Infraestructura de Clave Pública (PKI) y Criptografía en la práctica

Alejandro Hevia
ahevia@dcc.uchile.cl

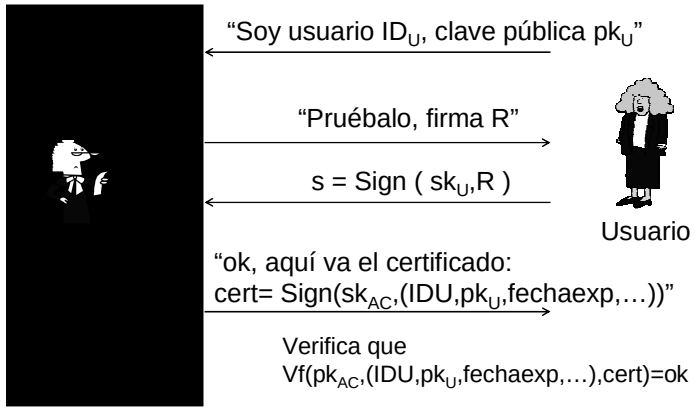
De quien es ésta clave pública?

- No hay garantias de confidencialidad y autenticidad sin certeza de usar la clave publica correcta
  - Ej. Ataque vía suplantación de clave
- Posible solución: PGP web of trust
  - Alicia recibe una clave de otra persona (Roberto?)
  - Si Alicia "cree" que viene de Roberto, la firma y distribuye la clave firmada a otros
  - Proceso se repite varias veces

PKI

- Public-Key Infrastructure o Infrastructura de Clave Pública
  - Entidades y algoritmos que permiten implementar directorios de claves públicas (identidades y atributos)
- Certificado: documento que "enlaza" una clave pública con una identidad y otros datos
  - Otros datos: fecha expiración, uso de la clave, etc.
  - Certificado = documento (identidad + clave publica) firmado por *autoridad certificadora*, (AC)
  - Supone AC es confiable

Ejemplo de certificación



Operación de PKI / certificados

- Distribución de certificados
  - Pull model, push model, on-request
- Y si se pierde/roban la clave privada?
  - *Revocación* de la clave
  - AC emite periódicamente una "lista de revocación de certificados" (o *certificate revocation list*, CRL)
- ACs pueden certificarse entre ellos
  - Relacionadas en forma jerárquica, dominios separados, certificación cruzada
- Certificado de atributos
  - Enlaza ID con "autorizaciones" asignadas al usuario (por ej. "ID está en grupo A", "ID puede hacer contratos")

Certificados X.509

- Estándar para certificados
- Desarrollado por la International Telecommunications Union (ITU)

(Detalles en auxiliar)

Referencias

- Internet X.509 Public Key Infrastructure – Certificate Management Protocol (CMP), Internet draft.

  http://www.ietf.org/internet-drafts/draft-ietf-pkix-rfc2510bis-09.txt
- C.Elison and B.Schneier "Ten Risks of PKI"

  http://www.schneier.com/paper-pki.html

Criptografía en la práctica

Muchos errores usuales

- No usar un buen algoritmo
  - Red de cajeros usaba cifrador de bloque "CHURN" con tamaño de bloque de 4 bits y clave de 8 bits (!)
  - Común al crear "mi propio cifrador"
- Usar esquemas sin demostraciones de seguridad (o con clara evidencia de inseguridad)
  - Modo ECB y RSA plano todavía son comunes

Criptografía en la práctica

- No usar la herramienta correcta
  - Creer que encriptacion tambien entrega autenticidad
  - Ej: primeras versones de SSH, IPSec y WEP no usaban esquemas de autentificacion de mensajes
    - Se encontraron ataques
- No implementar "exactamente" el esquema indicado
  - Incluso un pequeño cambio puede tornar un *esquema con demostración de seguridad* en esquema totalmente inseguro
  - Ej: MS Word y Excel implementaban CBC$ pero no elegían un valor R al azar cada vez

Implementaciones correctas?

- Generación de números pseudo-aleatorios
  - Cómo implemento $K \leftarrow^R \{0,1\}^k$?
  - Crucial en todo esquema criptográfico (claves, Ivs, etc)
  - Algoritmo que hace esto es conocido como *Pseudorandom number generator* (PRG o PRNG)

Solución 1

Usar generador provisto en biblioteca de C

```
procedure srand(seed) // seed es de 32 bits
    state = seed;

function rand( )
    state = ((state * 1103515245) + 12345) mod
    2147483648;
    return state;
```

2^{31}

Problemas c/sol. 1

Pudieramos implementar $K \leftarrow^R \{0,1\}^{128}$ como

Pero entonces

key[1] = ((key[0] * 1103515245) + 12345) mod 2^{31}

key[2] = (((key[0] * 1103515245) + 12345)* 1103515245) + 12345) mod 2^{31}

Key[3] = (((((key[0] * 1103515245) + 12345)* 1103515245) + 12345)* 1103515245) + 12345) mod 2^{31}

Y hay sólo 2^{32} posibilidades para la clave!

Caso del PRG de netscape

Problema: NSseed original era adivinable

Un buen PRG?

- Cifrador de bloque en modo CTRC

  Procedure secureSrand(seed)

  Dividir seed en (ctr || K); guardar ctr y K;

  Function secureRand( )

  return E_K(ctr++);

- Otras posibilidades: Variantes
  - FIPS 186 estándar (~ SHA1 en variante de modo CTR)
  - ANSI X9.17 estándar (~ DES en variante de modo CBC)
  - Blum-Blum-Shub PRG (basado en factorización)

Cripto en la práctica

- La mayor parte de los problemas no provienen de la parte criptográfica
  - Buenas noticias: Cripto funciona!
  - Malas noticias: gente comete errores, cripto no los resuelve
- Ejemplos
  - Instalación y administración de sistemas para cajeros automáticos
  - Protocolo WEP para 802.11b

Problemas

- Fallas de seguridad NO se dan a conocer
  - Gobierno: por “seguridad nacional” (terrorismo, etc)
  - Militares: idem pero aún más
  - Compañías privadas: secreto por vergüenza, efecto en precio de las acciones, posibles demandas
- Artículo en la seguridad de la industria bancaria en UK en los 90’s (por Ross Anderson)
  - Consultor en juicio a los bancos por clientes acusados de fraude (1992)

Estudio de Anderson

Leyes en UK permitían a bancos negar problemas y fomentar negligencia

- Ejemplos:
  - Adolescente en Ashton condenada por robarle a papá, obligada a declararse culpable → luego se determino que el error fue del banco
  - Sargento de policía de Sheffield acusado de robo y suspendido → error del banco
  - Terminaron en demanda en 1992

Fuentes del fraude en cajeros

- Fraude causado por empleados
  - PINs entregados vía oficinas de bancos, empleados podían saber PINs de clientes
  - Caso de ingeniero que modificó cajero para grabar números de cuentas y PINs
  - Un banco daba tarjetas "maestras" para empleados
- Fraude externo
  - Números de cuentas en recibos, observar PIN
  - Cajeros falsos, graban PIN
  - Robo de correo con tarjetas, info privada de cuentas

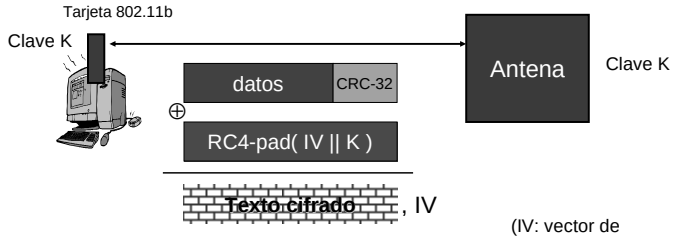
Más fraude en cajeros

- Ataques de fuerza bruta en tarjetas robadas
  - Explota sugerencia de cómo escribir PINs para recordarlos
  - Error de programación: todos los clientes recibieron mismo PIN
  - Banco guardaba PIN encriptados en archivo
    - Programador podía encontrar su propio PIN encriptado, y buscar otras cuentas con mismo PIN
  - Banco guardaba PIN encriptadas en banda magnetica
    - Se podía cambiar número de cuenta, dejar el mismo PIN; permitía girar dinero de la cuenta!

(Ataques más sofisticados en artículo)

Ataque a WEP

- WEP: protocolo para "asegurar" conecciones inalámbricas (802.11b)
  - Usa clave de 40 bits o de 128 bits usando RC4



Errores en el diseño de WEP

- CRC-32 $\Rightarrow$ no garantiza integridad del paquete
  - CRC-32 es lineal
  - Adversario puede modificar paquetes en tránsito, por ej. inyectar "rm –rf *"
  - Debería usar un MAC para integridad!
- Agregar IV al comienzo no era suficiente
  - Fluhrer-Mantin-Shamir: RC4 es inseguro si IV es prefijo de la clave
    - Teniendo 106 paquetes se puede extraer la clave
    - Implementado por Stubblefield: AirSnort, WEPCrack, …
  - Implementación corregida?
    - clavePaquete = SHA-1(IV || K)
    - usar IV más largo y aleatorio

Referencias

- Y. Kohno, "Implementation Pitfalls" http://www.cse.ucsd.edu/~mihir/cse107/yoshi.pdf
- R. Anderson, "Why cryptosystems fail" http://www.cl.cam.ac.uk/~rja14/wcf.html
- S.R. Fluhrer, I. Mantin, A. Shamir, "Weaknesses in the key scheduling algorithm of RC4" http://downloads.securityfocus.com/library/rc4_ksaproc.pdf
- A.Desai, A.Hevia, Y.L.Yin, "A Practice Oriented Treatment of Pseudorandom Number Generators" http://www.cse.ucsd.edu/~ahevia/publications/dhy-ec02-abs.html

CC51D Seguridad de Datos