

Igual que en el matrimonio, en la seguridad SSL lo que importa son los detalles

ATAQUES A SSL

Algo viejo, algo nuevo, algo prestado y algo azul. **POR KURT SEIFRIED**

El 2009 fue un año muy interesante en lo que respecta a seguridad SSL. Se publicaron varios ataques prácticos nuevos, la mayoría de los cuales, afortunadamente, se solucionaron en un período de tiempo relativamente corto. El año comenzó con un ataque efectivo contra certificados SSL basados en MD5 [1]. Se trataba de un ataque muy sofisticado, tecnológicamente hablando, y probablemente no sea el único que veamos de aquí en adelante, ahora que los certificados MD5 han quedado en entredicho. Luego, en la BlackHat de 2009 en Las Vegas, Moxie Marlinspike expuso varios ataques contra SSL, incluido un antiguo problema que se ha agravado recientemente.

Algo Viejo: SSL Strip

Este ataque es bastante simple e imaginativo. Debido a que la mayoría de los usuarios visitan páginas no protegidas con SSL antes de ser llevados a páginas cifradas con SSL (procesamiento de pago, servidor de acceso, etcétera), un atacante puede evitar fácilmente que éstos usen el cifrado para capturar así todo su tráfico. En el pasado, un atacante habría intentado ejecutar un ata-

que de MITM (*man-in-the-middle*) capturando el tráfico no cifrado o usando un certificado firmado por sí mismo para hacerse pasar por un sitio legítimo, pero SSL se diseñó para evitar precisamente este tipo de cosas.

¿Pero qué ocurre si la víctima nunca llega a comunicarse realmente con el sitio web seguro? Muchos sitios (probablemente el banco en el que operamos, comercios en línea, etcétera) utilizan páginas iniciales no cifradas en las que el usuario lleva a cabo la mayor parte de la actividad. Un atacante puede visualizar, e incluso modificar, el contenido de la comunicación sin levantar sospechas en el usuario final, tan sólo reescribiendo todos los enlaces de esas páginas que lleven a páginas protegidas o cifradas. A no ser, claro está, que el usuario sea lo suficientemente avisado como para percatarse de que faltan el candado y el resto de detalles que se muestran cuando se está visitando una página cifrada. El programa SSL Strip automatiza este proceso de eliminación o modificación de enlaces, e incluye un falsificador de ARP que afectará a cualquier persona en la misma red local o compartiendo el mismo punto de acceso, y redirigirá a nuestro sistema todo el tráfico [2].

Instalación de SSL Strip

SSL strip es una aplicación escrita en Python que usa el framework Twisted [3]. Por tanto, lo único que hace falta es instalar *twisted*, descargar y descomprimir el tarball de SSL Strip y, opcionalmente, instalarlo. Se puede ejecutar desde el directorio local donde se descomprima sin necesidad de instalarlo:

```
yum install twisted-web
apt-get install twisted-web
python setup.py install
```

Luego, basta con habilitar la redirección de paquetes, añadir una regla de *iptables* para redirigir el tráfico HTTP, y ejecutar el programa Python SSL Strip. Para redirigir las máquinas locales a nuestro sistema podemos usar el programa *arp-spoof* incluido en la suite de *dsniff* [4]. También se pueden utilizar otros trucos distintos del ARP spoofing (ataques de DHCP, envenenamiento de DNS, etcétera).

Algo Nuevo: Certificados SSL con Caracteres NULL

Al crear un certificado SSL y enviarlo a una autoridad certificadora para que lo firme (por ejemplo, VeriSign), el único campo al que la gente suele prestar atención es a *CN* o *common name*. El campo *CN* especifica el nombre del servidor, como pueda ser *www.example.org*, o **.somecompany.com*. Moxie Marlinspike descubrió que los estándares de certificado X.509 y SSL definen la cadena de *CN* como una cadena PASCAL; por tanto, esencialmente, se declara la longitud de la cadena en la posición 0 y se pone la cadena en sí en las posiciones siguientes. Sin embargo, y dado que la mayoría (básicamente todo) del software de pro-

cesamiento de certificados está escrito en C, dicho software suele manejar la cadena como una cadena de C; poniendo un NULL (\0) al final de la cadena para indicar dónde termina ésta. La mayoría de los programadores no se percató de las implicaciones que esto tenía y simplemente pasaban la cadena CN a una estructura de cadena en C.

El problema llega cuando alguien (que legítimamente posee *www.example.com*, por ejemplo) obtiene un certificado para *www.unbancocualquiera.com\0www.example.org*. Ya en los primeros días de los certificados SSL, cuando aún eran muy caros, eran los humanos quienes revisaban las peticiones. No sólo se solicitaban certificados, sino que también se registraban negocios, así como otros modos de probar la potestad sobre un dominio concreto. Por tanto, cualquier petición malformada o extraña era muy probable que fuese rechazada y no se procesara.

Los tiempos han cambiado. Ahora es posible obtener un certificado SSL en minuto, a través de un proceso completamente automatizado (que principalmente lo que hace es un WHOIS del dominio y envía un email al contacto administrativo o al contacto técnico). Por tanto, podemos solicitar un certificado que parezca que pertenece a un dominio de nuestra titularidad (por ejemplo, *example.org*). Sin embargo, cuando la aplicación es procesada por un navegador, debido a que maneja el CN como una cadena de C, sólo se leerá la primera parte, *www.unbancocualquiera.com*, ya que al encontrar el terminador NULL, dejará fuera *www.example.org* (permitiendo falsificar fácilmente *www.unbancocualquiera.com*).

Lo bueno de todo esto es que tiene fácil arreglo; las entidades certificadoras comenzaron a rechazar todos los certificados que contuvieran un carácter NULL – algo que en un escenario real nunca debería ocurrir – e implementaron filtros para evitar que volviera a pasar. En cuanto a la parte cliente, la mayoría de los navegadores web y de los clientes y servidores con soporte para SSL (como *mutt*, *PostgreSQL*, *fetchmail* u *Opera*) han publicado actualizaciones para lidiar con el problema. Para saber más sobre este ataque y sus derivados, véase la presentación en BlackHat de Moxie Marlinspike (en formato PDF) [5].

Algo Prestado: El Nombre

Todos hemos ido alguna vez a la página equivocada (*gogole.com* en vez de *google.com*). En el pasado, las cosas eran relativamente simples: Si un atacante quería registrar un dominio que imitaba a uno real, le bastaba con saltarse o añadir alguna letra. Para defenderse de estas amenazas, lo más común (como ha hecho Google con el dominio *gogole.com*) era registrar todos los dominios que se prestaban a confusión. Por suerte, el número de combinaciones no era exageradamente alto.

Sin embargo, esta circunstancia cambió con la llegada de los Nombres de Dominio Internacionales. Ahora hay varias docenas de caracteres virtualmente idénticos a las letras Romanas, como pueden ser la letras Cirílicas Es (c), Shha (h), Ye (e), Je (j), On (o), Er (p), Dze (s), Kha (x) y U (y), o las griegas omicron (O) y nu (N). Dado que no hay una forma sencilla de verificar los nombres de dominio que estamos visitando, se espera que los navegadores comiencen a proveer colas visuales con la apariencia de los nombres de dominio. Una forma de hacerlo sería marcando el texto con un color diferente en base a su país [6].

Algo Azul: Renegociación SSL

Nos quedamos sin espacio, así que este tema será corto y agradable. Básicamente, cuando se escribieron las especificaciones de SSL y TLS en 1990, éstas contaban con una ingeniería algo superior a la necesaria, de forma que permitían comportamientos (como la renegociación) que resultaron innecesarios e indeseados por la mayoría de las personas. Explotando el comportamiento de la renegociación, un atacante puede insertar contenido que le permita llevar a cabo una nueva clase de ataque de CSRF (*Cross-Site Request Forgery*). Pero no hay de qué preocuparse; la mayoría de las aplicaciones modernas poseen robustas protecciones contra ataques de CSRF, como los tokens de usar y tirar, que mutan en cada transacción para evitar la inclusión de transacciones falsas [7].

Por desgracia, algunos sitios web permiten determinados comportamientos que pueden derivar en problemas. El primer ejemplo de ataque en el mundo real fue Twitter. La API de Twitter (ahora par-

cheada) permitía que un atacante introdujese cabeceras HTTP nuevas en la petición. De esta forma podía dirigir el contenido de la petición original de la víctima, de modo que se enviase como HTTP POST. Así, lo que Twitter recibía eran datos HTTP POST (la cookie del usuario) y los publicaba como un tweet público [8]. La solución a este problema es realmente fácil: Muchos fabricantes simplemente deshabilitan la renegociación SSL en su software. Si no hay renegociación, no hay ataque.

Conclusión

La cosa se complica ahí fuera. Por suerte, los investigadores de seguridad son cada vez mejores leyendo especificaciones, comparando los sistemas creados y hallando los agujeros de seguridad. Quizá algún día los que escriben las especificaciones y los que escriben el software acaben entendiéndose. ■

RECURSOS

- [1] "Cadena de Confianza Rota", por Kurt Seifried, Linux Magazine, Número 51, pág. 8: http://www.linux-magazine.es/issue/51/008-009_InseguridadesLM51.pdf
- [2] SSL Strip: <http://www.thoughtcrime.org/software/sslstrip/>
- [3] Twisted: <http://twistedmatrix.com/trac/>
- [4] dsniff: <http://www.monkey.org/~dugsong/dsniff/>
- [5] Más Trucos para Vencer a SSL en la Práctica: <http://www.blackhat.com/presentations/bh-usa-09/MARLINSPIKE/BHUSA09-Marlinspike-DefeatSSL-SLIDES.pdf>
- [6] IDN Homograph Attack: http://en.wikipedia.org/wiki/IDN_homograph_attack
- [7] "Ataque CSRF" por Kurt Seifried, Linux Magazine, Número 50, pág. 8: http://www.linux-magazine.es/issue/50/008-009_Inseguridades50.pdf
- [8] Vulnerabilidad de Renegociación de TLS (CVE-2009-3555): <http://www.securegoose.org/2009/11/tls-renegotiation-vulnerability-cve.html>

EL AUTOR

Kurt Seifried trabaja desde 1996 como Consultor TI especializado en Linux y redes. Suele preguntarse cómo es posible que la tecnología funcione a gran escala cuando falla tanto a pequeña escala.

