

Pruebas de conocimiento nulo

Armando Faz Hernandez

armfazh@cloudflare.com

27 de septiembre de 2025

CatioCrypto 2025



Esquemas de comprometimiento

Pruebas

Sistemas de prueba interactivos

Pruebas de conocimiento

La propiedad de conocimiento-nulo

Esquemas de comprometimiento

Esquemas de comprometimiento

1. Alicia escribe un valor en un papel y lo pone en una caja cerrada con candado.
2. Alicia le da la caja a Beto.
3. Después, si Alicia lo desea puede abrir la caja enviando la llave del candado a Beto.

Propiedades:

Ocultante Cuando Beto recibe la caja no puede saber el valor hasta que reciba la llave.

Vinculante Cuando se abra la caja no se puede mostrar un valor diferente al almacenado.

Un esquema de comprometimiento no interactivo define los algoritmos:

- **Inicializa**(1^λ). Retorna parámetros del sistema para un nivel de seguridad λ .
- **Compromete**(m, r) $\rightarrow (c, d)$. Toma el mensaje m al que se va a comprometer y una entrada aleatoria r , retorna un compromiso c y un valor de apertura d .
- **Verifica**(c, m, d) $\rightarrow \{0, 1\}$. Retorna un bit indicando si verificación es exitosa.

Esquemas de comprometimiento



Compromete $(m, r) \rightarrow (c, d)$

c \longrightarrow c

m, d \longrightarrow $\text{Verifica}(c, m, d) \rightarrow \{0, 1\}$

Esquemas de comprometimiento



$\text{Compromete}(m, r) \rightarrow (c, d)$

$\xrightarrow{c} c$

$\xrightarrow{m, d}$
 $\text{Verifica}(c, m, d) \rightarrow \{0, 1\}$

Ocultante El compromiso c no revela información sobre el mensaje m .


Vinculante Ningún adversario puede generar c , $m \neq m'$ y d, d' tal que ambos acepten

Verifica(c, m, d)

Verifica(c, m', d')

¿Cómo lanzar una moneda por teléfono?

Alicia y Beto deciden otorgar un premio a aquel que gane un volado, pero no confían en el otro.

¿Qué es un volado? 

Juego de azar en el que se emplea una moneda, gana quien acierte qué lado de la moneda (de los dos posibles) caerá cara arriba.

¿Cómo lanzar una moneda por teléfono?

Alicia y Beto deciden otorgar un premio a aquel que gane un volado, pero no confían en el otro.

Protocolo para el volado:

1. Alicia se compromete al valor de un bit aleatorio b_A .
2. Alicia envía su compromiso $C = \mathbf{Compromete}(b_A)$ a Beto.
3. Beto escoge un bit b_B aleatoriamente y lo envía a Alicia.
4. Alicia abre C para que Beto conozca b_A .
5. Ambos calculan $b = b_A \oplus b_B$.

Comprometer un bit con RSA

Alicia tiene un par de llaves RSA: pública (e, N) y privada (d) .

Alicia se compromete a un bit b :

1. Alicia escoge un número $x_b \leftarrow \{0, \dots, N-1\}$ aleatoriamente tal que su bit menos significativo es b .
2. Alicia envía $C = x_b^e \bmod N$ a Beto.

Beto no puede saber x_b a menos que pueda romper RSA.

¿Es ocultante? y ¿es vinculante?

Tipos de compromisos:

1. Incondicional vinculante y computacional ocultante.
2. Computacional vinculante e incondicional ocultante.

Incondicional Aun con poder de cómputo infinito no puede violar la propiedad.

Computacional Un atacante limitado polinomialmente le es difícil violar la propiedad.

¿Podrían ser ambas propiedades incondicionales?

Comprometer un bit con QR_N

Alicia escoge aleatoriamente un módulo RSA $N = pq$.

Envía (N, x) a Beto tal que

x es escogido aleatoriamente de $\begin{cases} QR_N & \text{si } b = 0, \\ \mathbb{Z} \setminus QR_N & \text{si } b = 1, \end{cases}$

Alicia abre el compromiso enviando (p, q) .

Beto verifica que:

- $N = pq$
- Si x es QR_N , retorna $b = 0$.
- Si x no es QR_N , retorna $b = 1$.

Sea $G = \langle g \rangle$ un grupo cíclico de orden q y sean $g, h \in G$ dos generadores.

Compromiso de Pedersen para **exponentes** $m \in \mathbb{Z}/q\mathbb{Z}$.

- Alicia escoge $r \leftarrow \mathbb{Z}/q\mathbb{Z}$ aleatoriamente y calcula

$$c = g^m h^r$$

- Alicia abre el compromiso enviando (m, r) a Beto.
- Si Beto verifica que

$$c = g^m h^r$$

entonces retorna m , sino retorna un error.

Sea $G = \langle g \rangle$ un grupo cíclico de orden q y sean $g, h \in G$ dos generadores.

Compromiso de ElGamal para elementos $m \in G$.

- Alicia escoge $r \leftarrow \mathbb{Z}/q\mathbb{Z}$ aleatoriamente y calcula

$$c_0 = g^r, c_1 = mh^r$$

- Alicia abre el compromiso enviando (m, r) a Beto.
- Si Beto verifica que

$$c_0 = g^r, c_1 = mh^r$$

entonces retorna m , sino retorna un error.

Pruebas

Se desea brindar acceso a un recurso sólo a ciertos usuarios.

Los usuarios se pueden identificar con una contraseña.

Un usuario envía su contraseña al servidor para verificar su validez.

Problema:

- Cualquiera escuchando el medio puede saber la contraseña.

¿Cómo remediar la situación?

El servidor sólo necesita saber si el usuario sabe su contraseña.

Es decir debe saber sólo un bit **y nada más**.

Queremos diseñar un protocolo donde al final de la interacción entre el usuario y el servidor:

- el servidor obtiene un bit que indica si el usuario pudo identificarse exitosamente.

Un protocolo es de **conocimiento nulo** si comunica exactamente el conocimiento previsto y ningún (cero) conocimiento adicional.

Perla (probador) quiere convencer a Víctor sobre la verdad de alguna declaración.

Víctor (verificador) está interesado en verificar que la declaración sea verdadera.

Construyamos un protocolo de conocimiento nulo

Usaremos un criptosistema de llave pública.

El usuario A posee una llave pública P_A y una llave privada k_A .

Declaración: Perla desea probar que ella es el usuario A .

1. Víctor escoge un mensaje M aleatoriamente y envía $C = E(M, P_A)$ a Perla.
2. Perla descifra C usando k_A y envía el resultado M' a Víctor.
3. Víctor acepta la declaración si $M' = M$.

¿Se encuentran tanto Perla como Víctor convencidos?

Un atacante actúa como el verificador y envía un C' al probador.

El probador descifrá C' pues sigue el protocolo.

Observa que:

- El atacante abusa del protocolo haciendo que el probador **descifre** arbitrariamente.
- El probador no tiene defensa ante esta situación.

Fallamos: el protocolo no es de conocimiento nulo.

Un atacante actúa como el verificador y envía un C' al probador.

El probador descifrá C' pues sigue el protocolo.

Observa que:

- El atacante abusa del protocolo haciendo que el probador **descifre** arbitrariamente.
- El probador no tiene defensa ante esta situación.

Fallamos: el protocolo no es de conocimiento nulo.

Un atacante actúa como el verificador y envía un C' al probador.

El probador descifrá C' pues sigue el protocolo.

Observa que:

- El atacante abusa del protocolo haciendo que el probador **descifre** arbitrariamente.
- El probador no tiene defensa ante esta situación.

Fallamos: el protocolo no es de conocimiento nulo.

Cuando el verificador envía C no existen garantías de que también conozca el mensaje M .

¿Y que pasaría si envía C y M ?

No funciona porque el propósito es que el probador encuentre M .

Resumiendo:

- si el verificador envía M se pierde el propósito del protocolo;
- sino lo envía, entonces el protocolo no es de conocimiento nulo.

Habría que enviar M *encerrado* de alguna forma.

Idea ¡Usemos las cajas con candado!

El probador envía un compromiso al valor de M al verificador.

Construyamos un protocolo de conocimiento nulo (Parte II)

El usuario A posee una llave pública P_A y una llave privada k_A .

Declaración: Perla desea probar que ella es el usuario A .

1. Víctor escoge un mensaje M aleatoriamente y envía $C = E(M, P_A)$ a Perla.
2. Perla descifra C usando k_A y envía un compromiso c de M' a Víctor.
3. Víctor envía M a Perla.
4. Perla revisa si $M = M'$ y retorna error sino se cumple.
Cuando si se cumple, Perla abre el compromiso enviando (r, M') a Víctor.
5. Víctor acepta la declaración si $M' = M$ y si el compromiso pasa la verificación.

Sistemas de prueba interactivos

Ahora los protocolos se llevan a cabo como interacciones entre dos máquinas interactivas de Turing.

El probador (P) tiene poder computacional infinito.

El verificador (V) tiene un tiempo de cómputo limitado polinomialmente.

El par (P, V) recibe una entrada x y al final de la iteración el verificador acepta o rechaza x .

Existe un lenguaje $L \subset \{0, 1\}^*$.

El par (P, V) es un sistema de prueba interactivo para L si satisface:

Complejitud Si $x \in L$, entonces la probabilidad de que (P, V) rechaze x es desdeñable en la longitud de x .

Solidez Si $x \notin L$, entonces para cualquier probador P^* , la probabilidad de que (P^*, V) acepte x es desdeñable en la longitud de x .

La interacción y las probabilidades de falla hacen a la clase de complejidad IP más interesante que NP.

Un **argumento interactivo** para un lenguaje L es similar a un sistema de prueba interactivo, excepto que:

- Probador tiene tiempo de cómputo limitado polinomialmente y es probabilístico.
- En la defición de completitud: se requiere que para cada $x \in L$ exista una entrada auxiliar que permita al probador convecer al verificador.
- En la defición de solidez: se cambiar por cualquier probador que tenga tiempo de cómputo limitado polinomialmente y que sea probabilístico.

Pruebas de conocimiento

En una prueba es una **prueba de conocimiento** si el probador afirma saber una pieza de información.

Las propiedades se renombran como: Completitud de conocimiento y solidez de conocimiento.

Algoritmo Extractor

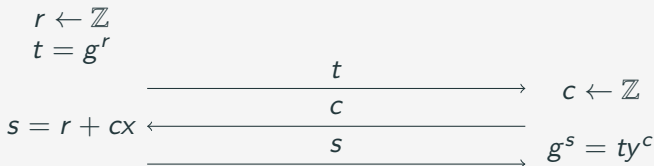
El extractor interactúa con un probador (malicioso) para extraer el valor que dice saber.

Reconstruye el valor mediante varias repeticiones del protocolo.

Prueba de conocimiento de Schnorr

Declaración: Sé el logaritmo discreto $x = \log_g(y)$, i.e., $y = g^x$.

Prueba de conocimiento de Schnorr



Extractor para la prueba de conocimiento de Schnorr

1. Ejecuta el probador para que calcule $t = g^r$. Sea $(*)$ el estado del probador en este punto.
2. Genera $c_1 \in \mathbb{Z}$ aleatoriamente, dáselo al probador y que produzca

$$s_1 = r + c_1 x$$

3. Rebobina el probador al estado $(*)$.
4. Genera $c_2 \in \mathbb{Z}$ aleatoriamente, dáselo al probador y que produzca

$$s_2 = r + c_2 x$$

5. A partir de s_1 y s_2 , retorna

$$x = \frac{s_1 - s_2}{c_1 - c_2}$$

Prueba de conocimiento de Schnorr

El protocolo de Schnorr es una prueba de conocimiento.

Usando la notación de Camenisch-Stadler:

$$PK\{(x): y = g^x\}$$

Protocolo de tres etapas:

1. Compromiso
2. Desafío
3. Respuesta

El protocolo posee **solidez especial** si es posible calcular el valor a partir de dos ejecuciones válidas con compromisos idénticos.

La propiedad de conocimiento-nulo

Ya sea un sistema de prueba interactivo o un argumento interactivo, puede tener una propiedad extra llamada **conocimiento-nulo**.

Aunque el verificador quiera hacer trampa, no le será posible saber mas allá que la veracidad de la declaración del probador.

Algoritmo Simulador

Toda interacción entre el probador y el verificador puede ser **simulada** cuando $x \in L$ *sin interactuar* con el probador.

La distribución de las conversaciones generadas por el simulador debe ser indistinguible de la distribución de las interacciones reales cuando $x \in L$.

1. El simulador configura la cinta de números aleatorios.
2. El simulador envía t al verificador el cual responde con desafío c .
3. Si el simulador puede responder al desafío, lo hace y regresa al paso 1.
4. Sino, rebobina y calcula $s \leftarrow \mathbb{Z}$ y $t = g^s/y^c$.
5. Reinicia el verificador con la misma cinta y envía este nuevo t .
6. El verificador arrojará el mismo desafío c .
7. Por lo tanto, el simulador puede responder con s .

Este simulador presentado sólo funciona si el verificador se comporta honestamente.

Entonces decimos que este protocolo posee la propiedad de **conocimiento-nulo** ante un **verificador honesto**.

¿Se podría arreglar el simulador para que permita verificador malicioso?

Sí, usando un esquema de comprometimiento para comprometer el desafío c .

Fiat-Shamir muestran un método que dada una prueba

- de conocimiento
- de tres etapas
- con la propiedad conocimiento-nulo con verificador-honesto

lo *transforma* en una prueba

- de conocimiento
- de 1 etapa
- usando una función de resumen (en el modelo de oráculo aleatorio).

Sea $H: \{0, 1\}^* \rightarrow \mathbb{Z}$ una función de resumen.

- Perla calcula:
 1. $r \leftarrow \mathbb{Z}$
 2. $t = g^r$.
 3. $c = H(t)$
 4. $s = r + cx$
- Envía la prueba (t, s) a Víctor
- Víctor acepta la prueba si

$$g^s = ty^c$$

Podemos obtener un esquema de **firma digital** a partir del protocolo de Schnorr.

Alicia desea firmar un mensaje $m \in \{0, 1\}^*$ con su llave privada x .

1. Alicia usa la versión no interactiva del protocolo de Schnorr.
2. Incluye el mensaje m en el cálculo del desafío

$$c = H(t, m)$$

3. Entonces (t, s) es la firma digital de m .

Beto actúa como el verificador y acepta la firma si la prueba verifica correctamente.

La pruebas de conocimiento no-interactivas pueden ser vistas como credenciales públicamente verificables.

Posibles pruebas:

- Prueba de bit: $PK\{(x): x^2 - x = 0\}$
- Prueba de rango: $PK\{(x): a < x\}$
- C es un circuito aritmético: $PK\{(x): C(x) = 0\}$

Tópico actual: ARC [!\[\]\(d3fb9f94af8b26d1c844efa9a98805b0_img.jpg\)](#) y ACT [!\[\]\(78eb1652b591ce460bbb1a853a52e223_img.jpg\)](#) son credenciales usadas para generar N cupones no rastreables.

¡Muchas gracias por su atención!



Fuente [🔗](#)

Disfruten de CatioCrypt, ASCrypto y Latincrypt 2025.

Pruebas de conocimiento nulo

Armando Faz Hernandez

armfazh@cloudflare.com

27 de septiembre de 2025

CatioCrypt 2025

