VCR Y PEO
DOS PROTOCOLOS CRIPTOGRÁFICOS SIMPLES

Ariel Futoransky
CORE, Seguridad de la información
FCEyN, Universidad de Buenos Aires
e-mail: ariel_futoransky@core-sdi.com

Emiliano Kargieman
CORE, Seguridad de la información
FCEyN, Universidad de Buenos Aires
e-mail: emiliano_kargieman@core-sdi.com

Keywords: Criptografía, Seguridad, VCR, PEO.
Authentication-Key: SKoZjEPn3cOUg

ABSTRACT

Este trabajo tiene por objetivo presentar dos protocolos criptográficos originales con importantes aplicaciones en el campo de seguridad de redes, sistemas y auditoría informática.

El primero, PEO, basado en funciones unidireccionales (one way hash functions), define un método que permite garantizar el almacenamiento de datos en forma incremental, aún cuando el medio no es seguro. El segundo, VCR, basado en criptografía de clave simétrica reproduce la funcionalidad del primero y asegura la confidencialidad de los datos.

Se mencionan aplicaciones de los protocolos para mejorar la auditoría de base de datos relacionales, para mejorar el esquema de seguridad en redes de comunicaciones y para auditar procedimientos de administración de computadoras.

INTRODUCCIÓN

Es corriente la creencia de que la documentación almacenada en forma digital es más fácil de adulterar que su equivalente impresa en papel.

Si bien es cierto que la tecnología brinda las facilidades para modificar de manera indiscriminada la información almacenada digitalmente, el uso de las herramientas adecuadas pueden hacer del medio digital un sistema mucho más seguro que el papel. Algunos ejemplos son la criptografía de clave pública, los sistemas de firma digital, etc.

Sin embargo es necesario ser muy cuidadoso al elegir las herramientas a integrar en cada desarrollo. El solo hecho de incluir criptografía no soluciona todos los problemas de seguridad.

Los protocolos introducidos en este trabajo están orientados a determinar si la información ingresada al sistema antes de que se produzca una determinada intrusión, ha sido adulterada.

Es decir, dado un sistema que incorpora registros en forma periódica a su base de datos, y una persona que adquiere acceso en un instante determinado, un ente auditor puede determinar si los datos ingresados al sistema antes de que se produzca el acceso en cuestión han sido modificados.

Los protocolos no garantizan lo que ocurra con la información en forma posterior al acceso, ya que el intruso cuenta con herramientas para modificar todas las entradas y salidas del sistema a partir de ese momento.

La seguridad de estos protocolos esta basada en el hecho de que el estado del sistema en el instante en que se da de alta un registro no puede ser reproducido en base a estados posteriores.

La complejidad de un ataque a este sistema depende directamente de la complejidad de los algoritmos utilizados. Una implementación de PEO utilizando MD5, considerando que la mejor manera de atacarlo es por fuerza bruta, implicaría la prueba de 2^{128} posibles estados, impensable con la tecnología informática usada actualmente.

Estos protocolos han sido desarrollados por los autores de este trabajo, durante el año 1995, mientras trabajaban en el Grupo de Investigación en Seguridad Informática, Departamento de Proyectos Especiales, Dirección General Impositiva.

DESCRIPCIÓN

A continuación se especifican los protocolos PEO y VCR, desde el punto de vista teórico.

Las partes intervinientes son:

- Auditor: el que audita la autenticidad de la información almacenada.
- Fuente: el que genera la información a almacenarse.
- Sistema: el que almacena la información.

En ambos protocolos la seguridad depende en gran medida de la confidencialidad del estado o clave inicial (que llamaremos K_0), que debe ser un conjunto de bits generados al azar.

El instante en que K_0 es generada por el auditor será el instante 0, los instantes intermedios serán nombrados genéricamente con la letra i y el momento en que el auditor procede a verificar los datos almacenados será nombrado con la letra n. De esta manera los instantes quedan ordenados $0 < i <= n$.

Los estados intermedios (K_i) son desechados por el sistema[1], que conserva únicamente el último estado (K_n). K_0 solo permanece en poder del auditor.

Durante el proceso de verificación el auditor regenera a partir de K_0 y los datos ingresados al sistema todos los K_i.

Es importante que K_i, para cada uno de los instantes se encuentre almacenada en forma segura, que no tenga acceso directo desde la fuente ni de cualquier posible atacante. De todas formas el hecho de que un intruso adquiera acceso a Ki no compromete directamente la seguridad de los datos D_j ($j <= i$)

La función unidireccional (one-way hash function) H(K,D) procesa los datos (D) con el estado inicial (K) para obtener un *digest* (resumen). La realimentación se produce al tomar como estado inicial el digest de la iteración anterior. Un ejemplo de one-way hash function es MD5.

$E_K(D)$ es una función de encriptado simétrico (de bloque) que encripta el mensaje D utilizando para ello la clave K. $E^{-1}_K(C)$ es la inversa de E_K, que desencripta el mensaje encriptado C utilizando la clave K. Ejemplos de algoritmos de criptografía simétrica son DES, IDEA, RC4[2].

Protocolo PEO (Primer Estado Oculto)

Instante	Acción	Descripción
0	INICIALIZACIÓN $K_0 = Random()$	El auditor genera K_0 en forma aleatoria y lo almacena en un lugar seguro.
i	ALTA $K_i = H(K_{i-1}, D_i)$	La fuente genera D_i, el sistema lo almacena y computa K_i utilizando para ello D_i y K_{i-1}. K_{i-1} es destruido.
n	VERIFICACIÓN $V_0 = K_0$ $V_j = H(V_{j-1}, D_j)$ $V_n == K_n$?	El auditor verifica K_n computando V en función de K_0 y los D.

1. Es necesario poner especial énfasis en asegurarse que los K_i sean borrados en forma segura, en cualquier medio en el que se los almacene. [Sch1994]

2. Una variante interesante se puede lograr utilizando una función de encriptado de clave pública en lugar de E_K, donde la clave privada la conserva en secreto el auditor. Esta variante mejora algunos aspectos de la seguridad del protocolo considerados en la sección análisis, sin embargo las implementaciones de este tipo de funciones son por lo general sensiblemente menos eficientes que las de funciones especificadas por el protocolo.

El protocolo VCR (Vector de Claves Remontante)

Instante	Acción	Descripción
0	INICIALIZACIÓN $K_0 = Random()$	El auditor genera K_0 en forma aleatoria y lo almacena en un lugar seguro.
i	ALTA $K_i = H(K_{i-1}, D_i)$ $C_i = E_{Ki-1}(D_i)$	La fuente genera D_i. El sistema computa K_i a partir de K_{i-1} y D_i, calcula C_i (encriptando D_i con K_{i-1}) y lo almacena. K_{i-1} y D_i son destruidos.
n	LECTURA y VERIFICACIÓN $V_0 = K_0$ $D_j = E^{-1}_{Vj-1}(C_j)$ $V_j = H(V_{j-1}, D_j)$ $V_n = = K_n$?	El auditor reconstruye D_j en función de C_j y K_0, y verifica K_n.

En el caso en que el resultado de H sea de dimensiones menores que la clave de E_K, será necesario tomar las precauciones necesarias durante la implementación para que esta particularidad no reduzca el espacio de clave de E_K.

ANÁLISIS

A continuación analizaremos algunas de las características de los protocolos basándonos en posibles ataques a los mismos y a algunas de las implementaciones.

Consideremos una implementación de PEO, donde un intruso gana acceso al sistema en el instante i, con intenciones de modificar el registro D_i por el registro M.

El intruso solo cuenta con la siguiente información: K_i y D_1 a D_i, no cuenta con K_0 a K_{i-1} ya que fueron destruidos específicamente por el sistema. Si el intruso pretende que el auditor no pueda notar el cambio, este debe calcular un nuevo K_i valido, o bien insertar o modificar algún registro para que el actual K_i sea valido. K_i es valido si es igual a $H(K_{i-1}, D_i)$.

Si H es criptograficamente fuerte, no hay forma de calcular K_{i-1} en función de la información disponible en tiempo razonable utilizando recursos razonables; como tampoco es factible encontrar un conjunto de modificaciones sobre los registros, con idéntico K_i. [Sch1994]

Si la intención del atacante fuese modificar otro registro la complejidad del ataque seria al menos igual al caso estudiado.

Si consideramos un caso análogo utilizando el protocolo VCR, podemos observar que el atacante cuenta a lo sumo con la misma información que en caso anterior. El sistema almacena C_1 a C_i en lugar de D_1 a D_i, manteniendo la misma sucesión K_j $(0<j<=i)$.

Si el intruso pretende modificar un determinado registro, suponiendo que no conozca D_j $(0<j<=i)$ (lo cual parece lógico entendiendo que la razón de utilizar VCR en lugar de PEO es con vistas a mantener D_j confidencial), la única información con la que contara es K_i. Si E y E^{-1} son funciones criptográficamente fuertes, el atacante no puede obtener ninguna información sobre las K_j o los D_j a partir de los C_j. Un choosen-plaintext atack [Sch1994] sobre E resulta imposible teniendo en cuenta que la clave de encripción K_j cambia con cada alta y al mismo tiempo existe un feedback con los D_j.

Llegado este momento afirmamos que la única manera de modificar los registros anteriores y poder pasar la verificación es "remontando" las claves a partir de K_0.

De todas maneras, debe quedar claro que si el atacante llega a conocer una determinada K_i no se puede asegurar por estos protocolos la autenticidad de los registros generados con posterioridad al ataque. En este caso el atacante será capaz de generar una nueva secuencia de D_j (C_j respectivamente) y K_j validas. Sin embargo si el atacante tuviera acceso a modificar el sistema podría modificar los datos de entrada antes de cualquier computación sobre ellos, esto nos dice que no es posible protegernos contra este tipo de ataque y no tiene sentido resguardar K_i mas allá del nivel de seguridad del propio sistema.

Es, como siempre, estrictamente necesario contemplar el aspecto de la seguridad al desarrollar una implementación de estos protocolos[3]. Entre los factores a tener en cuenta se encuentran: el tamaño de las K_j, el generador de azar con el que se computa K_0, la manera de guardar K_n, el método utilizado para desechar los K_j, etc[4].

IMPLEMENTACIÓN

A mediados del año 1995 fue desarrollada para la D.G.I. una aplicación que hace uso del protocolo VCR para mantener pistas de auditoría. La implementación asegura la confidencialidad y autenticidad de transacciones de administración en la red interna (TCP/IP rs/6000 AIX). Una idea mas acabada sobre el funcionamiento de esta aplicación se desprende de la lectura de "VCR, auditando a root" en el capitulo de "Aplicaciones".

La elección de las funciones para una implementación depende profundamente del tipo de aplicación al que se incorporen: es necesario considerar tanto los requerimientos de performance (dependientes directamente de la velocidad de los datos de entrada en aplicaciones en tiempo real), como los tamaños de los registros a procesar (Por una cuestión de simplicidad, los protocolos están siendo estudiados con un tamaño de registro fijo, lo cual no implica que no puedan ser implementados con tamaño de registro variable).

---

3. Hasta el one-time pad más largo es inseguro en manos de un programador inexperto.

4. Para ampliar estos temas ver [Sch1994].

La decisión sobre el tamaño del resultado de la función H dependerá en primera instancia de los algoritmos elegidos para la implementación, pero debe ser tal que no permita realizar un brute-force attack[5] para encontrar K_0.

Paralelamente es necesario considerar la disponibilidad de buenos algoritmos de criptografía, cabe destacar que los algoritmos patentados dentro de los Estados Unidos están sujetos a restricciones de exportación por ser considerados secreto militar. Para un seguimiento actualizado de este tema consultar [EFF].

Es importante concluir que no es necesario imponer ningún tipo de restricción especial a las funciones por el hecho de formar parte de estos protocolos.

A modo de ejemplo presentamos una implementación de VCR, en código C utilizando IDEA como función E y MD5 como función H.

```
.....
        IDEAKEY K;

        void
        agrega_registro(r)
                struct registro *r;
        {
                struct registro aux;

                ideacipher(r,&aux,sizeof(struct registro),K);
                write (ciphertext_fd,&aux,sizeof(struct registro));
                md5hash(K,r,K);
        }
        int
        verifica_todo(K0)
                IDEAKEY *K0;
        {
                IDEAKEY V;
                struct registro aux,r;

                memcpy(&V,K0,sizeof(IDEAKEY));
                while( (read(entrada_fd,&aux,sizeof (struct                registro))>0)
                {
                        ideadecipher(&aux,&r,sizeof(struct registro),V);
                        write(plaintext_fd,&r,sizeof(struct registro));
                        md5hash(V,&r,V);
                }
                return (memcmp(&V,K0,sizeof(IDEAKEY)));
        }
.....
```

Se trata de implementaciones muy simplificadas, destinadas a ejemplificar el esquema general de funcionamiento de los protocolos. La elección de IDEA como función de encriptado simétrica, y MD5 como función de hash redundan en beneficio de la simpleza de este ejemplo, ya que el tamaño de clave de IDEA y el del resultado generado por MD5 coinciden (ambos son de 128 bits).

---

5. Un ataque por fuerza bruta (Brute-force attack) consiste en probar todas las posibles soluciones sin ningún tipo de heurística que guié la búsqueda. [Sch1994] [GJ1979]

El costo en performance proveniente de la implementación de estos protocolos depende en principio de la elección de los algoritmos, pero es estimativamente bajo. Como ejemplo una implementación de IDEA en una computadora 386 de 33MHZ encripta a una velocidad cercana a los 880 kbps. PEO, al hacer uso únicamente de una función de hash por iteración, resulta particularmente apto para aplicaciones en tiempo real.

APLICACIONES

A continuación se sugieren tres aplicaciones en las que los protocolos PEO y VCR contribuyen con una importante mejora en el aspecto de la seguridad.

PEO como complemento de Bases de Datos:

Incorporar este protocolo junto al motor de base de datos utilizado normalmente, puede elevar la confiabilidad de las aplicaciones.

Una implementación simple y útil sería mantener un estado PEO por cada tabla en la que no deban realizarse bajas ni modificaciones. De esta forma un ente auditor puede determinar si el tipo de acceso a las tablas fue respetado. Ejemplos de esto pueden ser: base de datos de facturación, control de personal, etc.

PEO como complemento de syslog:

El sistema syslog de uso común en ambientes UNIX, tiene como objetivo centralizar el sistema de logs. Este almacena, en forma incremental, información sobre el funcionamiento del sistema operativo y aplicaciones. Esto hace que la implementación del protocolo PEO sea directa.

Durante el funcionamiento normal no se producen modificaciones en ninguno de los archivos, sin embargo, es de práctica común entre los intrusos que alcanzan el control total del sistema realizar modificaciones en estos archivos con el fin de ocultar sus rastros.

En este caso la utilización de PEO provee una solución definitiva a este problema, permitiendo a un ente auditor (o incluso al mismo administrador) detectar un acceso no autorizado registrado por estos medios.

Esta noción podría extenderse a todo un esquema de seguridad de una red corporativa. Podría completar el paradigma "Firewalls"[6] en las áreas en las que este es mas débil: intrusiones consumadas, ataques desde adentro, etc.

VCR, auditando a root:

Una aplicación novedosa de VCR dentro del ámbito de los sistemas abiertos es la auditoría del usuario administrador.

Por las características implícitas en el rol de administrador, es común que este posea acceso irrestricto a los recursos del sistema. Esto implica grandes riesgos si la información almacenada es valiosa. Los métodos directos de registro electrónico de las actividades del administrador en la misma maquina son vulnerables frente a sus propios poderes.

---

6. Ver [CB1994] [FK1996]

El uso de un sistema criptográfico corriente, de clave simétrica o publica, no impide que el administrador rescate los elementos necesarios para adulterar los registros de auditoría.

Imaginemos que todas las acciones que efectúa el administrador son almacenadas por un sistema basado en VCR. Todo intento de acceder al estado actual del sistema (como única manera de violarlo) quedaría almacenado utilizando el mismo protocolo, cambiando el estado en que este se encuentra, y por lo tanto impidiendo su modificación transparente.

Es decir, cualquier intento de conocer K_i daría por enterado al auditor y solo daría a conocer K_{i+1}[7].

CONCLUSIONES

La simpleza de los protocolos introducidos en este trabajo, y su facilidad de implementación, junto con el valor practico de las mejoras que introducen, hacen de ellos una útil herramienta a tener en cuenta en el desarrollo de aplicaciones donde la autenticidad de la información tenga un valor tangible.

7. *"Todo intento de salir del sistema esta condenado a fracasar"*.
Es necesario definir cualquier acción que el administrador pudiera realizar con vistas a manipular los recursos para eludir el sistema de logs, como un intento de ataque.

BIBLIOGRAFÍA

[CB1994] "Firewalls and Internet Security". W.R. Cheswick & S.M. Bellovin. Addison Wesley 1994.

[EFF] http://www.eff.org

[FK1996] "Seguridad detrás del Firewall, un enfoque criptográfico", Ariel Futoransky & Emiliano Kargieman. Inédito.

[GJ1979] "Computers and Intractability: a guide to the theory of NP-Completeness". M.R. Garey & D.S. Johnson. W.H. Freeman & Co 1979.

[Knu1981] "The art of computer programming: Volume II. Seminumerical Algorithms". D. Knuth. Addison Wesley 1981.

[Sch1994] "Applied Cryptography". Bruce Schneier. John Wiley & sons 1994.