Contramedidas en la suplantación de autoridades de certificación. Certificate pinning

Alfonso Muñoz
Telefonica Digital Identity - Privacy
Security researcher
Email: alfonso.munoz@11paths.com

Antonio Guzmán
Telefonica Digital Identity - Privacy
Security researcher
Email: antonio.guzman@11paths.com

Sergio de los Santos
Telefonica Digital Identity - Privacy
Security researcher
Email: sergio.delossantos@11paths.com

***Resumen*—La importancia de asegurar la comunicación entre personas ha crecido a medida que se ha avanzado en la sofisticación y el alcance de los mecanismos provistos para ello. Ahora, en la era digital, el alcance de estas comunicaciones es global y surge la necesidad de confiar en infraestructuras que suplan la imposibilidad de identificar a ambos extremos de la comunicación. Es la infraestructura de autoridades de certificación y la gestión correcta de certificados digitales la que ha facilitado una aproximación más eficiente para cubrir esta demanda. Existen, sin embargo, algunos aspectos de esta infraestructura o de la implementación de algunos de sus mecanismos que pueden ser aprovechados para vulnerar la seguridad que su uso debe garantizar. La presente investigación profundiza en alguno de estos aspectos y analiza la validez de las soluciones propuestas por grandes productores de software frente a escenarios realistas.**

***Palabras clave*—certificate pinning, certificado digital, identificación, confidencialidad**

I. Introducción

La presente investigación revisa algunos de los aspectos que son relativos a la seguridad de la identificación digital mediante certificados digitales. Para ello se propone un estado del arte actualizado sobre esta cuestión y se presta especial interés al concepto de *certificate pinning* y cómo éste se está utilizando en la actualidad para asegurar las comunicaciones en Internet.

Este artículo está estructurado en cinco apartados. El primer apartado refleja la estructura del artículo. El apartado segundo analiza la identificación mediante certificados digitales focalizado en tres grandes retos (criptográficos, de interfaces y de cadena de certificación), así como las contramedidas posibles a las amenazas sobre los mismos. El tercer apartado introduce el concepto de *certificate pinning*. Por último, en el apartado cuarto se reflejan nuestros estudios sobre la implementación actual del *certificate pinning* en navegadores web, concluyendo en el apartado quinto con diferentes conclusiones sobre la investigación realizada.

II. Amenazas y contramedidas definidas sobre la infraestructura para la gestión de certificados digitales

La identificación de entidades en las comunicaciones en Internet es un paso fundamental para proporcionar toda una serie de servicios apoyados en los clásicos mecanismos de seguridad, como son la confidencialidad, integridad y autenticidad, tanto de la información intercambiada como de los actores que participan en una comunicación. Hoy día, es posible establecer comunicaciones seguras en Internet de una manera escalable gracias a la criptografía de clave pública, los certificados digitales, típicamente certificados X.509v3, y las infraestructuras de clave pública. Todas estas tecnologías y algoritmos permiten a un navegador web, utilizando protocolo HTTP/TLS=HTTPS, establecer comunicaciones seguras garantizando su confidencialidad (se negocia una clave simétrica para cifrar la información), integridad y autenticidad. En la actualidad, escenarios tan importantes como el comercio electrónico o la firma electrónica no serían posibles sin estas garantías. Es tal la importancia de estas tecnologías que es vital analizar las amenazas existentes que pudieran vulnerar sus principios, así como proponer contramedidas siempre que fuera posible.

II-A. Amenazas

En la actualidad, las amenazas vienen fundamentalmente de tres vías: problemas en la implementación de los mecanismos que garantizan estos escenarios o debilidades en los algoritmos utilizados en la gestión de los certificados digitales, vulnerabilidades de la interfaz de usuario y la posibilidad de suplantación de elementos en la cadena de certificación que garantiza la autoría de una clave pública.

- **Problemas derivados de fallos criptográficos e implementación**

Toda la seguridad de los certificados digitales recae en la robustez e implementación adecuada de los algoritmos criptográficos utilizados (cifrado y hash). En los últimos años se han documentado casos graves de vulneración de comunicaciones cuando esto no se produce. Entre los más significativos destacan la investigación de Luciano Bello demostrando la implementación incorrecta de OpenSSL que permitía invertir procesos criptográficos [1], la investigación de Alexander Sotirov et al. [2] falsificando certificados digitales aprovechando ataques de colisión al algoritmo MD5 o la investigación de Moxie Marlinspike [3] que descubrió que las autoridades de certificación no validan adecuadamente el campo CN (Common Name), al firmar un certificado, y por tanto era factible utilizar una codificación especial para hacer enmascarar un dominio ilegítimo haciendo pensar al usuario que está viendo uno legítimo.

ISBN: 978-84-9717-323-0

- **Vulnerabilidades de las interfaces de usuario**

Los terminales que se utilizan para comunicaciones en Internet suelen ser terminales inseguros por definición, clásicamente, el computador de sobremesa o los dispositivos móviles. Si es posible manipular su configuración o instalar malware, cualquier sistema de seguridad será anulado. Por ejemplo, en el caso del eDNI, esto ha sido puesto de manifiesto en diversas ocasiones [4]. Si se parte del supuesto que la seguridad del sistema no se ha comprometido, entonces se puede concluir que parte de las amenazas estarán basadas en que es posible engañar al usuario aprovechándose de particularidades de la interfaz de usuario (su configuración y el modo que opera el usuario con ella), es decir, típicamente el navegador web. Posiblemente el ejemplo más representativo es el intento de hacer aceptar a un usuario un certificado como válido cuando el navegador le indica que no lo es (al final es decisión del usuario aceptarlo o rechazarlo). Por desgracia, en determinados escenarios esta decisión podría ser transparente al usuario y no ser consciente de la suplantación. El investigador Moxie Marlinspike descubrió en 2009 que una configuración incorrecta del protocolo OCSP (Online Certificate Status Protocol) simplificaría ataques al protocolo SSL [5]. OCSP es un protocolo de consulta online para saber si un determinado certificado digital ha sido revocado o no. Para ello, el cliente envía la petición a la dirección de la CRL (Certificate Revocation List), que viene indicada en el propio certificado digital. Si un atacante está haciendo un ataque de hombre en el medio para utilizar uno de estos certificados digitales, entonces también puede interceptar las peticiones OCSP y utilizarlas en su provecho. En un funcionamiento normal, un servidor mediante este protocolo podría enviar una respuesta *Try Later* indicando al cliente que ahora no puede atender una petición. El atacante podría simular esta contestación, que tiene asignado el código 3, para indicar al cliente que ahora no puede atender su petición. Ante esta situación muchos clientes web aceptaban el certificado digital al no poder corroborar su validez. En la práctica muchos esfuerzos se han realizado en el pasado, ataques y herramientas, para engañar al usuario. Un ejemplo significativo es la herramienta SSLstrip, del investigador Moxie Marlinskpe [6], que intenta engañar al usuario de la siguiente forma: cuando se llama a una página web, se sustituyen todos los enlaces https por http, con la intención que la comunicación entre el cliente y el atacante sea por http y la comunicación entre atacante y servidor por https. Para engañar a usuarios menos formados se simula el *candado amarillo* cargando esta imagen en el favicon. En los últimos años se ha documentado también un especial interés en la detección de malas implementaciones por parte del interfaz del usuario, navegador web, de protocolos de seguridad. Los ataques más significativos han sido BEAST [7], CRIME [8] y BREACH [9], que se apoyaban en implementaciones inadecuadas del protocolo TLS/SSL permitiendo, bajo ciertas condiciones, descifrar una comunicación (cookies de sesión).

- **Suplantación de la cadena de certificación**

La seguridad en Internet se apoya en la confianza en autoridades intermedias que certificarán la autenticidad de un certificado digital. Esta confianza ejecutada de manera recursiva a diferentes niveles, hasta resolver la autenticidad de un certificado digital de un usuario, se conoce como cadena de certificación. En los últimos años multitud de ataques se han centrado en suplantar a autoridades o certificados pertenecientes a la cadena de certificación con un objetivo claro: un certificado falso validado por la cadena de certificación será dado como bueno y por tanto se podrán suplantar dominios válidos. Algunos sucesos significativos han sido: compromiso de la CA Comodo, compromiso CA Diginotar [10], CA TurkTrust [11], virus Flame [12] (firmado de código y suplantación de Windows update), certificado Adobe (firmado de herramientas ilegítimas) [13], etc.

II-B. Contramedidas

Las amenazas reflejadas en el apartado anterior tienen soluciones diferentes. Las dos primeras amenazas aunque no resueltas de manera global pueden ser mitigadas con buenas prácticas y sistemas robustos de actualización. Por ejemplo, en el caso de la navegación segura en Internet utilizando el protocolo HTTP/SSL existen recomendaciones que son necesarias conocer [14]. No obstante, el usuario siempre puede tomar algunas medidas adicionales, que aunque requieran cierta configuración, pueden ser de utilidad en la mitigación. Por ejemplo, el uso de extensiones en el navegador web para forzar siempre de manera automática la conexión https a un dominio si ésta existe (add-on Firefox https everywhere [15]), la configuración adecuada de los protocolos que verifican el estado de revocación de un certificado digital o una postura más activa que permita al usuario comprobar la seguridad del servidor web con el que se comunica. Para ello una herramienta de gran utilidad es la herramienta TLSSLed [16].

Por otro lado, si se centra la atención en la última amenaza destacada en el apartado anterior, "suplantación de elementos en la cadena de certificación", la bibliografía publicada en cuanto a contramedidas y la evolución de las mismas en entornos reales es, a nuestro entender, escasa. En la actualidad este tema se ha abordado de manera individual en los extremos de la comunicación. Desde el punto de vista del desarrollador de una PKI, cómo implementarla y protegerla para evitar suplantaciones [17] [18]. Desde el punto de vista del usuario instalar y configurar contramedidas para detectar posibles plagios y modificaciones en certificados digitales. Un ejemplo de lo anterior consistiría en reducir los vectores de ataque bloqueando autoridades de certificación en función de su procedencia geográfica [20]. Esto minimiza el impacto de certificados (y por tanto dominios web) firmados por organizaciones que podrían ser más sobornables o manipulables en determinados países. En el caso más extremo, algunos investigadores como Moxie Marlinspike han propuesto, con la extensión Firefox denominada Convergence [21], no sin inconvenientes, delegar la confianza de si un certificado es válido en la decisión de un número de usuarios en red,

conceptualmente en una aproximación similar al concepto de anillo de claves de confianza en GPG.

De todas las contramedidas documentadas una técnica que está comenzando a mostrar gran utilidad es garantizar de manera transparente y automatizada la cadena de certificación para evitar ataques derivados de suplantación de entidades intermedias. Esta contramedida se conoce como *certificate pinning*.

III. Certificate Pinning. Reduciendo vectores de ataque

El problema principal en la suplantación de autoridades intermedias en una cadena de certificación, necesaria para validar la autenticidad de un certificado digital asociada a un dominio, reside en que los navegadores web no tienen la capacidad, por defecto, de detectar que la cadena de confianza se ha modificado (que no comprometido), ya que el certificado/CA comprometido estará firmado por una CA válida que a su vez depende de una CA de nivel superior de la cual se confía y no ha sido comprometida.

La idea detrás del *certificate pinning* reside precisamente en poder detectar cuándo una cadena de confianza ha sido modificada. Para ello se busca asociar inequívocamente un certificado digital a un dominio concreto (se recuerda certificados presentes en una cadena de certificación). De esta forma, un dominio A, por ejemplo www.google.com, estará vinculado a un certificado/autoridad de certificación B específico. Si una autoridad de certificación B' diferente (que depende de una autoridad de certificación raíz de la que se confía) intenta emitir un certificado asociado al dominio A este hecho generará una alerta. La cadena de certificación no ha tenido por qué ser comprometida, pero sí se detecta que ha sido modificada. Esta característica hubiera permitido detectar ataques recientes derivados del compromiso de entidades intermedias, como fue el caso del compromiso de la CA de Comodo [10].

En la práctica, a día de hoy no existe consenso en cómo llevar estos principios al mundo real. Aunque todavía no existe ningún estándar se están comenzando a vislumbrar trabajos más maduros en esta dirección [22][25][26]. En febrero del 2014 se publicó un borrador de RFC [22] en el que el IETF estudia la posibilidad de que el concepto de *certificate pinning* se incluya directamente en el protocolo HTTP. Con esta futura modificación al protocolo los dominios enviarían información HTTP al navegador sobre sus certificados utilizando cabeceras HTTP:

```
Public-Key-Pins:
pin-sha1=''4n972HfV354KP560yw4uqe/baXc='';
pin-sha1=''qvTGHdzF6KLavt4PO0gs2a6pQ00='';
pin-sha256=
''LPJNul+wow4m6DsqxbninhsWHlwfp0JecwQzYpOLm\\CQ='';
max-age=10000;
includeSubDomains
```

En esta propuesta de RFC, en esencia, se propone enviar un hash por clave pública a *pinear* (pin-sha1 o pin-sha256) y el tiempo máximo (max-age) en el cual se debería confiar en esa información. En cualquier caso, y mientras se extiende su utilización, las soluciones existentes se centran en la realización de cambios, más o menos complejos, a los navegadores y protocolos utilizados para acercarse al concepto de *pinning*. Entre las propuestas analizadas son destacables:

Proyecto DANE (DNS-Based Authentication of Named Entities) [26]. El IETF normaliza en la RFC 6698 la posibilidad de vincular el protocolo TLS a dominios específicos realizando ciertas modificaciones al protocolo DNS. Lógicamente se debe confiar en la validez del firmador de ese certificado, y como ya se ha visto éste es uno de los problemas actuales de las autoridades de certificación en Internet. Una aproximación a la solución de este problema consistiría en la utilización de DNSSEC (DNS Security Extensions), pudiendo vincular claves criptográficas (certificados) a nombres de dominio DNS (en lugar de a cadenas de texto más o menos arbitrarias presentes en un certificado). En este sentido, el proyecto DANE (DNS-Based Authentication of Named Entities) proporciona la posibilidad de utilizar la infraestructura DNSSEC para almacenar y firmar claves/certificados que serán utilizados en TLS.

Proyecto TACKS (Trust Assertions for Certificate Key) [27]. Moxie Marlinspike y T. Perrie en su propuesta TACKS (internet-draft) proponen una extensión al propio protocolo TLS para permitir el registro de la cadena de certificación de los certificados digitales. La idea es que un cliente intentando conectar a un servidor protegido pudiera resolver esta asociación a nivel de conexión TLS.

IV. Analisis de implementaciones de certificate pinning. Navegadores web

Las propuestas reflejadas en el apartado anterior muestran diferentes intentos para llevar a la práctica el concepto del *certificate pinning* como contramedida para los problemas derivados de ataques a la cadena de certificación. Por desgracia, como se ha indicado anteriormente, no existe a día de hoy una solución estándar, aunque diferentes fabricantes están realizando implementaciones propietarias buscando las ventajas de esta contramedida. En este apartado se indaga en la implementación del *certificate pinning* en algunos de los navegadores más utilizados, destacando sus ventajas e inconvenientes.

Para el análisis de todos ellos se parte del siguiente escenario genérico de prueba. En el cual se considera que las comunicaciones que utilicen protocolo SSL utilizarán certificados digitales que podrán ser validados por una cadena de certificación. Típicamente una autoridad certificadora hoja y raíz, y una o más autoridades de certificación intermedias.

Este escenario genérico implementado en nuestros análisis permitirá analizar diferentes escenarios donde la cadena de certificación, sin comprometerse, se modifica de alguna forma. Escenarios clásicos son cuando o el certificado hoja cambia por algún motivo o alguna autoridad certificadora intermedia

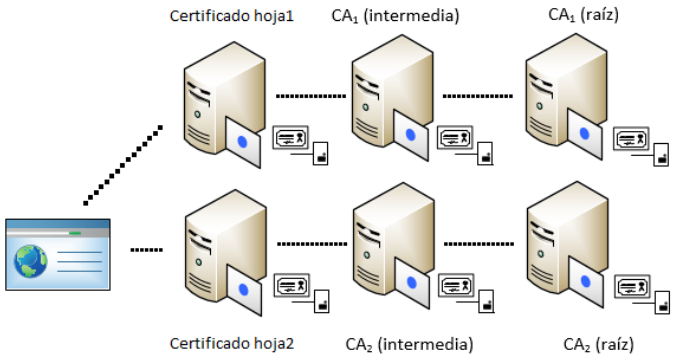

Figura 1. Escenario genérico de prueba con autoridades de certificación

es comprometida. Por ejemplo, un escenario de prueba es en el que originalmente el navegador confía en las autoridades certificadoras raíz (CA1 y CA2) y se compromete la autoridad certificadora intermedia CA2, generando un certificado para el dominio X. Dominio y certificado que fue generado originalmente por la autoridad intermedia CA1. Este escenario ha sido el más habitual en los ataques publicados en los últimos años [10].

IV-A. Microsoft Internet Explorer

Microsoft implementa una aproximación a la funcionalidad de *certificate pinning* en su herramienta de seguridad EMET [28]. El kit de herramientas de experiencia de mitigación mejorada (EMET) es una utilidad que ayuda a prevenir la explotación de vulnerabilidades de seguridad en el software.

Esta herramienta se aproxima al concepto de *certificate pinning* facilitando la unión de dominios con certificados raíz del repositorio de certificados de confianza del usuario o de la máquina. Mediante la herramienta Process Explorer [29] se puede observar cómo se inyecta la librería EMET_CE.DLL en el proceso de Internet Explorer lo que permite, mediante su configuración, analizar si el dominio al que supuestamente pertenece un certificado digital coincide con una regla de configuración definida en EMET. Si no fuera así se avisa con una pequeña ventana emergente. Por desgracia, la apuesta de Microsoft resulta ser un mecanismo poco usable para usuarios no formados.

Aunque por defecto vienen preconfigurados algunos dominios populares (Facebook, Twitter, etc.) esta solución todavía presenta una serie de inconvenientes reseñables:

- EMET realiza *pineo* exclusivamente a certificados creados por autoridades raíz registradas en el almacén de certificados del sistema operativo Windows. Cualquier compromiso de una CA intermedia/hoja que tenga una CA raíz de la cual se confía no podrá ser detectado. Por tanto, se podrían generar certificados para dominios válidos, por ejemplo www.google.com, desde una CA que no sea la creadora original del mismo (cadena de certificación modificada aunque no comprometida).
- Esta solución no está integrada directamente en el navegador web y su usabilidad es cuestionable. Requiere configuración local y manual. La solución debe instalarse y configurarse explícitamente. Esta tarea puede simplificarse mediante la herramienta EmetRules [30].
- EMET podría ser utilizado con otros navegadores pero suele estar contraindicado. Por ejemplo, en el caso del navegador Chrome, la recomendación oficial es no utilizarlo ya que afecta negativamente al rendimiento y no proporciona mayor seguridad que las contramedidas implementadas por defecto en el navegador [31].

IV-B. Google Chrome

Google aborda el problema de la confianza en un certificado digital combinando dos principios en su navegador web: HSTS y *pineo* de certificados.

Chrome implementa el estándar HSTS (Http Strict Transport Security). Se trata de una especificación que permite obligar a que, en una página, se use siempre https aunque el usuario no lo escriba en la barra del navegador. La idea fundamental es que el servidor web mediante cabeceras http fuerce al navegador web a conectarse directamente por https. Para ello, el servidor que lo desee debe enviarle una cabecera al navegador, del tipo:

```
Strict-Transport-Security:
    max-age=16070400;
    includeSubDomains
```

Esta característica aunque interesante tiene un problema. Si la primera vez que se realiza una conexión a un servidor (o cuando la información enviada expire) se realiza en una red hostil, ataque de hombre en el medio, se demuestra que la información enviada vía cabeceras http puede ser suprimida o modificada, engañando al navegador (hasta que expire la información: max-age). Precisamente por este motivo es útil utilizar una protección extra que propone Chrome y es la posibilidad de incorporar dominios bajo petición. La idea es que Chrome bajo petición (alg@chromium.com) incorpora en su código fuente [32] forzar la conexión HTTPS para los dominios reflejados. Es los que se conoce como *Preloaded HSTS sites*. Lógicamente esta solución no parece escalable a largo plazo.

En cualquier caso, aunque interesante, la protección anterior no protege frente a un escenario de ataque basado en certificados intermedios diferentes o emitidos de forma fraudulenta. La cadena de certificación se ha modificado pero no se ha roto. En este caso Chrome, complementa la medida anterior, con una aproximación al concepto de *certificate pinning*.

Antes de analizar las formas en las que Chrome puede realizar *pinneo* es importante entender cómo valida los certificados digitales con los que opera.

En la validación de certificados Chrome no comprueba el certificado completo para calcular su validez, sólo la clave

pública asociada a él (SubjectPublicKey+ SubjectPublicKeyInfo). Tradicionalmente los navegadores, y otras herramientas, calculan el hash del certificado completo como huella digital (fingerprint) para identificarlo, esto incluye el hash de todos y cada uno de los datos del certificado, no el hash de la clave pública en sí. Por ejemplo, si se cambia la fecha de caducidad o cualquier otro dato Chrome no se dará cuenta. Esto no es necesariamente negativo, en función del entorno esta flexibilidad podría evitar falsas alarmas.

Una vez resuelto cómo valida Chrome los certificados es importante responder a la siguiente pregunta ¿cómo realiza el *pinneo*? Chrome soporta *pineo* de certificados mediante dos mecanismos:

1. Servidores que soportan el borrador de RFC [22]. Mediante envío de cabeceras HTTP un servidor enviará a un navegador información de los certificados que tiene que recordar y por tanto *pinear* (en este escenario surge el mismo problema de posible ataque MitM visto con HSTS). Por ejemplo:

```
curl -i --insecure https://stalkr.net
Strict-Transport-Security: max-age=315360000
Public-Key-Pins: max-age=31536000;
pin-sha256="byhRQJ1xBQSjURWry5pt0/JXfsk3ye8ZO
```

2. *Pineo* establecido en el código fuente del navegador. En *https://src.chromium.org/svn/branches/1312/src/net/base/transport_security_state_static.h* puede comprobarse como Chrome *pinea* por defecto una serie de certificados digitales almacenando el hash de la clave pública, más exactamente el hash sha1 en base64 con el SubjectPublicKeyInfo y la propia clave pública. Por ejemplo: VeriSignClass3, Google2048, GeoTrustGlobal, etc. Al menos, *pinea* autoridades de certificación responsables de certificados digitales de servicios variados de Google.

Debe tenerse en cuenta que Chrome ignorará el *pineo* si un certificado raíz es instalado por el usuario. La razón fundamental recae en posibilitar instalar certificados cuando soluciones antivirus o proxies instalados en entornos corporativos necesitan inspeccionar el tráfico SSL [24].

Por tanto, conocido lo anterior, ¿qué certificados puede *pinear* Chrome? En principio, Chrome podría asociar un dominio a cualquier certificado en cualquier punto de la cadena de certificación. En la práctica, se demuestra que Chrome solo *pinea* el certificado de una autoridad certificadora intermedia y el de una autoridad certificadora de backup. Es posible comprobar este hecho introduciendo la siguiente url chrome://net-internals/#hsts en el navegador Chrome y realizando consultas a un dominio deseado. Por ejemplo, para www.google.com se observa: *domain:google.com pubkey_hashes: sha1/vq7OyjSnqOco9nyMCDGdy77eijM=, sha1/Q9rWMO5T+KmAym79hfRqo3mQ4Oo=*.

Este análisis permite inferir las siguientes conclusiones respecto a este navegador:

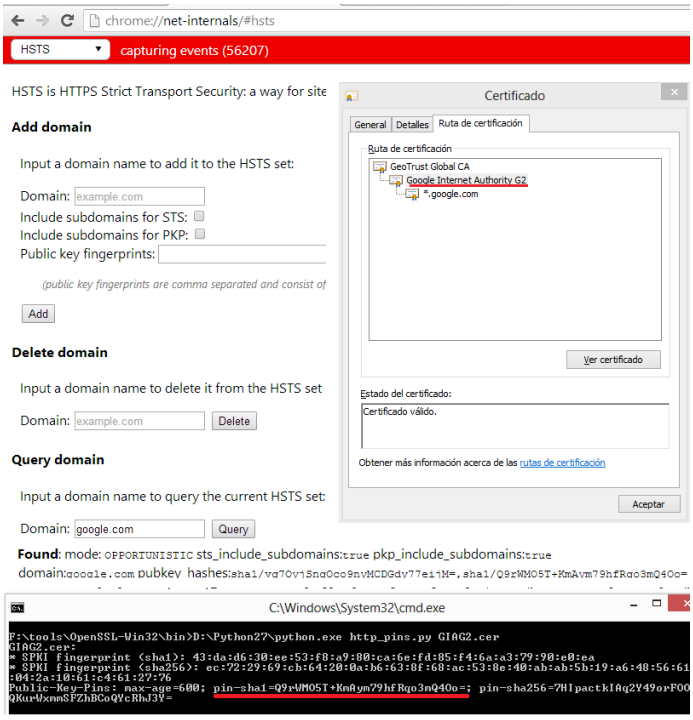

Figura 2. Detección de certificado pineado por Google

- **¿Detecta Chrome autoridades de certificación suplantadas?**

El objetivo de Chrome es intentar detectar cuándo un dominio válido, emitido originalmente por una autoridad certificadora intermedia (la cual se pinea), es emitido por otra autoridad certificadora diferente pero en la que también se confía porque se confía en una autoridad superior (típicamente raíz). Es posible detectar este escenario de cadena de certificación modificada pero no por ello comprometida, lo que supone un escenario real, y de gran utilidad como resultado de los recientes ataques documentados [10].

- **¿Qué cambios no detecta Chrome?**

Chrome, por defecto, no detecta cambios en ningún elemento de la cadena diferente a la autoridad de certificación intermedia que *pinea*. Por tanto, cualquier ataque que modifique la cadena a nivel de certificado hoja, raíz o cualquier otro elemento intermedio no será detectado. Para minimizar este efecto los servidores deberían implementar los principios del borrador de RFC [22] y enviar información de *pineo* al navegador.

IV-C. Firefox

Firefox no implementa ningún mecanismo intrínseco de *pineo* de certificados. Lo más parecido a esta aproximación es el uso del add-on *Certificate Patrol* [19] que monitoriza los cambios en certificados digitales de servidores a los que un usuario se conecta habitualmente. De todos los navegadores analizados es la única herramienta que alerta de cualquier

modificación en la cadena de certificación. Aunque esto puede resultar muy interesante tiene varios inconvenientes claros:

a) En función de la complejidad de la organización que proporcione acceso web a un servicio es común que los certificados hoja cambien por múltiples cuestiones, por ejemplo, varios certificados para balanceo de carga, etc. No parece muy conveniente el *pineo* de certificados hoja. Alertar el cambio puede saturar al usuario con alertas, muchas de las cuales no sabrá discernir como ataque.

b) La herramienta recuerda el primer certificado visto. Si se conecta por primera vez a una servidor web desde una red hostil, hombre en el medio que inyecta tráfico, se podría recordar un certificado inválido.

V. Conclusiones

El presente artículo establece un estudio del estado arte centrado en la problemática de ataques a la identificación basada en certificados digitales. El estudio se centra en el análisis y experimentación de las protecciones actuales frente a la modificación, que no compromiso, de las cadenas de certificación que certificarán o no la validez de un certificado digital.

Se ha analizado las herramientas más comunes de comunicación web en Internet, navegadores web, y se demuestra que ninguna de ellas está exenta de problemas ni permite solucionar de manera completa el problema del compromiso de cadenas de certificación. La propuesta más realista es la implementada por el navegador web Chrome si se tiene en mente una serie de limitaciones. Se ha comprobado que muchas webs populares a nivel mundial su cadena de certificación está compuesta solo por 3 niveles: certificado hoja, autoridad intermedia y autoridad raíz. Si se presupone que en general *pinear* un certificado hoja es mala idea porque puede cambiar en entornos reales (varios certificados para balanceo de carga, etc.) y una autoridad raíz debería ser muy difícil de comprometer, *pinear* solo la autoridad intermedia puede ser una solución escalable y transparente al usuario. Al final, se define un capa más de protección que, aunque no perfecta, proporciona mayores garantías en el uso de los servicios más famosos.

Mientras diferentes organismos de normalización, entre ellos el IETF [22][23], y fabricantes intentan diseñar la mejor contramedida basada en el concepto de *certificate pinning*, a día de hoy *pinear* una cadena de certificación completa no es posible en escenarios reales (cambiantes) y si no se *pinea* toda la cadena, como se ha demostrado, quedan escenarios de ataque que no se pueden detectar.

Referencias

[1] L. Bello, M. Bertacchini, “Predictable PRNG In The Vulnerable Debian OpenSSL Package. The What And The How,” *25th Chaos Communication Congress*, Berlin, Germany. December 27-30, 2008 http://events.ccc.de/congress/2008/Fahrplan/attachments/1245_openssl-debian-broken-PRNG

[2] J. Appelbaum, A. Lenstra, D. Molnar, D. Arcne, D. Weger, ‘MD5 considered harmful today Creating a rogue CA certificate‘,” *http://www.win.tue.nl/hashclash/rogue-ca/*, December 30, 2008.

[3] M. Marlinspike, ‘New Tricks For Defeating SSL In Practice‘,” *BlackHat-DC09*, https://www.blackhat.com/presentations/bh-dc-09/Marlinspike/BlackHat-DC-09-Marlinspike-Defeating-SSL.pdf

[4] G.Gonzalez, ‘Man-in-remote: PKCS11 for fun and non-profit‘,” *RootedCon 2011*, 3-5 Marzo 2011 Madrid. http://www.slideshare.net/rootedcon/gabriel-gonzalez-maninremote-pkcs11-for-fun-and-nonprofit-rootedcon-2011

[5] M. Marlinspike, ’Defeating OCSP with the character ’3”,” ,Julio 2009. http://www.thoughtcrime.org/papers/ocsp-attack.pdf

[6] M. Marlinspike, ’more tricks for defeating ssl in practice’,” *Defcon 17*, 2009. https://www.defcon.org/html/links/dc-archives/dc-17-archive.html

[7] J.Rizzo, T. Duong ‘BEAST: Surprising crypto attack against https‘,” *Ekoparty Security Conference 9º edición. 2011*

[8] J.Rizzo, T. Duong, ‘The Crime Attack‘,” *Ekoparty Security Conference 10º edición. 2012*

[9] A.Prado, N. Harris, Y. Gluck, ‘BREACH: Browser Reconnaissance & Exfiltration via Adaptive Compression of Hypertext‘,” http://breachattack.com/#howitworks

[10] R. Mandalia, ‘Security Breach in CA Networks -Comodo, DigiNotar, GlobalSign‘,” , April 2012, http://blog.isc2.org/isc2_blog/2012/04/test.html

[11] C. Wisniewski, ‘Turkish Certificate Authority screwup leads to attempted Google impersonation‘,” , April 2013, http://nakedsecurity.sophos.com/2013/01/04/turkish-certificate-authority-screwup-leads-to-attempted-google-impersonation/

[12] S. de los Santos, ‘TheFlame, el sueño de todo creador de malware‘,” , Junio 2012. http://unaaldia.hispasec.com/2012/06/theflame-el-sueno-de-todo-creador-de.html

[13] Security @adobe, ‘Inappropriate Use of Adobe Code Signing Certificate‘,” , Sept 2012. http://blogs.adobe.com/security/2012/09/inappropriate-use-of-adobe-code-signing-certificate.html

[14] C. Meyer, J. Schwenk, ‘Lessons Learned From Previous SSL/TLS Attacks. A Brief Chronology Of Attacks And Weaknesses‘,” https://eprint.iacr.org/2013/049.pdf

[15] EFF, ‘Https everywhere‘,” https://www.eff.org/https-everywhere

[16] R. Siles, ‘TLSLED‘,” , Feb 2013. http://www.taddong.com/en/lab.html#TLSSLED

[17] ETSI, ‘Policy requirements for certification authorities issuing qualified certificates (ETSI TS 101 456)‘,” , http://www.etsi.org/deliver/etsi_ts/101400_101499/101456/01.01.01_60/ts_101456v010101p.pdf

[18] Webtrust, ‘WebTrust Program for Certification Authorities‘,” , http://www.webtrust.org/homepage-documents/item27839.aspx

[19] C. Loesch, ‘Certificate Patrol‘,” , https://addons.mozilla.org/es/firefox/addon/certificate-patrol/

[20] Y. Jesus, ‘SSLcop tool‘,” , http://www.security-projects.com/?SSLCop

[21] M. Marlinspike, ‘Convergence‘,” http://convergence.io/index.html,

[22] C. Evans, C. Palmer, R. Sleevi, ‘Public Key Pinning Extension for HTTP‘,” https://tools.ietf.org/html/draft-ietf-websec-key-pinning

[23] B. Laurie, A. Langley, E. Kasper, ‘Certificate Transparency‘,” http://tools.ietf.org/html/draft-laurie-pki-sunlight-00

[24] ImperialViolet, ‘Public Key pinning‘,” https://www.imperialviolet.org/2011/05/04/pinning.html

[25] J. Walton, J. Steven, J. Manico, K. Wall, ‘Certificate and Public Key Pinning‘,” https://www.owasp.org/index.php/Certificate_and_Public_Key_Pinning,

[26] P. Hoffman, ‘The DNS-Based Authentication of Named Entities (DANE). Transport Layer Security (TLS) Protocol: TLSA‘,” https://www.rfc-editor.org/rfc/rfc6698.txt,

[27] M. Marlinspike, T. Perrin, ‘Tacks‘,” http://tack.io/-http://tack.io/draft.html,

[28] Microsoft, ‘Kit de herramientas de Experiencia de mitigación mejorada‘,” http://technet.microsoft.com/es-es/security/jj653751,

[29] M. Russinovich, ‘ProcessExplorer‘,” http://technet.microsoft.com/es-es/sysinternals/bb896653.aspx,

[30] S. de los santos, ‘EMET Rules‘,” https://www.elevenpaths.com/labs-tools-emetrules.html,

[31] Chromium projects, ‘Chromium and EMET‘,” http://dev.chromium.org/Home/chromium-security/chromium-and-emet

[32] Chrome, ‘Transport_security_state_static‘,” https://src.chromium.org/svn/branches/1312/src/net/base/transport_security_state_static.h,