IIC2523
Sistemas Distribuidos

Hernán F. Valdivieso López
(2025 - 2 / Clase 19)

Cifrado de mensajes
¿Beneficios de aplicar criptografía?

Temas de la clase

1. Criptografía Simétrica
2. Intercambio de Claves: *Diffie-Hellman*
3. Criptografía Asimétrica
4. Funciones *Hash* y Firmas Digitales
5. Protocolos Híbridos

Fuente: OpenAI. (2025). [Definición de Criptografía] [Imagen generada por inteligencia artificial con Sora]. https://openai.com/sora

Cifrado - Introducción

- Proporciona un mecanismo para ocultar los mensajes para garantizar su confidencialidad e integridad.
- Es la base para la autenticación de mensajes.

Conceptos Básicos

- **Texto plano**: El mensaje original, inteligible.
- **Texto cifrado**: El mensaje transformado, ininteligible sin la clave correcta.
- **Llave**: Un parámetro secreto utilizado en el algoritmo de encriptar/desencriptar.
- **Encriptar**: Proceso de transformar texto plano en texto cifrado.
- **Desencriptar**: Proceso inverso, transformar texto cifrado en texto plano.

Cifrado simétrico

Cifrado simétrico

- Se utiliza una misma clave secreta tanto para encriptar como para desencriptar un mensaje.
- También se le llama criptografía de clave secreta.

Cifrado simétrico

- Se utiliza una misma clave secreta tanto para encriptar como para desencriptar un mensaje.
- También se le llama criptografía de clave secreta.

Ventajas

- Es extremadamente rápida y eficiente para encriptar grandes volúmenes de datos.

Cifrado simétrico - *Pipeline*

Nodo A

K_{ak}

Ambas partes se ponen de acuerdo de usar la misma clave

Nodo K

Cifrado simétrico - *Pipeline*

Nodo A

Nodo K

Encriptar(mensaje, K_{ak})

Texto cifrado

Cifrado simétrico - *Pipeline*

Nodo A

Nodo K

Texto cifrado

Desencriptar(mensaje, K_{ak})

Cifrado simétrico - Algoritmos

- **AES (*Advanced Encryption Standard*)**: El estándar de cifrado simétrico actual más ampliamente adoptado, que puede usar claves de 128, 192 o 256 bits.
- **DES (*Data Encryption Standard*)**: Un estándar antiguo de EE. UU. que ahora se considera inseguro debido a su pequeña longitud de clave (56 bits), vulnerable a ataques de fuerza bruta modernos.
  - Existe Triple-DES (*3DES*) que aplica DES tres veces para mayor seguridad... pero que también fue retirado.
- **IDEA (*International Data Encryption Algorithm*)**: Un sucesor de DES, que usa una clave de 128 bits.

Cifrado simétrico

Desafío Principal

- La distribución segura de la clave secreta inicial es un problema fundamental. ¿Cómo se aseguran las dos partes (Nodo A y B) de que un tercero (Nodo 🐸) no intercepte la clave cuando la intercambian por primera vez.

Cifrado simétrico

Desafío Principal

- La distribución segura de la clave secreta inicial es un problema fundamental. ¿Cómo se aseguran las dos partes (Nodo A y K) de que un tercero (Nodo 🐸) no intercepte la clave cuando la intercambian por primera vez.
- **Necesitamos un mecanismo para compartir la llave de forma segura.**

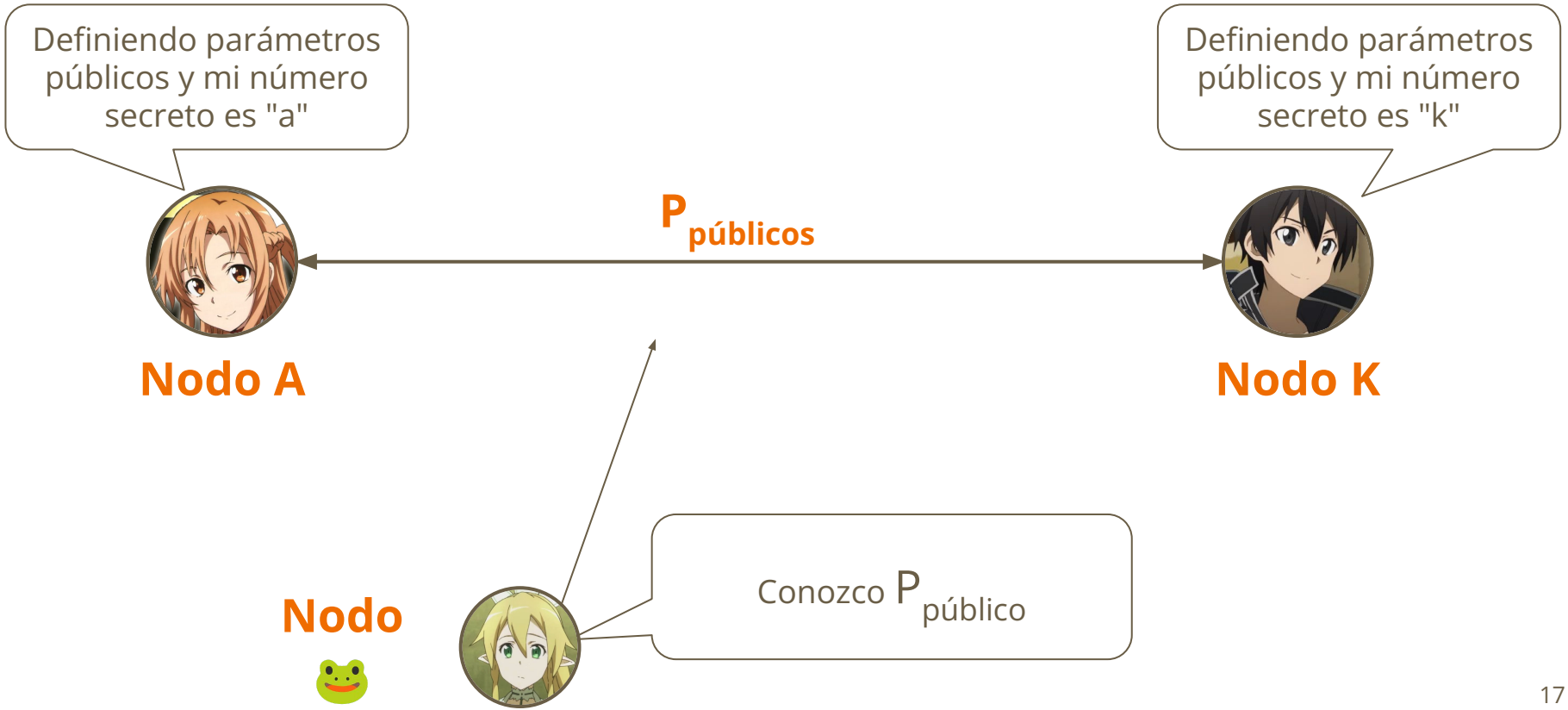
Intercambio de llaves

Intercambio de llaves

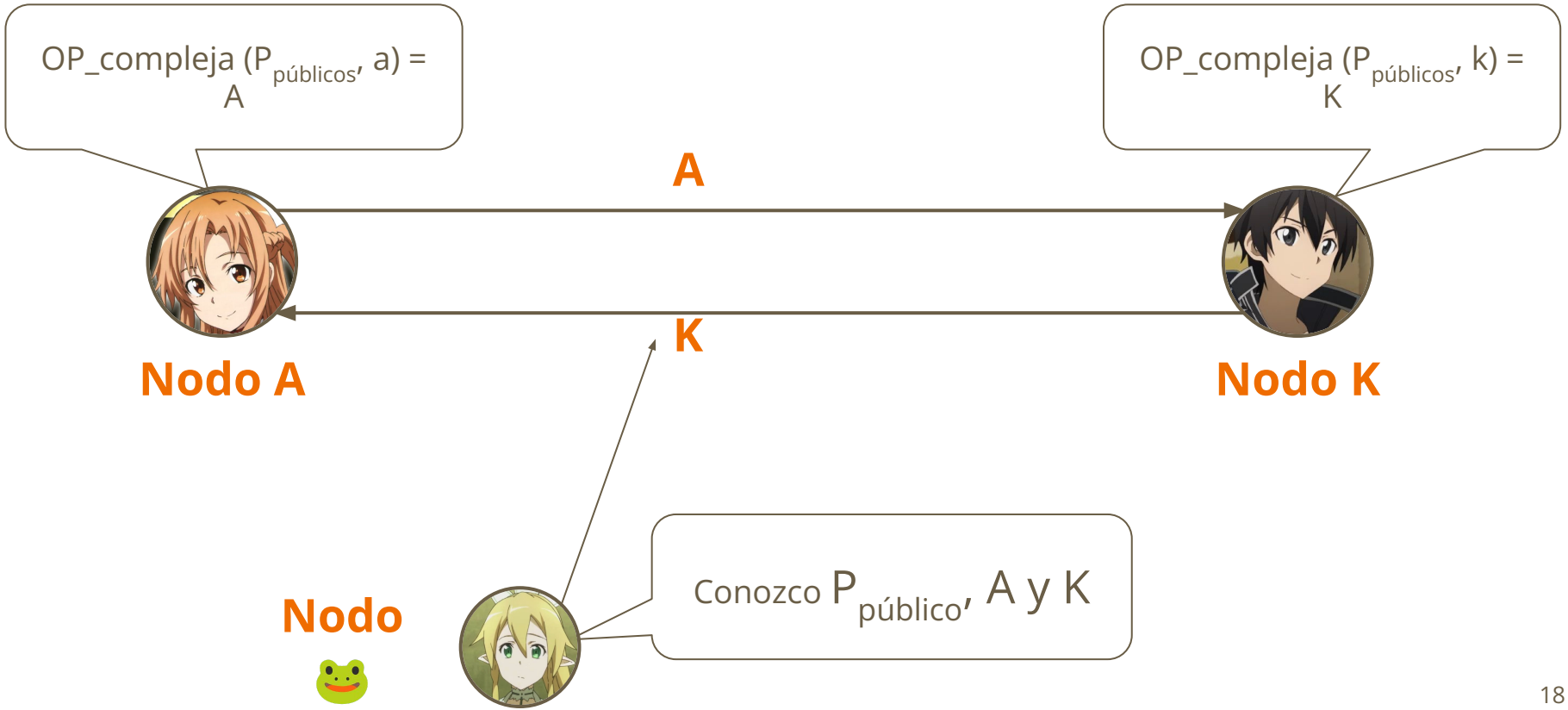
Diffie-Hellman

- Protocolo criptográfico creado por Whitfield Diffie y Martin Hellman en 1976
- Permite que dos partes establezcan una clave secreta compartida a través de un canal de comunicación inseguro sin que nunca se transmita explícitamente la clave secreta.
- Solo se transmiten datos públicos, pero construidos de tal forma, que combinándolo con un número secreto que cada parte conoce, llegan al mismo valor.
- Un atacante que observe el intercambio sólo verá los datos públicos compartidos, pero no podrá derivar la clave secreta sin conocer uno de los números secretos originales.

Intercambio de llaves - *Pipeline*



Intercambio de llaves - *Pipeline*



Intercambio de llaves - *Pipeline*

OP_compleja_2 ($P_{públicos}$, a, K) = S

Nodo A

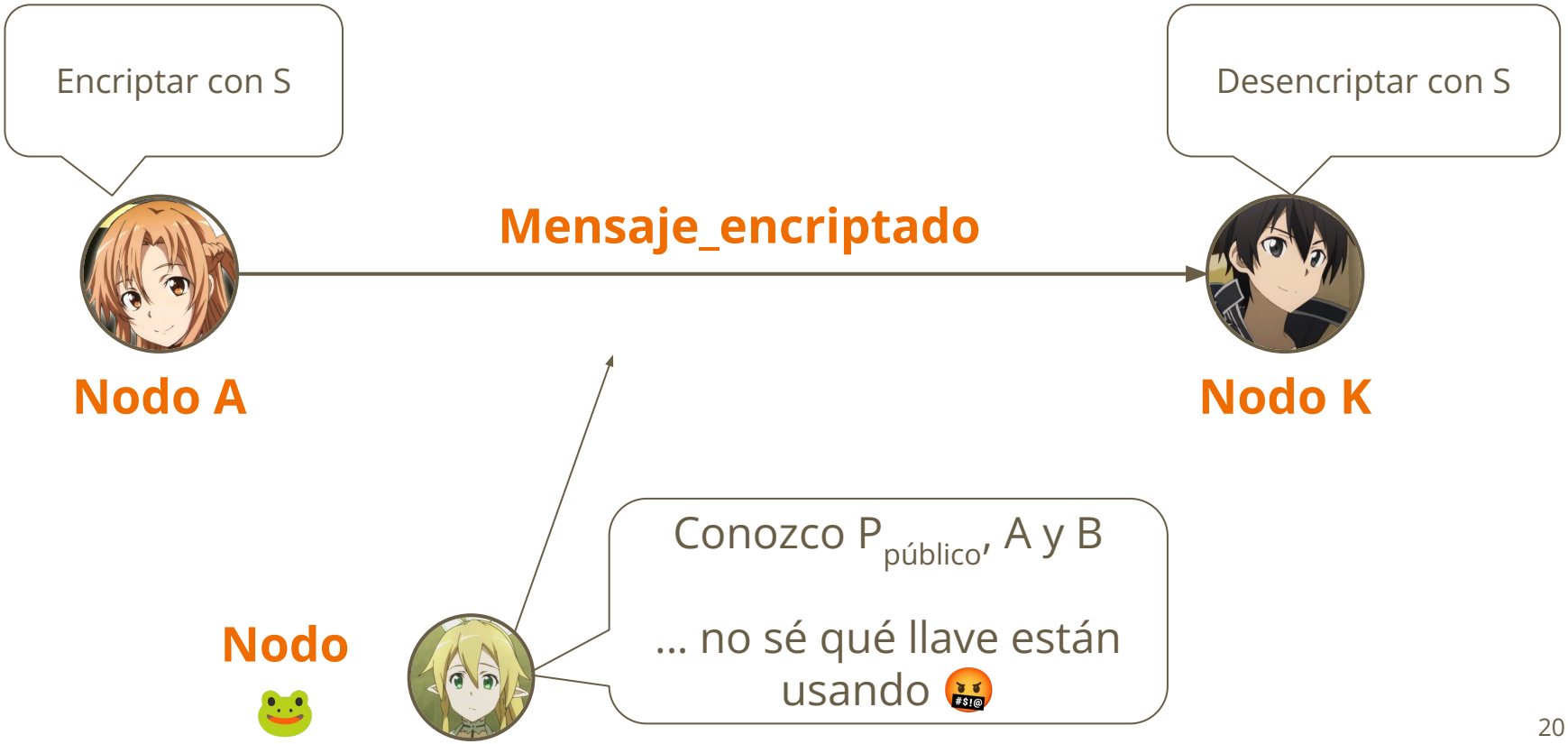
OP_compleja_2 ($P_{públicos}$, k, A) = S

Nodo K

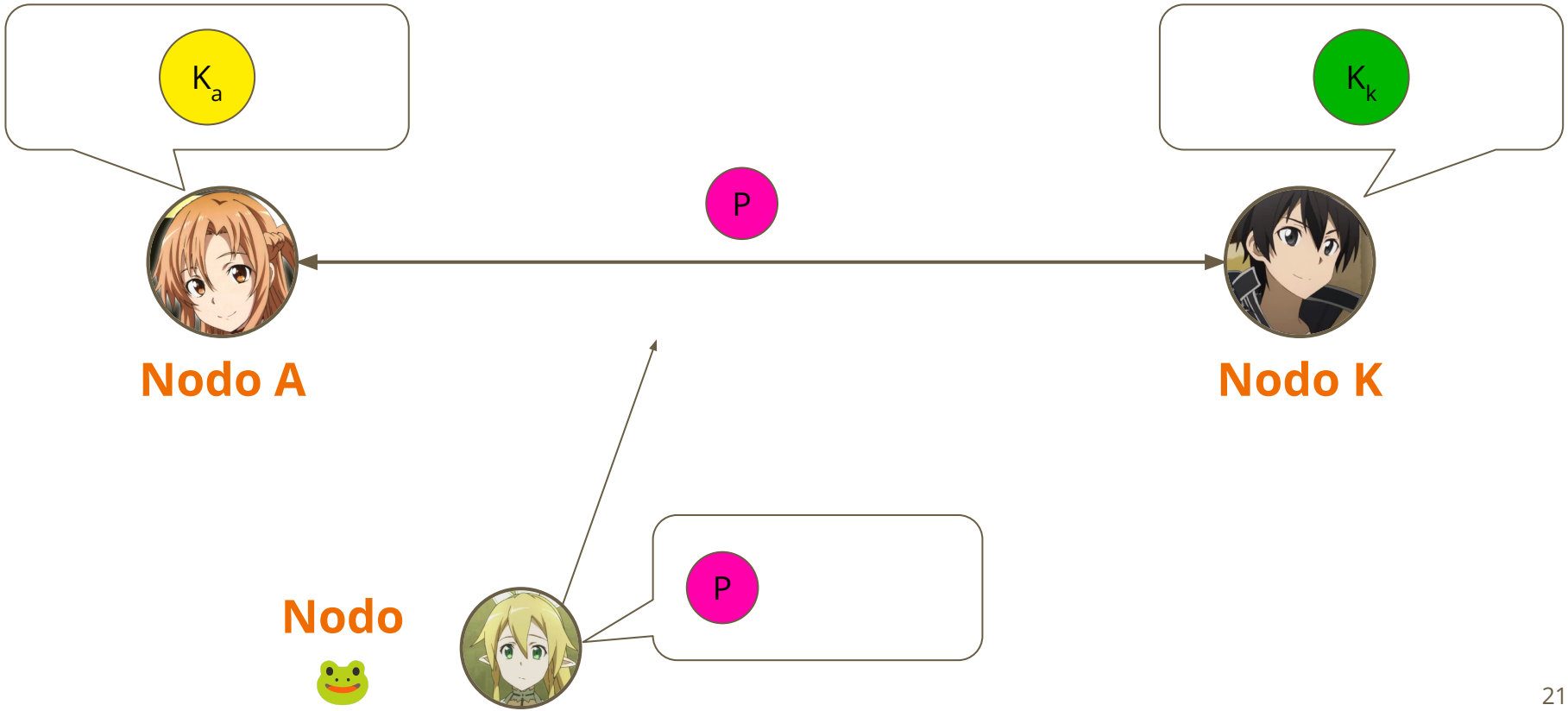
Nodo

🐸

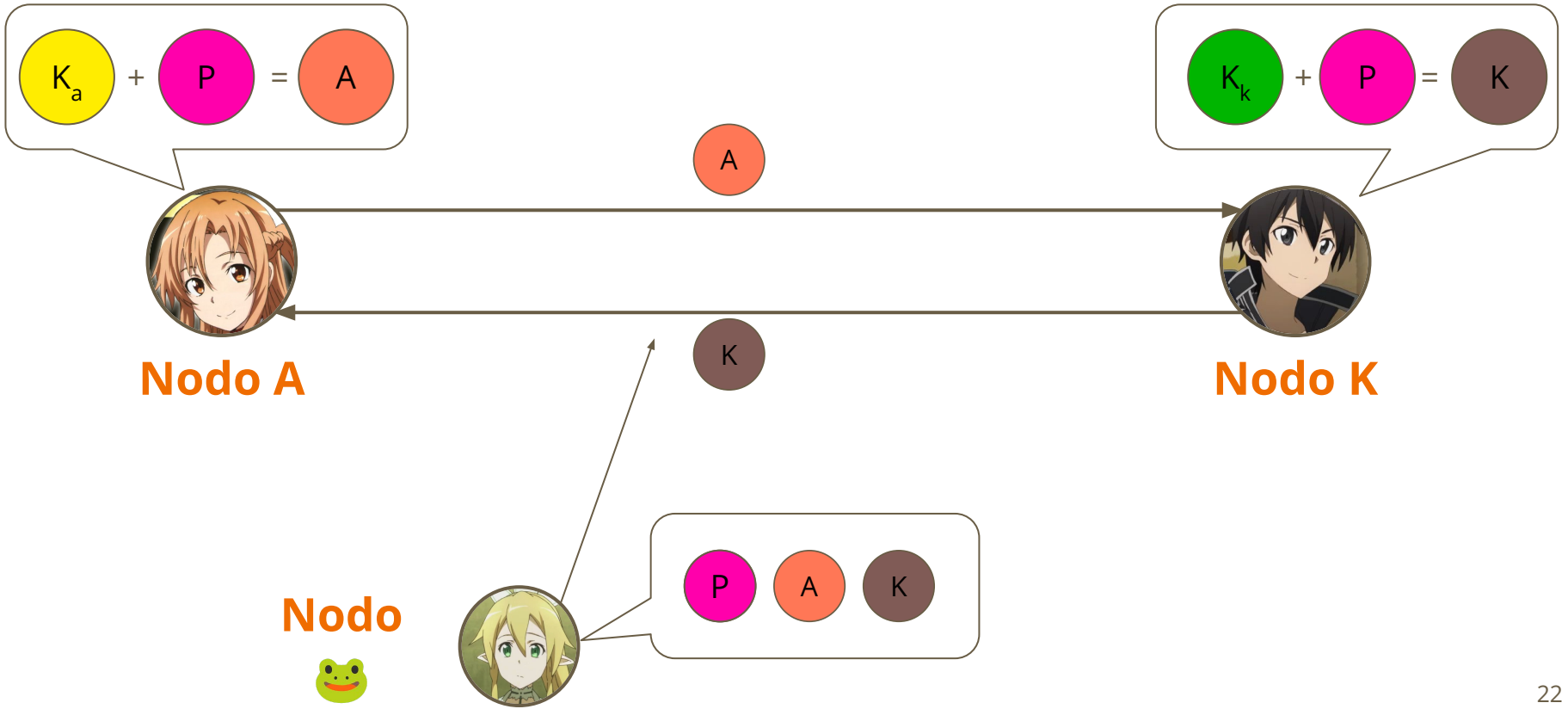
Intercambio de llaves - *Pipeline*



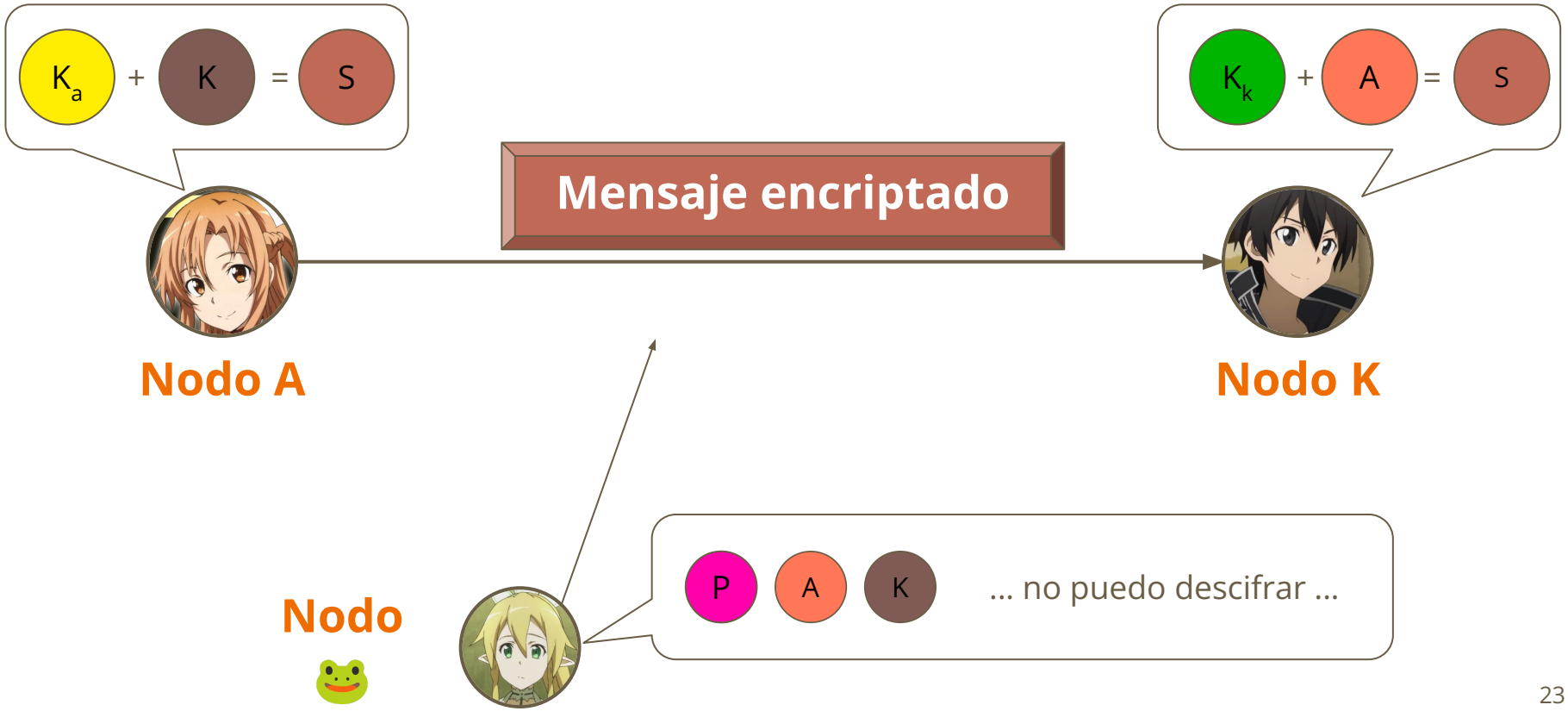
Intercambio de llaves - Ejemplo más "intuitivo"



Intercambio de llaves - Ejemplo más "intuitivo"



Intercambio de llaves - Ejemplo más "intuitivo"



Intercambio de llaves

Ataque *Man-in-the-Middle* en Diffie-Hellman

- Diffie-Hellman no verifica la identidad de las partes.
- Cualquiera puede interceptar y hacerse pasar por otro.

Intercambio de llaves

Ataque *Man-in-the-Middle* en Diffie-Hellman

- Diffie-Hellman no verifica la identidad de las partes.
- Cualquiera puede interceptar y hacerse pasar por otro.
- **Necesitamos un mecanismo para verificar la identidad.**

Cifrado Asimétrico

Cifrado Asimétrico

- Se utiliza un par de claves **matemáticamente relacionadas**: una llave pública (PK) y una llave privada (SK).
- La clave pública se puede compartir libremente, mientras que la clave privada debe conocerla únicamente su creador.
- También se le llama criptografía de clave pública.

Cifrado Asimétrico

- Se utiliza un par de claves **matemáticamente relacionadas**: una llave pública (PK) y una llave privada (SK).
- La clave pública se puede compartir libremente, mientras que la clave privada debe conocerla únicamente su creador.
- También se le llama criptografía de clave pública.

Ventajas

- Otra forma de resolver el problema de la distribución de claves inicial en la criptografía simétrica.
- Permite la autenticación y el no-repudio.
    - No-repudio: Quien "firma" un mensaje con su llave privada no puede negar ser el dueño porque solo él tiene esa llave.

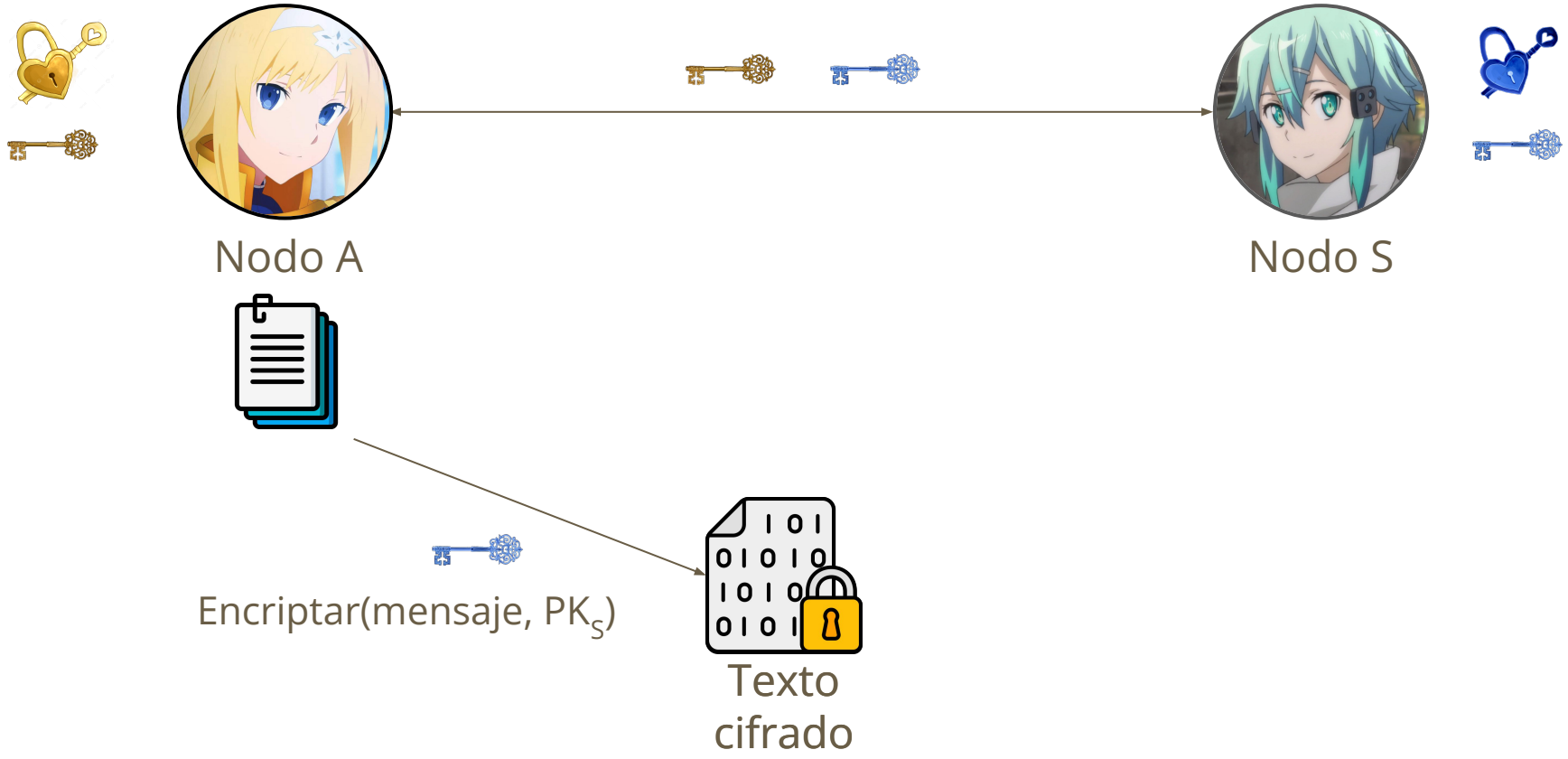
Cifrado Asimétrico - *Pipeline*

Nodo A

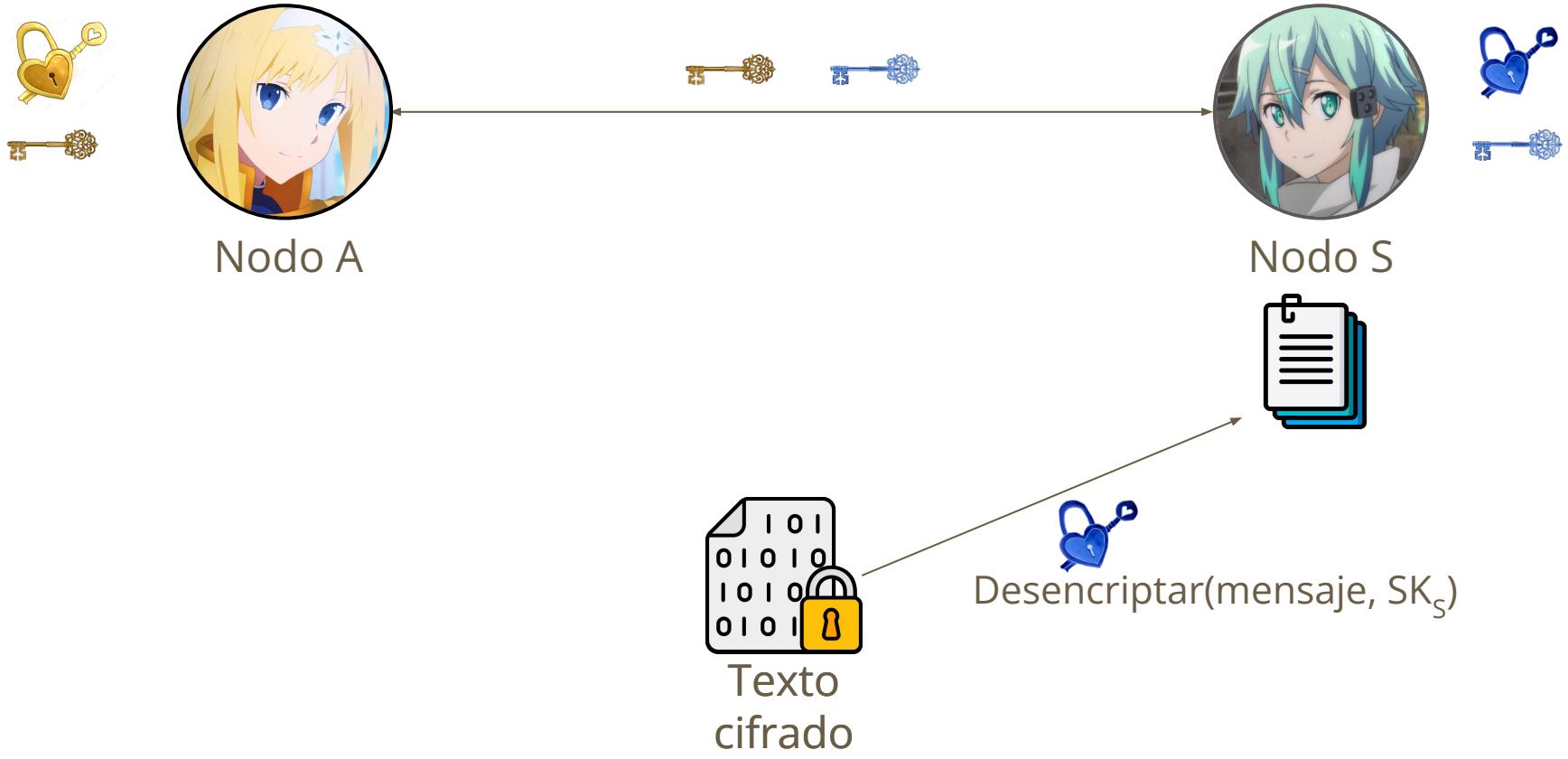
Ambas partes conocen la llave pública

Nodo S

Cifrado Asimétrico - *Pipeline*

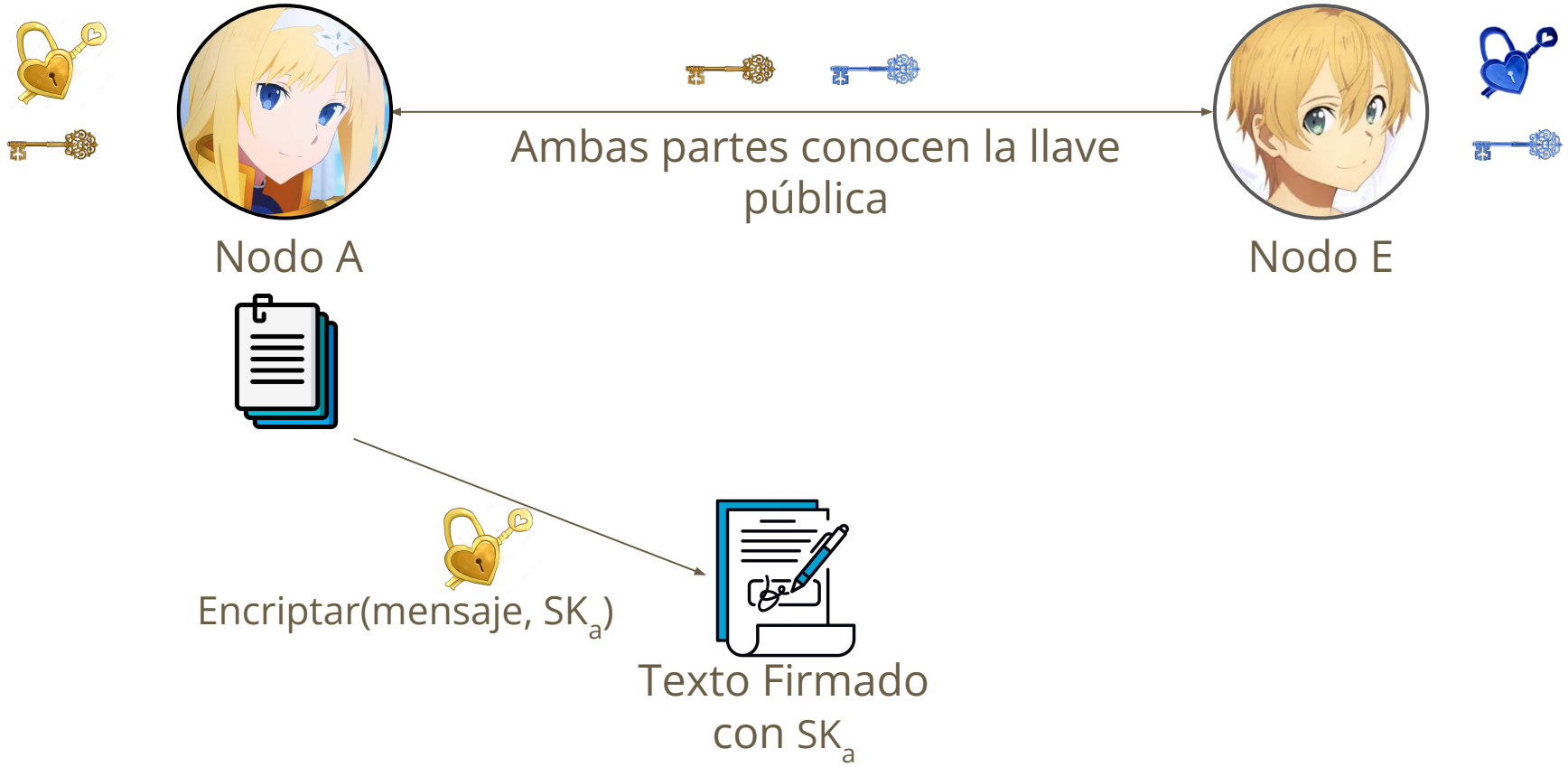


Cifrado Asimétrico - *Pipeline*



Cifrado Asimétrico - *Pipeline para Firmas Digitales*

Cifrado Asimétrico - *Pipeline para Firmas Digitales*



Cifrado Asimétrico - *Pipeline para Firmas Digitales*

Nodo A

Ambas partes conocen la llave pública

Nodo E

Texto Firmado con SK_a

Desencriptar(mensaje, PK_a)

Solo se logra desencriptar con la llave correcta

Cifrado Asimétrico - Algoritmos

- **RSA (*Rivest, Shamir y Adelman*)**: El algoritmo de clave pública más conocido y ampliamente utilizado. Su seguridad se basa en la dificultad de factorizar números primos muy grandes.
- ***ECC (Elliptic Curve Cryptography)***: Ofrece un nivel de seguridad comparable al de RSA pero con claves más cortas y requisitos de procesamiento más bajos. Es ideal para dispositivos con recursos limitados, como móviles.
  - *Bitcoin* y *Ethereum* utilizan ECC, específicamente el algoritmo de firma digital de curva elíptica (ECDSA), para firmar transacciones.

Cifrado Asimétrico

Desafío Principal

- Es computacionalmente mucho más lenta que la criptografía simétrica (100 a 1000 veces más).
- Para la firma de documentos, mientras más grande es dicho documento, más lento y costoso es el proceso.

Cifrado Asimétrico

Desafío Principal

- Es computacionalmente mucho más lenta que la criptografía simétrica (100 a 1000 veces más).
- Para la firma de documentos, mientras más grande es dicho documento, más lento y costoso es el proceso.
- **Necesitamos otro mecanismo para formar documentos que sea más rápido.**

Funciones *Hash* y Firmas Digitales

Funciones *Hash* y Firmas Digitales

- Una función *hash* toma un mensaje de cualquier longitud como entrada y produce una cadena de *bits* de longitud fija como salida.
- Garantiza la integridad de los datos (cualquier cambio en el mensaje original producirá un *hash* diferente).
- Son la base de las firmas digitales.

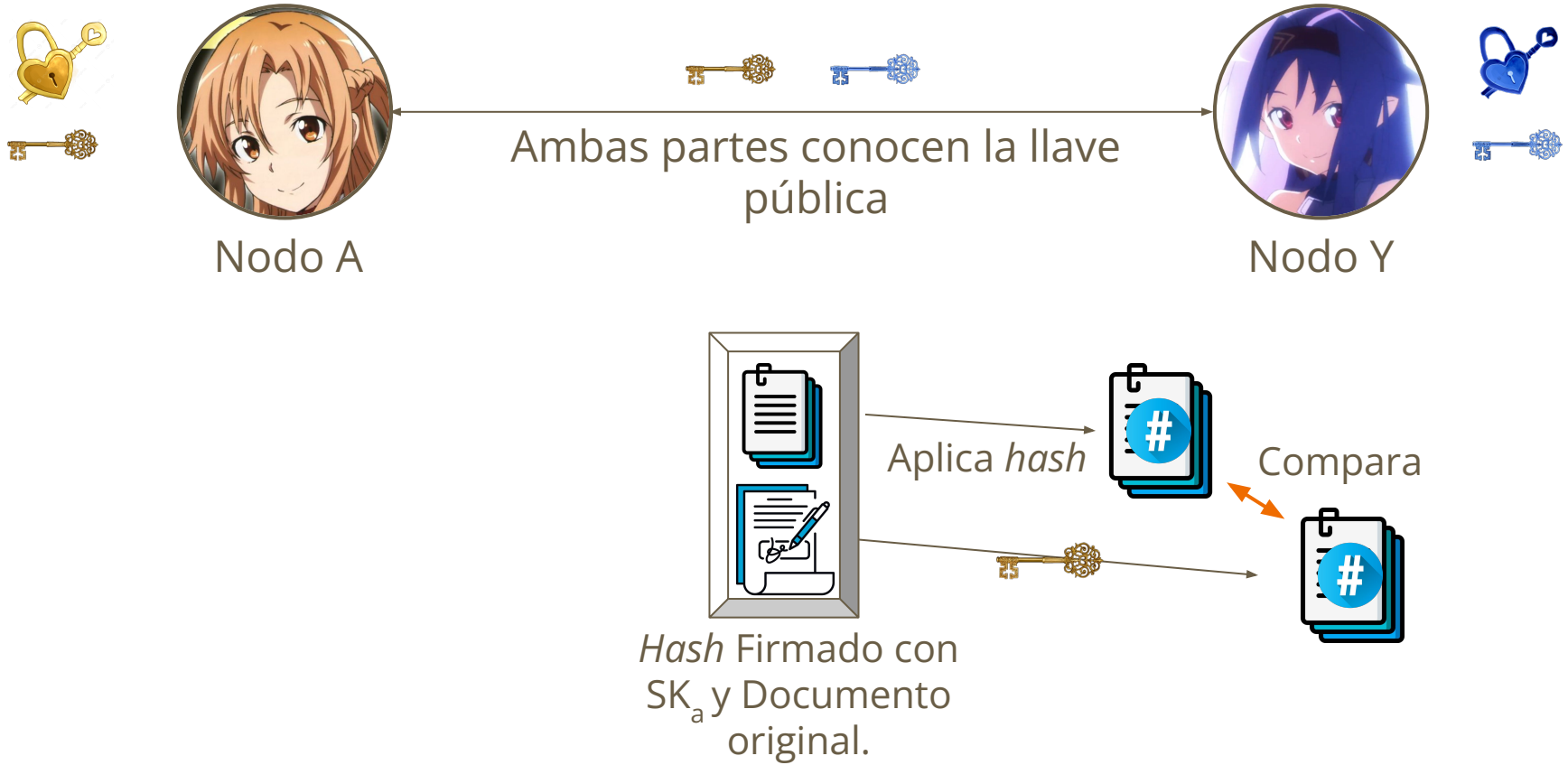
Funciones *Hash* y Firmas Digitales

Propiedades Esenciales

- **Unidireccional (*One-way*)**
  - Es computacionalmente inviable revertir el *hash* para encontrar el mensaje original.
- **Resistencia a colisiones débiles (*Weak Collision Resistance*):**
  - Dado un mensaje (m) y su hash (h), es computacionalmente inviable encontrar otro mensaje diferente (m') que produzca el mismo hash (h).
- **Resistencia a colisiones fuertes (*Strong Collision Resistance*):**
  - Es computacionalmente inviable encontrar cualquier par de mensajes diferentes (m, m') que produzcan el mismo *hash*.

Funciones *Hash* y Firmas Digitales - *Pipeline*

Funciones *Hash* y Firmas Digitales - *Pipeline*



Funciones *Hash* y Firmas Digitales

Resultados Clave

- **Integridad del Mensaje**: Si el mensaje fue alterado en tránsito, el *hash* calculado no coincidirá con el de la firma.
- **No-Repudio**: El emisor de la firma no puede negar haber firmado el documento, ya que solo él posee la clave privada utilizada para crear la firma.

Protocolos Híbridos

Protocolos Híbridos

- La criptografía simétrica es rápida pero la distribución de claves es un desafío.
- La criptografía asimétrica resuelve la distribución de claves pero es lenta.
- Los protocolos híbridos combinan las ventajas de ambas.

Protocolos Híbridos - *Pipeline* resumido

1. **Establecimiento de Sesión**

- Las partes utilizan la criptografía de clave pública para intercambiar, de forma segura, una clave de sesión secreta de uso único.
- Este proceso inicial es lento, pero solo se realiza una vez por sesión.
- La clave de sesión es efímera, es decir, se genera para una sesión específica y luego se descarta, lo que aumenta la seguridad.

2. **Comunicación de Datos**

- Toda la comunicación siguiente (datos masivos) se cifra y descifra utilizando esta clave simétrica.

Protocolos Híbridos - TLS 1.3

- *Transport Layer Security* (TLS) es el protocolo estándar para asegurar comunicaciones en redes informáticas, garantizando confidencialidad, integridad y autenticación.

¿Por qué enfocarnos en TLS 1.3?

- Es la versión más reciente y segura del protocolo (publicada en 2018).
- Mejora significativa en rendimiento y privacidad respecto a versiones anteriores.
- Corrige vulnerabilidades y elimina características inseguras presentes en TLS 1.2 y anteriores.
- Amplio soporte en navegadores y sistemas modernos.

Protocolos Híbridos - TLS 1.3

Saludo del Cliente

Protocolos Híbridos - TLS 1.3

Respuesta del Servidor

Cliente

Te mando 

Usaremos la encriptación X

También incluyo certificado 

Y con eso estoy listo ✅

Servidor

Protocolos Híbridos - TLS 1.3

Procesos internos

Cliente

Construir secreto

Verificar certificado 

Servidor

Construir secreto

Protocolos Híbridos - TLS 1.3

Cliente confirma todo ya utilizando llave secreta

Protocolos Híbridos - TLS 1.3

Comunicación encriptada

Extra - Enlaces de interés

- CryptoHack - Plataforma de desafíos prácticos en criptografía.
- Hash Table - VisuAlgo - Ver visualmente algunas implementaciones de *hash*.
- CryptoZombies - Curso gamificado para aprender criptografía y contratos inteligentes (en Ethereum).
- TLS 1.3 Handshake: Taking a Closer Look

Poniendo a prueba lo que hemos aprendido 👀

Considera las siguientes afirmaciones sobre mecanismos criptográficos:

I. El no-repudio asegura que el receptor no pueda negar haber recibido el mensaje.

II. El protocolo *Diffie-Hellman* presenta vulnerabilidad frente a ataques de intermediario porque no autentica la identidad de las partes.

III. El cifrado simétrico resuelve el problema de distribución inicial de claves utilizando un par de llaves pública y privada.

¿Cuáles de las afirmaciones anteriores son **correctas**?

a. Solo I

b. Solo II

c. Solo III

d. I y II

e. II y III

Poniendo a prueba lo que hemos aprendido 👀

Considera las siguientes afirmaciones sobre mecanismos criptográficos:

I. El no-repudio asegura que el receptor no pueda negar haber recibido el mensaje.

II. El protocolo *Diffie-Hellman* presenta vulnerabilidad frente a ataques de intermediario porque no autentica la identidad de las partes.

III. El cifrado simétrico resuelve el problema de distribución inicial de claves utilizando un par de llaves pública y privada.

¿Cuáles de las afirmaciones anteriores son **correctas**?

a. Solo I

b. Solo II

c. Solo III

d. I y II

e. II y III

Próximos eventos

Próximas clases

- Lunes: Resumen de clase como estudio de la I2
- Miércoles: No hay clases, solo trabajar en el control
- Post I2: Autenticación y autorización como mecanismo de seguridad.

Evaluación

- Hoy se publica control 5, no evalúa esta clase.
- Próxima semana es la I2 🎃

IIC2523
Sistemas Distribuidos

Hernán F. Valdivieso López
(2025 - 2 / Clase 19)

Créditos (animes utilizados)

Sword Art Online