Provable security for Key Establishment Protocols

Workshop Interconsolider ARES & I-Math

María Isabel González Vasco

Universidad Rey Juan Carlos

Indice

- Introducción
- Seguridad demostrable (modelo formal)
- (Muy) breve repaso histórico
- Algunas Construcciones
- Conclusiones

Introducción

Esquemas de establecimiento de clave

- Permiten a un conjunto de entidades acordar un secreto (clave) para su posterior comunicación.
  - Dos entidades (KE, AKE)
  - N› 2 entidades

- Distintos procesos de elección de la clave:
  - **Transporte de clave**: una entidad elige la clave y la transmite de manera segura a las demás.
  - **Acuerdo (agreement) de clave**: todas las entidades participan en la elección de la clave.

Esquemas de establecimiento de clave

- **Distintos modelos según el tipo de entidades implicadas:**
  - Participantes = grupo de entidades que acordará la clave
  - Pueden existir entidades externas al grupo de participantes (servidores), que
    - Ayudan en el proceso de autenticación
    - No deben acceder a la clave acordada

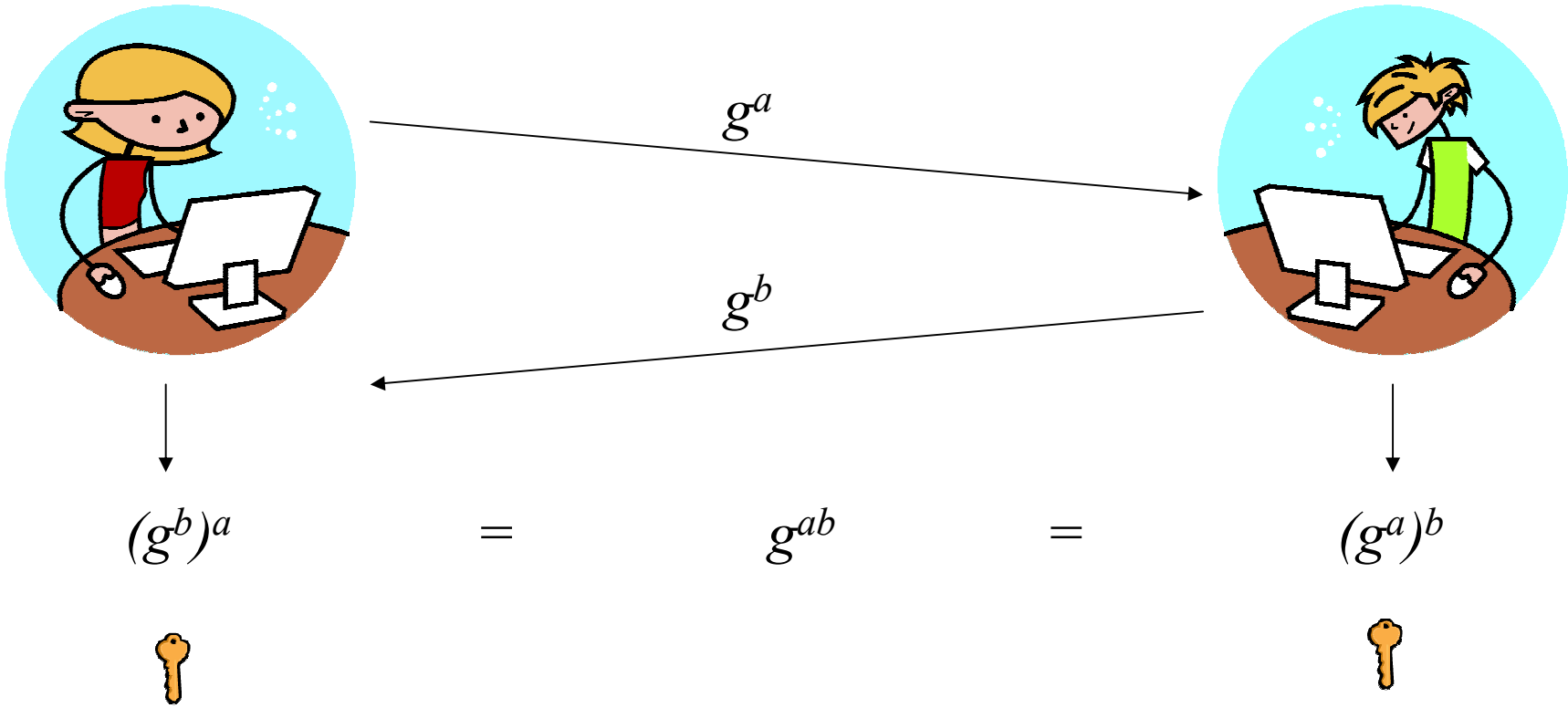
- **Según el modelo de adversarios que consideremos**:
  - Externos: Pasivos o Activos, según un enfoque ‘’clásico’’.
  - Internos: con objetivos ‘’retorcidos’’

Esquemas de establecimiento de clave

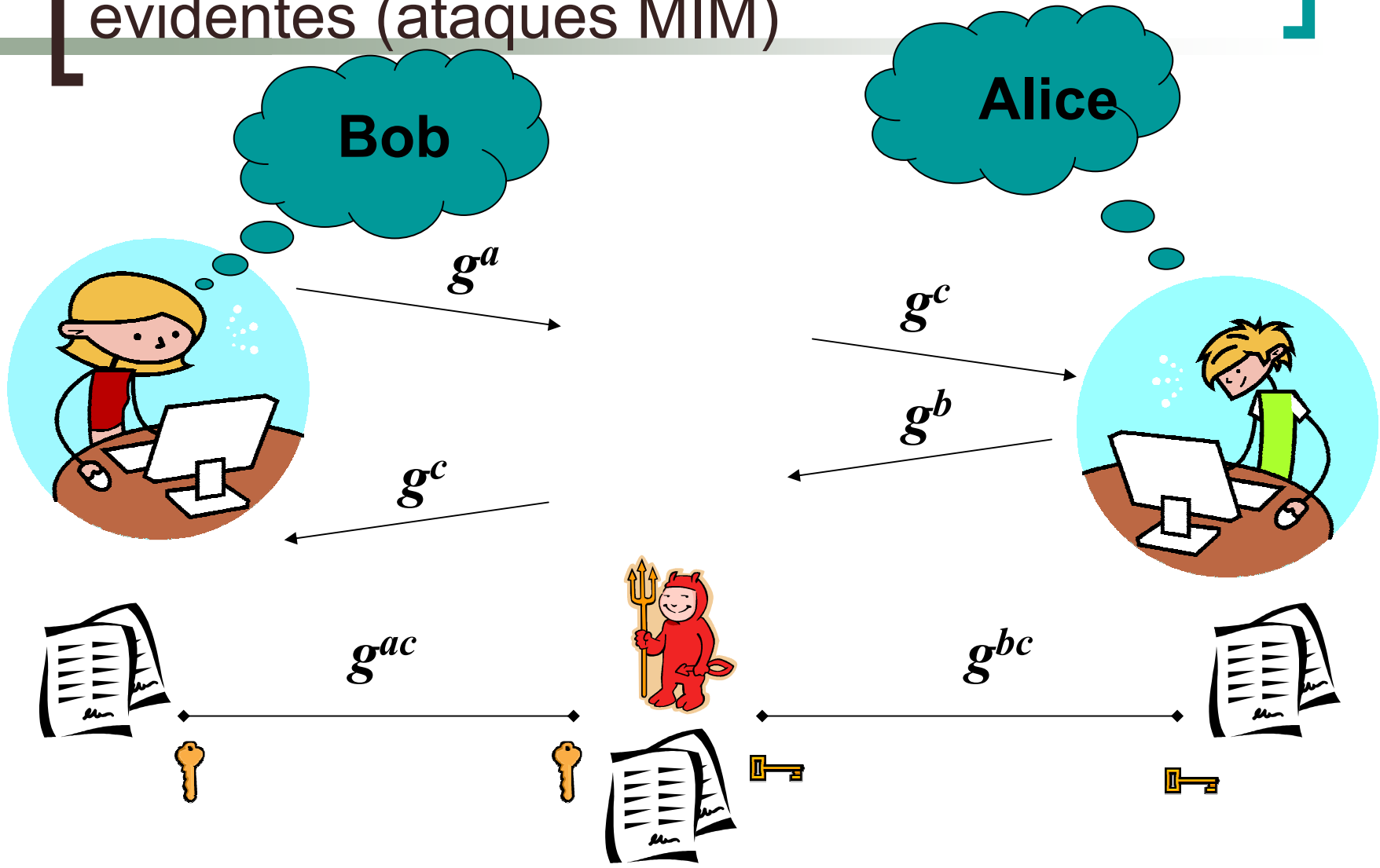
- Permiten a un conjunto de entidades acordar un secreto (clave) para su posterior comunicación.
  - N› 2 entidades
- Distintos procesos de elección de la clave:
  - **Acuerdo (agreement) de clave**: todas las entidades participan en la elección de la clave.
- **Distintos modelos según el tipo de entidades implicadas:**
  - Participantes = grupo de entidades que acordará la clave
- **Según el modelo de adversarios que consideremos**:
  - Externos: Pasivos o Activos, según un enfoque ‘’clásico’’.

Diffie-Hellman (1976)

G un grupo cíclico, con DDH difícil, g un generador

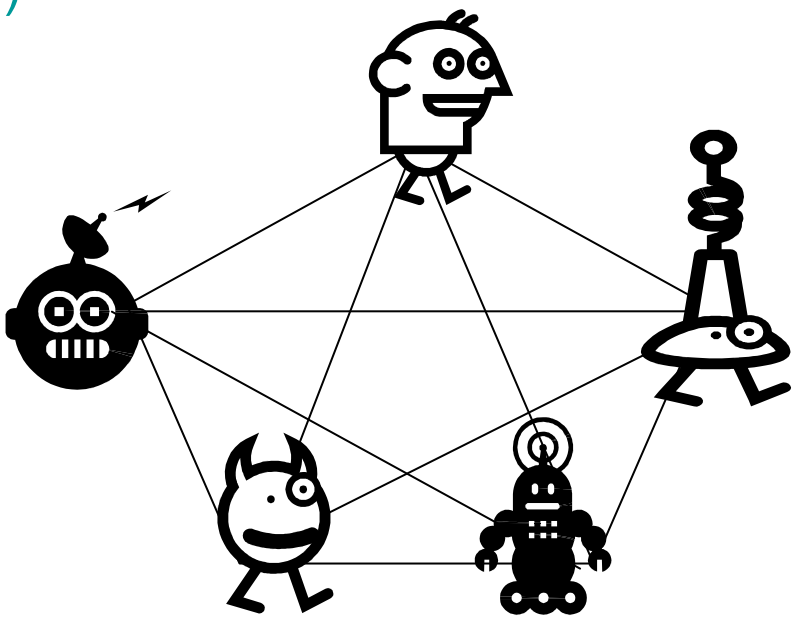


Sin autenticación, hay problemas evidentes (ataques MIM)



Esquemas (autenticados) de establecimiento de claves para grupos

- Objetivo: n›2 entidades quieren construir una clave común de manera autenticada
- Métodos para conseguir autenticación: PKI, técnicas simétricas, secretos compartidos de baja entropía (passwords)



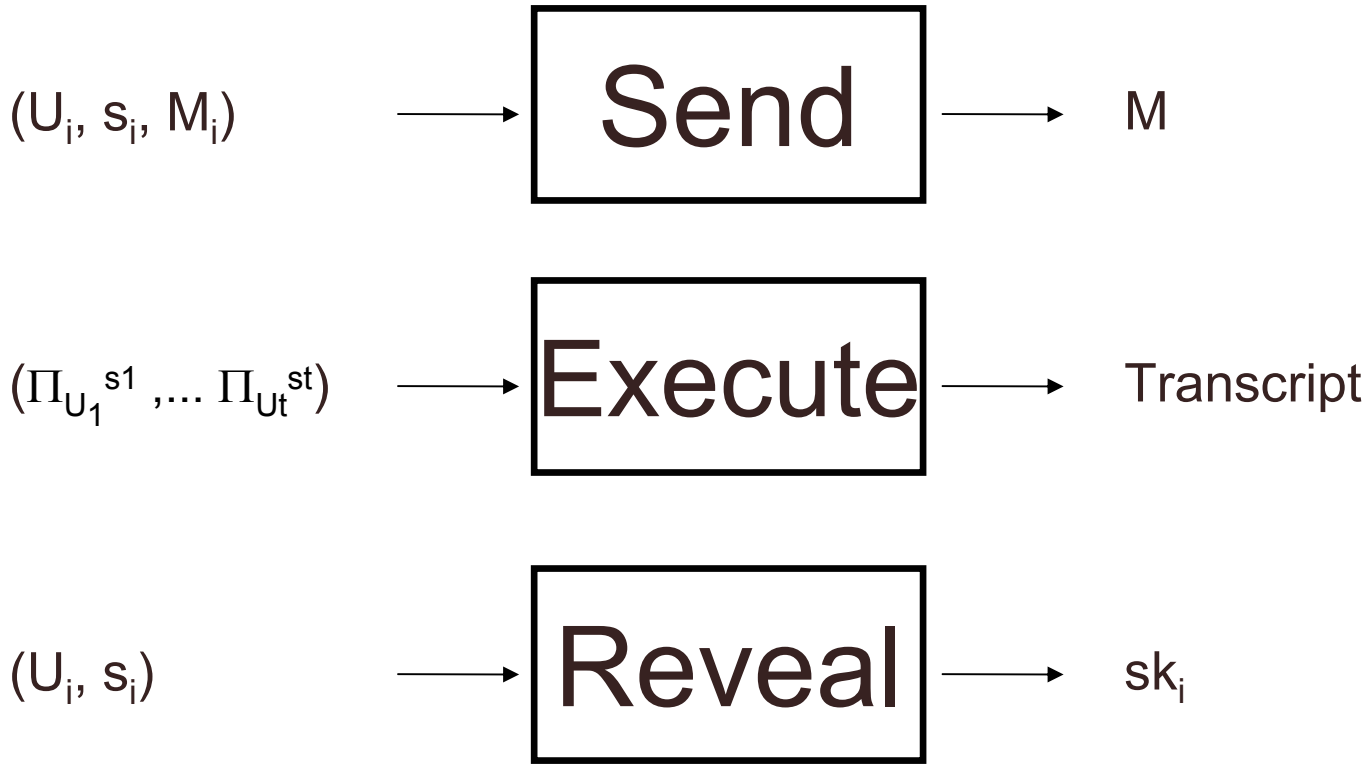
Seguridad Demostrable: Modelo Formal

Modelo

- **Red:** existe un canal conectando a cada par de participantes (no hay broadcast). La red es no privada y totalmente asíncrona.
- **Participantes:**
  - Usuarios del sistema: $U_1,\ldots, U_n$, todos compartiendo el mismo password pw de un diccionario $\mathcal{D}$ público (y polinomial)
  - Cada usuario está involucrado un número de ejecuciones, quizá en paralelo: $U_i \sim \Pi_i^1 \ldots \Pi_i^{s_i}$
  - Cada ejecución lleva asociadas variables:
    - $\Pi_i^{j_i} \sim$ sid: identificador de sesión,
    - pid: participantes involucrados,
    - sk: clave de sesión,
    - acc: éxito/aceptación de la ejecución

Adversario

Su actuación se modela a partir de llamadas a oráculos, que formalizan la intervención del adversario en distintas ejecuciones del protocolo.



Propiedades deseables (informalmente)

- Corrección
- Privacidad de la clave
- Integridad (Unknown key-share resilience, Key confirmation, Entity-Authentication)
- Forward Secrecy
- Key Control (Agreement)

Consideraremos al menos:

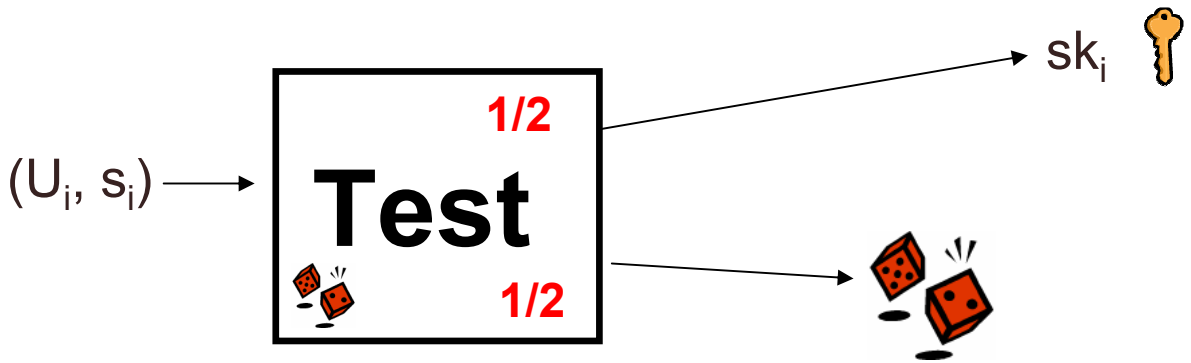
- Correción: Si el adversario $\mathcal{A}$ es pasivo (sólo tiene acceso al oráculo Execute) se verifica que si dos ejecuciones $\Pi_{U_i}^{si}$,$\Pi_{U_i}^{sj}$ aceptan ($acc_{U_i}^{si} = acc_{U_j}^{sj} = 1$) con el mismo identificador de sesión ($sid_{U_i}^{si} = sid_{U_i}^{sj}$) entonces han construido la misma clave de sesión y la asocian al mismo grupo de participantes ($pid_{U_i}^{si} = pid_{U_j}^{sj}$, $sk_{U_i}^{si} = sk_{U_j}^{sj}$).

- Integridad: idem, pero con $\mathcal{A}$ activo.

Privacidad de la clave

Se formaliza a través del siguiente planteamiento:

Considerar un oráculo de simulación Test al que el adversario sólo puede llamar con una ejecución *fresca* $\Pi_{U_1}^{s_i}$

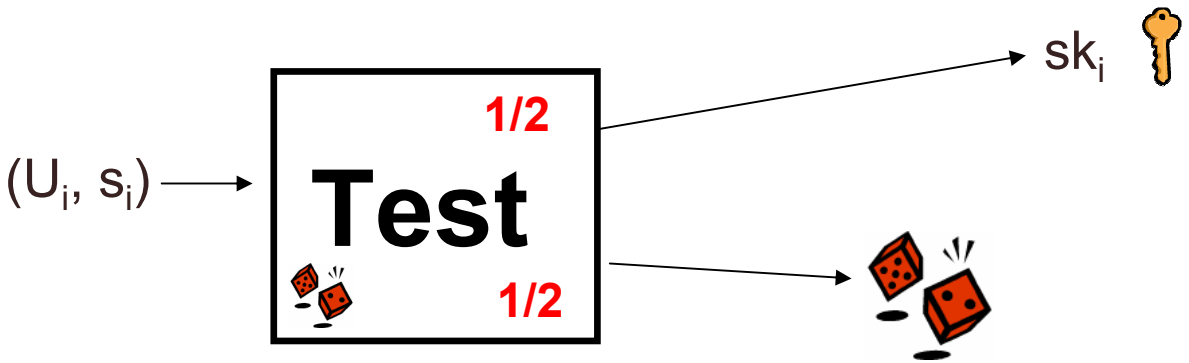


Privacidad de la cl

Aquí, *fresca* quiere decir que $\mathcal{A}$ no conoce la clave que se construye en esta sesión por razones triviales

Se formaliza a través del siguiente planteamien

Considerar un oráculo de simulación Test al que el adversario sólo puede llamar con una ejecución *fresca* $\Pi_{U_1}^{s_i}$



Privacidad de la clave

(caso de autenticación con passwords)

Sea P la probabilidad de que $\mathcal{A}$ adivine correctamente cuál fue la salida de Test.
Decimos que el esquema es seguro si

$$|2P - 1| \leq \quad C^{te}(q / |\mathcal{D}|) \; + \quad \text{negl}(l),$$

donde q denota el número de llamadas a Send que hace $\mathcal{A}$.

(Muy) breve repaso histórico

GAKE: historia incompleta (en una transparencia)

- [Burmester-Desmedt94, 05]
  - Diffie-Hellman para grupos (constant round)
- [Katz-Yung 03]
  - Compilador para construir GAKE a partir de GKE usando firma digital.
- [Kim-Lee-Lee 04]
  - Variante del esquema de BD en el ROM.
- [Joux00]
  - Diffie-Hellman para tres participantes en una ronda
- [Li-Pieprzyk99, Bresson-Catalano04]
  - Esquema basado en secret-sharing.

GPAKE: historia incompleta (en 2 transparencias)

- [Bresson et al. 02, 05]
  - Diffie-Hellman para grupos, número de rondas lineal en el número de participantes.
- [Lee-Hwang-Lee 04]
  - Seguridad demostrable en el ROM.
  - Roto por Abdalla et al. (06)
- [Dutta-Barua 06]
  - Seguridad demostrable en el ROM/ideal cipher model
  - Roto por Abdalla et al. (06)
- [Abdalla et al 06, Tang Choo 06]
  - Basado en BD
  - Seguridad demostrable en el ROM/ideal cipher model.

GPAKE: historia incompleta (en 2 transparencias)

- [Kwon-Jeong-Lee06]
  - Simplificación de Abdalla et al. 06, modelo standard.
  - Aparentemente inseguro…
- [Abdalla-Pointcheval 06]
  - Basado en el esquema de Gennaro-Lindell
  - Seguridad en el modelo estandar.
- [Bohi-G.V-Steinwandt – eprint]
  - Similar a Abdalla et al, más eficiente (de 5 a 3 rondas)
- [Abdalla-Bohli-G.V.Steinwandt07]
  - Compilador para pasar de 2-PAKE a GPAKE
  - Seguridad demostrable en el modelo estandar.

Algunas Construcciones

Burmester-Desmedt 94,05

Parámetros públicos: g, p, donde g genera de un grupo cíclico de orden p

Round 1:
Computación: Cada U_i elige $k_i \in Zp$,
Broadcast: Cada U_i envía (U_i, g^{k_i})

Round 2:
Computación: Cada U_i computa una clave Diffie-Hellman con
U_{i+1} y U_{i-1}, $Z_{i,i+1}$ $Z_{i-1,i}$, y calcula el cociente
$X_i = Z_{i,i+1}/Z_{i,i-1}$
Broadcast: Cada U_i envía (U_i, Xi)

Computacion: Cada U_i calcula la clave sK como el producto de todas las claves dos a dos intercambiadas;

$$sk_i = (Z_{i,i-1})^{nk_i} X_i^{n-1} \ldots X_{i+n-2}$$

GPAKE basado en el esquema dos parte de Gennaro-Lindell

Bohli, G-V, Steinwandt [preprint]

Herramientas

Smooth Projective Hashing (Cramer-Shoup 02)

Sean X, Π, S conjuntos no vacíos, $L \subseteq X$, y K un conjunto de índices finitos. Considerar

$$H:=\{ H_k : X \mapsto \Pi \}, k \in K$$

y la proyección

$$\alpha : K \mapsto S.$$

A **H** = (H, K, X, L, Π, S, α) se le llama *Familia hash proyectiva* - PHF - asociada a (X, L) si

$$\alpha (k) \approx H_{k|L} ()$$

Herramientas (II)

Asumimos que existen algoritmos eficientes para elegir (u.a.a.) elementos de K, describir y evaluar α y evaluar H_k en un valor $x \in X$ siempre y cuando:

- (x, k) sean conocidos
- (x, w, α(k)) sean conocidos, donde w sea un testigo de $x \in L$

Smoothness → Conociendo sólo x y α(k), H_k(x) is "indistinguible" de un valor aleatorio.

Un ejemplo (basado en El Gamal)

G grupo cíclico de orden primo q, g generador, $h = g^s$

Espacio de clave: $K := \mathbb{Z}_q^2$

Proyección: $\alpha(k1, k2) := g^{k1} h^{k2} = g^{k1 + sk2}$

Familia H: $H_k(x, y, z) = x^{k1}(yz^{-1})^{k2}$

Si (x, y, z) es de la forma $(g^r, h^r m, m)$, conocer r y $\alpha(k1, k2)$, permite computar $H_k(x,y,z)$ como $\alpha(k1, k2)^r$

Smoothness: si (x,y,z) es de la forma $(g^r, h^j m, m)$ entonces $H_k(x,y,z) = g^{k1 + jsk2}$. es un elemento aleatorio del grupo.

Herramientas (III)

Smooth projective hashing a partir de compromisos no maleables

Sea C un esquema de compromiso apropiado, y denotar por $\mathcal{C}(pw, r)$ a un compromiso al password pw que usa aleatoriedad r.

Sea C un conjunto (eficientemente reconocible) conteniendo a todos los posibles outputs de C.

Definimos $X := C \times \mathcal{D}$

$$L := \{ (c, pw) \in X \mid \exists\, r : c = \mathcal{C}(pw, r) \}$$

Un testigo para $x \in L$ es la aleatoriedad r tal que

$$c := \mathcal{C}(pw, r).$$

Herramientas (III)

Smooth projective hashing a partir de compromisos no maleables

Recordamos: X:= C x D and L:= { (c, pw) $\in$ X | $\exists$ r : c = C (pw, r)}

Supongamos que se dispone de una smooth PHF para (X,L):
Para cada k $\in$ K se define H_k : X $\mapsto$ G, (G grupo).
La seguridad del ejemplo siguiente se basa en la siguiente propiedad:

Dada una proyección α(k) y dos compromisos c1 y c2 asociados al mismo password pw, los valores H_k(c1, pw) y H_k(c2, pw) son computacionalmente indistinguibles de valores elegidos independientemente u.a.a. en G, siempre que no se conozcan testigos asociados.

Esquema del Protocolo (I)

Round 1:

Computación: Cada U_i elige u.a.r. $k_i \in K$, una aleatoriedad r_i y construye un compromiso al password

$c_i := \mathcal{C}(pw, r_i)$, y la proyección $S_i := \alpha(k_i)$

Broadcast: Cada U_i envía

$Mi^1 := (U_i, S_i, c_i)$

Round 2:

Computación: Cada U_i computa:

$Z_{i,i+1} := H_{ki}(pw, c_{i+1}) \cdot H_{ki+1}(pw, c_i)$

$Z_{i-1,i} := H_{ki-1}(pw, c_i) \cdot H_{ki}(pw, c_{i-1})$

$X_i := Z_{i,i+1} \cdot Z_{i,i-1}^{-1}$

$c_i := \mathcal{C}(X_i, r_i')$

Broadcast: Cada Ui envía

$Mi^2 := (U_i, c_i)$

Protocol Sketch (II)

Round 3:

Broadcast: Cada U_i envía

$Mi^3:=(U_i, X_i , r'_i)$

Check: Cada U_i comprueba $X_n \cdots X_1 = 1$ y la corrección de los commitments recibidos.

Computación:

Cada U_i computa los valores $Z_{i,j}$ y obtiene la master key $K:=(Z1,2,\ldots,Zn,1,$ $pid)$.

A partir de K construirá la clave de sesión y el identificador de sesión; haciendo

$UH(K):=\lambda$, $sk:=F\lambda(v_1)$, $sid:= F\lambda(v_0)$.

Compilador 2-AKE a PAKE

Abdalla, Bohli, G-V, Steinwandt, [TCC 07]

Idea de la construcción:

- Partimos de cualquier esquema AKE para dos entidades, construímos un GAKE seguro
- Cada usuario Ui ejecuta el esquema AKE con sus vecinos U_{i-1} and U_{i+1}, estableciendo claves K_i^L y K_i^R
- Cada usuario Ui construye un compromiso al valor $X_i = K_i^L \oplus K_i^R$ antes de distribuirlo
- La relación $X_1 \oplus \ldots \oplus X_n = 0$ y los compromisos sirven para verificar la autenticidad de los mensajes recibidos
- La clave de sesión se construye a partir de las claves AKE intercambiadas y de la pid de la sesión (para preservar la integridad)

Compilador (I)

Round 0 [2-AKE] :

Para i=1…n, se ejecuta 2-AKE(U_i, U_{i+1}). Al término cada usuario U_i tiene dos claves K_i^L y K_i^R que comparte con U_{i-1} y U_{i+1} resp.

Round 1:

Computación: cada U_i computa $X_i = K_i^L \oplus K_i^R$ y elige una aleatoriedad r_i para calcular un compromiso a X_i, $C_i = \mathcal{C}(i, X_i, r_i)$

Broadcast: cada U_i envía el mensaje

$$Mi^1 := (U_i, C_i)$$

Compilador (II)

Round 2:

Broadcast: cada U_i envía $Mi^2:=(U_i, X_i, r_i)$

Check: cada U_i comprueba $X_1 \oplus \ldots \oplus X_n = 0$ y la validez de los compromisos.

Computación: each U_i hace $K_i: = K_i^L$ y calcula los n-1 valores:

$$K_{i-j}:= K_i \oplus X_{i-1} \ldots \oplus X_{i-j} = 0.$$

Luego define una "master key" $K:=(K_1,\ldots,K_n, pid)$ y obtiene $UH(K):=\lambda$. (UH función hash universal)

Finalmente, construye

$$sk:=F\lambda(v_1),\ sid:= F\lambda(v_0)$$

Compilador (II)

Rou

λ selecciona una función dentro de una familia "especial" de funciones pseudoaleatorias – Katz-Shin 05)

os compromisos.

Comp valores:

$\oplus X_{i-1} \ldots \oplus X_{i-j} = 0$.

Luego define una "master key" $K:=(K_1,\ldots,K_n, pid)$ y obtiene $UH(K):=\lambda$. (UH función hash universal)

Finalmente, construye

$sk:=F\lambda(v_1)$, $sid:= F\lambda(v_0)$

Análisis de Seguridad

- Integridad: Viene de la resistencia a colisiones de la familia de funciones pseudoaleatorias: a igual λ habrá igual sid, pid y sk.
- Privacidad de la clave: La demostración explota la no-maleabilidad de los compromisos utilizados y las propiedades de la familia de funciones pseudoaleatorias.
- Otras propiedades especiales del 2-AKE se transladan al GAKE (forward secrecy)

Conclusiones

- Al hacer una propuesta es imprescindible delimitar el modelo de seguridad (adversarios considerados, objetivos perseguidos) y demostrar formalmente las propiedades del esquema en ese escenario.
- Es difícil construir esquemas demostrablemente seguros y eficientes en el modelo estándar, aunque ya existen varias aceptables.
- Actualmente, la mayoría de los trabajos se encaminan a conseguir esquemas en modelos más restrictivos:
  - UC – composibilidad universal
  - Composibilidad con dependencias en aleatoriedades
  - Usuarios maliciosos

¡¡Gracias!!

¿preguntas?