PANDA
CLOUDINTERNETPROTECTION

*Simply... Evolution*

# SU NAVEGADOR ESTA DESNUDO:
## POR QUÉ LOS NAVEGADORES PROTEGIDOS SIGUEN SIENDO VULNERABLES

PANDA CLOUD
OFFICE PROTECTION

PANDA CLOUD
EMAIL PROTECTION

PANDA CLOUD
INTERNET PROTECTION

PANDA SECURITY

# INDICE

## INTRODUCCIÓN

Como usuarios de tecnología, nos han enseñado que aunque Internet no sea siempre un lugar seguro, podemos protegernos aplicando parches y reforzando la seguridad de nuestros sistemas. A pesar de que hasta ahora la aplicación de parches y la protección de los sistemas han sido los pilares básicos de la seguridad de las empresas, las reglas están cambiando gracias a la evolución tecnológica y a los nuevos tipos de ataques. Hoy en día, es algo habitual que una máquina totalmente protegida se infecte. A veces esto se debe a sofisticados ataques de ingeniería social o a nuevas vulnerabilidades (amenazas de día 0).

Sin embargo y cada vez más, es el resultado de ataques que en vez de aprovechar vulnerabilidades específicas de plataformas individuales, explotan la funcionalidad y estructura de la propia Internet. Esta evolución hacia los "ataques sobre navegadores desnudos" supone un cambio radical. De la misma forma que los atacantes están constantemente mejorando sus técnicas, las empresas deben adaptar su estrategia de seguridad si quieren mantenerse por delante en la eterna carrera armamentística de la seguridad Web.

Actualmente estamos inmersos en la era de las aplicaciones Web. La gran mayoría de desarrollos de las empresas se diseñan directamente como aplicaciones Web, independientemente de que sean para uso interno o externo. Este cambio ha traído una cierta uniformidad al panorama informático. Sea cual sea nuestro sistema operativo y nuestro hardware –móvil o no–, dispondrá de un navegador Web que se ajuste a unos estándares básicos. Los motores JavaScript se han convertido en la norma incluso en los navegadores de los teléfonos móviles, y la mayoría de plataformas soportan las tecnologías Rich Internet Application más populares como Adobe Flash.

Esta estandarización -sutil y voluntaria- no sólo ha facilitado la aparición de tecnologías 2.0 como AJAX, sino que también ha creado una amplia base de objetivos potenciales para los atacantes. Mediante el empleo de ataques de JavaScript, por ejemplo, en vez de sobre una versión concreta de Internet Explorer, el número de victimas potenciales aumenta de un X% de los usuarios de Internet a un 100%.

En el pasado, los hackers enfocaban sus ataques hacia los navegadores. Ahora, sin embargo, los navegadores son simples elementos que facilitan los ataques. Simplemente son una puerta que da acceso a los datos que busca el atacante. Ladiferencia está en que ya no hace falta identificar y explotar una vulnerabilidad concreta en un navegador. Actualmente, muchos de los ataques son 'multi-navegador' o 'multi-plataforma', ya que no se dirigen sobre el navegador, sino sobre su funcionalidad, que es la misma independientemente del navegador.

La Web se creó para ser una plataforma abierta, no una plataforma segura. Y esto no ha pasado desapercibido para los atacantes, quienes dedican gran parte de su tiempo a manipular la funcionalidad de la Web a su favor. Esto se ha traducido en una oleada de ataques con gran éxito incluso contra máquinas totalmente protegidas y actualizadas. Llamamos a estos ataques 'ataques sobre navegadores desnudos', ya que no se espera la aparición de parches que protejan a los usuarios. En este nuevo orden mundial, debemos replantearnos nuestro enfoque con respeto a la seguridad si queremos seguir luchando.

Al igual que en la naturaleza aquellos que se adaptan al medio natural son los que sobreviven y prosperan, en el mundo de los ataques informáticos sucede lo mismo. El entorno informático de las empresas ha evolucionado de forma significativa en la última década, lo que ha obligado a los atacantes a adaptar sus tácticas. El objetivo de los ataques ha pasado de los servidores a las aplicaciones Web, y los navegadores se han convertido en un medio de llevar a cabo actividades delictivas. Los hackers ya no son individuos motivados por la curiosidad, sino organizaciones criminales que buscan beneficios económicos. Ante este panorama, es hora de que las empresas suban el nivel de la seguridad de su red.

Los atacantes que anteriormente atacaban los servidores de las empresas se han dado cuenta de que es mucho más sencillo fijarse en las máquinas cliente, gracias a su escasa protección, la ingenuidad de sus usuarios y la omnipresencia de los navegadores Web. Los desbordamientos de buffer en servicios públicos del lado del servidor han sido reemplazados por ataques diversificados en el lado del cliente que utilizan la ingeniería social, las vulnerabilidades de los navegadores Web y aplicaciones Web de confianza, pero vulnerables.

Las tecnologías Web 2.0, que han permitido a los desarrolladores generar aplicaciones intuitivas y fáciles de utilizar y a los usuarios tomar el control de los contenidos de la Red, han hecho que las aplicaciones Web vulnerables se hayan convertido en una parte más de la vida en Internet. El navegador Web se ha transformado en un punto de entrada para los atacantes, permitiéndoles acceder a información sensible en dispositivos de sobremesa y móviles, así como tomar el control de las máquinas integrándolas en sus redes de bots. Las empresas deben adaptarse para defender sus redes contra los ataques actuales.

## ANTECEDENTES

En el pasado, el Ciclo de Ataque era sencillo – los hackers descubrían una vulnerabilidad en una aplicación o sistema operativo y la explotaban hasta que el fabricante publicaba un parche que resolvía el problema. Aunque ésta es una explicación algo simplista, describe el escenario en el que ha trabajado el sector de la seguridad durante mucho tiempo. Un avance prometedor, sin embargo, fue la reducción de la ventana de tiempo que aprovechaban los atacantes para explotar una vulnerabilidad. Hace una década, las empresas podían pasarse semanas o incluso meses realizando pruebas de regresión antes de tener la seguridad necesaria para aplicar los parches publicados. Así, había un amplio período de tiempo durante el cual los detalles de la vulnerabilidad eran de dominio público y millones de ordenadores quedaban vulnerables a posibles ataques. Afortunadamente, a medida que las empresas han llegado a ser más conscientes de esta amenaza y los fabricantes han mejorado el proceso de distribución de parches y la comunicación de los riesgos asociados a cada caso concreto, la ventana de riesgo ha disminuido, siendo ésta de unos pocos días, o incluso horas, a partir de la publicación de información sobre una vulnerabilidad.

Durante la última década, como se puede ver en la Imagen 1, ha habido tres épocas distintas en cuanto a los ataques. Hasta aproximadamente el año 2004, muchos ataques se dirigían a servicios Web como servidores Web, de correo y FTP. Los hackers aprovechaban vulnerabilidades en estos servicios, provocando epidemias protagonizadas por gusanos de rápida propagación. En la siguiente fase, los atacantes pasaron a fijar su atención en los navegadores Web. Con servidores cada vez más seguros, los ataques buscaban vulnerabilidades en los navegadores. Y como había muchas, los usuarios finales seguían cayendo víctimas de los ataques. Aunque muchos de estos ataques utilizaban la ingeniería social para convencer a un usuario de que visitase una página o pinchase un enlace, esto no suponía un gran reto para los hackers. Hoy, sin embargo, estamos entrando en una nueva era, la de los 'ataques sobre navegadores desnudos'. Se llaman así porque no existe una vulnerabilidad específica en el navegador, aun así, los usuarios finales siguen siendo víctimas de los mismos.

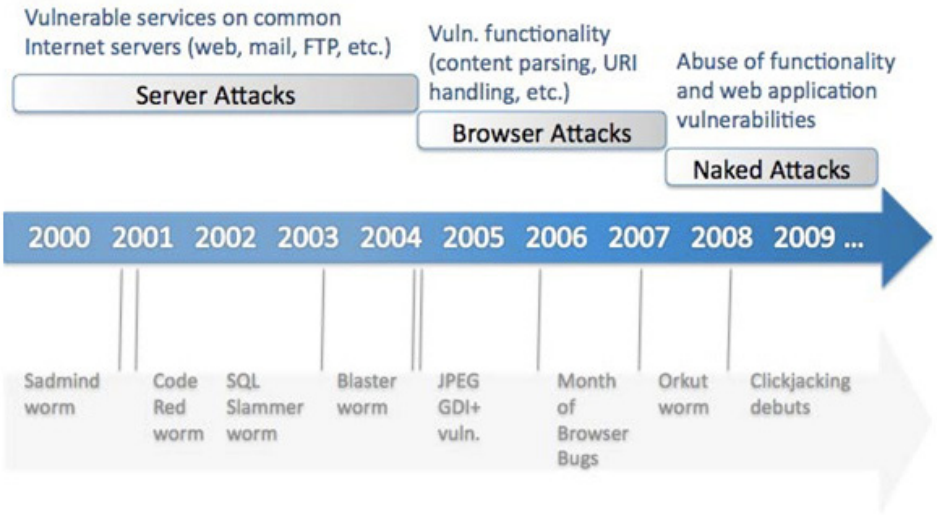



Imagen 1 – Evolución de los ataques

## ATAQUES SOBRE NAVEGADORES DESNUDOS

El éxito de este tipo de ataque contra navegadores protegidos y actualizados se debe a que aprovechan la relación de confianza existente entre un navegador y una aplicación Web vulnerable, o porque explotan la funcionalidad de la propia Web, utilizándola de una forma no contemplada. Hay muchos ataques que entran dentro de esta categoría, y no podemos incluir aquí una explicación detallada de todos ellos. Explicaremos este concepto mediante una descripción del ataque de cross-site scripting (XSS), que aprovecha la confianza existente entre el navegador y el servidor, así como la técnica de 'clickjacking', que explota una funcionalidad legítima de una forma irregular.

## CROSS SITE SCRIPTING

Los ataques XSS siguen siendo unos de los más frecuentes en la actualidad, a pesar de que llevan varios años en el ojo público. Han figurado en la lista OWASP Top Ten [1] que detalla las vulnerabilidades existentes en las aplicaciones Web desde su aparición en el año 2004.

Además, el informe WhiteHat Website Security Statistics Report [2] de diciembre de 2008 apuntaba ya que un 67% de los sitios Web podía estar afectado por agujeros XSS. Es una cifra estremecedora, y subraya el riesgo que corre cada usuario cada vez que se conecta a Internet.

Los ataques XSS son un claro ejemplo de uno de los cambios principales producidos por la interconectividad de la Web. Los ataques XSS explotan vulnerabilidades no en el navegador del usuario, sino en las aplicaciones Web de terceros a las que accede el usuario. A pesar de ello, el usuario es la víctima del ataque ya que su navegador responde como debe ante el código JavaScript inyectado, es decir, interpretándolo. En este tipo de ataque el navegador no puede distinguir entre el contenido que un usuario haya podido incluir en una petición Web, y el contenido inyectado a través de un ataque XSS.

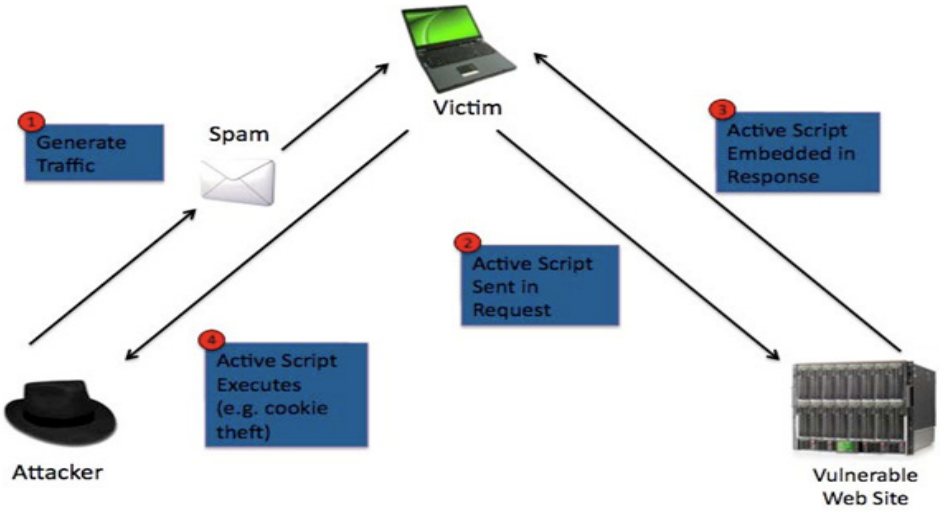



Imagen 1 – Evolución de los ataques

1 http://www.owasp.org/index.php/OWASP_Top_Ten_Project

2 http://www.whitehatsec.com/home/resource/stats.html

## Ataque típico

La imagen 2 muestra el escenario típico de un ataque XSS. A continuación detallamos como funciona un ataque XSS, y por qué no necesita de una vulnerabilidad en el navegador para tener éxito.

1. **Generación de tráfico:**
   El ataque XSS requiere que un usuario envíe una petición especialmente formada a un servidor Web vulnerable. Esta petición contiene contenido activo embebido (normalmente código JavaScript), diseñado para realizar la acción elegida por el hacker. En muchos casos, el script intenta enviar al hacker las cookies del usuario correspondientes al sitio Web atacado. Un método típico para conseguir que las víctimas envíen la petición maliciosa es mediante mensajes de spam. Normalmente, el enlace está embebido en un mensaje HTML que incluye texto diseñado para convencer al usuario de que pulse sobre él.

2. **Envío del script activo en la petición:**
   Si el usuario pincha en el enlace incluido en el mensaje de spam, estará enviandouna petición al servidor Web vulnerable. Sin embargo, en lugar de una simple petición solicitando la URL de una página Web, la solicitud también incluirá el código JavaScript inyectado, bien como parámetros de la misma URL (petición GET) o dentro del cuerpo de la solicitud (petición POST).

3. **Script activo embebido en la respuesta:**
   La página Web vulnerable incluye funcionalidades que aceptan input del usuario y lo incluye en la página generada dinámicamente devuelta al mismo usuario. Este comportamiento es correcto, siempre que el input del usuario esté adecuadamente sanitizado para asegurar que el contenido devuelto sea el esperado. Es la ausencia de tales controles la que permite los ataques XSS. Por ejemplo, imaginemos que una página permite a un usuario introducir su nombre de forma que la página resultante muestre un mensaje del tipo "Hola [nombre]". Aunque se espera una cadena de texto, en caso de que no haya una sanitización adecuada se podría usar ese mismo vector para inyectar JavaScript malicioso.

4. **Ejecución del script activo:**
   Cuando el navegador recibe la respuesta, el contenido de la página incluye el JavaScript malicioso, que el navegador interpreta y ejecuta.

## Impacto

Aunque este sencillo escenario implica una única víctima y un atacante, los ataques de XSS suelen ser más complejos. En enero de 2008, se supo que hackers habían aprovechado una vulnerabilidad XSS en la página de acceso del banco italiano Banca Fideuram para inyectar un formulario falso en un IFRAME [3]. Este ataque enviaba los credenciales del usuario a un servidor controlado por un atacante. El ataque era especialmente peligroso ya que estaba hospedado en una página de confianza, protegida por SSL. Los ataques XSS están evolucionando rápidamente; ya no son estáticos. Hemos visto ya la aparición de gusanos XSS en redes sociales populares como Orkut [4] y MySpace [5].

Es importante tener en cuenta que los ataques XSS prosperan tanto si los usuarios han aplicado todos los parches pendientes como si no. El éxito de los ataques XSS radica en que los navegadores están diseñados para interpretar código JavaScript. Mediante esta técnica, el atacante explota una vulnerabilidad que afecta a la validación de input por parte de una aplicación Web, sin embargo, es el usuario que visita la página el que se convierte

3 http://news.netcraft.com/archives/2008/01/08/italian_banks_xss_opportunity_seized_by_fraudsters.html
4 http://www.washingtonpost.com/wp-dyn/content/article/2007/12/19/AR2007121900781_pf.html
5 http://en.wikipedia.org/wiki/Samy_(XSS

en la víctima. Además, el tópico de que mientras el usuario se limite a visitar páginas 'de confianza' se mantendrá seguro, ya no es cierto. Casi todos los sitios importantes han tenido vulnerabilidades de XSS, y considerando el componente de ingeniería social de un ataque XSS, los sitios con más tráfico suelen figurar entre los objetivos de los ataques más exitosos

## CLICKJACKING

El 'clickjacking' se convirtió en noticia cuando Adobe pidió a unos investigadores que suspendiesen una ponencia sobre el tema unos pocos días antes de que tuviera lugar. El ataque consiste en ocultar contenido malicioso debajo de contenido inocuo, y, con una pequeña dosis de ingeniería social, engañar al usuario para que realice alguna acción sin querer.

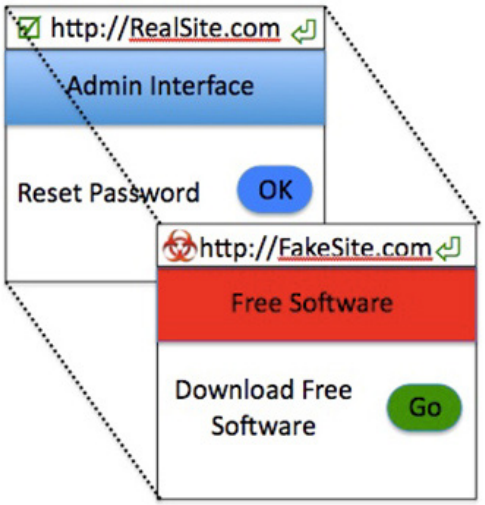



Imagen 3 – Clickjacking

En la Imagen 3 se observa una interfaz para cambiar una contraseña tapada por un sitio falso, que oculta el contenido real. Así, en este caso, un usuario que pinche para descargar software gratis, estará en realidad cambiando su contraseña. La técnica del 'clickjacking' requiere los siguientes componentes:

1. **Contenido embebido:**
   La acción deseada (p.ej. el cambio de una contraseña), que se encuentra en una página no controlada por el atacante, es embebida en una página que el atacante sí controla. Esto se suele conseguir mediante un IFRAME en la página controlada por el hacker. Se recomienda utilizar código 'frame busting' – que impide que se muestre contenido en un IFRAME-, en el lado del servidor como defensa contra el clickjacking.

2. **Ofuscación:**
   El contenido ajeno, incluyendo el botón para cambiar la contraseña, no será visible ya que su valor de opacidad (su transparencia) está fijado a cero.

3. **Superposición de capas:**
   El contenido controlado por el atacante se esconde debajo del contenido del tercero, estando perfectamente alineado con el mismo de forma que los dos botones estén exactamente en el mismo lugar. Sin embargo, es el botón 'Go' para la descarga del software el que es visible en lugar del botón 'OK' para cambiar la contraseña, gracias a la configuración de la opacidad (ver Ofuscación). La superposición se basa en los valores del índice z, propiedad que define el valor de profundidad de los elementos de una página Web. En este caso, el contenido de la tercera parte tendría un valor de índice z superior al del contenido controlado por el atacante. Así, aunque la victima está viendo el botón 'Go', en realidad está pulsando el botón 'OK'.

## IMPACTO

El 'clickjacking' es una técnica que facilita los ataques de ingeniería social. Como sucede con los ataques de XSS, el usuario no se convierte en víctima simplemente con visitar una página Web. Debe pinchar un enlace para desencadenar elataque. Mediante la combinación de diversas técnicas HTML legítimas, el clickjacking introduce un elemento de ingeniería social, haciendo que el usuario piense que está pinchando un enlace distinto del enlace con el que realmente está interactuando su navegador.

límite a los daños que podrían conllevar un ataque de clickjacking sólo está en la imaginación del atacante. La compañía Adobe se ha visto gravemente afectada por los ataques de clickjacking, no porque ofrezca un navegador que permita este tipo de ataque, sino porque su software Flash Player Settings Manager es vulnerable a esta técnica. La configuración de Flash Player se gestiona a traves de contenido Flash embebido en una página Web de Adobe [6] .La pestaña para la configuración de la privacidad (ver Imagen 4) permite establecer los permisos de un sitio Web de forma que siempre sea un sitio de


Imagen 4 - Adobe Flash Player – Configuración de la privacidad Web

Una vez que un atacante consigue influenciar las acciones del usuario de esta forma, las posibilidades de ataque son prácticamente ilimitadas. El hecho de que un usuario – sin su conocimiento- suba datos a un sitio controlado por un atacante, por ejemplo, puede comprometer su privacidad. Se podría esquivar un sistema de autenticación, abrir una cuenta falsa o reducir el nivel de seguridad preconfigurado de un sitio. En el caso de que un usuario acabase descargando un binario malicioso, se podría poner en peligro la seguridad de todo un sistema. En resumen, el

confianza y pueda acceder así a la cámara Web y al micrófono del usuario sin su permiso.
Simplemente con ofuscar la consola con contenido falso y convencer al usuario de que haga clic en ciertas partes de la pantalla, un atacante podría tomar el control de la cámara Web y el micrófono de un usuario confiado y espiarle. Este tipo de ataque fue revelado en el blog Guya.net, lo que llevó a desvelar públicamente información sobre los ataques de clickjacking [7]. La prueba de concepto que Guya.net utilizó fue un juego, implementado con JavaScript, que pedía al usuario

6 http://www.macromedia.com/support/documentation/en/flashplayer/help/settings_manager.html

7 http://ha.ckers.org/blog/20081007/clickjacking-details/

que siguiera los movimientos de un botón a través de la pantalla e hiciese clic sobre él. Mientras que alguno de los clics era inocuos, otros cambiaban la configuración de Flash Manager de forma que se seleccionase 'Always Allow' ('Permitir siempre'), tal y como se muestra en la Imagen 4.

Aunque Adobe ha resuelto este ataque específico de clickjacking, añadiendo código 'frame-busting' al Settings Manager, la técnica del clickjacking sigue funcionado perfectamente con la mayoría de navegadores Web. Y lo hace porque los IFRAMEs y las propiedades como los valores de índice z y la opacidad son estándares empleados por la mayoría de navegadores más conocidos. Se trata de herramientas muy potentes en sí mismas y que permiten el diseño de aplicaciones Web impresionantes. Sin embargo, en caso de ser combinadas de forma maliciosa por un hacker, pueden convertirse en un vector viable de ataque. Una vez más estamos hablando de ataques que no aprovechan vulnerabilidades individuales, sino que explotan funcionalidades existentes utilizándolas de una forma no contemplada inicialmente.

## OTROS ATAQUES

Los ataques de XSS y clickjacking no son los únicos que no dependen de vulnerabilidades presentes en aplicaciones en el lado del cliente. Son sin embargo dos de los más conocidos y por eso los hemos destacado aquí. Existen numerosos ataques con características similares. Por ejemplo, la falsificación de peticiones en sitios cruzados, la falsificación de contenido, la redirección de URLs, la división de respuesta HTTP, etc. Todos estos ataques incluyen elementos de los 'ataques sobre navegadores desnudos'. Además, muchos de estos ataques tienen que ver con nuevos temas emergentes y de los que sólo estamos viendo el principio.

## DESAFÍOS

No se puede parar lo que no se conoce. Si un ataque se combina con tráfico legítimo, siempre será más difícil de detectar. Los exploits de los navegadores se prestan a la detección mediante identificadores, ya que estos ataquesm implican normalmente el envío de tráfico anómalo. Tomemos el ejemplo de un desbordamiento de bufer en un navegador Web. En general, será necesario crear contenido Web con datos que activen el ataque, shellcode para ejecutar comandos una vez se haya tomado el control y algunos datos de relleno para asegurar que todo llegue al sitio adecuado. Nada de esto es contenido normal de una página Web, y por lo tanto es detectable. El clickjacking, en cambio, es mucho más difícil de detectar utilizando identificadores, ya que ninguno de sus componentes son maliciosos en sí mismos. Todos son propiedades legítimas que están a disposición de los desarrolladores de aplicaciones Web. Por lo tanto, se pueden encontrar en una gran variedad de páginas Web. Es la combinación de varios atributos legítimos lo que hace posible el ataque. En consecuencia, resulta poco probable que vayamos a encontrar una defensa infalible para combatir estos ataques.

## DEFENSA EN PROFUNDIDAD

La idea de tener que defenderse de ataques que prosperan a pesar de que los navegadores estén convenientemente protegidos y actualizados es ciertamente desconcertante. Nuestra formación nos ha hecho creer que dispondríamos de una buena defensa si mejorábamos los procedimientos de aplicación de parches, y de repente nos encontramos con ataques que esquivan todo este proceso. Es más, los 'ataques sobre navegadores desnudos' suelen aprocharse de la ingeniería social y es difícil, sino imposible, prevenir un ataque en el que un empleado ayuda de forma inconsciente al atacante. Los textos que tratan sobre la manera de defenderse contra los ataques XSS o CSRF describen el modo de proteger la aplicación Web, no el navegador afectado por el ataque. Hasta ahora, la mayor parte del capital destinado a la seguridad se concentraba en defender los servidores, no los navegadores. Sin embargo, una empresa típica puede tener cientos de navegadores por cada servidor y la mayoría se encuentran en portátiles que a menudo salen de las instalaciones de la empresa. Lo que es más, las personas que utilizan dichos navegadores suelen tener pocos conocimientos de seguridad. Cuando se contempla la seguridad de las empresas desde esta perspectiva, es obvio que tenemos que cambiar nuestras prioridades.

## SOLUCIONES ACTUALES

No existen parches que protejan a los navegadores de estos ataques, ya que los navegadores en sí no son vulnerables. Es más, actúan conforme a su diseño. Dicho esto, existen algunas aplicaciones del lado cliente que pueden ayudar a la hora de proteger contra estas amenazas. NoScript, por ejemplo, es una excelente extensión para Firefox y otros navegadores basados en Mozilla, que permite establecer un control granular sobre la ejecución de contenido activo como JavaScript, Java, Flash y otros complementos. Además, incluye controles específicos para identificar y bloquear ataques XSS y clickjacking. Sin embargo, hay que recordar a los administradores que NoScript ha sido diseñado para usuarios avanzados, y muchas de sus opciones pueden llegar a confundir a un usuario normal. Aunque algunos administradores optan por la desactivación total de los motores script en los navegadores, no se trata de una opción viable dada la importancia que tiene JavaScript en las aplicaciones Web actuales.

En el futuro, los desarrolladores de navegadores ampliarán su funcionalidad para incluir características que protejan contra estos ataques, a pesar de que aprovechen agujeros de seguridad en las aplicaciones Web en vez de en los propios navegadores. Microsoft dará un paso adelante con el lanzamiento de Internet Explorer 8, que incluirá una funcionalidad capaz de detectar ataques XSS reflejados [8].

## CÓMO DEFENDERSE CONTRA LOS ATAQUES SOBRE NAVEGADORES DESNUDOS

No es de sorprender que no exista una defensa infalible para protegerse contra los ataques sobre navegadores desnudos. No existen parches para resolver los problemas que permiten estos ataques, y por lo tanto no se prevé que haya soluciones a corto plazo. Teniendo esto en cuenta, es importante que las empresas implementen controles de detección y prevención para combatir este tipo de ataque.

**Monitorización:**
Una adecuada monitorización y registro de la actividad de la red puede servir para aislar los ataques sobre los navegadores desnudos y facilitar su seguimiento. Estos registros deben estar unificados de forma que no se mantengan de forma individual en cada pasarela Web. De este modo, se favorecerá la correlación de las incidencias en las distintas ubicaciones físicas. Analizar los registros Web para identificar patrones anómalos de tráfico. Esto podría servir para detectar picos de tráfico a una página concreta y averiguar si los atacantes están dirigiendo a los usuarios a un sitio específico para atacarles. También una caída repentina en el volumen de tráfico podría ser sospechosa, ya que indicaría que se está bloqueando el acceso a ciertos sitios de las maquinas infectadas. Esto podría ocurrir, por ejemplo, para evitar que los usuarios pudieran descargarse los últimos identificadores antivirus, y podría servir para identificar maquinas comprometidas. Sin embargo, la monitorización no es suficiente en sí misma. Alguien tiene que ser responsable del proceso para asegurarse de que los informes se generan, analizan y escalan cuando sea necesario.

**Gestión:**
Una norma generalmente aceptada en el sector de la seguridad indica que los usuarios sólo deben disponer de los niveles de acceso correspondientes a las necesidades de sus puestos. Entonces, ¿por qué cuando se trata de Internet las empresas ofrecen a sus usuarios acceso universal? (Con la posible excepción del filtrado de URLs con contenido ofensivo). Las aplicaciones Web se llaman así por un motivo – son aplicaciones, y por lo tanto podemos –y debemos– restringir el acceso a las mismas en base a su funcionalidad, no sólo al destino. Por ejemplo, aunque quizás no nos importe que nuestros empleados vean contenido en Facebook, tal vez sería conveniente que algunos, o incluso todos los usuarios, no pudiera subir información a la red social, evitando así la fuga de datos. Busque soluciones que permitan controlar lo que hacen los usuarios, no sólo a donde van.

8 http://msdn.microsoft.com/en-us/library/cc994337(VS.85).aspx

**Integración:**
Existen varias fuentes de datos, tanto comerciales como freeware, (p.ej. Phishtank, Google Safe Browsing, OpenDNS, etc.), que identifican contenido potencialmente malicioso. Dichas fuentes pueden incluirse en soluciones de filtrado Web para bloquear el acceso a sitios que formen parte de ataques basados en navegadores, como phishing o redes de bots. Con este tipo de contenido, es importante hacer un seguimiento adecuado de las métricas para asegurarse de que las listas están aportando valor y no creando niveles inaceptables de falsos positivos.

**Educación:**
Tampoco se debe pasar por alto la formación de los usuarios, pese a ser cierto que por muy diligentes que sean éstos, los controles técnicos siempre van a ser imprescindibles.
Es importante que los usuarios dispongan de los conocimientos adecuados no sólo para evitar los ataques, sino también para escalar incidencias cuando sea necesario. A la hora de diseñar un programa de formación, asegúrese de que la educación sea continua y variada. La gente aprende de distintas maneras, pero la repetición es esencial para retener los conocimientos.

## CONCLUSIÓN

En el pasado, los hackers concentraban sus esfuerzos en atacar los servidores de las empresas, buscando agujeros que les proporcionaran las 'llaves del reino'. Sin embargo, y en la medida en que los servidores se han vuelto más seguros, los PCs de los empleados se han convertido en objetivo de los ataques, sobre todo los navegadores Web, cuya seguridad siempre ha sido dudosa. Hoy en día, muchos de los ataques dirigidos a navegadores son "ataques desnudos", que no requieren de vulnerabilidades. La naturaleza interconectada de la Web hace que el peligro no quede aislado entre el navegador Web y el servidor. Algunos de los ataques descritos en este documento dejan al descubierto los datos de los usuarios debido a vulnerabilidades en las aplicaciones Web. Otros simplemente se aprovechan de alguna funcionalidad de una forma no prevista.

De todos modos, lo importante es que un número creciente de ataques va a prosperar, incluso aunque los navegadores contra los cuales se dirigen estén totalmente protegidos y actualizados. Por lo tanto, las empresas deben replantearse su enfoque de la seguridad Web. La gestión de parches ya no es la solución infalible que habíamos esperado.

## SUITE PANDA CLOUD PROTECTION

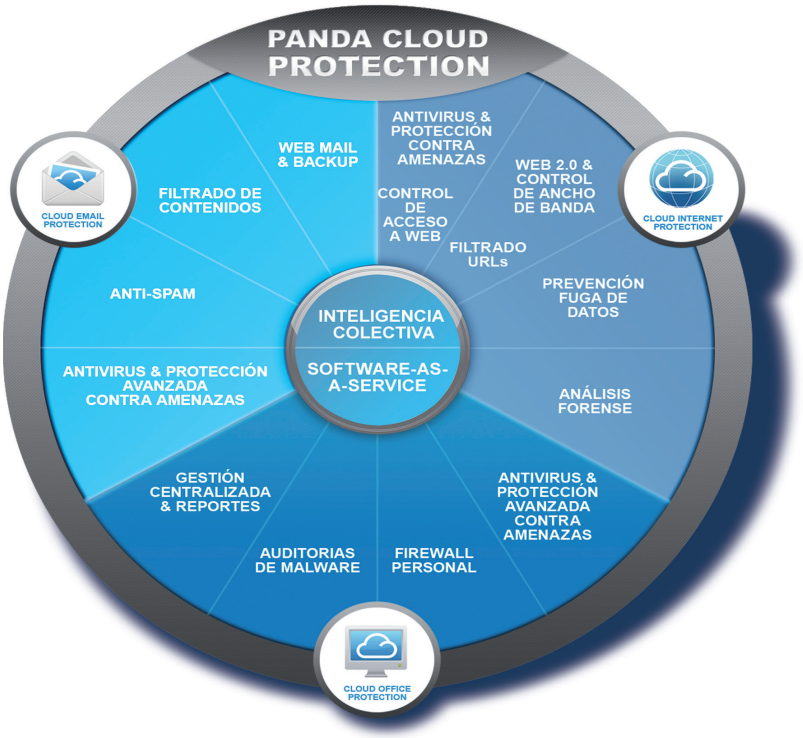
Panda Cloud Internet Protection es parte de la suite Panda Cloud Protection, una completa solución de seguridad SaaS que protege los principales puntos de entrada de las amenazas - endpoints, correo electrónico y tráfico Web- contra el malware, spam, cross-site scripting y otros ataques avanzados de Web 2.0, mediante una solución ligera, segura y sencilla.

Esta suite de seguridad está basada en la nube, ofreciendo máxima protección, reduciendo el gasto y aumentando la productividad. La puesta en marcha de la solución es inmediata y su intuitiva consola Web hace que sea muy fácil de gestionar.

La suite Panda Cloud Protection se beneficia de la gran capacidad de la Inteligencia Colectiva:

un sistema basado en la nube que almacena 21 terabytes de conocimiento y experiencia obtenidos directamente de millones de usuarios. Panda Cloud Protection ofrece protección completa para el mundo real, no intrusiva e instantánea contra el malware conocido y desconocido.

Panda Cloud Protection explota el poder de la nube y proporcionar protección en tiempo real contra las amenazas conocidas y desconocidas en cualquier momento y en cualquier lugar, gracias a su Consola de Administración en la Nube.



PANDA SECURITY

EUROPE
Ronda de Poniente, 17
28760 Tres Cantos. Madrid. SPAIN

Phone: +34 91 806 37 00

USA
230 N. Maryland, Suite 303
P.O. Box 10578. Glendale, CA 91209 - USA

Phone: +1 (818) 5436 901

www.pandasecurity.com



PANDA SECURITY