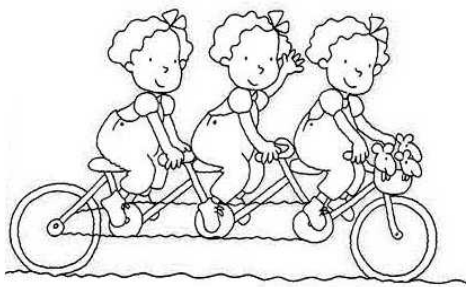
Protocolos Criptográficos

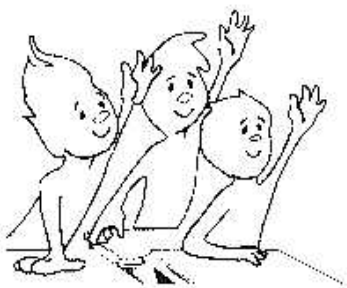
Compartición de Secretos



Canal Subliminal

Lanzamiento de monedas

Elecciones



Poker Mental

Firma de Contratos

Transferencia Inconsciente

Demostraciones de Conocimiento Nulo

Protocolos Criptográficos

Algoritmos utilizados por un conjunto de participantes con una meta común e implementados en entornos distribuidos inseguros.

- Todos los participantes deben conocer su descripción y deben estar de acuerdo en usarlo.
- Debe estar claramente definido lo que gana cada usuario con su ejecución.
- Debe estar prevista cualquier posible situación
- Normalmente incluyen intercambios sucesivos de mensajes entre los participantes
- A menudo se supone en su diseño una capacidad de cálculo limitada para los participantes

Protocolos Criptográficos

- Cada participante es modelizado como una **Máquina de Turing** interactiva con cinta de entrada privada (de sólo lectura) y cinta de salida (de sólo escritura).
- Todas las máquinas comparten una cinta de entrada común (de sólo lectura) y una cinta de salida (de sólo escritura).
- Un protocolo especifica un programa para cada máquina, de forma que al final cada máquina tendrá en su cinta de salida su mensaje de salida particular, y todos los programas coordinan todos los mensajes.
- Se suele suponer que son máquinas de Turing probabilistas con potencia de cálculo limitada polinomialmente
- El caso peor de participantes deshonestos se modeliza suponiendo que pueden llevar a cabo un ataque coordinado.

Protocolos Criptográficos

- Se considera adversario **pasivo** a aquel que puede leer las cintas privadas y los mensajes de todas las máquinas intervenidas.
- Se llama adversario **activo** a aquel que puede además especificar los mensajes que dichas máquinas enviarán al resto, de manera que puede forzar la violación del protocolo.
- Un adversario **rushing** sería aquel que puede leer toda la información recibida por las máquinas intervenidas antes de decidir los mensajes que éstas enviarán al resto.
- Un adversario **estático** controla siempre al mismo conjunto de máquinas durante el protocolo.
- Un adversario **dinámico** puede incrementar el número de máquinas intervenidas durante cada iteración del protocolo
- En la mayoría de protocolos se considera sólo el caso de un posible adversario activo estático, y en la demostración formal de su seguridad se requiere alguna suposición de complejidad

Primitivas y Algoritmos Bipartitos y Multipartitos

- Primitivas
  - Transferencia Inconsciente
  - Compromiso de Bits
- Algoritmos Bipartitos
  - Conocimiento Nulo
  - Firma de Contratos
  - Lanzamiento de Monedas

- Algoritmos Multipartitos
  - Distribución de Secretos
  - Compartición de Secretos
  - Elecciones Electrónicas
  - Póquer Mental
  - Dinero Digital
  - Sellado Temporal

Transferencia Inconsciente

- A (Alicia) quiere transferir un secreto a B (Benardo) de forma que dicha información se transfiera con una probabilidad 1/2, y que al final B sepa con certeza si la ha recibido, pero A no.
- *Rabin, M., How to exchange secrets by oblivious transfer. Technical Report TR-81, Harvard Aiken Computation Laboratory, (1981).*
- Se puede usar como base un cifrado probabilístico con probabilidad 1/2
- Puede usarse como primitiva para cualquier protocolo bipartito

TI: Protocolo de Rabin

Secreto: Factorización del producto de dos grandes primos.

1.- A escoge al azar dos grandes números primos p y q, y calcula y envía a B, $N=p \cdot q$.

2.- B verifica si N es primo, potencia de primo o par, y si cualquiera de estos casos ocurre, entonces B anuncia que A hace trampas. En otro caso, B escoge un entero x al azar entre 1 y N-1 y primo con N, y envía $x^2 \pmod N$ a A.

3.- A calcula las cuatro raíces cuadradas diferentes de $x^2 \pmod N$, {x, N-x, y, N-y}, escoge una al azar y se la envía a B.

4.- Si B recibe y o N-y, entonces él puede calcular p y q gracias a que ((x+y), N) es p o q. Si B recibe x o N-x, entonces él no puede calcularlos.

TI: Protocolo de Rabin

Secreto: Factorización de 77=11*7

1.- A escoge p=7 y q=11, y envía a B N=77

2.- B verifica que 77 no es primo, potencia de primo ni par, escoge un entero x=13 entre 1 y 76, y primo con 77, y envía a A $x^2 \pmod N = 13^2 = 15 \pmod{77}$

3.- A calcula las cuatro raíces cuadradas diferentes de 15 (mod 77), {13, 77-13=64, 20, 77-20=57}, y envía una a B .

4.- Si B recibe 20 o 57, entonces puede calcular p y q gracias a que (13+20, 77)=11. Si B recibe 13 ó 64, entonces no puede calcularlos.

TI: Protocolo de Rabin

- Problema de la **residuosidad cuadrática**:

  Encontrar a tal que $x^2=a \pmod N$, siendo $a<N$

  - Si $N=p*q \Rightarrow$ Existen $(p-1)(q-1)/4$ residuos cuadráticos (mod N)
  - Ej: residuos cuadráticos mod 35: 1,4,9,11,16,29
- Estafa de A: A no sabe cuál es la raíz cuadrada que B conoce, luego elige una raíz al azar.
- Estafa de B: B sólo obtiene una raíz cuadrada de una potencia cuadrada de un número aleatorio, si realmente x lo es. Pero podría ser que B lo escogiera a partir de un valor z tal que $z=x^2 \pmod N$, lo que puede ser un ventaja para factorizar N. Por tanto sería conveniente que B demostrara a A que realmente x fue elegido al azar, por ejemplo usando una DCN.

TI: Protocolo de Pfleeger

1. A elige dos **claves públicas** y obtiene sus correspondientes claves secretas
2. B escoge una clave secreta k_B y una de las dos claves públicas de A al azar, cifrando con ésta k_B y enviando el resultado a A
3. A escoge al azar una de sus dos claves secretas, descifra con ella el mensaje recibido de B, cifra con el resultado de este descifrado un mensaje secreto y envía lo que obtiene a B
4. Si la clave elegida por A en el paso 3 se corresponde con la clave pública escogida por B en el paso 3, B podrá descifrar el mensaje secreto de A. Si no, no.

TI 1-2

A tiene dos secretos m_0 y m_1. Se pretende que B elija y obtenga exactamente uno de ellos, sin que A sepa cuál es.

•Generalización: Venta de secretos

•Protocolo de **Salomaa**:

1.A escoge un generador g del grupo cíclico G, y un elemento aleatorio $a \in \{0,1,...,|G|-1\}$, y calcula $A=g^a$. Su clave pública es $(G,g,|G|,A)$ y su clave secreta es a.

2.B escoge $j \in \{0,1\}$ y un elemento aleatorio $b \in \{0,1,...,|G|-1\}$. Calcula y envía a A, $B_0=A^j g^b$ y $B_1=A^{1-j}g^{-b}$

3.A comprueba que $A=B_0B_1$, escoge al azar $y_0,y_1 \in \{0,1,...,|G|\}$ y calcula $A_i=g^{yi}$, las claves $K_i=B_i^{yi}$, y $c_i=K_i m_i$. A envía a B (A_i,c_i)

4.B calcula la clave $K_j=A_j^{xj}$ siendo $x_j=(-1)^j b$ y el mensaje $m_j=K_j^{-1}c_j$

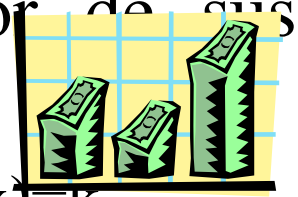
TI Combinada

Dos participantes A y B tienen sendos bits secretos a y b, y se pretende que A reciba f(a,b) sin aprender nada sobre b, y sin que B aprenda nada sobre a.

- Goldreich, O., Micali, S., Wigderson, A., *Proofs that yield nothing but their validity and a methodology of cryptographic protocol design*. Proceeding 27th IEEE Symposium on Foundations of Computer Science, 174-187, (1986).
- Aplicación en Elecciones Electrónicas y Póquer Mental

TI Combinada: Problema de los Millonarios

Dos millonarios desean saber cuál de los dos es más rico manteniendo en secreto el valor de sus respectivas pertenencias.



1. B escoge un número aleatorio x y calcula $E_A(x)=k$
2. B dice a A el número k-j
3. A escoge un primo p, calcula los números $z_u=D_A(k-j+u) \pmod p$, $1\leq u\leq 100$, y verifica que $\forall u\neq v:|z_u-z_v|\geq 2$ y $0<z_u<p-1$, y en caso contrario elige otro primo
4. A comunica a B por orden $z_1,z_2,...,z_i,z_{i+1}+1,...,z_{100}+1,p$
5. Si el j-ésimo elemento es congruente con x módulo p, B concluye que $i\geq j$, y en caso contrario que $i<j$

Compromiso de Bits

- A se compromete frente a B con un valor, de forma que A no puede cambiarlo, y B no puede descubrir el valor hasta que A abre el compromiso.
- Blum, M., *Coin flipping by telephone*, Proceedings IEEE Spring COMPCOM, 133-137, (1982).
- Util en Lanzamiento de Monedas y Demostración de Conocimiento Nulo.
- Analogía: Sobre cerrado (inalterable e ilegible)
- Apertura del compromiso: Correspondencia definida desde un gran dominio sobre {0,1}:
  - –Un bit resulta comprometido cuando se da un elemento aleatorio en el origen de la correspondencia para ese valor de salida.
  - –El esquema es inalterable cuando la correspondencia es función.
  - –Es ilegible cuando las distribuciones del conjunto origen del 0 y las del conjunto origen del 1 son indistinguibles.

Compromiso de Bits

- Propiedades:
  - Debe poderse comprometer cualquiera de los dos posibles valores para cada bit
  - No se puede modificar el valor comprometido variando la forma de abrir el compromiso.
  - B no debe aprender nada sobre cómo se abren los compromisos a partir de la apertura de varios
  - De la apertura del compromiso, B sólo obtendrá el valor del bit comprometido.

Compromiso de Bits

- Usando **transferencia inconsciente** para comprometer un bit b:
  - A escoge n bits aleatorios b_i, tales que $b_1+b_2+...+b_n=b$
  - **Compromiso**: A envía cada b_i por orden mediante transferencia inconsciente
  - **Apertura**: A envía a B los bits b_i
  - **Verificación**: B compara los bits b_i recibidos en el paso de compromiso con los correspondientes de la apertura
- La probabilidad de fraude de A es $\leq 2^{-1}$, pero puede reducirse a 2^{-k} si se ejecuta k veces el algoritmo en paralelo

Compromiso de Bits

- Usando una **función hash** h para comprometer un bit x:
  - Compromiso: A calcula y envía a B y=h(x).
  - Apertura: A envía a B el valor x.
  - Verificación: B comprueba que h(x)=y.
- Ventaja: Comunicación unidireccional

Compromiso de Bits

- Usando un **cifrado de clave secreta** para comprometer un bit b:
  - B genera y envía a A una secuencia seudoaleatoria R
  - Compromiso: A cifra con su clave secreta k, la secuencia R y el bit b, y envía a B el resultado $E_k(R,b)$
  - Apertura: A envía a B la clave k
  - Verificación: B descifra el mensaje, comprueba la secuencia R, y descubre el bit b

Compromiso de Bits

- Usando un **generador seudoaleatorio** para comprometer un bit b:
  - B genera y envía a A una secuencia seudoaleatoria R
  - Compromiso: A genera una semilla x_i para su generador, y envía a B:
    - para cada x_i=0, el correspondiente bit de salida del generador
    - para cada x_i=1, la XOR del bit de salida del generador y b
  - Apertura: A envía a B la semilla x_i
  - Verificación: B comprueba el compromiso.

Compromiso de Bits

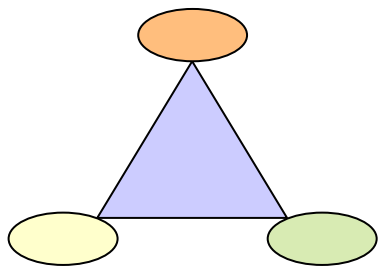
- Usando **logaritmos discretos** para comprometer un bit b:
  - A y B acuerdan un primo p y un elemento a generador de Z_p
  - B escoge al azar s en Z_p y se lo envía a A
  - Compromiso: A escoge al azar un entero y entre 0 y p-2, y calcula y envía a B el número $x=s^b a^y \pmod p$
  - Apertura: A envía a B el entero y
  - Verificación: B obtiene b y comprueba x

Compromiso de Bits

- Usando **residuosidad cuadrática** para comprometer un bit b:
  - A y B acuerdan n=p*q, siendo p=q=3 (mod 4) información secreta de A
  - Compromiso: A genera y envía a B:
    - un residuo cuadrático mod n, si b=0
    - un no residuo cuadrático mod n, si b=1
  - Apertura: A envía a B la factorización (p,q) de n
  - Verificación: B obtiene b comprobando si el número enviado es residuo o no

Compromiso de Bits

- Usando **grafos no isomorfos** para comprometer un bit b:
  - A y B acuerdan dos grafos no isomorfos G y H
  - Compromiso: A genera y envía a B un grafo isomorfo:
    - con G, si b=0
    - con H, si b=1



  - Apertura: A envía a B el isomorfismo
  - Verificación: B obtiene b y comprueba el isomorfismo

Protocolos Bipartitos

- En general la computación bipartita segura es imposible sin suposiciones sobre la complejidad.

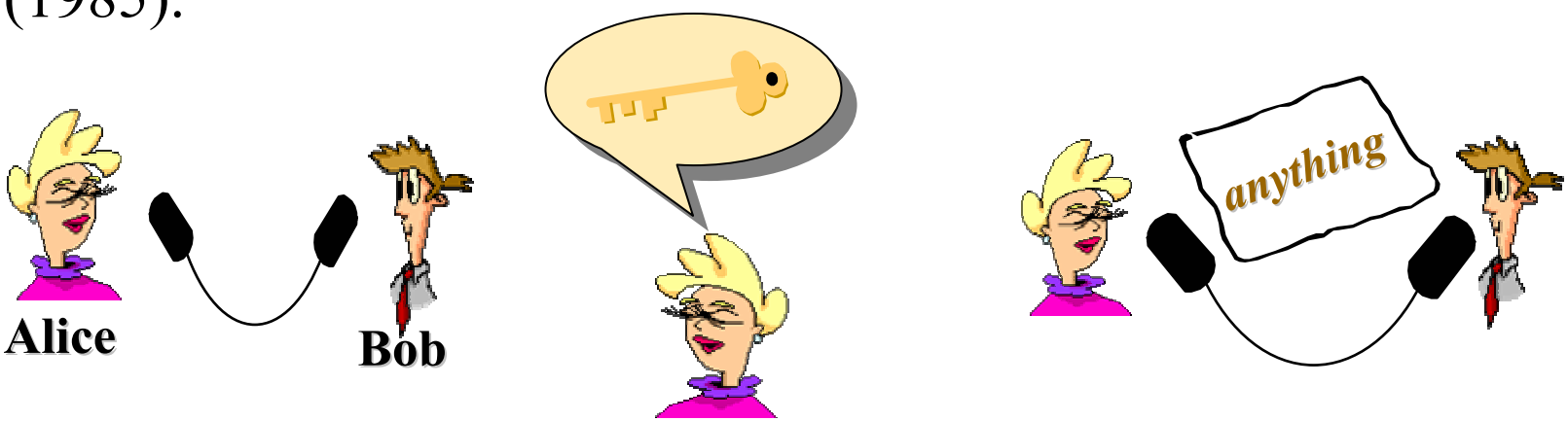
➢ Yao, A.C., *Protocols for Secure Computations*. Proceedings 23rd IEEE Symp. on Foundations of comp. Science, 160-164 (1982).

- Cualquier función f(x,y) puede ser calculada de forma privada por dos participantes suponiendo la dificultad de la factorización

Demostración de Conocimiento Nulo

Protocolo criptográfico que permite a una usuaria A convencer a otro usuario B de que ella tiene algún secreto sin revelarle absolutamente nada sobre él.

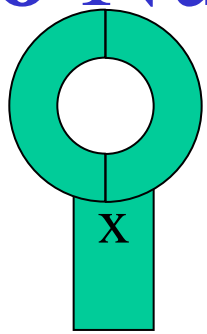
- Goldwaser, S., Micali, S., Rackoff, C., *The knowledge Complexity of Interactive Proof Systems*, Proceedings 17th ACM Symposium on Theory of Computing, 291-304, (1985).



Demostración de Conocimiento Nulo

- Estafa de A: A no puede engañar a B, ya que si en realidad no posee el secreto, entonces la probabilidad de convencer a B de lo contrario es ínfima
- Estafa de B: B no recibe ninguna información a partir de la demostración, salvo el convencimiento de que A posee el secreto

Demostración de Conocimiento Nulo



Ejemplo de la cueva:

A desea demostrar que conoce la palabras mágicas que permiten traspasar la pared dentro de una cueva:

1. A entra sola por uno de los dos pasadizos I (Izquierdo) ó D (Derecho) hasta el final
2. B entra hasta el punto X y le pide a A que vuelva por I o por D.
3. Si A está en el pasadizo que B eligió, sale sin más. Si B está en el otro pasadizo, utiliza las palabras mágicas y pasa al otro para salir.

– A tiene una probabilidad 2^{-1} de estafar a A, pero tras k repeticiones, esta probabilidad baja a 2^{-k}

Demostración de Conocimiento Nulo

Dependiendo de la relación entre las distribuciones de probabilidad de la perspectiva de la interacción entre A y B (todo lo que B observa durante el protocolo) y la de la perspectiva de la interacción entre el simulador y B:

- Conocimiento Nulo **Perfecto**: Si ambas coinciden.
- Conocimiento Nulo **Computacional**: Si ambas son polinomialmente indistinguibles.
- Conocimiento Nulo **Estadístico**: Si ambas son estadísticamente indistinguibles.

Demostración de Conocimiento Nulo

- En la mayoría de protocolos:
  - la seguridad se basa en un **problema difícil**
  - A y B **no** se intercambian **información previa**
  - participan **Terceras Partes Confiables**
  - se evita el uso de **cifrados**
  - existe interactividad entre A y B **(DCNI)**
- **Completitud**: La probabilidad de aceptación para un probador honesto es uno
- **Solidez**: La probabilidad de aceptación para un probador deshonesto es muy baja
- **Conocimiento nulo**: Paradigma de la simulación: Se puede definir un algoritmo polinomial que simule el comportamiento de A $\Rightarrow$ No degradación por uso

Demostración de Conocimiento Nulo

- El **número de iteraciones** debe ser consensuado por A y B, atendiendo a sus intereses contrapuestos
- **Corte y Elección**: A divide la información en partes descritas por preguntas y respuestas, de forma que unas demuestran su conocimiento de la información, mientras que otras sirven de garantía contra sus posibles trampas
- **Reto-Respuesta:** Los retos han de ser totalmente impredecibles para que tras cada iteración aumente la confianza de B.
- **Probabilístico**: Si el número de retos equiprobables en cada iteración es n, tras m iteraciones la probabilidad de estafa de A es como máximo n^{-m}
- **Compromiso de bits**: A se compromete con la información y envía a B un testigo, de forma que en la fase de verificación B comprueba que el compromiso no ha sido roto

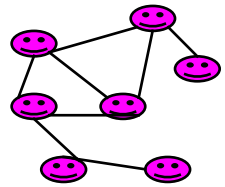
Demostración de Conocimiento Nulo

- Suponiendo la existencia de funciones de cifrado, todos los lenguajes en NP poseen demostraciones de conocimiento nulo (Goldreich, Micali y Widgerson, Crypto 86)

➢ Construcciones concretas con grafos usando problemas NP-completos como:

- ♦ Recubrimiento de vértices
- ♦ Conjunto independiente
- ♦ Subgrafo bipartido
- ♦ Subgrafo cúbico
- ♦ Circuito hamiltoniano

DCN basadas en Grafos



✓A elige como **identificación secreta** una solución de un problema difícil en un grafo G, y usa G como **identificación pública**

➢Se realizan m iteraciones de los pasos siguientes:

1.A elige como **compromiso** secreto una solución del problema en un grafo G' isomorfo a G, y envía G' a B como **testigo**

2.B elige aleatoriamente un bit e y envía este **reto** a A

3.A realiza una de las dos acciones según el valor de e:

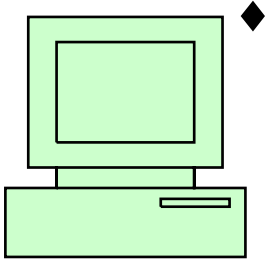
- Si e=0, envía a B el **isomorfismo** entre ambos grafos
- Si e=1, envía a B la **solución en el grafo isomorfo**

4.B **verifica**:

- Si e=0, el isomorfismo entre ambos grafos
- Si e=1, que la información recibida verifica las propiedades de una solución en el grafo isomorfo

DCN basadas en Grafos

- La **seguridad** se basa en un problema difícil de grafos y en la
  - Elección cuidadosa de G y de la solución, y la
  - Capacidad computacional del verificador polinomial
- **Completitud**: Si A es **honesto** el protocolo acaba bien ya que el isomorfismo y la solución en el grafo isomorfo son válidos
- **Solidez**: Un **suplantador de A** tendría éxito cuando e=0 si genera un grafo isomorfo, pero fracasaría con e=1 ya que no podría resolver el problema en dicho grafo. También podría generar una solución en un grafo falso respondiendo correctamente al reto e=1, pero entonces no podría responder a e=0. Luego la probabilidad de suplantación exitosa es 2^{-m}



- **Conocimiento Nulo**: B puede diseñar un **simulador** que juegue el papel de A generando aleatoriamente un grafo isomorfo G', o bien definiendo un grafo G' adecuado a partir de una solución. En tiempo polinomial se genera una perspectiva con distribución de probabilidad indistinguible

DCN basada en Circuitos

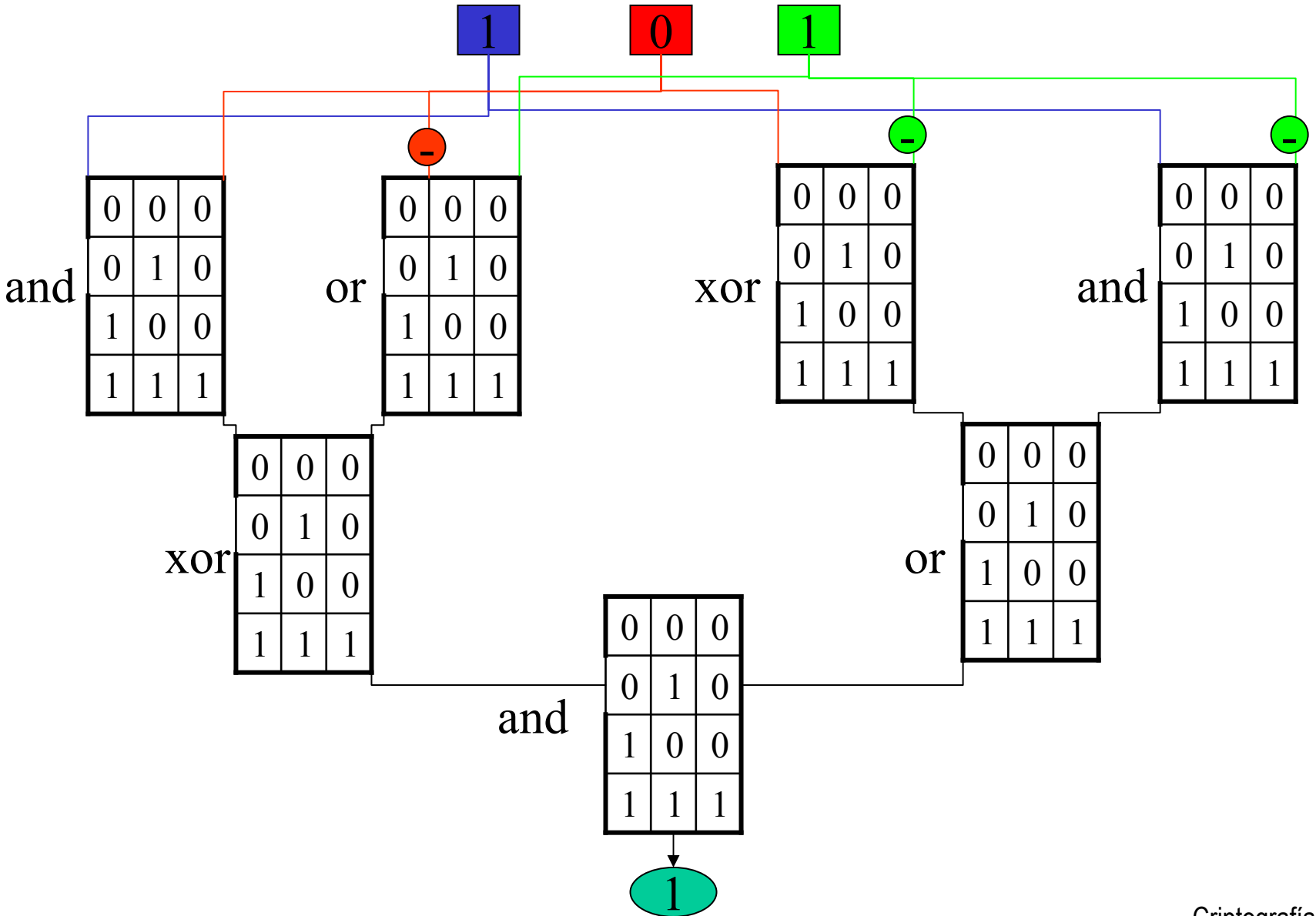
A elige como **identificación secreta** una asignación secreta de valores que satisfacen una fórmula booleana, que usa como **identificación pública**.

Se realizan m iteraciones de los pasos siguientes:

1. A *permuta* las tablas de verdad correspondientes a la fórmula y a la asignación, y elige como **compromiso** secreto el circuito que resulta, y envía las tablas resultantes a B como **testigo**
2. B elige aleatoriamente un bit e y envía este **reto** a A
3. A realiza una de las dos acciones según el valor de e:
   - Si e=0, indica a B las columnas complementadas
   - Si e=1, indica a B la filas correspondientes a la asignación secreta.
4. B verifica que el resultado final es un 1.

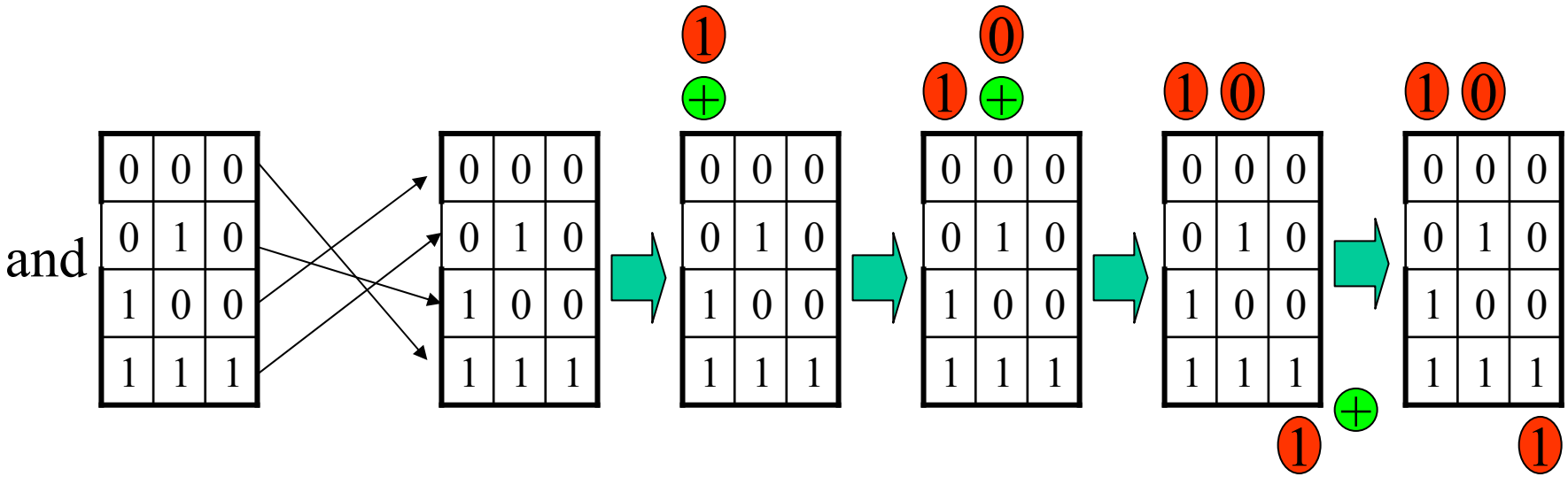
DCN basada en Circuitos

[(p and q)xor(¬q or r)] and (¬r xor q)or(p and ¬r)]



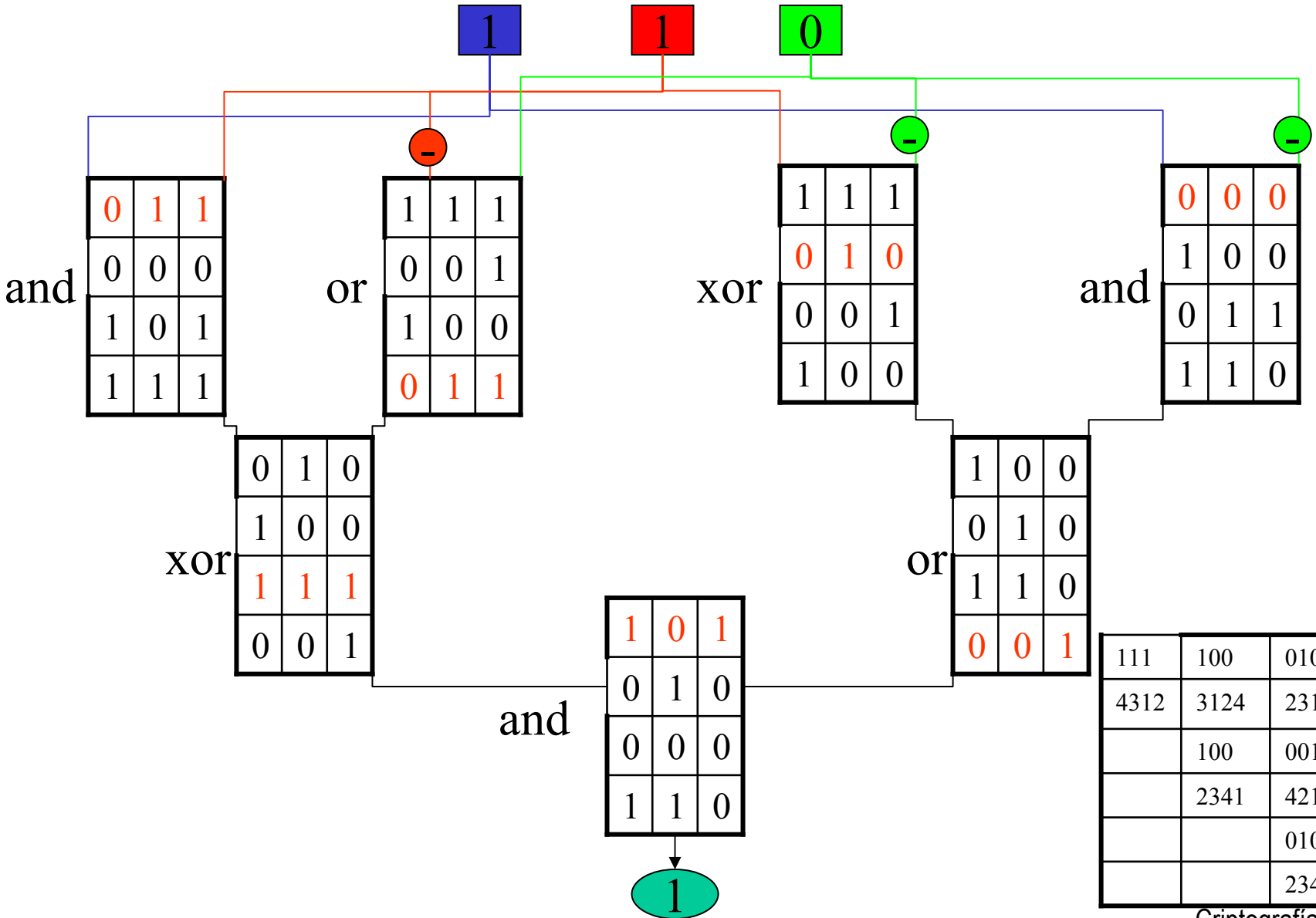
DCN basada en Circuitos

Permutación de las tablas de verdad



1. Permutar filas
2. Complementar aleatoriamente columnas de las primeras tablas
3. Complementar las primeras columnas de las tablas siguientes a las que se hayan complementado

DCN basada en Circuitos



111	100	010	100
4312	3124	2314	2413
	100	001	
	2341	4213	
		010	
		2341	

DCN: Protocolo de Fiat-Shamir

- ✓ Una TPC elige y publica un módulo RSA $n=pq$, manteniendo en secreto los primos p y q
- ✓ A elige como **identificación secreta** un entero s primo con n y tal que $0<s<n$, y calcula su **identificación pública** $v \equiv s^2 \pmod n$
- ✓ Se realizan m iteraciones de los pasos siguientes:
  1. A elige un **compromiso** secreto r tal que $0<r<n$, y envía a B el testigo $x \equiv r^2 \pmod n$
  2. B elige al azar un bit e y envía este **reto** a A
  3. A calcula y envía a B la **respuesta** y:
     - $y = r$, si $e=0$,
     - $y \equiv rs \pmod n$, si $e=1$
  4. B acepta la identificación si $y \neq 0$ e $y^2 \equiv xv^e \pmod n$

DCN: Protocolo de Fiat-Shamir

- La seguridad se basa en el problema de la **residuosidad cuadrática**
- Completitud: Con participantes honestos el protocolo acaba bien siempre ya que $y^2=r^2\equiv xv^0 \pmod n$ y $y^2 \equiv r^2s^2 \equiv xv^1 \pmod n$
- Solidez: Un suplantador de A que no conozca s tendría éxito si e=0, pero sería detectado cuando e=1. También podría elegir cualquier valor r y usar $x=r^2/v$ e y=r para responder al reto e=1 correctamente, pero entonces sería incapaz de responder cuando e=0. Luego la probabilidad de suplantación exitosa es 2^{-m}
- Conocimiento Nulo: La respuesta y=r no depende del secreto s. Por otra parte, la respuesta $y\equiv rs \pmod n$ no proporciona información alguna sobre s dada la aleatoriedad del número secreto r. De hecho, la información (x,y) recibida por B podría haberla generado él solo mediante un simulador que eligiera aleatoriamente y, y definiera $x=y^2$ ó bien $x\equiv y^2/v \pmod n$, ya que en un número de pasos polinomial estaría asegurada una simulación con idéntica distribución de probabilidad

DCN basada en Logaritmos Discretos

- ✓ A elige al azar como **identificación secreta** un entero x tal que $a^x=b \pmod p$, siendo p un primo y x relativamente primo con p-1, y usa como **identificación pública** (a,b,p)
- ✓ Se realizan m iteraciones de los pasos siguientes:
  1. A genera al azar un número r<p-1, que es su **compromiso** secreto, y envía a B el valor **testigo** $h=a^r$
  2. B elige al azar un bit e y envía este **reto** a A
  3. A calcula y envía a B la **respuesta** $s=r+ex \pmod{p-1}$
  4. B **verifica** que $a^s=hb^e$

DCN No Interactiva

1. A usa su secreto y algunos números aleatorios para transformar el problema difícil original en n problemas isomorfos diferentes, los resuelve y se compromete públicamente con sus soluciones
2. A usa dichos compromisos como entradas para una función **hash** y dependiendo del valor de los primeros n bits de la salida:
   - para cada 0, prueba que los correspondientes problemas son isomorfos
   - para cada 1, abre la solución correspondiente comprometida

Firma de Contratos

A y B quieren firmar simultáneamente un contrato a través de una red de comunicaciones, de forma que ninguno pueda obtener la firma del otro sin haber firmado el contrato, y que ninguno pueda repudiar su propia firma.

➢Even, S.,Goldreich,O.,Lempel,A., *A randomized protocol for signing contracts*. Communications of the ACM, 28, (1985).

•A y B se pueden intercambiar alternativamente bits de sus firmas digitales del contrato, de forma que si uno interrumpe el proceso ambos tienen prácticamente la misma cantidad de firma del otro. El problema es que si uno envía basura en lugar de su firma, el otro no lo nota hasta el final.

•No es posible diseñar un protocolo determinístico que no requiera una tercera parte de confianza.

Firma de Contratos

Propiedades de las firmas de contratos **aleatorizadas**:

- Los firmantes están obligados a culminar el proceso a partir de un punto del protocolo.
- Las firmas no son falsificables, y pueden comprobarse al final del protocolo.
- Ambas firmas se dividen en partes que son transmitidas alternativamente.
- Usan esquemas de compromiso de bits
- Existen protocolos basados en clave secreta y basados en clave pública.

FC basada en Transferencia Inconsciente

- Versión del protocolo de **Rabin** aplicado sucesivamente. El contrato se considera firmado al final si ambos usuarios conocen la factorización secreta del otro.
- Versión del protocolo de **Pfleeger** aplicado sucesivamente usando como mensaje secreto bloques de contrato firmados digitalmente.

FC basada en TI y Clave Secreta

El contrato se considera firmado si cada uno tiene la **mitad** de la firma del otro

1. A y B crean cada uno L y R, como sus firmas de la mitad izquierda y derecha respectivamente del contrato.
2. A y B eligen cada uno dos claves secretas de DES K^L y K^R, y las usan para cifrar L y R, obteniendo C^L y C^R, y enviando todos estos mensajes al otro.
3. Cada uno remite al otro por transferencia inconsciente 1 de 2 las claves K^L y K^R

FC basada en TI: Algoritmo de Even, Goldreich y Lempel

Si un participante logra descifrar (n-m) mensajes firmados del otro, se considera que ha obtenido su firma del contrato

1. A y B eligen cada uno resp. 2n claves secretas x_i e y_j, y 2n mensajes, M_i, y M'_j
2. A envía a B ordenadamente $E_{xi}(D_a(M_i))$ por transferencia inconsciente, y B hace lo mismo con $E_{yj}(D_b(M'_j))$
3. A envía a B n claves x_i por transferencia inconsciente, y B hace lo mismo con las y_j
4. A y B descifran lo que pueden

FC: Intercambio de Secretos

A y B quieren intercambiar sus secretos simultáneamente

- Even, S., Goldreich, O., Lempel, A., A randomised protocol for signing contracts. Communications of the ACM, 28, 637-647, (1985).
- Util en Correo Certificado y Firma de Contratos
- Combinación de TI iteradas

1. A y B se intercambian sus claves públicas de RSA, n_A y n_B, y escogen al azar cada uno k números secretos $a_i \leq n_A/2$ y $b_i \leq n_B/2$
2. A envía a B todos los $a_i^2 \pmod{n_B}$ y B envía a A todos los $b_i^2 \pmod{n_A}$
3. A calcula las cuatro raíces cuadradas mod n_A de los nº recibidos y elige las dos menores para cada, y B hace los mismo mod n_B
4. A y B se intercambian los k pares de raíces, de forma que ambos pueden factorizar n_A y n_B

FC: Correo Certificado

A quiere enviar un mensaje a B de forma que éste no pueda leerlo sin devolver un acuse de recibo a A

- Blum, M., Vazirani, U.V., Vazirani, V.V., *Reductibility among protocols*. Proc. Crypto'83.

- Intercambio de Secretos, Correo Certificado y Firma de Contratos son problemas reducibles unos a otros
- La Transferencia Inconsciente se puede usar como primitiva para estos protocolos

Lanzamiento de monedas

- Se trata de generar una secuencia aleatoria común a dos usuarios de forma que:
  - El generador de la secuencia no pueda elegir una secuencia no aleatoria particular
  - El otro usuario no pueda anticipar el resultado del lanzamiento

➢ Blum, M., *Coin flipping by telephone*, Proceedings IEEE Spring COMPCOM, 133-137, (1982).

- Tiene aplicaciones en Póquer mental, Correo certificado, Intercambio de secretos y Generación de claves

Lanzamiento de monedas

Propiedades:

- Las caras y las cruces deben tener probabilidad 0.5
- Cada participante debe saber en cada lanzamiento si el otro hizo trampas o no
- Si un participante coge en una trampa al otro debe poder probarlo ante un juez
- Después de que A lance una moneda, A sabe si fue cara o cruz, pero B no tiene ni idea
- Tras varios lanzamientos, A debe ser capaz de demostrar a B qué moneda fue cara y cuál cruz

Lanzamiento de monedas

- Se puede diseñar a partir de cualquier **transferencia inconsciente** de manera que gana B si logra recibir el secreto transferido

- A y B escogen al azar sendos bits aleatorios a y b, y se los intercambian mediante **compromiso de bits**. Tras la apertura y comprobación de los compromisos se toma como resultado del lanzamiento $a \oplus b$.

Lanzamiento de monedas

Suponiendo la existencia de una **función unidireccional completamente segura** (tal que a partir de f(x) se tiene una probabilidad 0,5 de saber si x tiene o no alguna propiedad).

1. A y B acuerdan usar una f.u.c.s. f
2. A elige un entero secreto x (lanzamiento), y calcula y envía a B, f(x)
3. B dice a A si cree que x es par o impar (apuesta de B)
4. A comunica a B si acertó o no, y para convencerle le envía x

Lanzamiento de monedas

Algoritmo basado en un cifrado de **clave pública** tal que $D_A(E_B(E_A(m)))=E_B(m)$.

1.B se compromete con A sobre su apuesta del lanzamiento

2.A genera dos mensajes m_1 and m_2 (Cara y Cruz), y los cifra con su clave pública, enviando a B los dos mensajes $E_A(m_i)$

3.B escoge al azar uno de los dos mensajes, lo cifra con su clave pública y envía el resultado a A: $E_B(E_A(m_i))$.

4.A descifra con su clave secreta el mensaje recibido, enviando el resultado a B: $D_A(E_B(E_A(m_i)))=E_B(m_i)$.

5.B descifra el mensaje obtenido con su clave secreta, recupera el resultado de su apuesta, y se lo envía a A.

6.A verifica la información recibida, comparándola con los datos que tiene.

Lanzamiento de monedas

A da a B un **residuo cuadrático** y B apuesta por que su raíz cuadrada es par o impar

1.A escoge al azar un entero Blum N (producto de dos primos congruentes con 3 módulo *4*), y $x \in GF(N)-\{0\}$, calcula $y=x^2 \pmod N$ y $z=y^2 \pmod N$, y envía a B N y z.

2.B anuncia su apuesta por que y es par o impar.

3.A comunica x e y a B, y le convence de que N es un entero Blum.

4.B verifica que $y=x^2 \pmod N$ y $z=y^2 \pmod N$.

LM: Protocolo de Blum

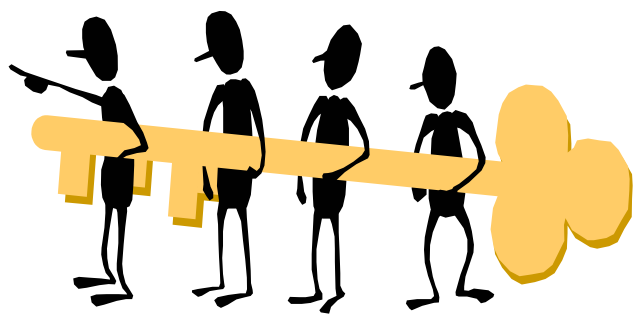
1. B elige aleatoriamente dos grandes primos distintos p_1 y p_2 congruentes con 3 (mod 4), calcula y envía a A $n=p_1*p_2$
2. A comprueba que $n=1 \pmod 4$ y que para algún x existe un y tal que $x^2 = y^2 \pmod n$
3. A elige valores aleatorios x_i y envía a B sus cuadrados (mod n) junto con n
4. B comprueba n, y envía a A n, $x_i^2 \pmod n$ y b_i
5. A comprueba n y $x_i^2 \pmod n$, y determina la secuencia aleatoria: 1 si B acertó sobre x_i, y -1 en otro caso
6. A envía a B los x_i
7. B comprueba los cuadrados

Compartición de Secretos

Un secreto se divide en partes (sombras) que se distribuyen entre varios participantes de forma que sólo cuando un número de ellas (no necesariamente todas) se reúnen es posible reconstruir el secreto.

- Para proteger información secreta importante se puede:
  - –hacer varias copias y distribuirlas entre los participantes
  - –dividirla y distribuir las partes entre los participantes

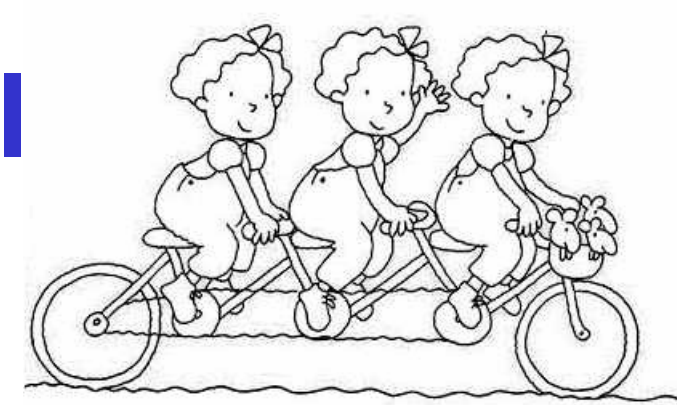
- *Blakley, G.R., Safeguarding cryptographic keys, Proceedings AFIPS 1979 National Computer Conference, 313-317, (1979).*
- *Shamir, A., How to share a secret. Communications of the ACM, 612-613, (1979).*



CS: Propiedades

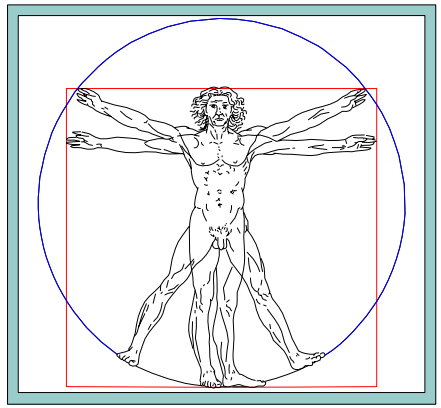
- Si P es el conjunto de sombras, se llaman estructuras de acceso $\Gamma \subseteq 2^P$ a los distintos subconjuntos de sombras que permiten calcular el secreto, y agrupación autorizada a cada uno de esos subconjuntos.
- Un esquema perfecto es el que:
  - –permite a las agrupaciones autorizadas obtener el secreto, y
  - –no permite a ninguna agrupación no autorizada conseguir ninguna información sobre el secreto.
- Siempre se puede diseñar un esquema perfecto a partir de cualquier estructura de acceso.
- La base del esquema es el subconjunto de agrupaciones autorizadas minimales.
- La tasa de información mide la cantidad de información que tienen los participantes.

CS: Esquema Umbral



- Posee tres fases:
  - División y distribución de las sombras secretas.
  - Reunión de una agrupación autorizada
  - Reconstrucción del secreto
- En todo esquema umbral *(t, w)* (con t≤w):
  - un secreto *k* puede reconstruirse a partir de cualesquiera *t* de las *w* sombras,
  - *k* no puede reconstruirse mediante ningún subconjunto de *(t-1)* o menos sombras.

CS: Esquema Umbral de Shamir

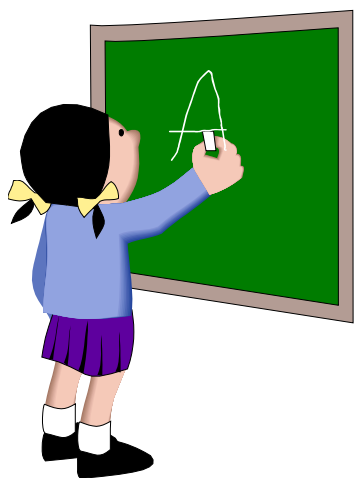


- Se basa en la interpolación polinomial para reconstruir una curva de grado *t-1* a partir de *t* puntos.

- Los polinomios tienen las propiedades de:
  - **Interpolación**: Dados *t* puntos del polinomio (x_i,y_i) con $y_i=f(x_i)$, es posible encontrar los coeficientes de $f(x)$.
  - **Secreto**: Dados cualesquiera *t-1* puntos del polinomio, no es posible deducir nada acerca de los coeficientes de $f(x)$.

CS: Esquema Umbral de Shamir

- Se obtienen las w sombras $y_i = f(x_i)$, $i=1,\ldots,w$, evaluando sobre w valores diferentes $x_1,\ldots,x_w$ un polinomio aleatorio de grado $t-1$, $f(x)=(a_{t-1}x^{t-1}+ \ldots+a_1x^1+a_0) \pmod p$ cuyo coeficiente constante $a_0=k= f(0)$ es el secreto.
- Los valores $x_1,\ldots,x_w$ pueden ser $1,\ldots,w$, o números aleatorios.
- $f(x)$, y por tanto k, pueden reconstruirse mediante cualesquiera t sombras resolviendo un sistema de ecuaciones en el que



$$a_0 = \frac{\begin{vmatrix} f(x_1) & x_1 & \ldots & x_1^{t-1} \\ f(x_2) & x_2 & \ldots & x_2^{t-1} \\ \vdots & \vdots & & \vdots \\ f(x_t) & x_t & \ldots & x_t^{t-1} \end{vmatrix}}{\begin{vmatrix} 1 & x_1 & \ldots & x_1^{t-1} \\ 1 & x_2 & \ldots & x_2^{t-1} \\ \vdots & \vdots & & \vdots \\ 1 & x_t & \ldots & x_t^{t-1} \end{vmatrix}}$$

CS: Esquema Umbral de Shamir

Ejemplo: $t=3, w=5, p=17, k=13, f(x)=(2x^2+10x+13) \pmod{17}$.

Evaluando $f(x)$ en $x=1,\ldots,5$ se obtienen las sombras:

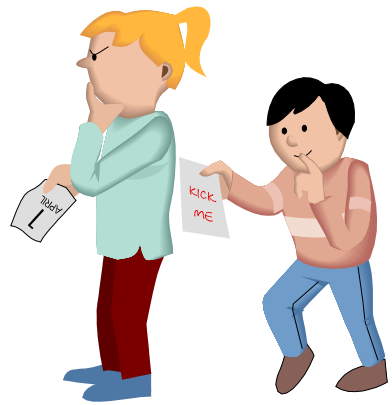
$y_1 = f(1) = (2 + 10 + 13) = 8 \pmod{17}$

$y_2 = f(2) = (8 + 20 + 13) = 7 \pmod{17}$

$y_3 = f(3) = (18 + 30 + 13) = 10 \pmod{17}$

$y_4 = f(4) = (32 + 40 + 13) = 0 \pmod{17}$

$y_5 = f(5) = (50 + 50 + 13) = 11 \pmod{17}$



$f(x)$ puede reconstruirse mediante cualesquiera 3 sombras. Por ejemplo, usando y_1, y_3 e y_5 se obtiene:

$$a_0 = \frac{\begin{vmatrix} 8 & 1 & 1 \\ 10 & 3 & 9 \\ 11 & 5 & 8 \end{vmatrix}}{\begin{vmatrix} 1 & 1 & 1 \\ 1 & 3 & 9 \\ 1 & 5 & 8 \end{vmatrix}} = 132 \cdot 18^{-1} = 132 = 13 \pmod{17}$$

CS: Esquema Umbral de Shamir

Propiedades:

- El tamaño de las sombras es similar al del secreto
- Es posible añadir o eliminar sombras
- Pueden cambiarse las sombras sin más que usar un nuevo polinomio con el mismo coeficiente constante
- Puede ser atacado por el repartidor o por tramposos independientes o coordinados
- Se puede generalizar a t secretos

CS: Esquema Umbral de las Sombras Congruentes

Dado un sistema de w ecuaciones $\boldsymbol{k=a_i\ (mod\ m_i)}, i=1,\ldots,w,$ cuya incógnita *k es el secreto,* si cada par de módulos m_i son primos entre sí, y el mayor producto de *t-1 módulos* m_i $<k<$ el menor producto de *t-1 módulos* m_i, entonces existe una solución

$$k = \sum_{t\ sumandos} a_i M_i N_i \ (\mathrm{mod}\ M)$$

con $N_i=M_i^{-1}(mod\ m_i)$ $M_i=M \bullet m_i^{-1}$ $M = \prod_{t\ multiplicandos} m_i$

CS: Esquema Umbral de las Sombras Congruentes

Ejemplo: k=123456, w=5, t=3,

m_i={82,83,85,87,89}, a_i={46,35,36,3,13} tales que

- 123456= 46(mod 82), 123456= 35(mod 83), 123456= 36(mod 85), 123456= 3(mod 87), 123456= 13(mod 89),
- cada par de módulos son primos entre sí y
- mayor producto de 2 m_i =7743<k=123456<menor producto de 3 m_i=578510,

Con 3 sombras cualesquiera, p.ej. 2ª, 3ª y 4ª

M_2= m_3m_4=7395, M_3= m_2m_4=7221 y M_4= m_2m_3=7055 y sus inversos (mod m_i) N_2=52, N_3=21 y N_4=11, de donde

$$k=35\cdot7395\cdot52+36\cdot7221\cdot21+3\cdot7055\cdot11=$$

$$=123456 \pmod{613785=m_2m_3m_4}$$

CS: Esquema Umbral de las Sombras Congruentes

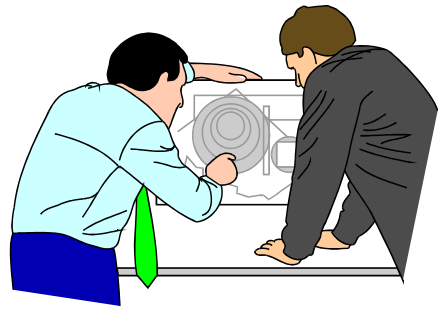
Tanto tramposos individuales como coordinados pueden modificar sus sombras para reconstruir un secreto falso, pero nunca pueden elegir el secreto falso concreto a reconstruir

CS: Ataques

- Un repartidor deshonesto puede dar sombras no válidas de forma que al reunirlas no definen unívocamente un secreto
- Participantes tramposos pueden no proporcionar sus sombras cuando son requeridas, o bien dar falsas sombras
- Grupos coordinados de tramposos pueden engañar al resto modificando sus participaciones de forma que se recupere un secreto falso

CS Verificable

CS en que cada participante puede verificar la validez de su sombra y de las sombras que reciba de los otros



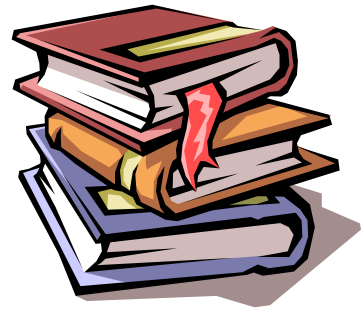
CSV Basado en Shamir y CB

- Fase de distribución: El repartidor que tiene el secreto k, calcula sombras $s_1,s_2,...,s_w$ de s según el esquema de Shamir, obtiene compromisos de s_i, $i=1,2,...,w$, publica los testigos C_i, y le da a cada participante i su sombra s_i y los parámetros para la apertura del compromiso C_i
- Fase de reconstrucción: Como en el esquema de Shamir, pero cada participante i no sólo entrega su sombra s_i sino también la información de apertura del compromiso C_i para la verificación de la sombra aportada

CSV: Esquema de Ben-Or, Goldwasser y Wigderson

1. El repartidor escoge un polinomio de grado *t-1* en dos variables $p(x,y)$ tal que el secreto $k=p(0,0)$
2. Cada participante *i* recibe dos polinomios de grado *t-1* en una variable $f_i(y)=p(i,y)$ y $g_i(x)=p(x,i)$
3. Cada participante *i* comprueba que $f_i(i)=g_i(i)$, y si no se cumple, sospecha del repartidor
4. Cualquier par de participantes *i* y *j* pueden generar conjuntamente la misma sombra secundaria $f_i(j)=g_j(i)=s_{ij}$
5. Si el participante *i* descubre que $f_i(j)\neq g_j(i)$
   - si hay más de *t-1* participantes *j*, sospecha del repartidor
   - si hay menos de *t* participantes *j*, sospecha de éstos

CSV: Esquema de Ben-Or, Goldwasser y Wigderson



Ejemplo: $k=1$, t=2, w=3,

El repartidor escoge p(x,y)=1+2x+3y y reparte

$f_1(y)=p(1,y)=3+3y$, $g_1(x)=p(1,i)=4+2x$, t.q. $f_1(1)=g_1(1)=6$

$f_2(y)=p(2,y)=5+3y$, $g_2(x)=p(2,i)=7+2x$, t.q. $f_2(2)=g_2(2)=11$

$f_3(y)=p(3,y)=7+3y$, $g_3(x)=p(3,i)=10+2x$, t.q. $f_3(3)=g_3(3)=16$

Si 1º y 2º participantes se intercambian sombras:

- comprueban $f_1(2)=g_2(1)=s_{12}=9$
- reconstruyen el secreto $k=1$ a partir de {k+a=3, k+2a=5} ó de {k+b=4, k+2b=7}

CS Jerarquizada

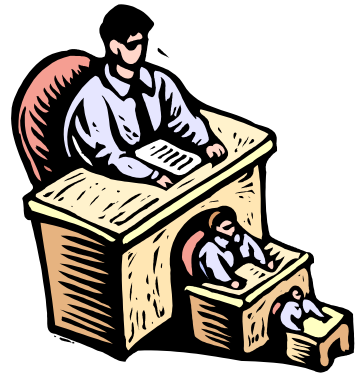
CS con diferentes clases de sombras de forma que la cantidad mínima necesaria para reconstruir el secreto es distinta según la clase

- Permite la existencia de un participante principal tal que sin su presencia sea imposible recuperar el secreto, pero que a la vez requiere de la colaboración de un número mínimo de otros participantes para llevar a cabo la reconstrucción

CS Jerarquizada

- Ejemplo: Presidente, Secretario y 2 Vocales
  - Estructura de acceso={(P,S),(P,V_1,V_2)}
  - Secreto=100
  - Sombras: de P=111, de S=011, de V_1=010, de V_2=001
  - 111+011=100,
  - 111+010+001=100
- Se puede usar un esquema umbral haciendo que el secreto k sea función de secretos parciales de diferentes clases n_i, $k=f(n_1,...,n_s)$, de forma que cada secreto parcial n_i se pueda reconstruir mediante un esquema umbral (t_i,w_i), con $t=t_1+...+t_s$ y $w=w_1+...+w_s$

CS Jerarquizada



Ejemplo: k=81, w=10= 4 Vicerrectores+6 Directores,

Estructura de acceso=2V+3D. El rector escoge

(s_v,s_d)=(26,38), (p_v,p_d)=(31,43), t.q. $k=s^2_v s^2_d (\text{mod } p_v p_d)$

$P_v(x)=26+29x(mod\ 31)$, $P_d(x)=38+16x+24x^2(mod\ 43)$

$(x_{v1},x_{v2},x_{v3},x_{v4})=(10,25,12,27)$

$(x_{d1},x_{d2},x_{d3},x_{d4},x_{d5},x_{d6})=(40,14,32,26,30,23)$ *y reparte*

$(x_{vi},P_v(x_{vi}))=\{(10,6),(25,7),(12,2),(27,3)\}$ *y*

$(x_{di},P_d(x_{di}))=\{(40,34),(14,21),(32,14),(26,37),(30,16),(23,30)\}$

Si se reunen $V_3+V_4+D_1+D_5+D_6$ reconstruyen el secreto con

$\{s_v+12x=2, s_v+27x=3\}$(mod 31) y $\{s_d+40y+40^2z=34,$

$s_d+40y+40^2z=34, s_d+40y+40^2z=34\}$(mod 43), de donde

(s_v,s_d)=(26,38), y $k=s^2_v s^2_d=81$(mod 1333)

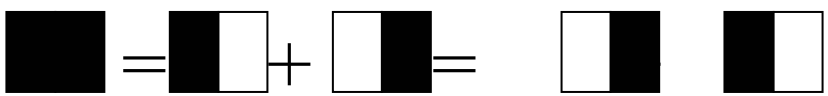
CS: Descubrimiento Parcial de Secretos

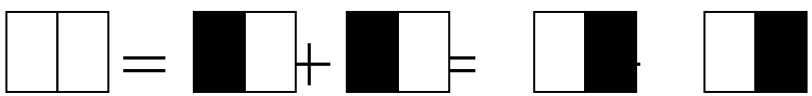
- Los participantes sólo descubren partes de sus sombras.
- Todos conocen la definición de una función *f*, que les permite utilizar su propia sombra e información parcial de las sombras de los demás para recuperar el secreto compartido.

CS: Criptografía Visual

CS usado con el propósito de cifrar una imagen para que pueda ser descifrada (reconstruida) mediante la superposición de un número mínimo de sombras (transparencias)

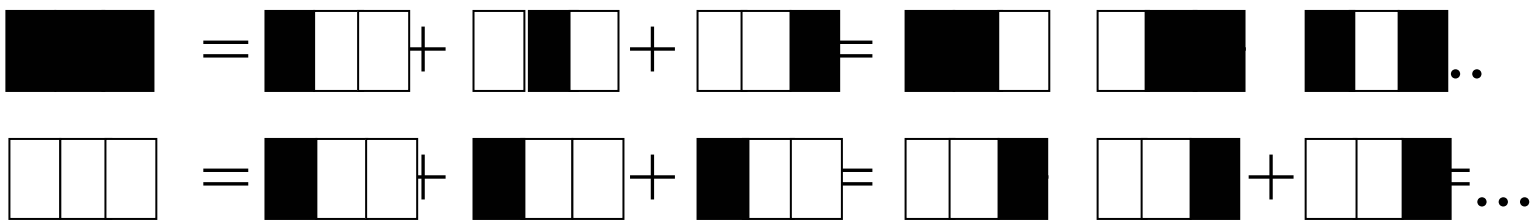
Esquema umbral visual (2,2)



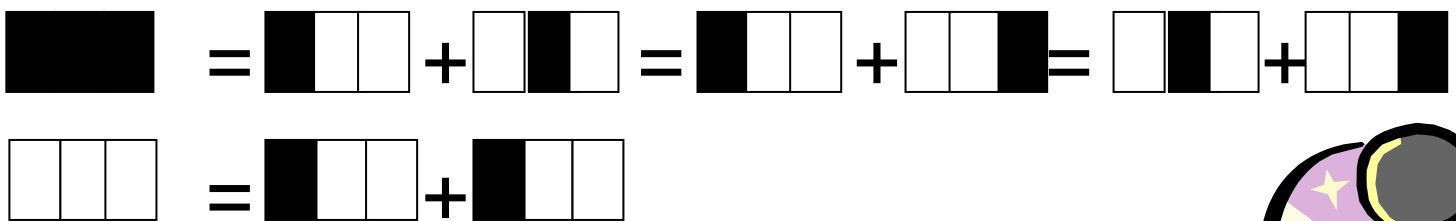


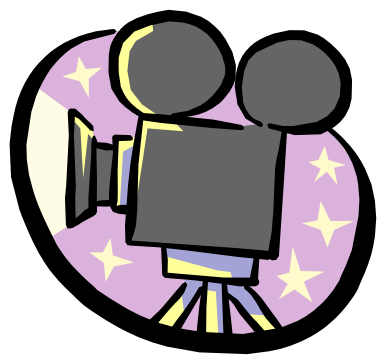
CS: Criptografía Visual

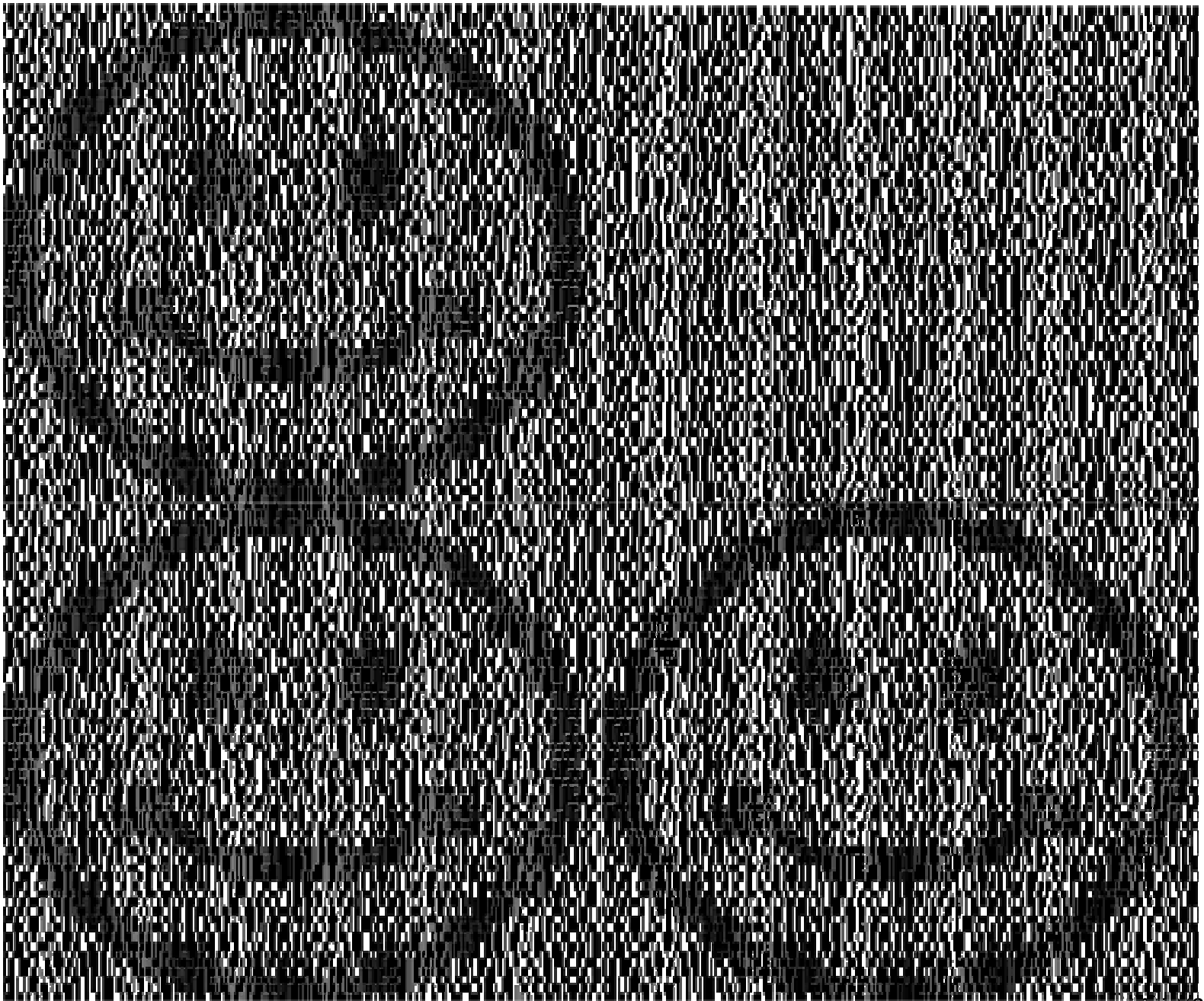
Esquema umbral visual (2,3)



Si se reúnen







Póquer Mental

Varios participantes juegan al póquer sin estar presentes físicamente, y con la garantía de que nadie puede hacer trampas

1. Todas las combinaciones de 5 cartas tienen la misma probabilidad
2. Todas las cartas escogidas por los distintos jugadores son diferentes
3. Cada jugador sólo conoce sus propias cartas
4. Cada jugador puede descubrir si otro hace trampas
5. Es seguro contra posibles coaliciones de tramposos
6. Permite cualquier número de jugadores
7. Utiliza técnicas criptográficas económicas y seguras
8. Requiere la participación de una TPC

PM: Basado en Clave Pública

1. Uno de los jugadores A cifra todas las cartas con su clave pública E_A.
2. Cada jugador escoge 5 cartas cifradas, las cifra con su clave pública y se las envía a A para que se las descifre con su clave privada D_A, y envía al siguiente jugador las cartas restantes

- Una vez la partida ha acabado, los jugadores publican sus claves privadas para realizar las verificaciones
- Las claves usadas deben determinar unívocamente los mensajes originales y cifrados
- El cifrado debe evitar posibles pérdidas de información

Distribución Anónima de Mensajes

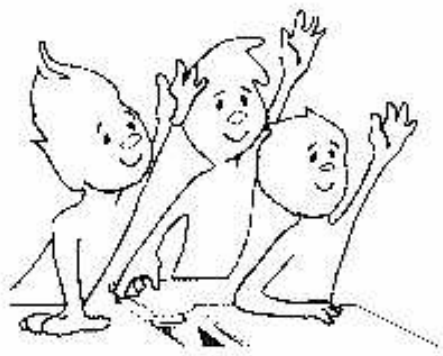
Para difundir de forma anónima un mensaje entre varios participantes

Algoritmo de la Cena de Criptógrafos

1. Los participantes se colocan en círculo.
2. Cada participante en secreto lanza una moneda y dice el resultado a su vecino derecho.
3. Cada participante dice si coinciden o no los dos lanzamientos que conoce.
4. El usuario que desea distribuir el mensaje (bit) dice la verdad o miente dependiendo del valor del bit.
5. Si el número total de diferencias es impar, entonces el bit difundido es un 1. Si es par, el bit es un 0.

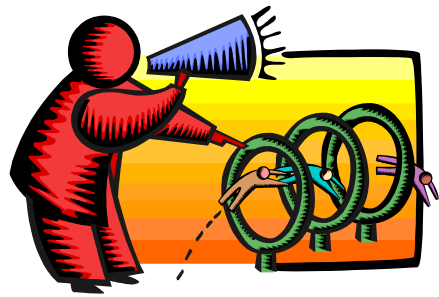
Elecciones Electrónicas

Los participantes (votantes) aplican una función (suma) sobre sus entradas binarias secretas (votos) y obtienen el resultado de la elección sin poner en peligro el secreto de sus votos



- Cada participante j vota con un bit de entrada v_j y la salida para cada uno es $f_i(v_1,v_2,...,v_k)=\sum v_j$, i=1,2,...,k
- *Chaum, D., Elections with unnconditionally secret ballots and disruption equivalent to breaking RSA, E'88*
- Suele haber 5 fases: registro, validación, recolección, recuento y verificación

EE: Propiedades



– Democracia:
  1. Debe impedir que votantes no censados puedan votar
  2. Debe impedir que nadie pueda votar más de una vez
– Privacidad:
  3. Debe mantener el secreto de los votos
  4. Debe impedir que nadie pueda duplicar el voto de otro usuario
  5. Debe imposibilitar el forzar a ningún votante a revelar su voto
– Exactitud:
  6. Debe asegurar que el recuento se hace correctamente
– Verificabilidad:
  7. Debe permitir comprobar que el voto ha sido contado en el escrutinio
– Tolerancia a fallos:
  8. Debe ser tolerante a fallos, es decir, debe funcionar correctamente incluso en presencia de varios participantes deshonestos

EE: Con 2 Mesas

- L: Comprueba la legitimidad de los votantes y recopila sus votos, elaborando el conjunto N con todos los números de identificación de quienes han votado, que envía a C
- C: Calcula y publica el resultado del escrutinio de los votos del conjunto N

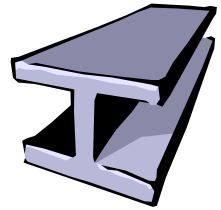
1. A envía a L un mensaje identificándose
2. Si A está censado, L le envía su número de identificación i(A) y borra a A de la lista de votantes censados. Si A no está censado, L lo rechaza.
3. A escoge una identificación secreta s(A) y envía a C la 3-upla (i(A), v(A), s(A)) donde v(A) es su voto.
4. La mesa C comprueba si i(A) está en el conjunto N, y en ese caso lo borra y añade s(A) al conjunto de votantes que han votado por v(A).
5. Cuando las elecciones acaban, C realiza el escrutinio y calcula y publica los resultados junto con la lista de las identificaciones secretas de quienes han votado por cada opción.

EE:Firma a Ciegas

Firmar sin ver lo que se firma

- Chaum,D., *Blind signatures for untraceable payments*, Crypto82
- Util en elecciones electrónicas y dinero digital
- Ejemplo: A genera n documentos válidos y envía a B cada uno en un sobre de papel carbón (blindados):
  1. B abre n-1 sobres al azar y verifica su contenido.
  2. Si los contenidos de los sobres abiertos son correctos, B firma por fuera el sobre restante y se lo envía a A.
  3. A abre el sobre y comprueba la firma
     - B no puede hacer trampas, porque si su firma es errónea o bien abre el último sobre, A lo detecta
     - La probabilidad de fraude de A es menor que n^{-1}, luego disminuye a medida que n crece

EE: Con Firma a Ciegas



1. Registro: Cada votante se identifica ante la Mesa y le envía su voto conteniendo un Número Aleatorio de Identificación (NAI), blindado
2. Validación: La Mesa comprueba la legitimidad del votante, le elimina de la lista de registrados y firma a ciegas el voto
3. Recolección: El votante desblinda su voto y envía de forma anónima el resultado a la Mesa cifrado con la clave pública de ésta
4. Recuento: La Mesa descifra, valida su firma a ciegas, comprueba la no duplicidad de NAI, y al final publica todos los votos
5. Verificación: El votante verifica mediante el NAI que su voto está entre los publicados

EE: Problemas

- El Problema de la Coerción:
  1. Antes de que empiecen las elecciones, forzar un voto particular
  2. Después de que terminen las elecciones, obligar a revelar el voto emitido
  - Una solución: Permitir que cada votante pueda intercambiar un número limitado de bits secretos a través de un canal seguro con la mesa.
- El Derecho al Veto:
  - Nadie debe saber si el resultado se debe o no al uso del derecho al veto.

EE: Protocolo de Chaum

- Los votos quedan asociados a pseudónimos digitales (NAI)
- Utiliza firmas a ciegas y un canal de comunicaciones anónimas
- Usa cifrados de claves múltiples (t.q. $E_{r'}(m')=E_r(m)$)
- No es práctico en elecciones a gran escala porque el fallo de un único votante puede interrumpir todo el proceso, aunque se descubre al culpable.

EE: Protocolo de Merritt

1. Cada mesa i=1,2,...,n genera sus claves privada y pública (E_i,D_i)
2. Cada votante j cifra su voto v_j y su NAI s_j, mediante $E_1(E_2(...E_n(v_j,s_j)...))=y_{n+1,j}$ y lo publica
3. Desde i=n hasta 1 cada mesa C_i para cada $y_{i+1,j}$ genera un número aleatorio $r_{i,j}$ y publica $y_{i,\Pi i(j)}=E_i(y_{i+1,j}, r_{i,j})$, siendo $\Pi_i(j)$ una permutación secreta, de manera que al final se publican $y_{1,j}=E_1(E_2(...E_n(y_{n+1,j},r_{n,j})...r_{2,j})r_{1,j})$
4. Recolección: Las mesas realizan dos iteraciones de descifrados desde i=1 hasta n sobre cada $y_{1,j}$, y publican (v_j,s_j)
5. Recuento: Se obtiene mediante la suma de los votos v_j
6. Verificación: Cada votante comprueba su NAI s_j

EE: Protocolo de Pfleeger

- Organizado sólo por los votantes (sin mesas)
- Basado en un cifrado de clave pública
- Con gran coste computacional y de comunicaciones
- Hace falta que todos los votantes estén presentes a la vez
- Si uno de los votantes no puede continuar, el proceso se interrumpe
- No es válido para elecciones a gran escala

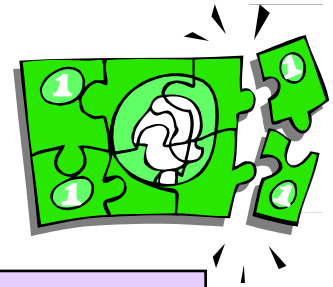
EE: Protocolo de Boyd

- Usa una Mesa TPC para registrar a los votantes
- Usa cifrados de clave múltiple
- El gobierno puede ver los votos, e incluso producir un falso recuento añadiendo votos de su elección

EE: Protocolo de Fujioka

- Es útil para elecciones a gran escala porque no requiere mucho cálculo ni muchas comunicaciones
- Ningún votante puede abstenerse porque en ese caso podrían conspirar las dos mesas que participan para añadir votos falsos
- Utiliza CB y firmas digitales de cada votante, un canal de comunicaciones anónimas entre la Mesa C y los votantes, y la Mesa L usa firmas a ciegas
- Hay 3 etapas: Registro, Votación y Apertura

Dinero Digital



Sistema de pago electrónico anónimo

– *Chaum, D., Untraceable electronic mail, return addresses and digital pseudonyms. Communications of the ACM 1981*

- Imitación electrónica del dinero en efectivo, y alternativa frente a otros métodos de pago como cheques digitales, tarjetas de crédito y de débito, monederos electrónicos
- Debe proporcionar privacidad, autenticidad, no repudio y **anonimato**
- Debe ser portátil, reconocible, aceptable, transferible, anónimo, y divisible, no debe dejar rastro, y debe permitir gastos sucesivos.

DD: Propiedades

- Hay tres clases de participantes:
  1. El pagador (A)
  2. El cobrador (B)
  3. La entidad financiera (el banco)
- Tipos de esquemas:
  1. **Protocolo on-line**: Cada transacción entre A y B se realiza a través del banco
  2. **Protocolo off-line**: B envía las monedas de A al banco para su verificación después de que la transacción se ha completado
- Subrutinas dentro del esquema:
  1. **Retirada** del dinero digital del banco
  2. **Pago** de bienes a un vendedor mediante dinero digital
  3. **Depósito** por parte del vendedor en su cuenta bancaria del dinero digital que ha cobrado

DD: Esquema General Basado en Clave Pública

Retirada:

1. A genera un número aleatorio grande N (número de serie del dinero)
2. A firma el número N junto al valor monetario que quiere gastar y envía el resultado a su banco
3. El banco verifica la firma de A
4. El banco comprueba que A tiene en su cuenta la cantidad solicitada, la sustrae como débito, firma el documento y se lo devuelve a A (dinero digital)

DD: Esquema General Basado en Clave Pública

Pago:

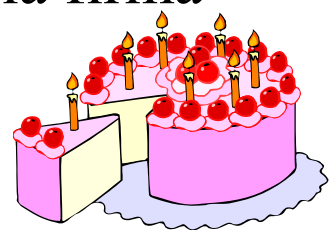
5. A envía el dinero digital a B
6. B verifica la firma del banco
7. B envía el dinero a su banco

Depósito:

8. El banco de B re-verifica la firma del banco de A en el dinero recibido
9. El banco de B comprueba que N no ha sido ya usado
10. El banco de B añade la cantidad de dinero a la cuenta de B

DD: Esquema de Chaum

- Fue el primero (1988)
- Protege el anonimato gracias a que cada usuario usa un Número Aleatorio de Identificación (NAI) y a que la firma del banco es a ciegas
- Detecta dobles gastos a posteriori
- Usa corte y elección en la retirada y reto-respuesta en el pago
- Se basa en RSA y en funciones unidireccionales libres de colisiones, como MD4 y MD5
- La moneda contiene el doble de la información que realmente se usa



DD: Esquema de Chaum

Retirada:

1. A crea una moneda electrónica que incluye su NAI
2. A blinda la moneda y la envía al banco
3. El banco firma a ciegas la moneda y se la devuelve a A
4. A desblinda la moneda

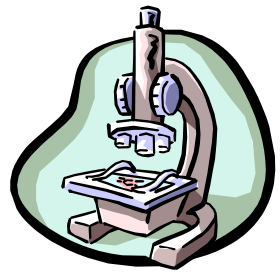
Pago:

5. A envía el dinero digital a B
6. B verifica la firma del banco y envía un reto a A, quien envía una respuesta a B (revelando información sobre su NAI)
7. B verifica la respuesta y envía el dinero a su banco junto con el reto y la respuesta

Depósito:

8. El banco de B re-verifica la firma del banco de A
9. El banco de B comprueba que el dinero no ha sido ya usado
10. El banco de B añade la cantidad de dinero a la cuenta de B

DD: Esquema de Brands



- Es el más eficiente.
- No usa RSA
- Detecta dobles gastos a priori
- Existe la figura de un “observador” que autoriza cada transacción
- Incluye un procedimiento de identificación y una firma digital
- Se basa en una DCN de la posesión de una clave secreta
- Utiliza exponenciación modular y el problema del logaritmo discreto
- El esquema divisible más eficiente, de Okamoto y Ohta, se basa en el esquema de Brands

DD: Actualidad

- Actualmente no hay estándares reconocidos
- Varias empresas privadas proporcionan el servicio: digicash, cybercash (de verisign), netcash, checkfree, open market, netscape, ecashtechnologies, millicent,...
- Digicash usa el esquema de Chaum
- Cybercash usa cifrado RSA y no preserva el anonimato
- Netcash opera por e-mail con monedas que son palabras de 15 caracteres, y tampoco preserva el anonimato

DD: La Discusión

- Con anonimato el dinero digital permite el "Crimen Perfecto"
- Sin anonimato el dinero digital permite el "Gran Hermano"