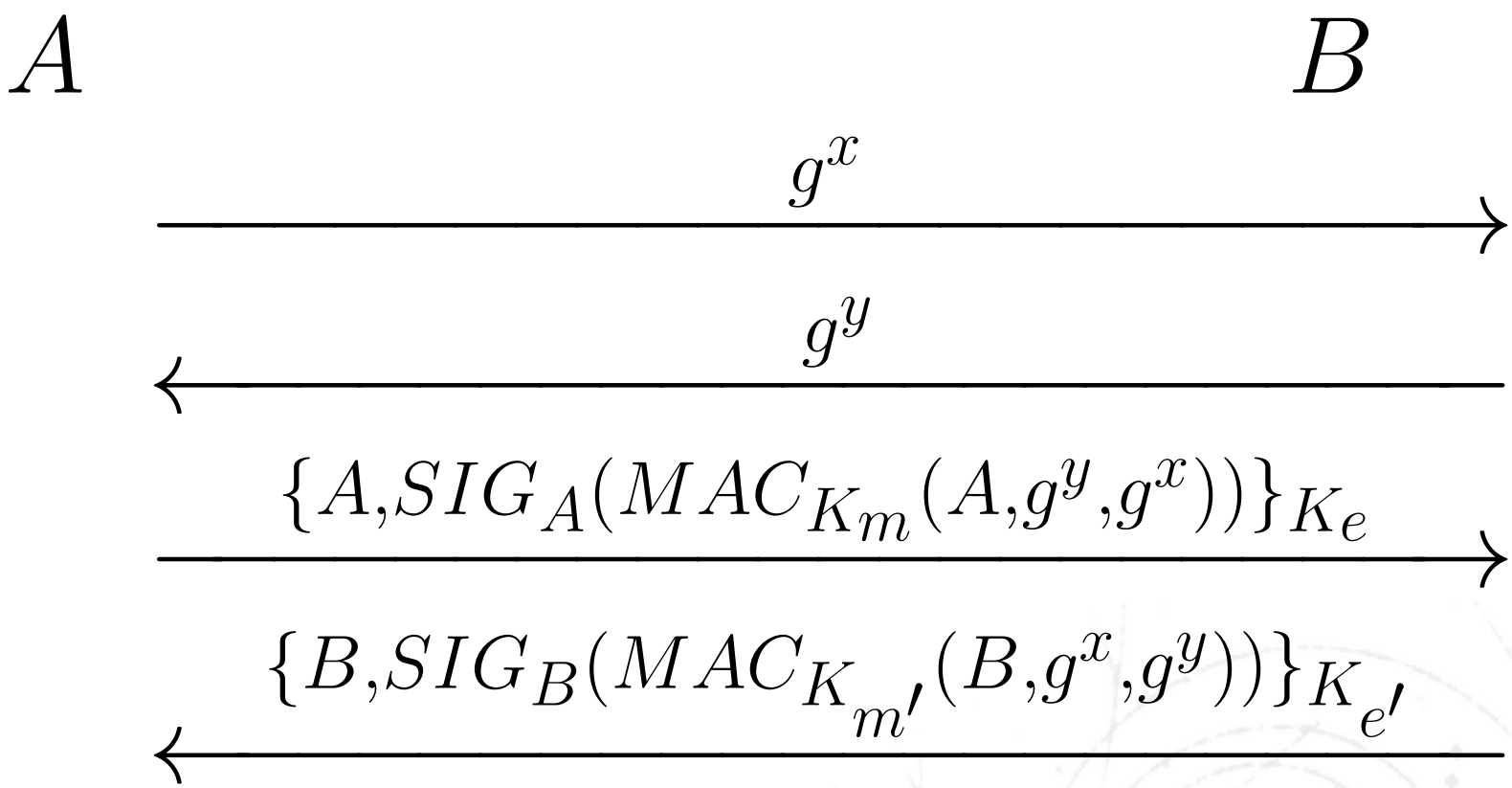
- Cuando un nodo A quiere comunicarse con B bajo la protección de un enlace IPSec, es necesario un intercambio de llave simétrica.
- La derivamos de SIGMA, visto recientemente.

Variante de IKEv1: MAC bajo SIG



- IKE modo “agresivo” (sin protección de ID).

IKE modo principal



- Se generan llaves K_m, $K_{m'}$, K_e, y K'_e a partir de la llave K