Redes de datos 1

Introducción a la seguridad

Facultad de Ingeniería – Universidad de la República

Instituto de Ingeniería Eléctrica

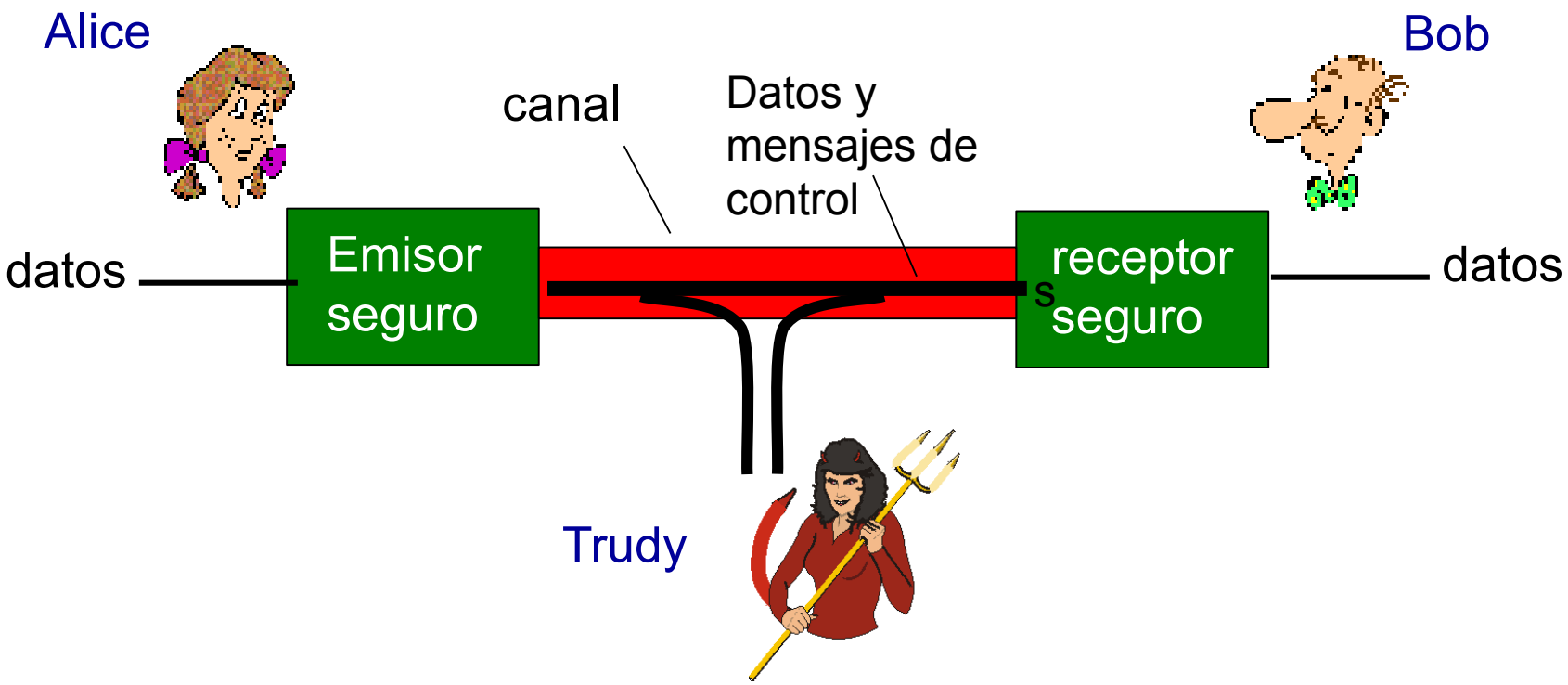
¿Qué podemos entender por seguridad en la red?

- Confidencialidad: solo el emisor y el receptor autorizado pueden “entender” el mensaje
  - Criptografía
- Autenticación
  - Determinar la identidad de mi contraparte
- Autorización
  - Determinar qué permito hacer a la contraparte
- Integridad de los mensajes
  - Receptor quiere asegurarse que los mensajes no fueron modificados sin detección
- Acceso y disponibilidad: los servicios deben ser accesibles y estar disponibles para el usuario legítimo

- La seguridad en la red es solo una parte de una solución completa de seguridad

Amigos y enemigos

- Alice y Bob (A y B): entidades que quieren comunicarse “de forma segura”
- Trudy: entidad maliciosa. Puede interceptar, borrar, o agregar mensajes



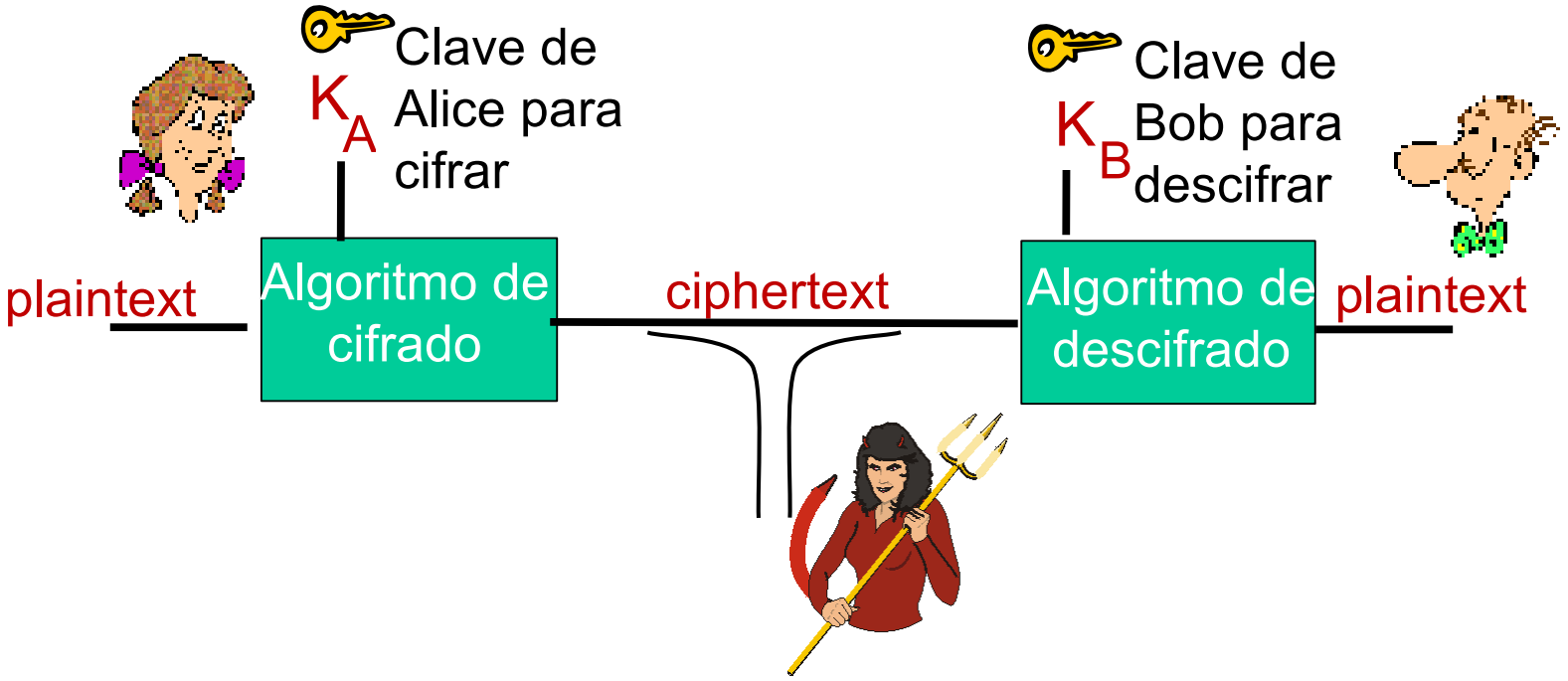
- En la vida real pueden ser personas, navegador y servidor web, entidades de correo electrónico seguro, DNS, routers intercambiando tablas de rutas….

¿Qué puede hacer (intentar hacer) un atacante?

- Espiar (eavesdrop): interceptar mensajes
- Insertar mensajes en la conversación (p. ej. rutas falsas)
- Suplantación de identidad (hacerse pasar por otra entidad)
- Secuestro (por ejemplo de una comunicación establecida)
- Negación de servicio (impedir que el servicio sea utilizado por otros)
- Etc., etc.

Criptografía

- Criptografía: Estudio de principios o métodos de cifrado
  - Y otras funcionalidades asociadas
- Cifrar: informalmente, proceso para “ocultar” la información mediante la aplicación de algoritmos matemáticos para ser transmitida a través de un canal inseguro
- Descifrar: proceso aplicado a la información “oculta” para obtener el mensaje original



- Texto plano (plaintext) m: mensaje original (no necesariamente “texto”)
- Texto cifrado (ciphertext) $K_A(m)$: mensaje “ocultado” por el cifrado

¿Difícil o imposible?

- Seguridad incondicional:
  - No importa que tiempo o poder computacional se disponga, el cifrado no podrá ser quebrado ya que no hay suficiente información para determinar de forma única el correspondiente texto plano
- Seguridad computacional:
  - Dado un poder de recursos computacionales, el tiempo necesario de cálculo esperado para quebrar un código es mucho más grande que el tiempo de vida del mensaje

- En la práctica trabajaremos con seguridad computacional

Criptosistema (clásico)

- Se llama criptosistema al conjunto de:
  - Un conjunto finito de posibles textos planos
  - Un conjunto finito de posibles textos cifrados
  - Un conjunto finito de posibles claves
  - Un conjunto de reglas de cifrado y otro de reglas de descifrado, que cumplen que:
    - $d_K(e_K(x)) = x$ para todos los posibles textos planos x

Criptoanálisis

- Es el estudio de métodos para obtener el significado de la información que está cifrada
- Es la ciencia de descifrar códigos, decodificar secretos, violar esquemas de autenticación y romper protocolos criptográficos
- El ideal del criptoanálisis es descubrir la clave secreta y no solo decodificar algunos mensajes
- Dos tipos de ataques:
  - Ataques criptoanalíticos
    - Se buscan debilidades en los algoritmos, procedimientos, o implementaciones
  - Ataques de fuerza bruta
    - Se intentan todas las combinaciones hasta encontrar la o las correctas
    - Teóricamente siempre funcionan. Pero se busca que sean imprácticos

Clasificación de acuerdo a la información disponible al atacante

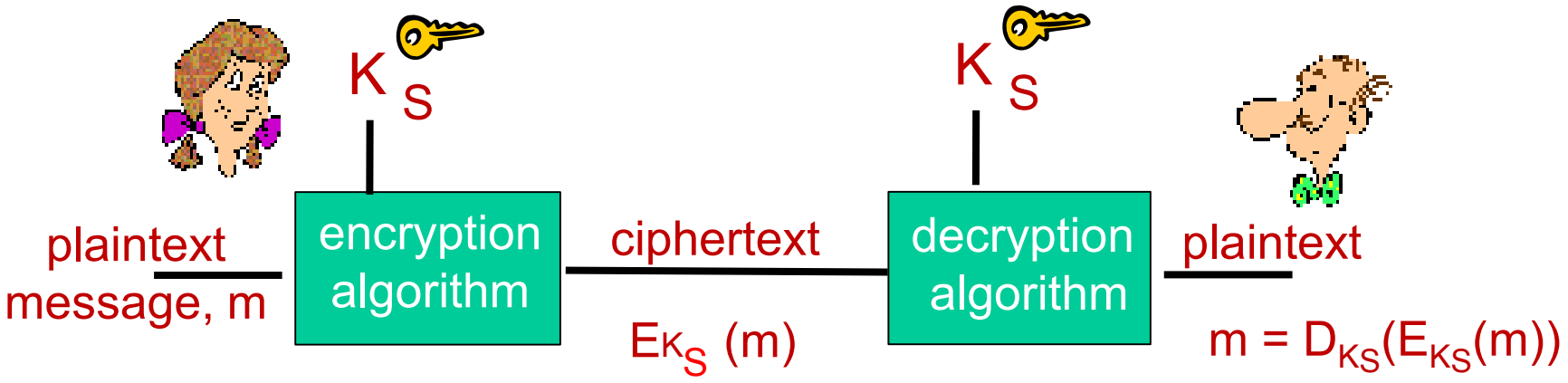
- cipher-text only attack (ataque de texto cifrado): Trudy tiene texto cifrado que puede analizar
- known-plaintext attack (ataque de texto plano conocido): Trudy tiene algunos mensajes antes y después de cifrar
- Chosen-plaintext attack (ataque de texto plano a elección): Trudy puede obtener el texto cifrado para textos de su elección
- Chosen-ciphertext attack (ataque de texto cifrado a elección): Trudy puede obtener el texto descifrado para textos cifrados de su elección

- En todos los casos el ataque por fuerza bruta funciona
  - La longitud de la clave es importante para la seguridad
  - Un ataque criptográfico es encontrar un mecanismo de ataque más eficiente que el ataque por fuerza bruta

Ejemplos de Criptografía tradicional (histórica)

- Cifrado por sustitución
  - Cada símbolo es reemplazado por otro símbolo
  - Ejemplo: Cifrado de Cesar: a->D, b->E, c->F, etc.
  - En general, sustitución monoalfabética:
    - abcdefghijklmnopqrstuvwxyz
    - qwertyuiopasdfghjklzxcvbnm
  - Debilidad: ataques estadísticos (por ejemplo estudiando frecuencia de los caracteres, de los pares...)
    - Por ejemplo, en inglés, letras más frecuentes: e t o a n i
  - Ataque adivinando palabra o frase
- Cifrado por Transposición
  - Reordenar los símbolos de un mensaje
  - Ataque:
    - Descubrir que el método es una transposición
    - Deducir el número de columnas
    - Probabilidad de pares y tríos

Criptografía simétrica



- Alice y Bob comparten la misma clave
- Tenemos 2 algoritmos, E y D, que reciben una clave y un bloque de bytes
- El algoritmo de cifrado y descifrado son inversos cuando utilizan la misma clave

- En el ejemplo del cifrado por sustitución, ¿cuál sería la clave?
- Precisamos un mecanismo para que Alice y Bob obtengan la clave compartida

Cifrados en bloque y stream

- Cifrados en bloque: actúan sobre un bloque de bits de tamaño fijo dependiente del cifrado (ej. 64, 128 bits)
  - Es una función de 2 variables, el bloque y la clave
  - Dado un bloque y una clave, la salida será siempre la misma
- Cifrados stream: el mensaje se cifra de a byte o de a bit
  - “a medida que va pasando”
  - Clave (y usualmente vector de inicialización) se utilizan para inicializar generador de números aleatorios

Ejemplo: DES (Data Encryption Standard)

- Cifrado en bloques
- 1977 – Bloques de 64 bits, claves de 56 bits
- NO es seguro hoy en día (clave muy corta)
- Hoy (2023) puede quebrarse por fuerza bruta en horas

- Se puede hacer seguro aplicándolo 3 veces (DES triple) con al menos 2 claves distintas
  - No se conocen ataques criptoanalíticos efectivos a DES triple
  - Igualmente se recomienda pasar a cifrados más modernos

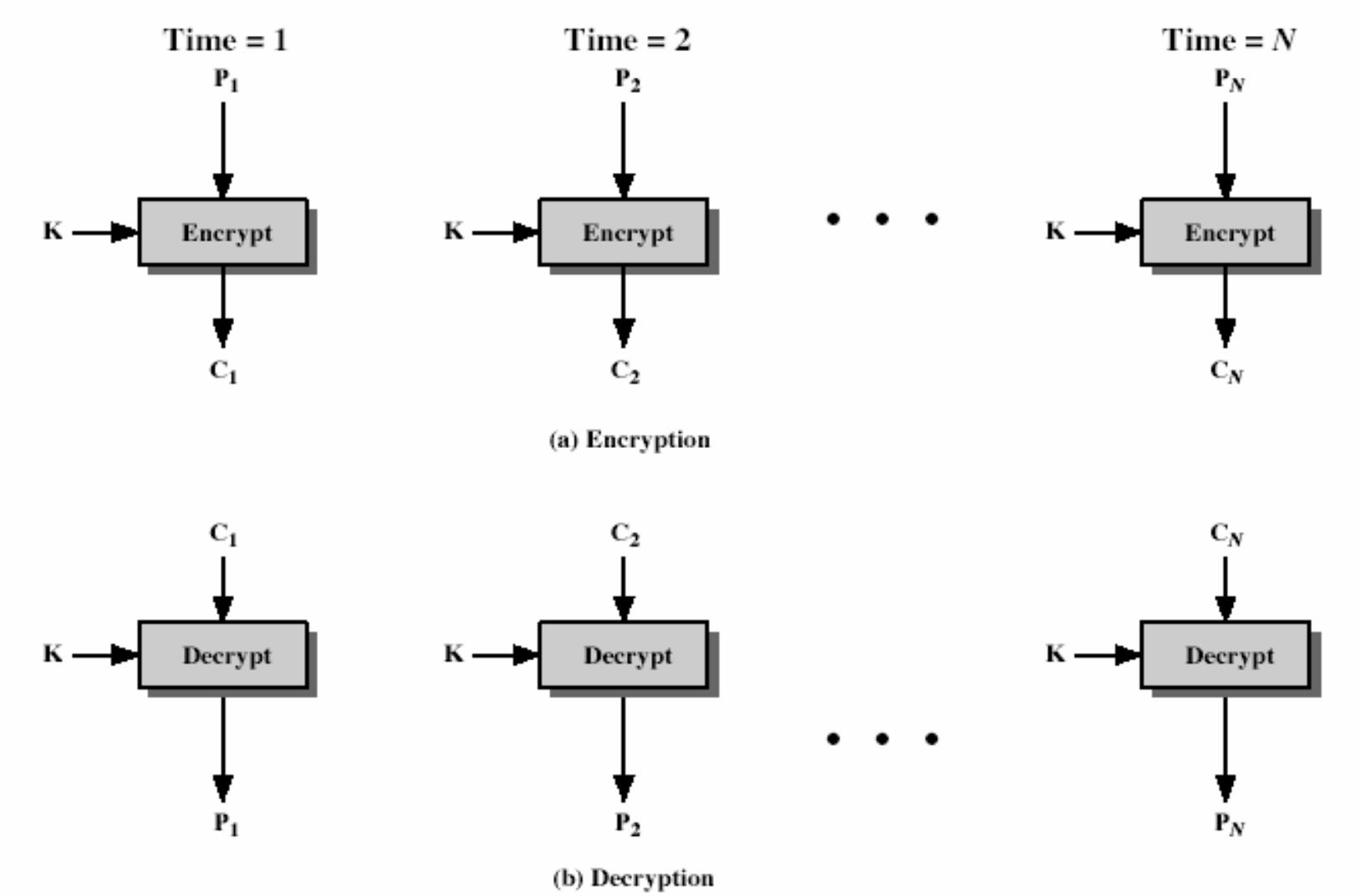
AES (Advanced Encryption Standard)

- Standard del NIST para clave simétrica, reemplaza DES (Nov 2001)
- Bloques de 128 bits
- Claves de 128, 192, o 256 bits
- Descifrado por fuerza bruta: si el ataque por fuerza bruta toma 1 segundo para DES, asumiendo la misma velocidad para realizar un cifrado, para AES con claves de 128 bits tomaría 149×10^{12} años
- El más utilizado hoy en día

Modos de operación (encadenamiento)

- ¿Qué hago si lo que quiero cifrar no tiene el tamaño de un bloque?
  - Modos de encadenado. Definen cómo dividir un mensaje en bloques (posiblemente rellenando el último) y cómo cifrar dichos bloques
- Múltiples modos de operación propuestos. Algunos ejemplos:
- ECB: Electronic Code Book
- CBC: Cipher Block Chaining
- CFB: Cipher FeedBack
- OFB: Output FeedBack

Ejemplo: ECB (Electronic Code Book)



- Los bloques se cifran de forma independiente
- Problema: iguales textos planos producen iguales textos cifrados

Ejemplo: Cipher Block Chaining

- Se hace depender el cifrado del bloque anterior
- Evita que dos bloques idénticos den el mismo cifrado

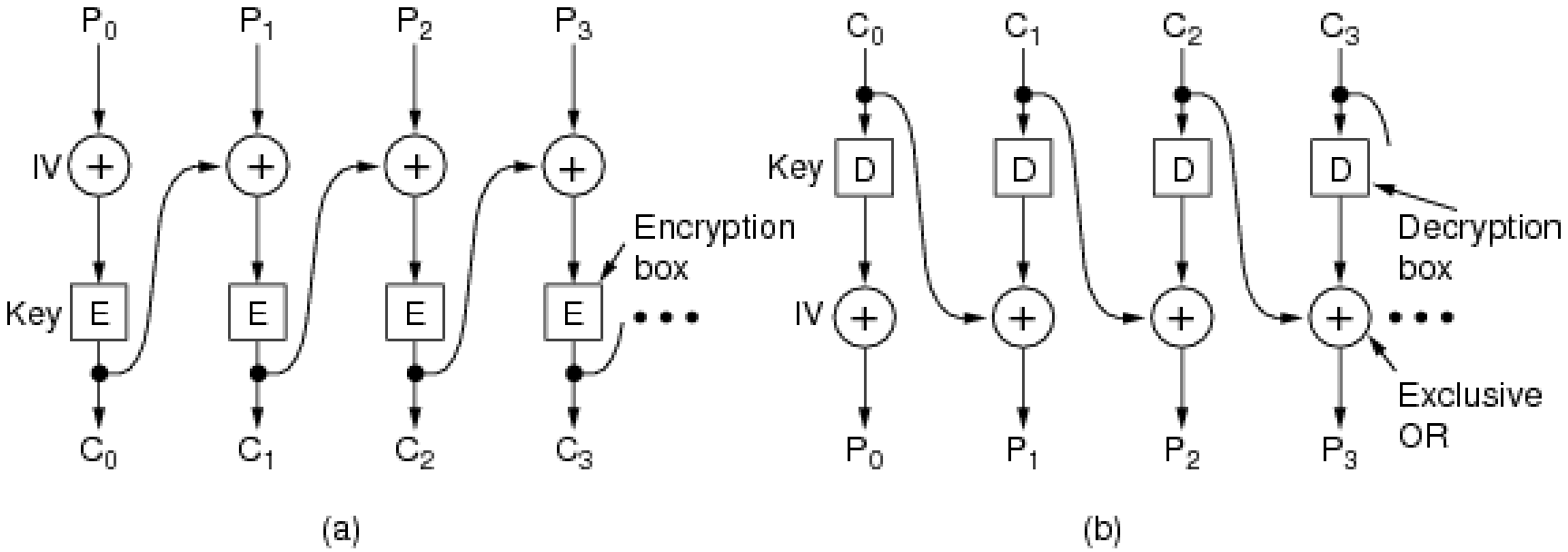
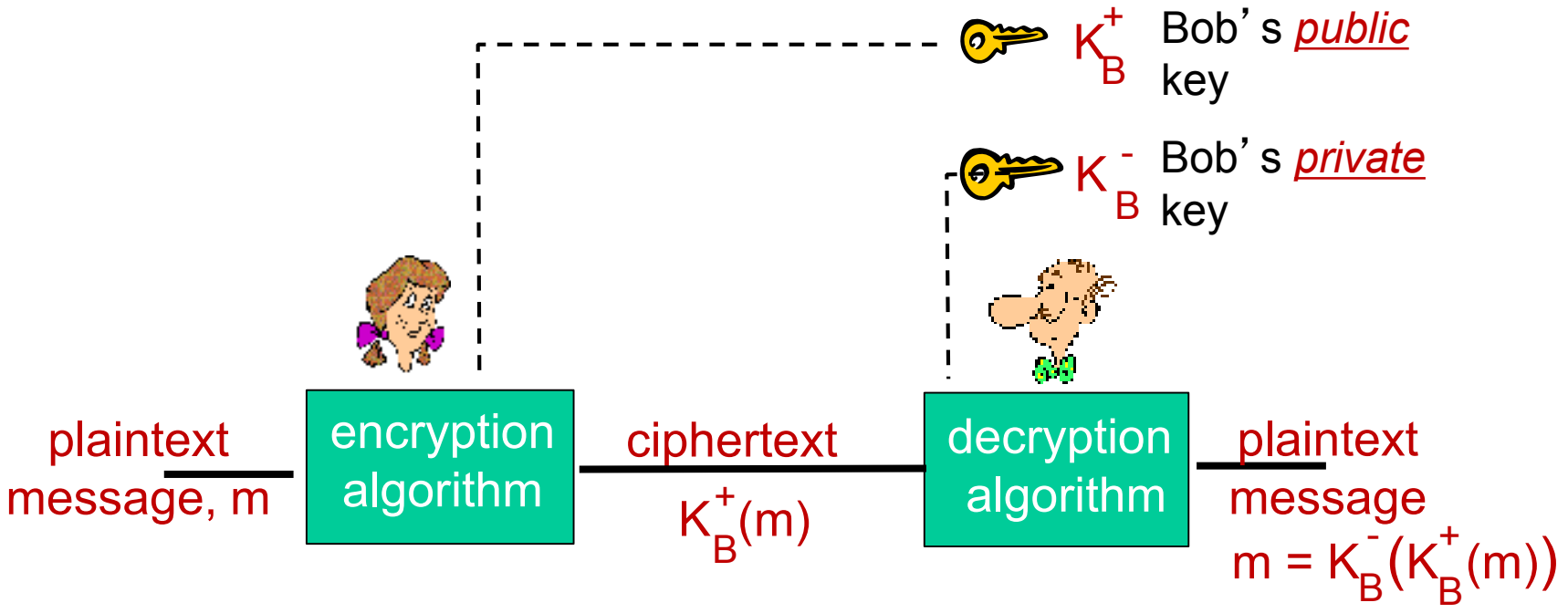


Fig. 8-12. Cipher block chaining. (a) Encryption. (b) Decryption.

Criptografía de clave pública

- En criptografía de clave simétrica, debemos compartir una clave previamente
  - Puede ser un problema
- Criptografía de clave pública: nuevo enfoque
  - Diffie & Hellman, 1976
  - Clave de cifrado y descifrado distintas
  - No se puede deducir una de la otra
  - Clave pública conocida por todos
  - Clave privada solo por su dueño
- La seguridad de los criptosistemas asimétricos se basa en que la clave privada solo puede ser computada a partir de la pública resolviendo un problema “difícil”
  - Mientras que generarlas juntas es “fácil”
- Se utilizan problemas matemáticos para los cuales no se conocen soluciones eficientes

Criptografía de clave pública (o asimétrica)



- Requerimientos:
  - $K_B^-(K_B^+(m)) = m$
  - Dada la clave pública, K_B^+, debe ser "imposible" deducir K_B^-

** Nueva nomenclatura: $K_B^+(M)$ significa cifrar/descifrar el bloque M con la clabe K_B^+*

Ejemplo: RSA

- RSA (1978): algoritmo de Rivest, Shamir y Adleman
- Se basa en dificultad de factorizar el producto de números primos grandes
  - Si p y q son números primos, pero solo tengo n=p*q, encontrar p y q es un problema matemático difícil
  - Hoy se acepta que claves con n de 2048 o más bits son seguras en el futuro (hoy en día tampoco se puede factorizar n general de 1024 bits)

Algunas propiedades de la aritmética modular

- x mod n = resto de dividir x entre n
- Resultados matemáticos:
  - [(a mod n) + (b mod n)] mod n = (a+b) mod n
  - [(a mod n) - (b mod n)] mod n = (a-b) mod n
  - [(a mod n) * (b mod n)] mod n = (a*b) mod n
- Se obtiene también:
  - $(a \bmod n)^d \bmod n = a^d \bmod n$

Creando las claves para RSA

- Elegir 2 números primos grandes p y q (p. ej. 1024 bits cada uno)
- Calcular n=p.q, z=(p-1)(q-1)
- Elegir e, con e<n, que no tenga factores comunes con z
- Elegir d tal que d.e-1 sea divisible por z (d.e mod z=1)
- Clave pública K^{+}_{B} =(n,e). Clave privada K^{-}_{B} =(n,d)

Cifrado RSA

- Dadas las claves (e,n) (d,n)
- Dado un mensaje m, que debe tener menos bits que n, representamos m como un entero
- Cifrado:

  $c = m^e \bmod n$

- Descifrado:

  $m = c^d \bmod n$

- Se cumple que

$$m = (\underbrace{m^e \bmod n}_{c})^d \bmod n$$

- No vamos a demostrarlo

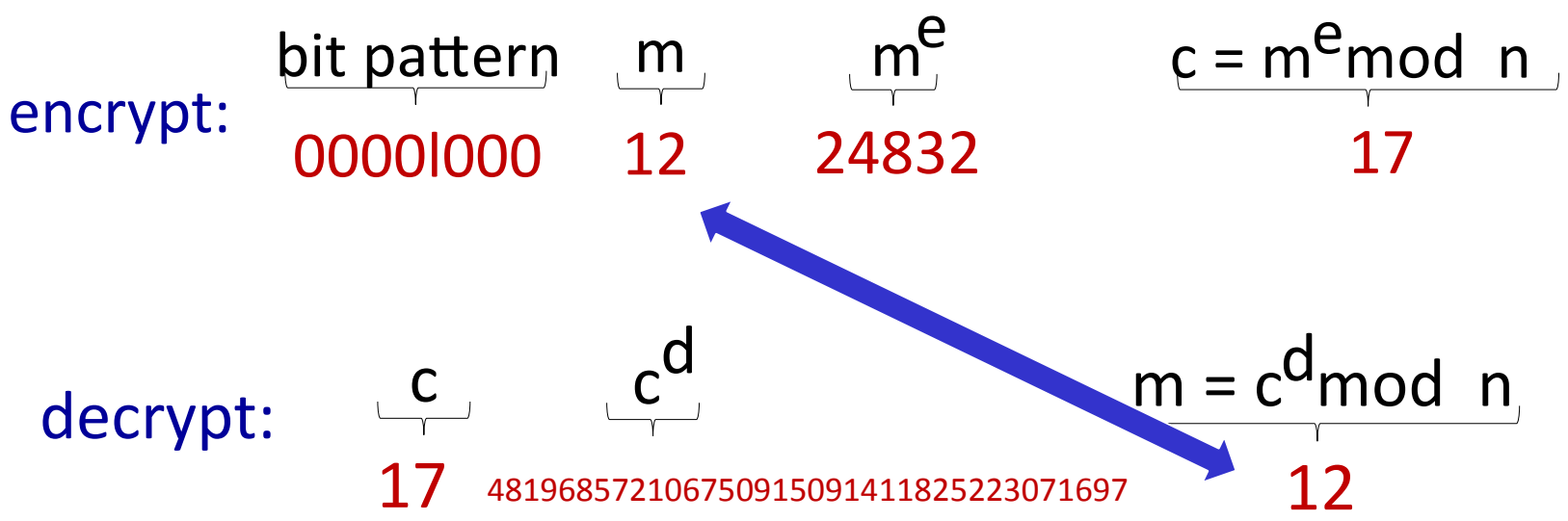
- Observar que podemos intercambiar las claves pública y privada

$$K_B^-(K_B^+(m)) = m = K_B^+(K_B^-(m))$$

Ejemplo de RSA

- Bob elige p=5, q=7. Entonces n=35, z=24.
- Elegimos e=5 (e, z primos entre si).
- Obtenemos d=29 (ed-1 divisible entre z).

Ver detalle de las cuentas en el Kurose:



Seguridad de RSA

- Encontrar d a partir de (e,n), es equivalente a obtener p y q
  - Factorizar n en sus factores primos
- Este es un problema matemático difícil
- En cambio encontrar números primos grandes es “fácil”

Otros problemas utilizados en criptografía asimétrica

- Logaritmos discretos en los enteros módulo p
  - Sea *p* un número primo muy grande, y α un generador del grupo multiplicativo Z_p^*:
  - $\alpha^0, \alpha^1, \ldots, \alpha^{p-2}$ reducidas módulo *p*, construyen todos los enteros entre 1 y *p*-1
  - En el criptosistema, los parámetros *p* y α son conocidos
  - Dado $\beta \in Z_p^*$, encontrar el único exponente *a* ($0 \leq a \leq p-2$) tal que $\alpha^a = \beta \pmod{p}$
- Logaritmos discretos sobre curvas elípticas

- Ambos métodos obtienen el mismo nivel de seguridad que RSA con claves más cortas

Intercambio de claves de Diffie-Hellman

- La versión original basa su seguridad en la dificultad de encontrar el logaritmo discreto
  - Alice y Bob acuerdan utilizar un módulo *p* y un generador α. Ambos son públicos.
  - Alice elige un número aleatorio secreto a, y envía A= α^a mod p
  - Bob elige un número aleatorio secreto b, y envía B= α^b mod p
  - Alice calcula s=B^a mod p
  - Bob calcula s=A^b mod p
  - ¿obtienen el mismo valor?
  - A^b mod p = $(\alpha^a)^b$ mod p = α^{ab} mod p =$(\alpha^b)^a$ mod p = B^a mod p
  - Solamente a y b son secretos
- Debilidad: ataques Man in the Middle.
  - Debe utilizarse junto con algún sistema de autenticación

Algoritmos de clave pública en la práctica

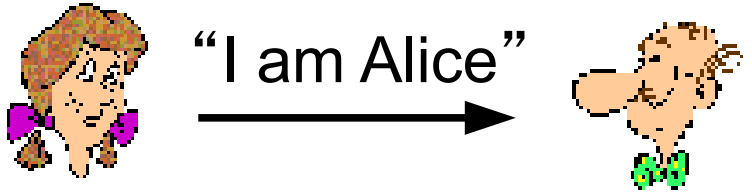
- RSA (y los otros algoritmos de clave pública) son caros computacionalmente
- En el uso práctico, se utilizan para autenticar y establecer una clave de sesión compartida
  - Y luego pasamos a utilizar un algoritmo simétrico, como AES

Autenticación

- Objetivo: Bob quiere que Alice “le demuestre” su identidad
  - Aunque Trudy pueda leer, e incluso reenviar, mensajes de Alice
- Puede realizarse tanto con claves compartidas, como con clave pública

Protocolo de autenticación. Primer intento

Protocolo ap1.0. Alice dice “Yo soy Alice”

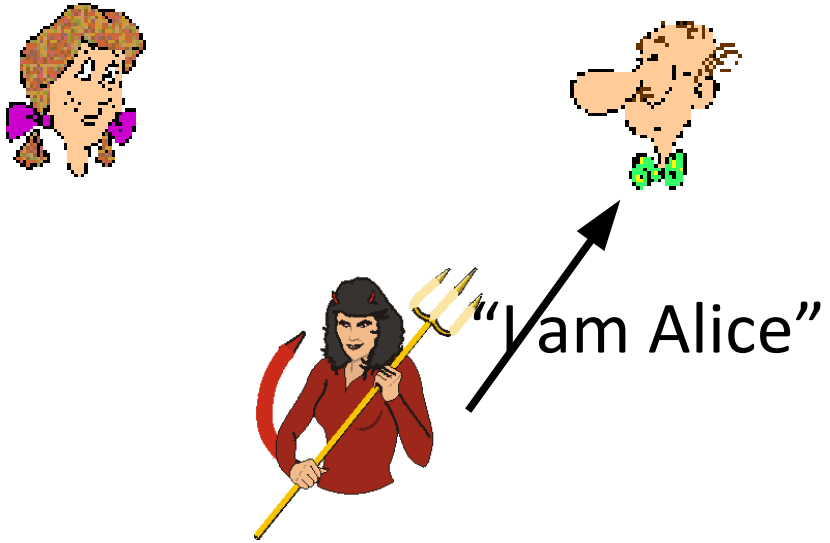


¿Cómo puede fallar?

Bob no está viendo a Alice

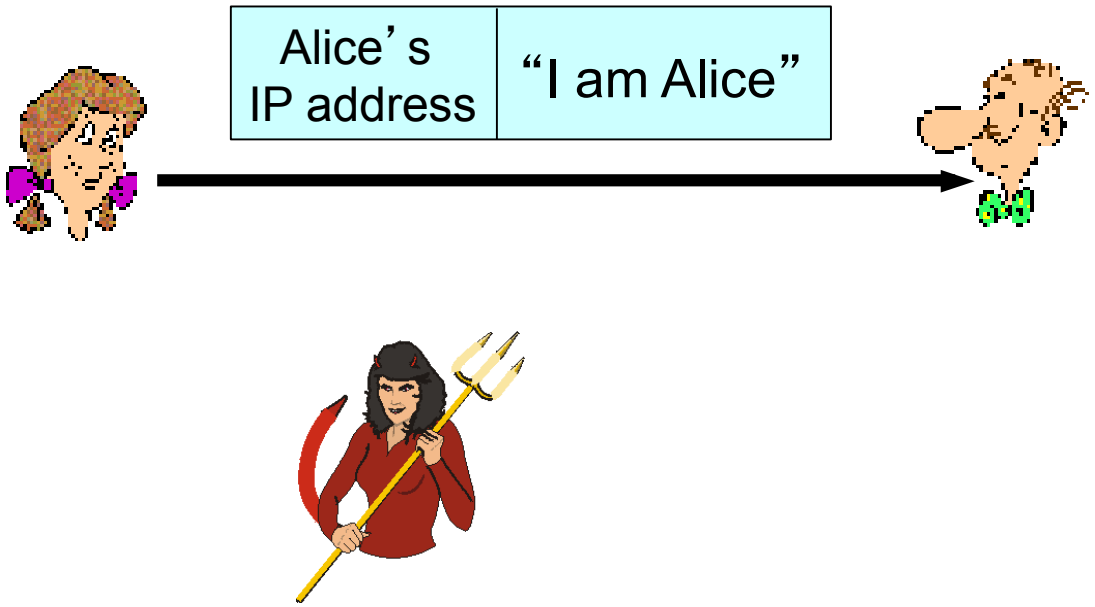
Trudy simplemente declara

ser Alice

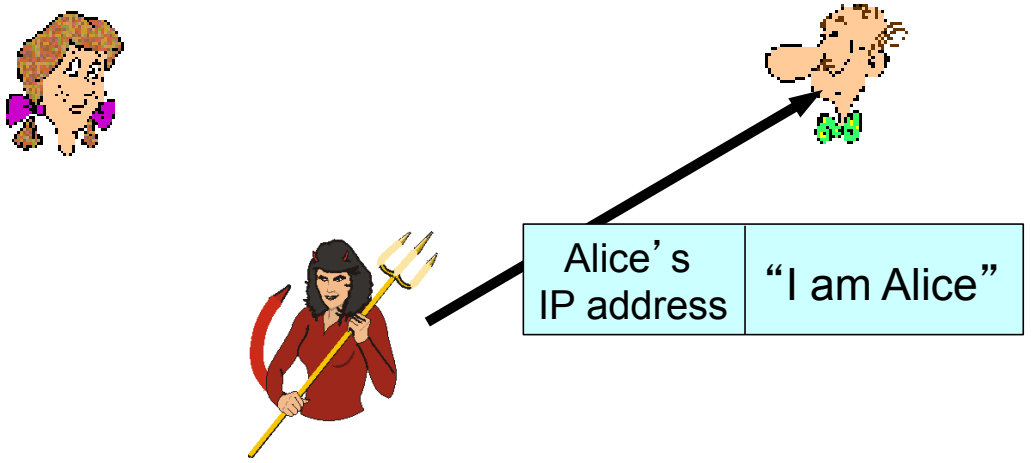


Protocolo de autenticación. Segundo intento

- Bob verifica que la IP de origen sea la de Alice

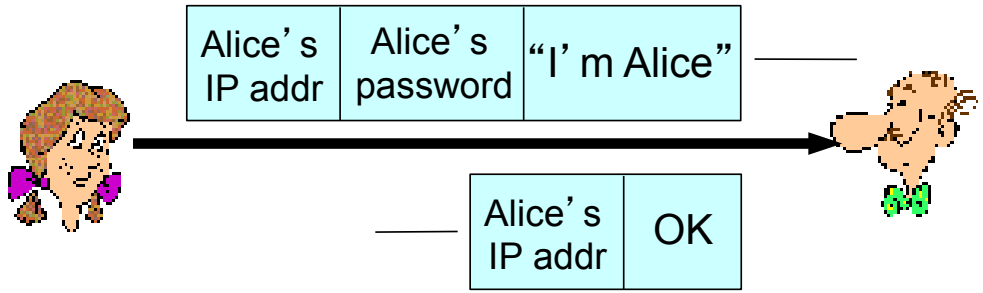


- Problema: en IP no se verifica la IP de origen. Trudy puede falsearla

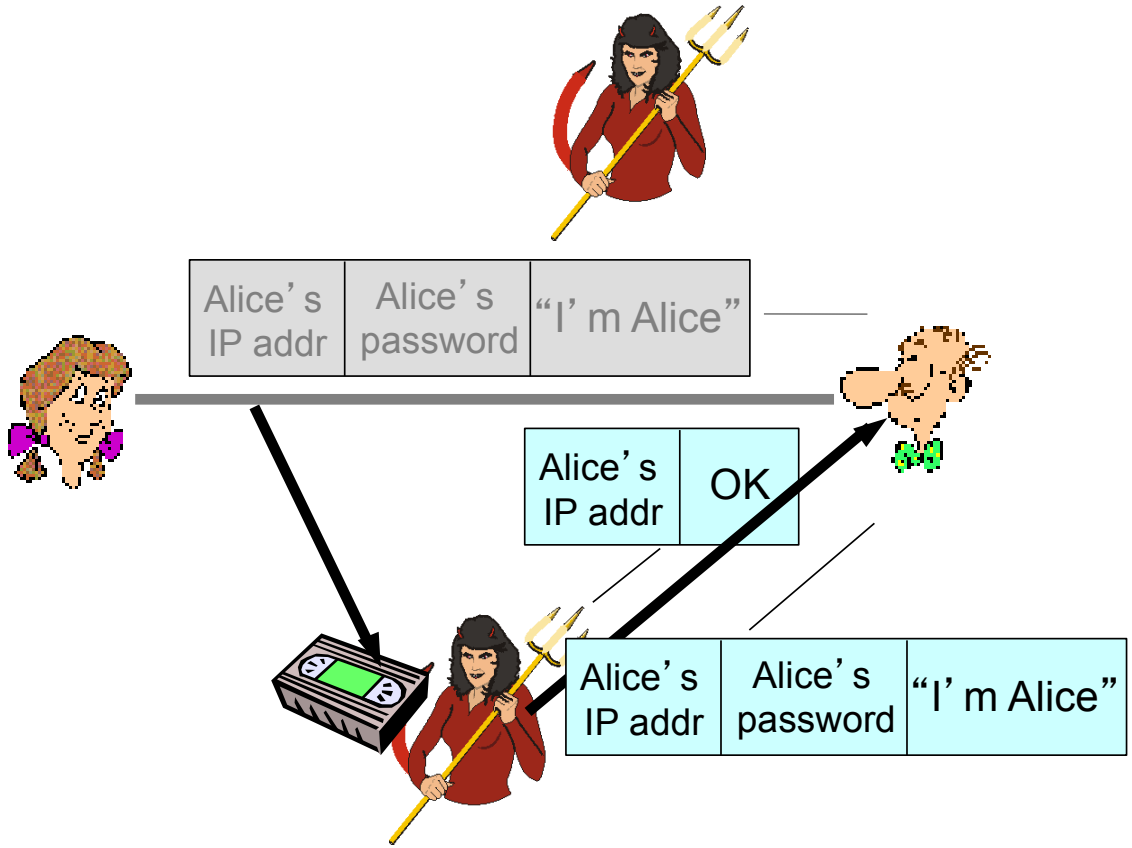


Protocolo de autenticación. Tercer intento

- Alice envía su password junto con el mensaje



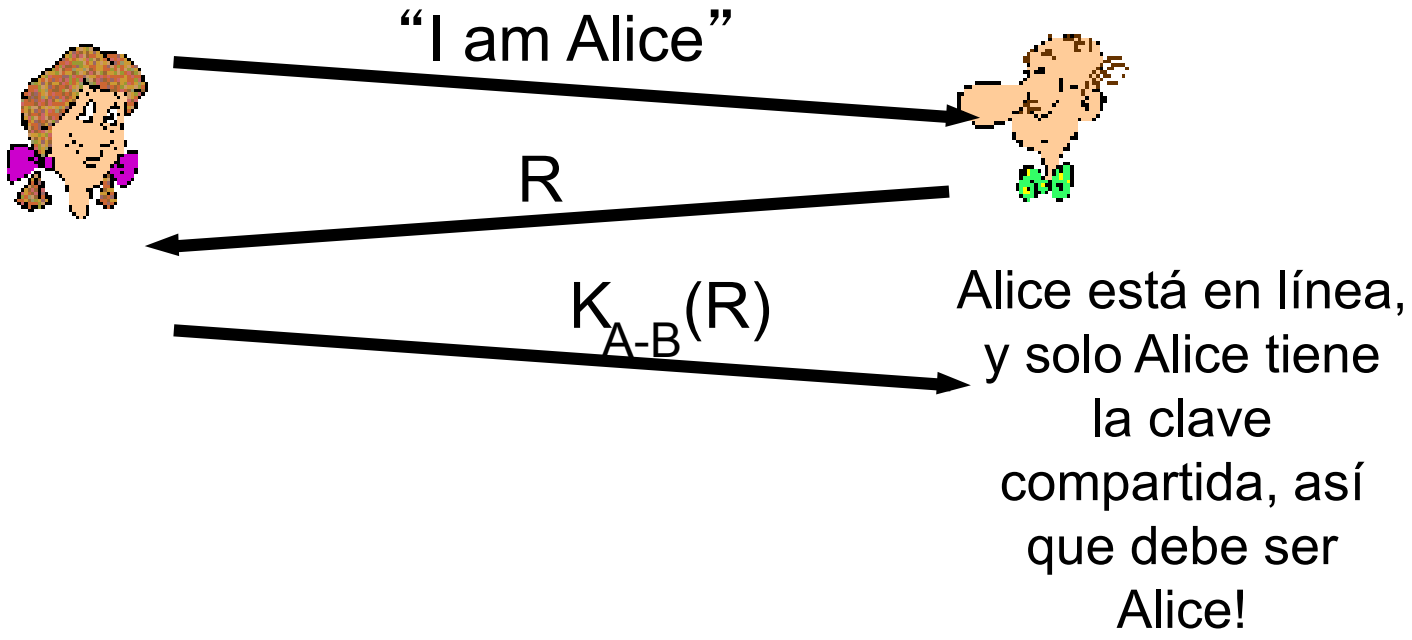
- Falla: Si Trudy puede ver el tráfico de Alice, puede obtener la password



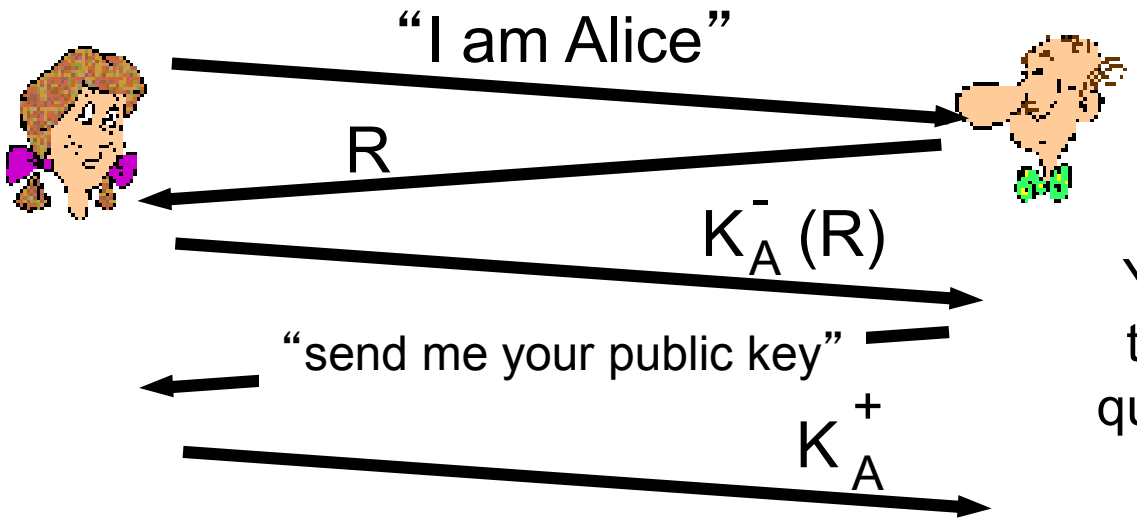
- ¿Mejora si ciframos la password?

Protocolo de autenticación. Cuarto intento

- Objetivo: evitar el replay
- Nonce: número (R) que solo se puede utilizar una vez (mientras usemos la misma clave)
- Para verificar que está hablando con Alice, Bob le envía un desafío (nonce) para que Alice lo cifre con la clave compartida.



Protocolo de autenticación. Clave pública



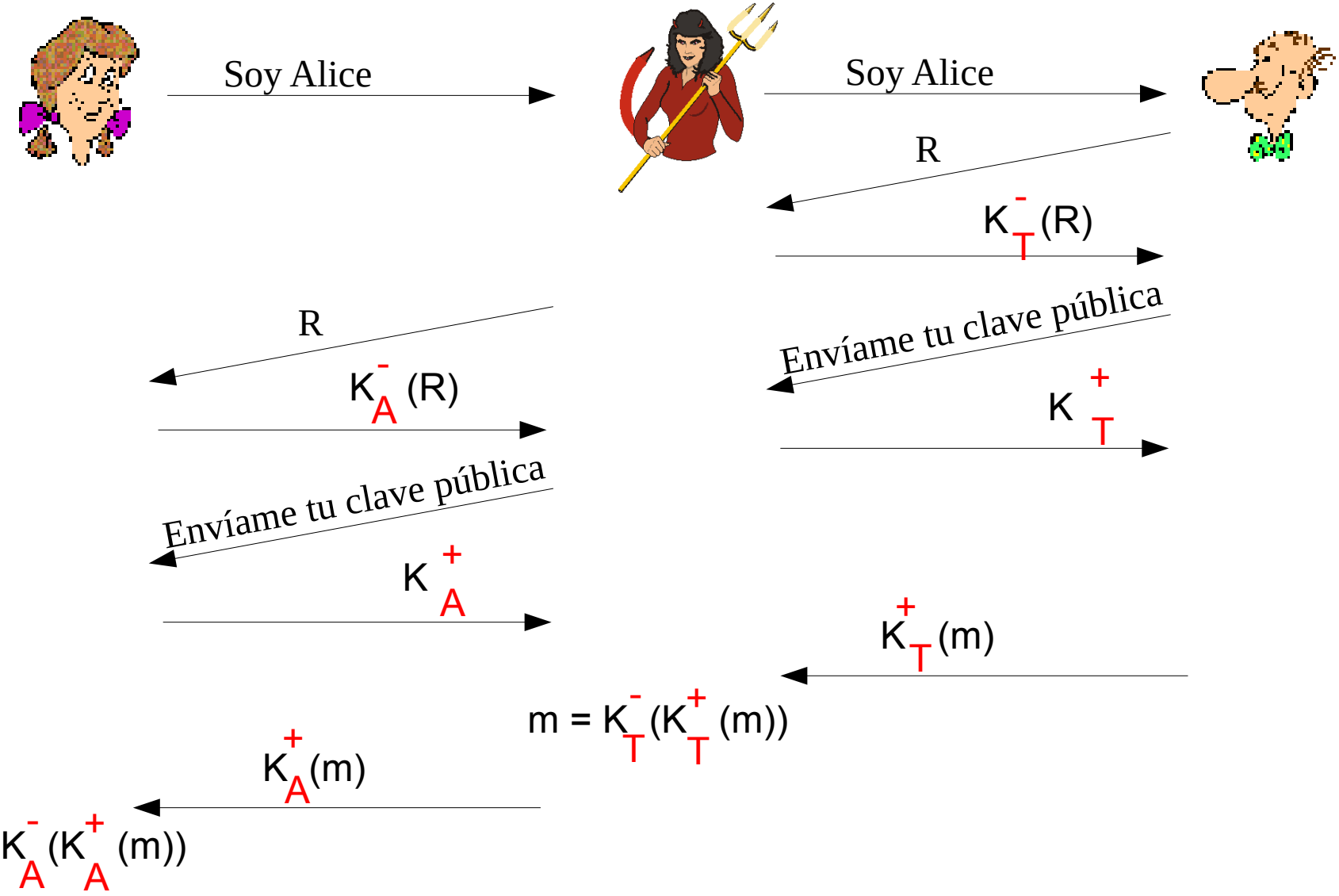
Bob calcula

$K_A^+(K_A^-(R)) = R$

Y sabe que sólo Alice tiene la clave privada que cifra correctamente R tal que

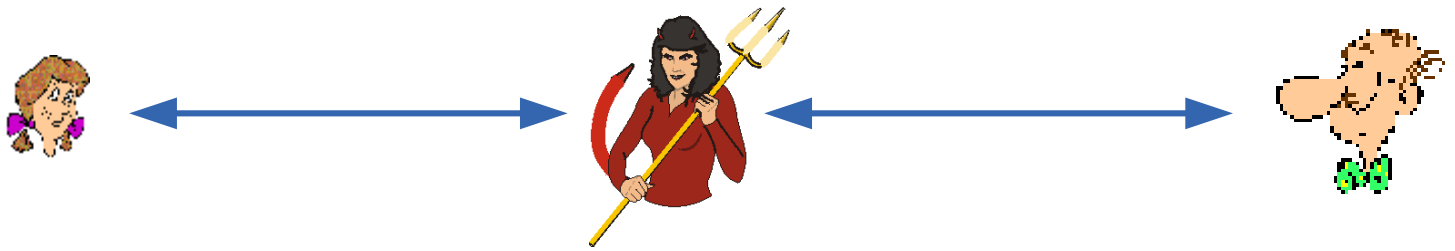
$K_A^+(K_A^-(R)) = R$

Problema del protocolo anterior: Man in the middle



Ataque “man in the middle”

- Trudy se hace pasar por la contraparte (Alice para Bob, y Bob para Alice)



- Difícil de detectar
  - Cada uno recibió los mensajes correctamente
  - Pero Trudy pudo interceptar los mensajes!!!
- **Problema: cómo estar seguro que la clave pública que tengo es la clave de Alice**
- Veremos cómo resolver este problema (certificados de clave pública)

Message digests (resumen)

- Función de hash segura
- “Huella digital” de un bloque de bits
  - De largo fijo
  - Fácil de calcular
- Se aplica la función hash H a mensajes de largo arbitrario m, obteniendo una huella digital de largo fijo H(m) (por ejemplo 256 bits)
- Claramente no es invertible (largo fijo)
- Propiedades para que una función de hash sea segura:
  - Dado un resumen x, debe ser computacionalmente inviable encontrar m tal que x=H(m)
  - Resistencia débil a las colisiones: dado M, es computacionalmente inviable encontrar M’ tal que H(M)=H(M’)
  - Resistencia fuerte a las colisiones: es computacionalmente inviable encontrar X e Y tales que H(X) = H(Y)

Algoritmos de Hash “populares”

- MD5: genera resumen de 128 bits. Obsoleto. No se recomienda usar
  - 2004-2005, se quebró la resistencia fuerte a las colisiones
- SHA-1: resumen de 160 bits. En proceso de dejarse de usar
  - 2017, colisión calculada (resistencia fuerte a las colisiones quebrada)
  - Google: “This attack required over 9,223,372,036,854,775,808 SHA1 computations. This took the equivalent processing power as 6,500 years of single-CPU computations and 110 years of single-GPU computations”
- SHA-256, SHA-384, SHA-512 mejoras, seguras por ahora
- SHA-3 (2012). Conveniente migrar a este nuevo algoritmo

Utilidad de los hashes

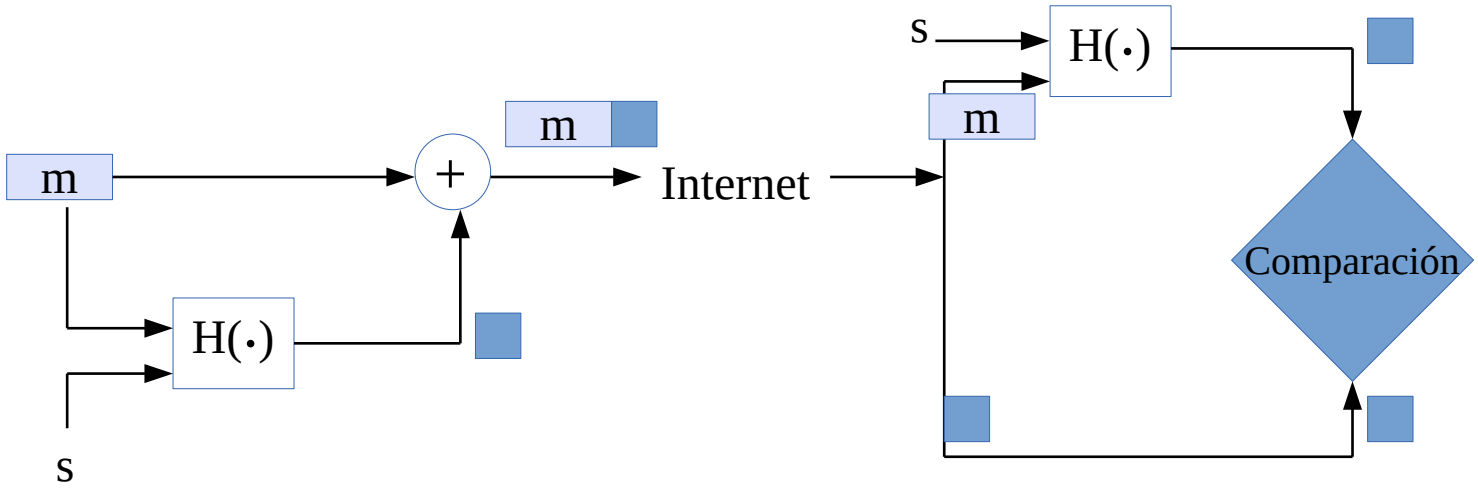
- **Función auxiliar para firmas digitales**
- Control de integridad
- Como base para funciones de MAC (Message Authentication Code)
- Sistemas de password
- Blockchain

Ejemplo (simplificado): distribución de archivo

- "A": mensaje formado por la actualización (posiblemente grande)
- H(A) : resumen de la actualización
- Se distribuye "A", y $K^-_B(H(A))$
- Cliente debe conocer la clave pública K^+_B (correspondiente a K^-_B)
- Cliente calcula H(A) y lo compara con $K^+_B(K^-_B(H(A')))$
  - Si son iguales, cliente asume que la actualización es correcta
- ¿Por qué esto garantiza la autenticidad de la actualización?
- Nuevamente surge el problema de estar seguros que la clave pública K^+_B realmente es la de B

Control de integridad: MAC (Message Authentication Code)

- Funciones similares a los hashes
- Reciben como parámetro una clave
- Ejemplo: MAC(m, s) = H(m + s) (aquí **+** significa concatenación)
  - Esta construcción tiene algunas vulnerabilidades
- Ejemplo normalizado: HMAC (RFC 2104)
- Independientemente del algoritmo de MAC utilizado, el esquema es el mismo



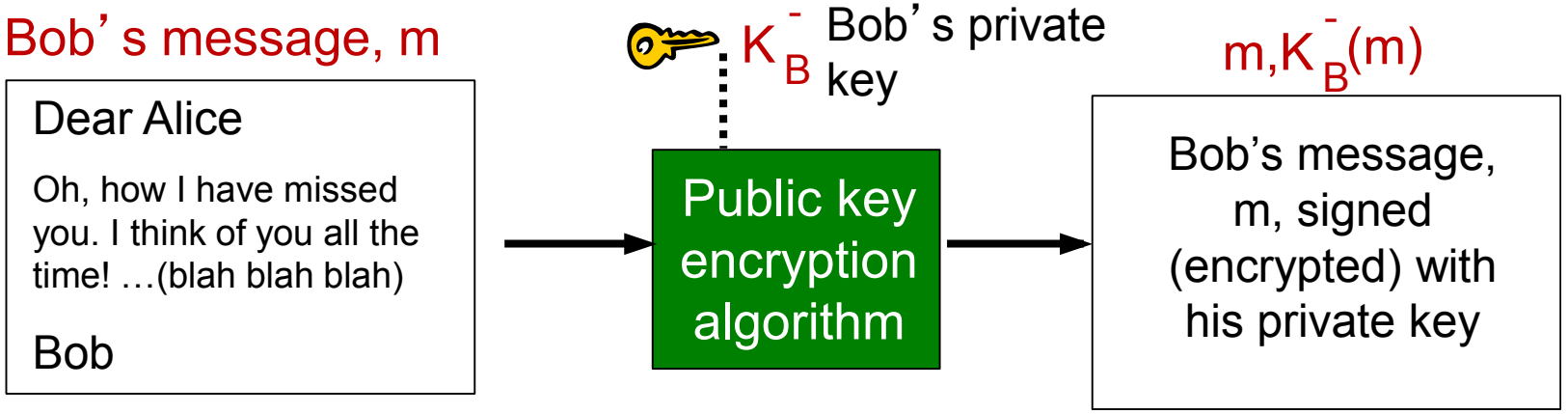
Firmas digitales

- Buscamos “algo” que pueda cumplir la función de una firma
  - Que Bob pueda firmar un mensaje, para asegurar que fue generado por el
  - Que pueda ser verificado por Alice
  - Que no pueda ser falsificable
  - Que pueda ser verificable por terceros (que Bob no pueda negar haber firmado)

- Utilizaremos algoritmos de clave pública

Firmas digitales. Primer intento

- Primer intento: Bob cifra el mensaje, m, con su clave privada



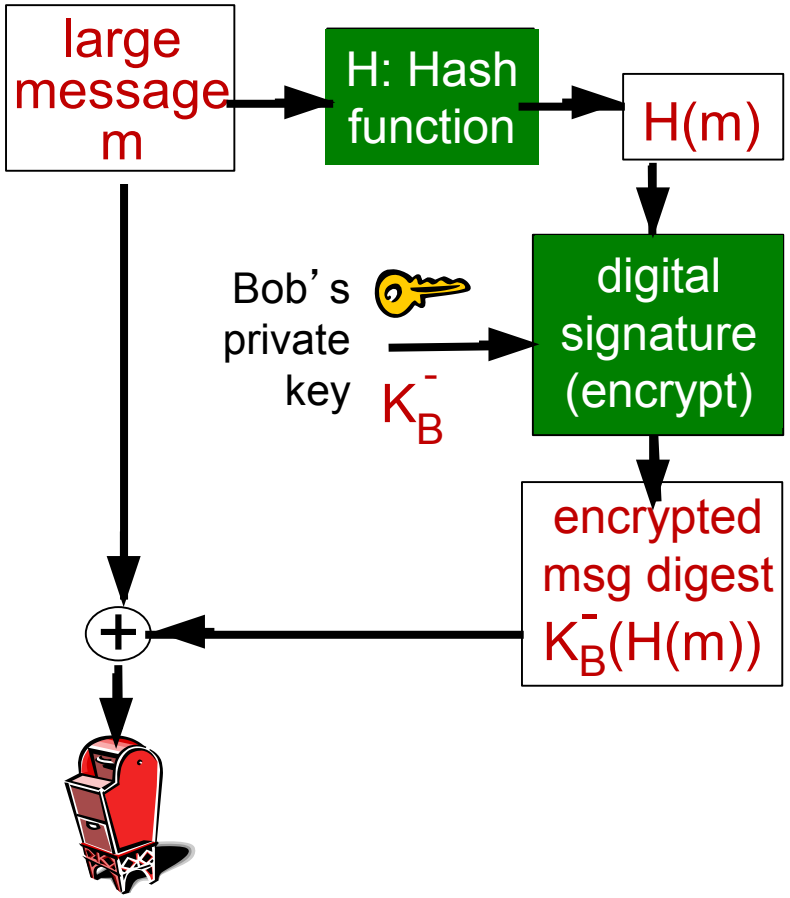
- Alice puede verificar la firma
- Problema: estamos cifrando un mensaje potencialmente grande con criptografía de clave pública
  - Veremos cómo resolver este problema en breve

Verificación de la firma

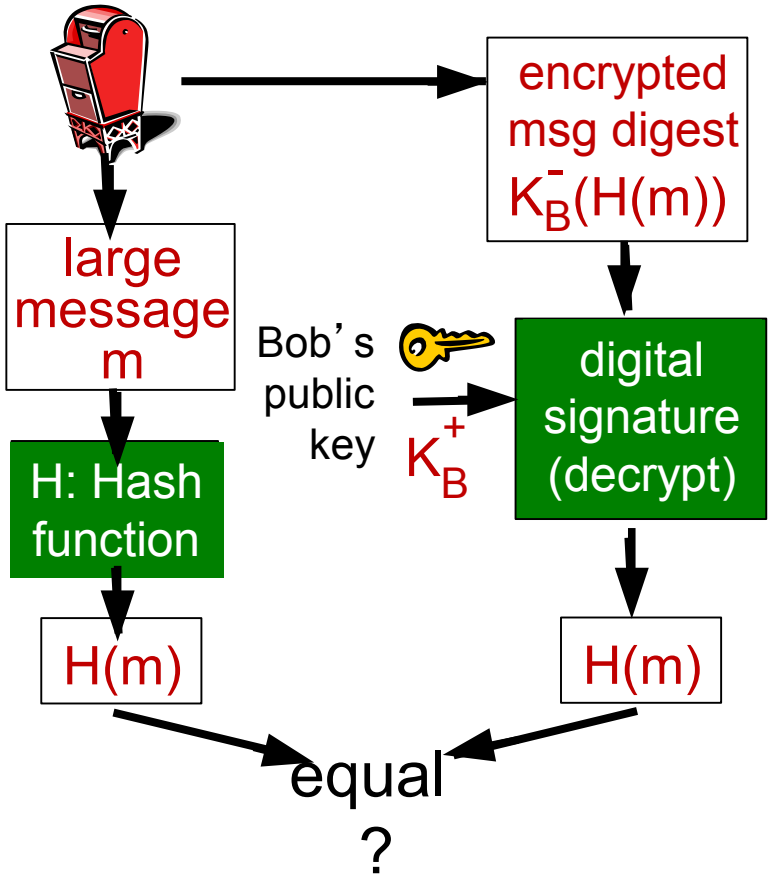
- Si Alice recibe el mensaje m y la firma $K^-_B(m)$
- Debe disponer de la clave pública de B
  - Nuevamente precisamos un mecanismo para asegurarnos que tenemos la clave pública correcta
- Si $K^+_B(K^-_B(m)) = m$, entonces quien realizó la firma debe poseer la clave privada de B
- Entonces Alice verifica que:
  - B firmó m
  - Nadie más pudo firmar m (si B no “perdió” su clave)
  - B firmó m y no m’
- No repudio:
  - Alice puede llevar m, $K^-_B(m)$ ante un juez y probar que Bob firmó m
- ¿En qué se diferencia de un algoritmo de MAC?

Firmas digitales: huella digital firmada

Bob envía mensaje firmado digitalmente:



Alice verifica la firma, y la integridad del mensaje firmado digitalmente:



Certificación de clave pública

- Vimos en más de una ocasión que precisamos estar seguros de poseer la clave pública de Bob, y no otra
  - Es un problema crítico para hacer viable los sistemas de clave pública
- Dos estrategias:
  - Autoridades de certificación, que certifican la relación entre una clave pública y una identidad
    - “Public Key Infrastructure (PKI)”
    - La más utilizada
  - PGP. Confianza basada en “reputación”

Autoridades de certificación (CA)

- Entidades “confiables”
- Encargadas de certificar la relación entre una clave pública y una identidad
- Deben validar la identidad (pedir documentación, etc.)
- Deben asegurarse que la entidad tiene acceso a la clave privada correspondiente
- La información se resume en un *Certificado de clave pública* firmado por la autoridad certificadora

Certificado de clave pública

- Documento firmado por una CA
- **Objetivo: asociar una clave pública con una identidad**
  - Identidad: Persona, IP, nombre de dominio….
- Formato: X509 (ITU) + RFCs (IETF)
- Incluye otros campos, como período de validez, usos permitidos…
- Para poder validar el certificado, debo conocer la clave pública de la autoridad certificadora
  - Por ejemplo, navegadores y sistemas operativos traen los certificados de clave pública de múltiples CAs
- Solucionamos así el problema de distribuir la clave pública
  - Puedo incluso aceptar su certificado de Bob!
  - Solucionamos el problema de man in the middle

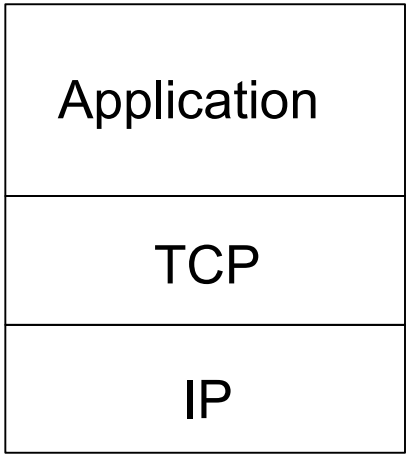
Campos comunes de un certificado

Versión	Versión de X509 en uso
Nº de serie	Nº de serie único en esa CA
Algoritmo de la firma	
Autoridad de certificación	Identidad de la CA que genera el certificado
Período de validez: not before, not after	
Nombre de entidad (DN)	Información de la entidad en formato DN
Información de la clave pública del DN	Algoritmo y parámetros de clave del DN
Clave pública del DN	
… Parámetros opcionales ….	
Firma del certificado	Firma con la clave privada de la CA

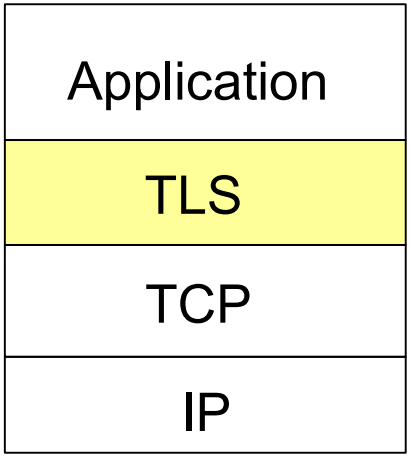
**DN: Distinguished Name. Campo donde se identifica al "sujeto" dueño de la clave pública*

Seguridad en TCP: TLS

- TLS: Transport Layer Security (anteriormente SSL, Secure Sockets Layer)
- Utilizado para navegación segura (HTTPS), pero disponible para cualquier aplicación que utilice TCP
- Provee confidencialidad (cifrado), autenticación, control de integridad, anti-replay…
- Versión actual: TLS 1.3



Aplicación normal



aplicación con TLS

Etapas de TLS

- Handshake: Alice y Bob negocian algoritmos a utilizar, intercambian certificados, se autentican, e intercambian secreto compartido
- Derivación de claves: A partir del secreto compartido, Alice y Bob derivan todas las claves necesarias para el intercambio (cifrado, integridad…)
- Transferencia de datos: Se transfiere la información de la aplicación
  - Fraccionada en “records”
- Cierre de la conexión: se da por finalizada la conexión

Cipher suites

- Cipher suite: conjunto de algoritmos a usar
  - Algoritmo de clave pública
  - Algoritmo de clave simétrica
  - Algoritmo de MAC para control de integridad
- TLS 1.2 soporta muchas combinaciones (cipher suites).
  - TLS 1.3 lo reduce a 5
- Integridad: HMAC utilizando SHA-256 o mejor
- Intercambio de claves: en TLS 1.2, RSA o Intercambio de Diffie Hellman
  - En TLS 1.3 solo Diffie Hellman
- Cliente envía lista soportada, servidor elige uno

Handshake TLS (RSA, TLS 1.2)

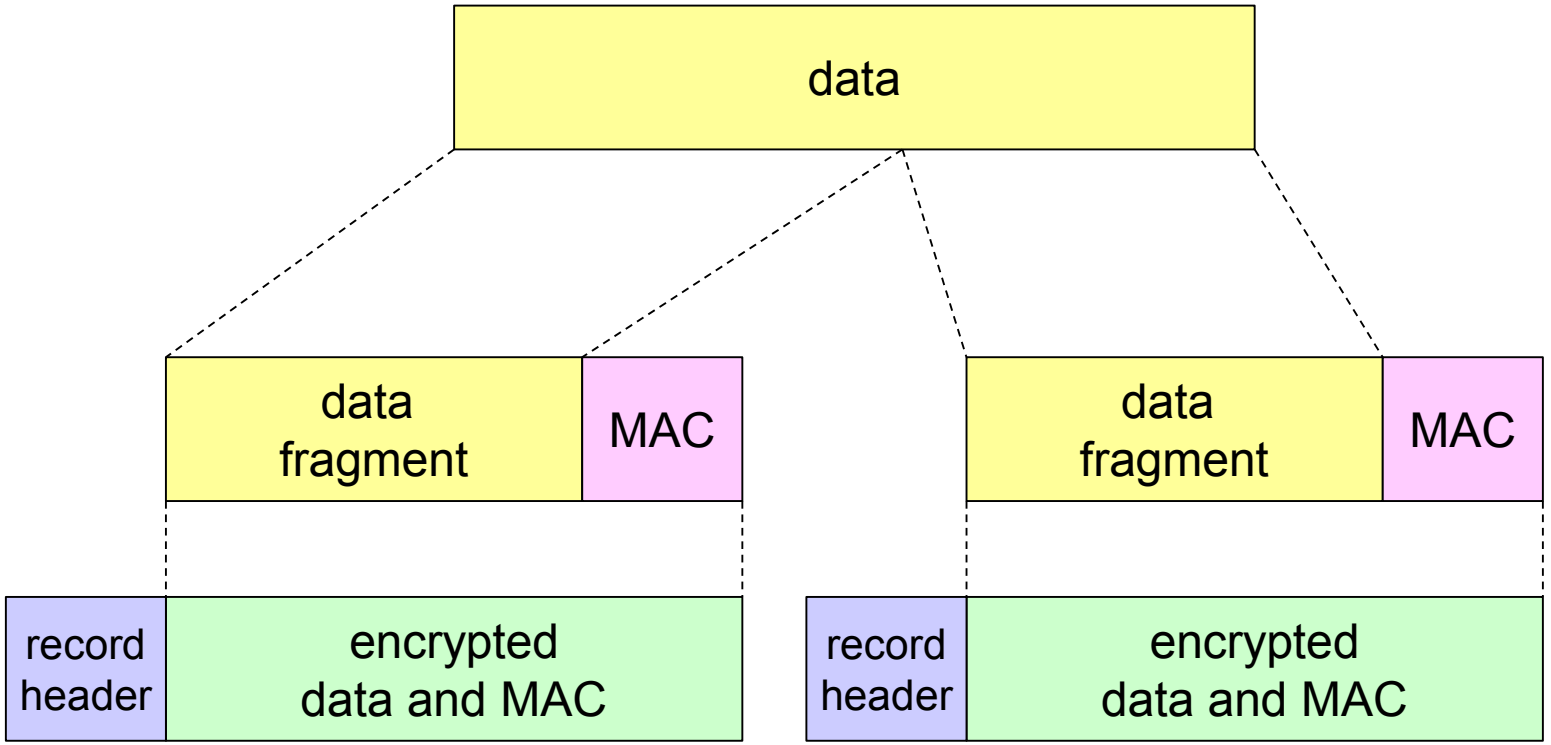
1) Cliente envía lista de algoritmos soportados (cipher suites), y un nonce
2) Servidor elige de la lista. Envía elección + certificado + server nonce
3) Cliente verifica el certificado, extrae clave pública del servidor, genera pre_master_secret, lo cifra con la clave pública y lo envía
4) Servidor recibe pre_master_secret, lo descifra con su clave privada
5) Cliente y servidor calculan las claves de cifrado y MAC a partir de pre_master_secret y los dos nonces
6) Cliente y servidor comienzan a cifrar
7) Cliente envía MAC de todos los mensajes del intercambio
8) Servidor envía MAC de todos los mensajes del intercambio

- Los puntos 7 y 8 evitan que Trudy pueda modificar los primeros mensajes que no van cifrados (por ejemplo lista de cipher suites soportados)
- nonces evitan replay

Derivación de las claves

- Conveniente tener distintas claves para distintos usos
- Se calculan a partir de: pre_master_key, client nonce, server nonce
- Se genera master secret
- A partir del master secret, se generan las claves que se precisen:
  - Clave para MAC de cliente
  - Clave para MAC de servidor
  - Clave para cifrado de cliente
  - Clave para cifrado de servidor
  - Vector de inicialización de cliente (IV)
  - Vector de inicialización de servidor (IV)

Record protocol



- Record header: tipo de contenido, versión y largo
- MAC calculado con clave de MAC correspondiente. Incluye número de secuencia
- Fragmentos: máximo 2^{14} bytes

Intercambio TLS

handshake: ClientHello

handshake: ServerHello

handshake: Certificate

handshake: ServerHelloDone

handshake: ClientKeyExchange

ChangeCipherSpec

A partir de aquí
todo cifrado

handshake: Finished

ChangeCipherSpec

handshake: Finished

application_data

application_data

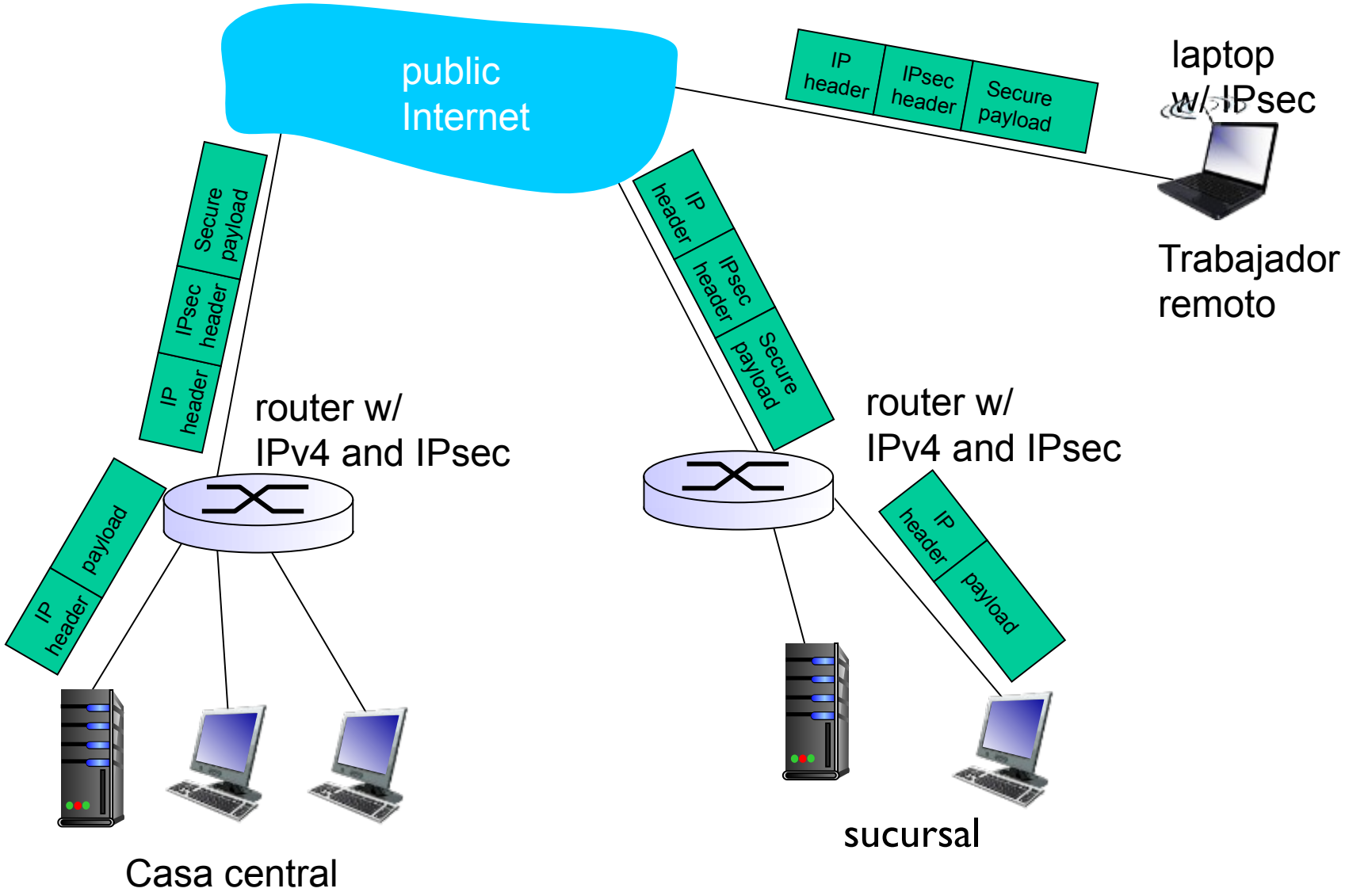
Alert: warning, close_notify

Seguridad en capa 3: IPSec

- Seguridad entre 2 entidades en capa de red
- Muy utilizado para crear VPNs (redes privadas virtuales)
  - Se interconectan locales a través de Internet
  - Utilizando IPSec para brindar confidencialidad, integridad, autenticación
- Servicios:
  - Control de Integridad
  - Confidencialidad (cifrado)
  - Autenticación mutua
  - Prevención de replay

- 2 protocolos:
  - AH: Autenticación, integridad
  - ESP: confidencialidad, autenticación, integridad
    - El utilizado habitualmente

Ejemplo de VPN



Modo transporte, modo túnel

- Modo transporte: protección del contenido en los extremos finales
  - Se mantiene el paquete original, se agrega encabezado IPSec entre IP y capas superiores
- Modo túnel: protección de todo el paquete
  - Se cifra el paquete original, y se encapsula en un nuevo paquete
  - El utilizado para realizar VPNs

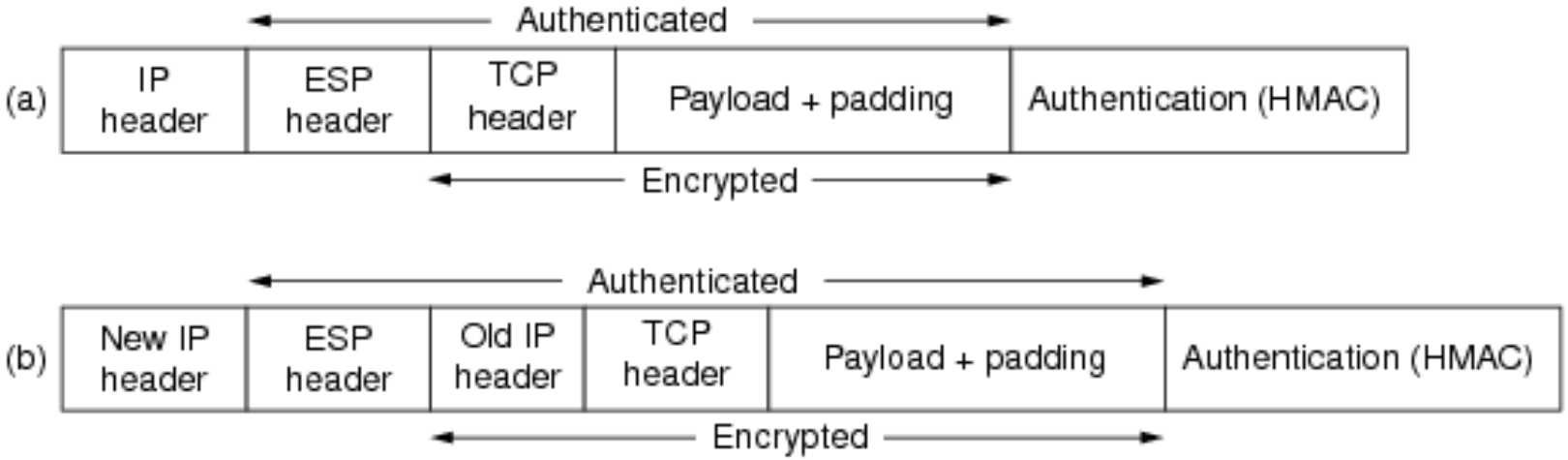
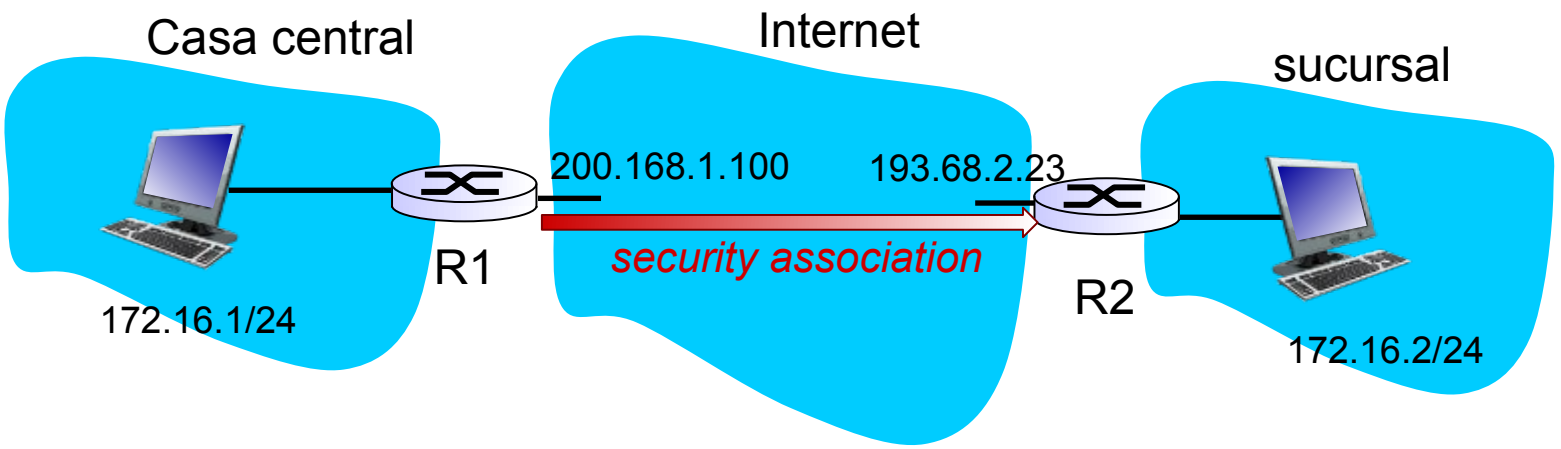


Fig. 8-28. (a) ESP in transport mode. (b) ESP in tunnel mode.

Asociaciones de seguridad (SA)

- Conexión lógica entre dos entidades
- Unidireccional
  - Precisaré 2 para tráfico bidireccional
- Mantienen estado entre los extremos de la comunicación



- En el ejemplo, R1 y R2 mantienen varios parámetros:

  Identificador de SA: 32 bits. Interfaces (IP) origen y destino

  Algoritmos para cifrado y control de integridad

  Claves para cifrado y control de integridad

Base de datos de asociaciones de seguridad y de políticas

- Security Association Database (SAD): contiene las SA activas
- Cada paquete lleva un índice (SPI) apuntando a la SA correspondiente

- Security Policy Database (SPD): Indica qué paquetes deben utilizar cada SA
  - Para cada datagrama, precisamos saber si debe utilizar IPSec
  - Si debe utilizar IPSec, con qué SA
  - Usualmente se elige en base a redes IP de origen y destino

IKE: Internet Key Exchange

- Impráctico manejar las SA manualmente
- IKE se encarga de autenticar y negociar las claves
- Autenticación:
  - Pre shared key (clave compartida).
  - PKI (Certificados y pares de claves privada/pública)
- Fases de IKE:
  - Fase 1: establecimiento de SA bidireccional de IKE (ISAKMP SA)
  - Fase 2: se usa ISAKMP para negociar de forma segura la asociación IPsec bidireccional

Protocolos y puertos de IPSec/IKE

- IKE: UDP puerto 500
- AH: IP protocolo 51
- ESP: IP protocolo 50
- Existe una opción, llamada “Nat Traversal”, para utilizar IPSec sobre UDP
  - Pensada para cuando uno de los extremos está detrás de un dispositivo que hace NAT
  - Encapsula los paquetes en un nuevo segmento UDP (puerto 4500)

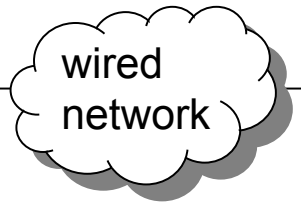
Seguridad en redes inalámbricas

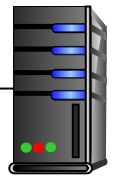
- En 802.11 original (1999): WEP (Wired Equivalent Privacy)
  - Problemas graves de seguridad
- 802.11i: provee buena seguridad
  - WPA2
  - WPA3
  - Opciones para usuarios hogareños (clave compartida) y empresariales (autenticación centralizada utilizando EAP)
  -

802.11i. Modo EAPoL

STA:
client station

AP: access point





AS:
Authentication
server

1 Discovery of
security capabilities

2 STA and AS mutually authenticate, together
generate Master Key (MK). *AP serves as "pass through"*

3 STA derives
Pairwise Master
Key (PMK)

3 AS derives
same PMK,
sends to AP

4 STA, AP use PMK to derive
Temporal Key (TK) used for message
encryption, integrity

Seguridad en redes TCP/IP

- No se tuvo en cuenta la seguridad en el diseño de la mayoría de los protocolos
- Tenemos problemas de seguridad en el propio diseño
- Además, problemas de seguridad en las implementaciones
  - Bugs en todos los sistemas operativos
  - No los estudiaremos
  - Podemos a veces protegerlos de los ataques por la red mediante dispositivos (ej. firewalls)

Capa física

- No lo veremos
- Con suficientes recursos, prácticamente cualquier capa física es pasible de ser intervenida
- Se requieren medidas en las capas superiores

Redes de área local (802.?)

- 802.11: seguridad dada por 802.11i
- 802.3: disponemos de 802.1x (Port Based Access Control) si quiero autenticar los equipos
- ARP?
  - Protocolo muy sencillo
  - A envía consulta por Broadcast pidiendo MAC correspondiente a la IP B
  - “B” responde
    - Cualquiera puede responder
  - Idea de ataque: modificar el mapeo en cache (ARP spoofing)
    - La mayoría de los sistemas aceptan respuestas a preguntas que no hicieron, o actualizan su cache ante un pedido
    - Si lo “refresco” suficientemente seguido, no hará un nuevo pedido broadcast
  - Permite ataques Man In The Middle, escuchas, Negación de servicio, etc.
  - Pocas soluciones
    - Aislar redes seguras de no seguras
    - Entradas ARP estáticas
    - Monitoreo

Capa de red en Internet

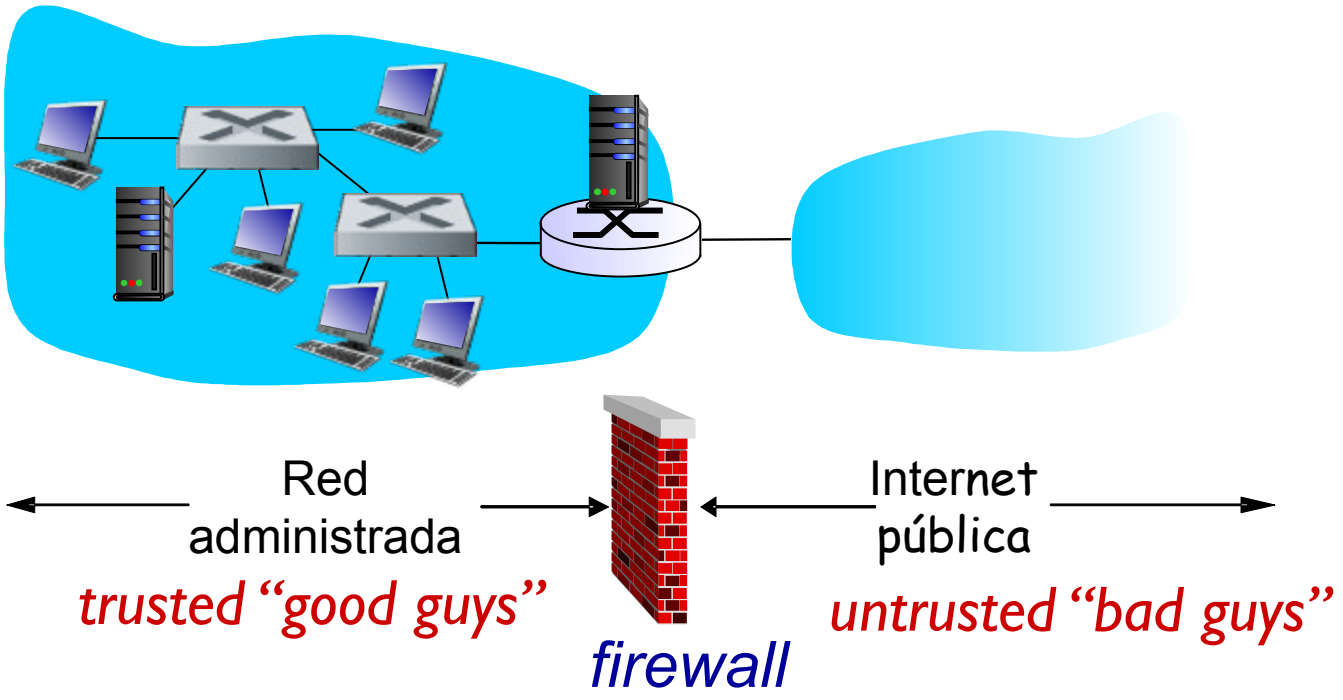
- IPv4 e Ipv6
- Muy similares en sus características de seguridad
- Servicio de datagramas. No hay garantías de entrega/duplicados/retardos. No hay garantías de origen
  - Cualquier equipo puede enviar un paquete con IP de origen arbitraria
  - Algunos proveedores filtran (BCP 38), otros no
- Salvo en ambientes muy controlados, no puede garantizarse la relación IP <-> máquina
  - Difícil rastrear ataques
- Protocolos de enrutamiento pueden ser atacados: habilitar medidas disponibles (autenticación/integridad)
- En enrutamiento externo (BGP), el mayor peligro hoy son publicaciones a través de proveedores que no filtren adecuadamente a sus clientes

Capa 4

- UDP
  - Muy fácil hacer spoofing de solicitudes
- TCP
  - Si puedo adivinar qué número de secuencia está usando otra conexión, puedo insertar paquetes
  - Se agrega aleatoriedad a la elección del número de secuencia inicial
- Negación de servicio: consumir recursos del servidor
  - Muchas conexiones a medio establecer
  - Muchas conexiones
  - Muchos pedidos a servidor DNS

Firewalls

- Dispositivo que permite filtrar el tráfico entre distintas redes
  - Por ejemplo, entre mi organización e Internet
  - Decidiendo qué tráfico dejo pasar y cual no



Tipos de firewall

- Sin estado: filtra paquete a paquete, basado en los encabezados de capas 3 y 4
- Con estado: mantiene el status de cada conexión TCP/flujo UDP y solo permite secuencias “lógicas”
  - Por ejemplo, “solo permitir conexiones destinadas al puerto 80, salientes de mi red”
    - Si no vi salir el SYN, no voy a permitir paquetes entrantes
- Firewalls “capa 7”
  - Agregan al estado la inspección del protocolo de capa de aplicación
  - Solo para determinados protocolos
- Gateways de aplicación
  - Participan del protocolo de aplicación (ejemplo proxy HTTP)
  - Específicos para aplicaciones determinadas

Sistemas de detección/prevención de intrusos

- IDS: Intrusion Detection System: genera alarmas
- IPS: Intrusion Prevention System: bloquea tráfico potencialmente malicioso
- Inspecciona el tráfico en capa de aplicación (deep packet inspection)
- Puede detectar tráfico malicioso, sospechoso mediante firmas o análisis de tráfico buscando anomalías
- Puede correlacionar tráfico de distintos flujos (por ejemplo para detectar port-scans)
- Puede estar integrado con otros dispositivos