Criptografía Avanzada

M.I. GONZÁLEZ VASCO / GRADO EN INGENIERÍA DE LA CIBERSEGURIDAD

UNIVERSIDAD REY JUAN CARLOS

Qué vamos a aprender

1. Esquemas de compartición de secretos
2. Esquemas de compromiso

Capítulo 23 (sección 5), Capítulo 24 (secciones 1 y 2)

1. Esquemas de Compartición de Secretos

Definición de SSE

Secret Sharing Schemes, Shamir, 1979

Un usuario U_0 distribuye información parcial sobre un secreto s a n participantes

$U_1, \ldots, U_n$

de modo que todos juntos tengan toda la información sobre s (pero cada individuo no tenga información en absoluto)

Esquemas umbral (TSSE)

Un usuario especial (dealer) comparte un secreto entre n participantes, de modo que:

- Cada parte $i \in [1,\dots,n]$ recibe un secreto parcial
- Cualquier grupo de k participantes puede cooperar y reconstruir el secreto
- Ningún grupo de $k-1$ participantes puede obtener ninguna información sobre el secreto.

Ejemplo (mala idea)

- Sea K una clave de 100 bits de un cifrador en bloque.

¿Por qué no compartirla entre dos usuarios dando a cada uno 50 bits?

Esquema de Shamir para compartición de secretos

Principio matemático utilizado:

Dados k puntos del planto $(x_1, y_1), \ldots, (x_k, y_k)$, donde los x_i son todos distintos, existe un único polinomio f de grado = k − 1, tal que

$$f(x_i) = y_i \text{ para } i=1,\ldots, k$$

Demostración (constructiva): Dados dichos k puntos, puede reconstruirse f usando la formula de interpolación de Lagrange

(¡Esto funciona también en el cuerpo Zp, siendo p primo!)

Shamir SSE; Fase 1

- Sea s un elemento (secreto) de Zp, p primo
- Elijamos al azar cualquier polinomio de grado k-1 con s como término independiente:
  - Elegir $f_{1}, \ldots, f_{k-1}$ u.a.a. en Zp
  - Fijar $f_0 := s$
  - El polinomio es $f(x) = f_0 + f_1 x + \ldots + f_{k-1} x^{k-1}$

- Para $i \in [1, \ldots, n]$, distribuir los valores

$$s_i = (i, f(i))$$

al participante i-ésimo

Shamir, corrección

El secreto s puede ser reconstruído a partir de cualquier subconjunto de k pares (i, y_i) ----siendo f(i)= y_i

Demostración: Interpolación de Lagrange, dados k puntos

(i , y_i), i = 1, ... , k,

$$f(x) = \sum_{i=1}^{k} y_i \prod_{j=1, j \neq i} \frac{x-j}{i-j} \mod p$$

Y, en particular

$$f(0) = \sum_{i=1}^{k} y_i \prod_{j=1, j \neq i} \frac{-j}{i-j} \mod p$$

Shamir, seguridad

Cualquier subconjunto de menos de k puntos no filtra ninguna información acerca del secreto.

Demostración: dados k-1 pares de la forma (x_i, y_i) para cada valor s0 de Zp podemos definir un polinomio f de grado k−1 tal que f(0) = s 0 .

Conclusión:

El esquema de Shamir es seguro (independientemente de las capacidades computacionales de los usuarios)

2. Esquemas de Compromiso

Aplicaciones

- En esquemas multiusuario, sirven para "corregir" la asincronía de la red y evitar abusos (subastas, concursos…)
- En herramientas criptográficas complejas, sirve para evitar ataques de usuarios del sistema que puedan elegir sus inputs *adaptándolos* a la información recibida durante la ejecución.

Definición

Un esquema de compromiso es una terna de algoritmos

(Setup, Commit,Open)

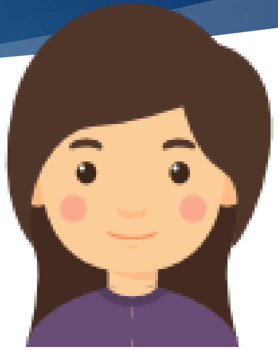
- **Setup**: pptm, recibe como entrada 1^n y genera la clave pública ck
- **Commit:** pptm, recibe como entrada un mensaje m y una clave ck y da como salida un par (c,d)
- **Open:** recibe como entrada un par (c,d) y la clave pública ck, da como salida un mensaje de error $\perp$ o un mensaje m.

Corrección: m= **Open**(**Commit**(m, ck), ck)

Estructura de uso

Ck público

¿¿De dónde sale?? ¿¿Quién ejecuta el Setup??



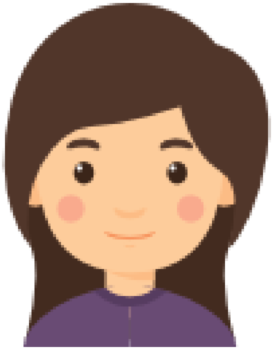
Icons from www.flaticon.com

Estructura de uso

Ck público

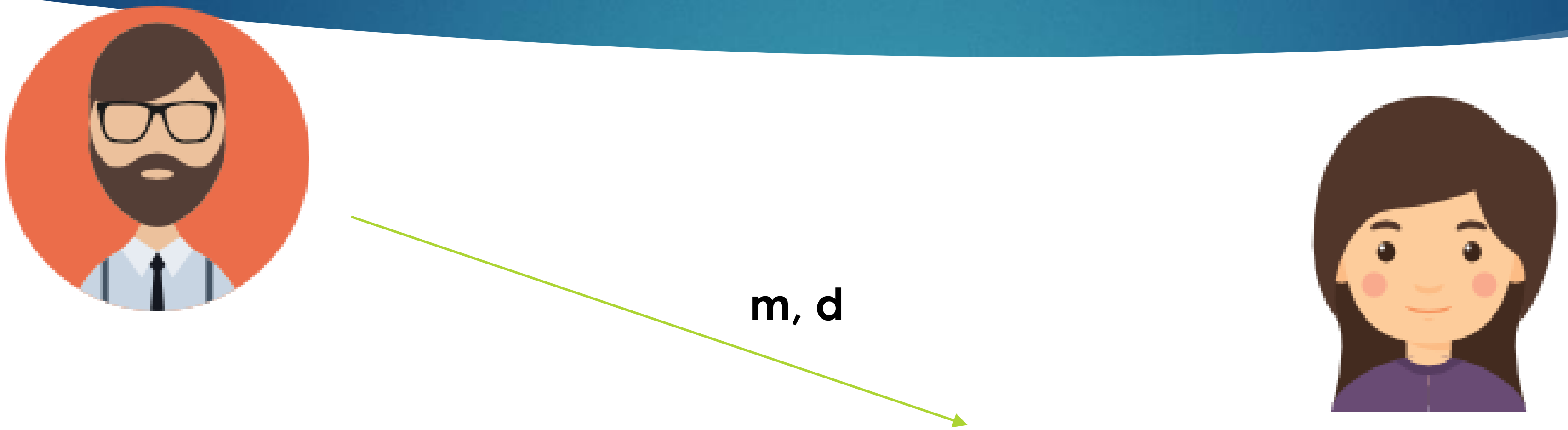
Fase de Compromiso:
Bob ejecuta **Commit(m,ck) = (c,d)**

c



Icons from www.flaticon.com

Estructura de uso



Fase de Apertura:
Alice comprueba la igualdad
$m = \textbf{Open}(\textbf{c}, \textbf{d}, ck)$

Icons from www.flaticon.com

Propiedades de seguridad

Hiding: el compromiso c no debe revelar nada del valor que “esconde”

…….más aún: dada la clave ck, no es posible generar dos mensajes
m0 y m1 tales que sus correspondientes compromisos puedan distinguirse

Propiedades de seguridad: binding

Binding: no podemos "echarnos atrás" una vez efectuado el compromiso…

… No es posible para un advesario constuir una terna

$$(c, d, d')$$

llamada colisión, de modo que

- (c,d) sea un compromiso válido para m
- (c,d') sea un compromiso válido para m'
- $m \neq m'$

Ejemplo - Pedersen

Esquema de compromiso basado en el problema del logaritmo discreto.

- **Setup**: recibe como entrada 1^n y genera un primo p de n bits, y un elemento u.a.a de Z_p^* y g un generador de Z_p^*

$$ck := (p, y, g)$$

- **Commit:** recibe como entrada un bit y selecciona al azar un exponente r en Z_p^* , construyendo $c = g^r y^b \bmod p$ y así, d:=r.
- **Open:** recibe como entrada un par (c, r) y da como salida el bit b tal que $c = g^r y^b$ ---- si la igualdad no se cumple para 1 ni para 0, devuelve $\perp$

Ejemplo – Pedersen: seguridad

- **Hiding** (¡¡es incondicional!) :

  tanto g^r como $g^r y$ son elementos u.a.a. en $Z_p{}^*$

- **Binding** : Si un adversario construye (c, r0) y (c, r1) de manera que:

  Open(c, r0) = 1 y Open(c, r1) =0

Tendría que cumplirse

$$g^{r0} = g^{r1} y \text{ y por tanto } y = g^{r0-r1}$$

Es decir, ¡¡el adversario ha resuelto el problema del logaritmo discreto para (g,y)!!!!

FIN

¡¡Enhorabuena!!

¡¡Hemos terminado la teoría en esta situación tan excepcional!!

Icons made by photo3idea_studio