SEGURIDAD INFORMÁTICA

PNUD - Colombia

Jordi Serra Ruiz
Guillermo Navarro Arribas
Sergio Castillo-Pérez
Jordi Herrera Joancomartí
Sergi Robles Martínez
Sergio Castillo Pérez
Joaquín García Alfaro

Contenido

La seguridad informática es en estos días una cuestión ya nacional, ha cobrado tal envergadura que se hace de mayor importancia para todos los estados, entidades, empresas y personas en general. Nos concierne ya a todos. Cada día existen más dispositivos conectados a la red que nos facilitan la vida, nos ayudan a mejorar ciertas tareas o a minimizar el tiempo empleado en ellas, reservar o comprar entradas por Internet, pedir un taxi, ver la predicción meteorológica…
Actualmente está creciendo la concienciación que todas las personas tenemos sobre la seguridad informática, ya casi a diario salen en los medios de comunicación nuevos incidentes de seguridad, lo que hace que las persones empiecen ya a pensar en esta problemática. Robo de tarjetas, suplantación de identidad, filtraciones de datos personales de webs, ataques directos a infraestructuras críticas de los estados… todos estoy problemas y más se dan a diario.

Vamos a introducir toda esta problemática de la seguridad informática desde las vulnerabilidades que pueden tener los sistemas informáticos, hasta las buenas prácticas que deben tener todos los usuarios de Internet.

1. Las Vulnerabilidades

1.1. Introducción a las vulnerabilidades.

La simple existencia de una vulnerabilidad en un sistema de información, ya sea del software o en un protocolo, es lo que posibilita que dicho sistema pueda ser atacado. Por ello, al hacer un repaso de las vulnerabilidades estamos dando una introducción al problema de la seguridad informática por su base. Dicho de otra manera, veremos el porqué de la seguridad informática.

Una vulnerabilidad de seguridad se puede ver como el punto de partida de todo el proceso que implica la seguridad en general. Por ejemplo, un ataque informático sobre un servidor web generalmente parte de una vulnerabilidad en alguno de los sistemas que implementan o dan soporte al servidor: errores en la instalación del mismo servidor, dejando usuarios y contraseñas por defecto, o en el sistema operativo, con privilegios excesivos, fallos en el diseño de protocolos de comunicación, errores propiciados por la inexperiencia del personal encargado de utilizar, configurar o administrar el servidor, etc. La dificultad de prever estos errores y, por tanto, los posibles ataques lleva a la implantación de medidas preventivas y sobretodo reactivas en el caso de no dispones de las primeras, como puede ser el uso de cortafuegos, sistemas de detección de anomalías o intrusiones (IDS), realización de auditorías de seguridad, planes de contingencia y continuidad de negocio, o educación a usuarios y administradores.

Poder definir que es una vulnerabilidad no es nada sencillo. Para ello es necesario poner en contexto a dicha definición, ya que existen multitud de casos en los que podrían existir vulnerabilidades. En este sentido, aquí nos centraremos en vulnerabilidades de seguridad en sistemas de información.

En este caso consideraremos los sistemas de información (SI) de manera muy general como cualquier sistema destinado a recoger, almacenar, procesar y/o distribuir

información, conectados o no a la Red. Aunque generalmente se asocia el concepto de sistema de información al mundo empresarial, adoptamos el sentido más amplio de SI abarcando su uso tanto personal como dentro de una organización, al fin y al cabo las computadoras que tenemos en casa también pueden ser clasificadas dentro de este grupo de sistemas. En general, al hablar de SI nos referiremos a un sistema informático, aunque no todos los SI son informáticos. Es importante resaltar que el estudio de la seguridad de los SI y sus vulnerabilidades abarca no solo a los SI propiamente, sino también el estudio de todas las entidades y los fenómenos que puede afectar directa o indirectamente a los SI. Esta es una visión muy amplia de SI que puede comprender entre otros: ordenadores de uso personal, teléfonos móviles, servidores de correo electrónico, servidores web, sistemas de almacenamiento de datos, sistemas de comunicación de información, redes telemáticas, a los usuarios de dichos sistemas, etc.

La seguridad en los SI comienza a partir de saber de la existencia de vulnerabilidades relativas a estos sistemas. Una vulnerabilidad no tendría sentido si luego no pudiese ser explotada por un ataque con el objetivo de violar la seguridad del sistema. Es muy común asociar el concepto de vulnerabilidad a error, aunque esta equivalencia conviene matizarla adecuadamente. En esta línea, adoptaremos la siguiente definición del concepto de vulnerabilidad.

> "Una vulnerabilidad de seguridad es un fallo o debilidad en el diseño, la implementación, la operación o la gestión de un sistema, que puede ser explotado con el fin de violar la política de seguridad del sistema"

Una política de seguridad es el conjunto de reglas y prácticas que definen y regulan los servicios de seguridad de una organización o sistema con el propósito de proteger sus recursos críticos y sensibles. En otros términos, es la declaración de lo que está permitido y lo que no está permitido hacer dentro de esa organización.

La política de seguridad es la base de la seguridad de un sistema. En ella se detallan los servicios de seguridad del sistema, se determina quién y/o qué se puede o no hacer con los recursos del sistema, y generalmente se especifica cómo se implementan dichos servicios. La implementación concreta de una política de seguridad se lleva a cabo mediante mecanismos de seguridad. La política no tiene por qué ser una declaración formal, a veces se trata de simples directrices sobre la seguridad del sistema en lenguaje informal.

Por lo general, hablamos de incidente de seguridad para referirnos a cualquier hecho que supone una violación de la seguridad del sistema. En el caso de que el incidente sea intencionado, nos referimos a él como ataque.

> "Un ataque es una agresión a la seguridad de un sistema fruto de un acto intencionado y deliberado que viola la política de seguridad de un sistema"

Existen dos tipos de ataques, los activos que intentan alterar el sistema, sus recursos u operaciones, y los pasivos que intentan aprender o utilizar información del sistema pero no afectan al propio sistema, ni a su funcionamiento.

A forma de resumen podemos ver la siguiente imagen, donde podemos ver la relación entre estos conceptos explicados hasta ahora.

Fig. 1 Relación entre conceptos

Otro aspecto importante que trataremos en este material es el de riesgo y amenaza. Toda vulnerabilidad implica una amenaza al sistema y, por tanto, entraña un riesgo.

Una amenaza es una violación de la seguridad en potencia, que existe a partir de unas circunstancias, capacidad, acción o evento que pueda llegar a causar una infracción de la seguridad y/o causar algún daño en el sistema.

Es muy importante distinguir entre estos dos aspectos (ataque y amenaza). Un ataque es una acción intencionada, realizada directa o indirectamente por un atacante (persona o máquina) al que se le atribuye cierta capacidad de acción inteligente. Por el contrario, una amenaza es la posibilidad de que ocurra una violación de la política de seguridad. Esta violación puede ser provocada por un ataque o por incidentes no deliberados causados de manera fortuita, como desastres naturales.

Una parte importante en la seguridad informática es evaluar el riesgo asociado a un servicio o sistema dentro de la organización. Este riesgo suele ser directamente proporcional a la existencia de vulnerabilidades y amenazas. Aunque hay que tener en cuenta que no siempre a mayor número de vulnerabilidades mayor es el riesgo asociado a un sistema. El riesgo vendrá determinado también por la criticidad o gravedad de la vulnerabilidad. En la figura 2 se muestran los principales conceptos vistos hasta ahora.

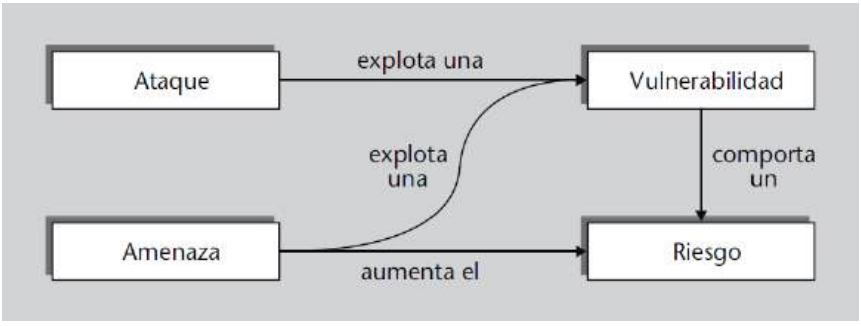


Fig. 2 Diferencias entre los conceptos

Podemos definir el riesgo como la expectativa de pérdida de información expresada como la probabilidad de que una amenaza explote una vulnerabilidad particular con resultados especialmente perjudiciales. Una vez evaluado el riesgo, considerando las posibles amenazas al sistema, el trabajo de los expertos en seguridad informática consiste en desarrollar contramedidas con el objetivo de mitigar dicho riesgo. Hay que tener en cuenta que, dada la actual complejidad de los sistemas informáticos, resulta prácticamente imposible disponer de un sistema libre de vulnerabilidades y amenazas. En esta línea, la figura 3 muestra el proceso de seguridad, el cual se suele percibir como un ciclo, donde se aplican medidas de prevención, detección y reacción.

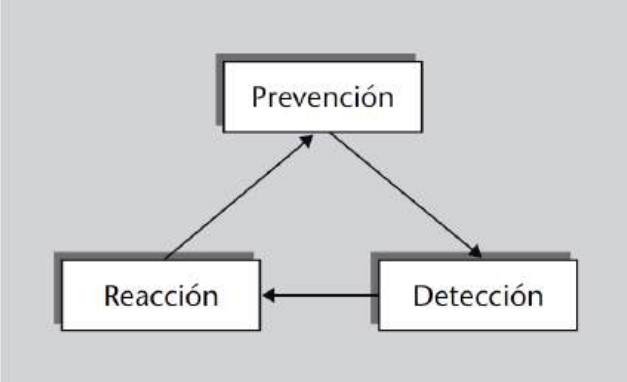
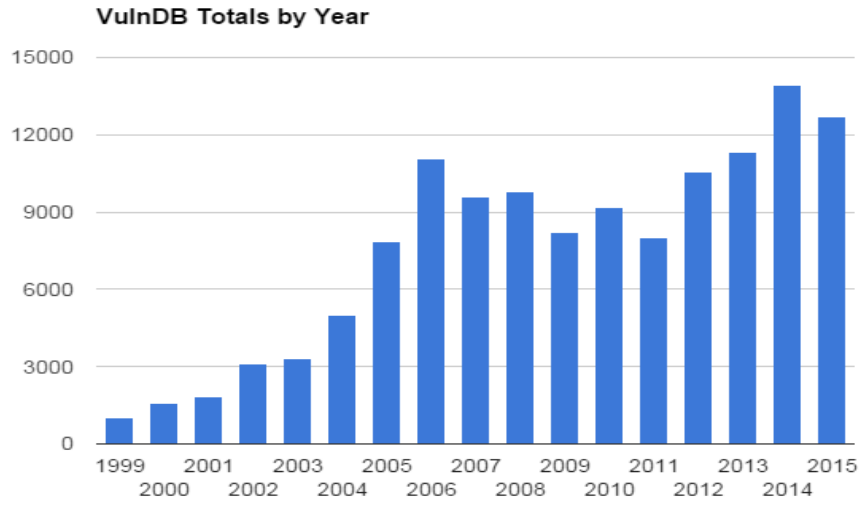


Fig. 3 El ciclo de la seguridad

Ejemplos de vulnerabilidades

El incremento de sistemas informáticos y especialmente de los sistemas conectados a Internet ha hecho crecer drásticamente el número de vulnerabilidades, ya que cada vez hay más sistemas y muchos de los antiguos se mantienen por ser complejo actualizarlos. Estimar el número de vulnerabilidades existentes es difícil, pero existen datos de, por ejemplo, el número de vulnerabilidades diferentes catalogadas, como las mostradas en la figura 4. Notar que no se muestra el número de vulnerabilidades, sino el de vulnerabilidades diferentes identificadas. El hecho de que una vulnerabilidad en concreto tenga mayor o menor presencia no se muestra en este gráfico.



1999	1024
2000	1568
2001	1831
2002	3108
2003	3327
2004	4989
2005	7865
2006	11054
2007	9589
2008	9808
2009	8191
2010	9178
2011	7992
2012	10536
2013	11335
2014	13905
2015	12701

Fig. 4 vulneravilidades diferentes nuevas por año

(Fuente) https://www.riskbasedsecurity.com

Vamos a describir algunos ejemplos de vulnerabilidades muy conocidos y destacados en su momento:

- Sony PSN (PlayStation Network): un ataque en mayo del 2011 a la red de usuarios PSN de Sony acabó con el posible robo de información personal de sus cerca de 70 millones de usuarios, que incluía información de tarjetas de crédito. En dicho ataque se explotó una vulnerabilidad conocida (no se ha revelado a qué sistemas afectan).

- Stuxnet: gusano que infecta sistemas industriales, los sistemas SCADA (Supervisory Control And Data Acquisition) de Siemens para la configuración y el control de procesos industriales. Fue descubierto en julio del 2010 y una particularidad de este gusano es que sus objetivos parecían ser muy concretos: centros de enriquecimiento de uranio de Irán. El gusano explota un total de 4 vulnerabilidades del sistema operativo Windows de Microsoft (2 de ellas eran conocidas, y 2 eran vulnerabilidades de zero-day, sin parche).

- Ataque de Mitnick: en 1994 se produjo uno de los ataques informáticos más publicitados y documentados, mediante el cual Kevin Mitnick consiguió acceder a los ordenadores de Tsutomu Shimomura, situados en la Universidad de California, para robar el código fuente de un teléfono móvil. Mitnick explotó vulnerabilidades en el protocolo TCP de denegación de servicio y de secuestro de sesión

Existen diferentes maneras de poder identificar y clasificar vulnerabilidades, que se pueden hacer públicas para que todo el mundo pueda aplicar medidas preventivas. Muchos problemas de seguridad vienen por no tener los sistemas informáticos actualizados con los últimos parches de seguridad que evitan vulnerabilidades conocidas. Aun así, siempre hemos de asumir que existen vulnerabilidades no conocidas y que pueden dar lugar a incidentes de seguridad. Estas vulnerabilidades no conocidas, de las que es muy difícil protegerse, se denominan vulnerabilidades de día-cero o en inglés zero-day vulnerabilities. Hemos de ser conscientes del riesgo que tiene no actualizar los sistemas, ya que si el fabricante desarrolla un parche para arreglar una vulnerabilidad para cerrarla, el hecho de no aplicar la actualización deja

vulnerable nuestro sistema, ya que los atacantes pueden conocer como entrar a nuestro sistema simplemente comparando el sistema con la actualización y sin ella para descubrirla y explotarla. Por eso es importante aplicar las actualizaciones.

Así mismo, también vemos que muchos ataques informáticos explotan más de una vulnerabilidad. Esto es especialmente latente en ataques que realizan varias acciones diferentes. Por ejemplo un ataque que consiga un acceso remoto y después realizan otro para una escalada de privilegios y obtener un usuario con más privilegios.

1.2. Gestión de vulnerabilidades

Un proceso muy importante dentro de la seguridad informática es la gestión de las vulnerabilidades que se van reportando. Dentro de la seguridad preventiva, uno de los puntos clave es encontrar vulnerabilidades en sistemas existentes que puedan suponer una amenaza ante ataques potenciales. A tal efecto se destinan muchos recursos a diferentes niveles, desde los propios fabricantes de hardware y software, hasta entidades gubernamentales o asociaciones altruistas.

Los equipos encargados de la gestión de vulnerabilidades e incidentes de seguridad suelen recibir el nombre de CERT (*Computer Emergency Response Team*) o CSIRT (*Computer Security Incident Response Team*). La principal tarea de estos equipos es la gestión de incidencias de seguridad. En la práctica, sirven como medio para la difusión de vulnerabilidades de seguridad a usuarios (particulares u organizaciones), que suelen ser reportadas por los propios usuarios

El primer centro de coordinación CERT fue creado en 1988, por DARPA (*Defense Advanced Research Projects Agency*), en el Instituto de Ingeniería del Software (*Software Engineering Institute, SEI*) de la Carnegie Mellon University en Estados Unidos, y actualmente continúa siendo una referencia internacional. La necesidad de crear este centro fue impulsada por el gran impacto que tuvo el gusano Morris. La aparición de Internet hacía posible la rápida distribución de código malicioso, dando lugar a la aparición de virus, gusanos, etc. Como respuesta a esta problemática se crearon estos equipos CERT, que rápidamente se han extendido por todo el mundo. Actualmente, el CERT original de la Carnegie Mellon University se conoce como CERT/CC (CERT Coordination Center), ya que actúa como coordinador y da soporte a equipos CERT (o CSIRT) de nivel nacional (generalmente gubernamentales) en todo el mundo. La gran proliferación de equipos CERT por el mundo ha hecho necesario establecer cierta coordinación entre ellos. Aparte de CERT/CC, existen centros de coordinación de equipos CERT a escala internacional, como FIRST (*Forum of Incident Response and Security Teams*), TF-CSIRT (*Computer Security Incident Response Teams Task Force*) en el ámbito europeo, o colCERT (Grupo de Respuesta a Emergencias Cibernéticas de Colombia, http://www.colcert.gov.co/) en Colombia.

Los equipos CERT disponen de bases de datos y canales de distribución (como listas de correo) para distribuir de forma inmediata la información relativa a las nuevas vulnerabilidades. Su principal tarea es recoger información sobre vulnerabilidades, clasificarlas y publicar su existencia, así como posibles medidas para mitigarlas. La información la suelen suministrar fabricantes, organizaciones o usuarios directamente a un CERT que luego suele distribuir a otros equipos CERT con los que esté coordinado. Así mismo, la distribución puede ser directa y abierta al público, restringida a organizaciones concretas, o a suscriptores. Como recogen y distribuyen información los CERT depende mucho del tipo de CERT y su ámbito de aplicación. En

general, se considera una buena práctica difundir información de vulnerabilidades de modo abierto y gratuito a toda la comunidad de usuarios de Internet con el objetivo de mejorar la seguridad general de la Red.

A la hora de decidir si se hace o no pública una nueva vulnerabilidad, por lo general se plantea un dilema. Por una parte, los partidarios de publicar libremente una vulnerabilidad una vez es conocida; de esta manera, todo el mundo puede conocer su existencia y aplicar las medidas preventivas adecuadas, si es que existen. Por otra parte, hay gente que piensa que hacer pública información relativa a vulnerabilidades equivale a dar armas al enemigo, ya que muchas veces son vulnerabilidades conocidas y aún no corregidas las que se utilizan en los ataques.

En el campo de la seguridad informática, por lo general, se considera como buena práctica el ofrecer la mayor claridad posible sobre problemas y mecanismos de seguridad, y no basar la seguridad de un sistema en la ocultación de información. Un atacante lo primero que hace al querer vulnerar un sistema concreto es recoger información sobre ese sistema: sistema operativo, software que utiliza, etc., y luego busca vulnerabilidades que puedan ser explotadas en dicho sistema. El intentar ocultar información sobre el sistema para impedir que el atacante pueda buscar vulnerabilidades en él es algo común. Sin embargo, esta es una práctica de dudosa eficacia y puede llegar a dar una falsa sensación de seguridad. Por ello, muchos expertos abogan por asumir que el posible atacante dispone de esa información y, por tanto, no es necesario ocultarla.

1.3. Identificación de vulnerabilidades

Para poder identificar vulnerabilidades, existen identificadores únicos que impiden que se publiquen duplicados y facilitan la posibilidad de hacer referencia a vulnerabilidades concretas para obtener toda la información posible. El sistema de identificación más importante a escala internacional es el CVE (*Common Vulnerabilities and Exposures*). CVE se presenta como un estándar de nombres de vulnerabilidades de seguridad informática de uso gratuito y público. Se autodefine como un diccionario de vulnerabilidades (no como una base de datos), donde cualquiera puede buscar el nombre (identificador) que recibe una vulnerabilidad concreta.

Según CVE, una vulnerabilidad es un estado de un sistema informático (o conjunto de sistemas) que cumple alguno de los siguientes casos:

- Permite a un atacante ejecutar comandos como otro usuario.
- Permite a un atacante acceder a datos violando las restricciones de control de acceso específicas para dichos datos.
- Permite a un atacante suplantar a otra entidad.
- Permite a un atacante llevar a cabo una denegación de servicio.

Como se puede ver, la definición de vulnerabilidad de CVE es bastante más restrictiva y específica que la definición genérica que hemos adoptado anteriormente. Esto es así para poder restringir un poco la aplicabilidad de CVE, ya que de otra manera resultaría muy difícil poder dar cabida a todas las vulnerabilidades posibles. Hay que tener en cuenta también que CVE nace y se gestiona bajo la esponsorización de agencias de inteligencia y defensa cuya principal preocupación es la defensa ante atacantes (de aquí la insistencia en el atacante en la definición de las vulnerabilidades).

1.4. Clasificación de vulnerabilidades

Existen varias clasificaciones de vulnerabilidades, el hecho de adoptar una clasificación u otra viene muchas veces condicionado por el propósito de dicha clasificación. En nuestro caso, y con el objetivo de proporcionar una visión global de la seguridad informática, adoptamos una clasificación que se basa en el tipo de sistema al que afecta la vulnerabilidad. De esta manera, distinguimos entre:

- **Vulnerabilidades de bajo nivel y software malicioso**. Serán las vulnerabilidades que afectan al sistema operativo y aplicaciones a bajo nivel propiciadas generalmente por errores en la programación, como *buffer overflows* o *race conditions*. También se incluye en este tipo de vulnerabilidades el estudio de software malicioso, como virus o gusanos que explotan este tipo de vulnerabilidades.
- **Vulnerabilidades de red**. Son vulnerabilidades que afectan al software y componentes de red o interconexión de redes. Estas vulnerabilidades pueden ir desde redes locales a Internet. Principalmente, se observan vulnerabilidades en los diferentes protocolos de red, así como vulnerabilidades derivadas del análisis de tráfico.
- **Vulnerabilidades en aplicaciones web**. La importancia de Internet y las aplicaciones web, desde el comercio electrónico, las redes sociales, o lo que actualmente se denomina como *cloud computing*, provoca que dichas aplicaciones merezcan un apartado propio. Estas son vulnerabilidades propias de aplicaciones pensadas para ser ofrecidas mediante una interfaz web directamente y a las que generalmente tienen acceso un gran número de usuarios. En general, se consideran vulnerabilidad de más alto nivel que las del primer apartado, e incluyen *cros-site scripting*, inyección de código, etc.
- **Vulnerabilidades de ingeniería social**. Este apartado concentra las vulnerabilidades asociadas a los propios usuarios de los sistemas informáticos. Tienen más relación con el aspecto psicológico de dichos usuarios que con problemas puramente técnicos. Ejemplos típicos son el *spam, phising*, etc. Éstas las trataremos en detalle.

Otra posible clasificación de las vulnerabilidades se basa en la identificación del servicio al cual afectan. De este modo, si tomamos la clasificación que tradicionalmente se aplica a los servicios de seguridad (confidencialidad, integridad y disponibilidad), tenemos vulnerabilidades que comportan:

- Pérdida de integridad.
- Pérdida de confidencialidad.
- Pérdida de disponibilidad.

En este caso, la clasificación va orientada a identificar qué pueden permitir hacer estas vulnerabilidades a un atacante. Por ejemplo, una vulnerabilidad que permita una denegación de servicio es manifiestamente una vulnerabilidad de disponibilidad, así como vulnerabilidades que permitan espiar comunicaciones lo son de confidencialidad.

1.5. Las vulnerabilidades de ingeniería social

Si tenemos en cuenta a las personas como parte integrante de estos sistemas de información, aspecto que muchas veces es descuidado en los análisis de seguridad de las organizaciones, resulta que suele ser, con diferencia, el eslabón más débil de toda la cadena de seguridad. Aunque la consideración de este factor humano en el estudio de las vulnerabilidades de un sistema parece obvio, hay circunstancias que favorecen su exclusión. El primer impedimento está en la propia naturaleza del problema, ya que puede pensarse que está fuera del dominio de las tecnologías de la información y de las comunicaciones y descartarlo en vez de buscar su integración. Sin lugar a dudas, considerar a las personas como parte del sistema dificulta enormemente su análisis. Puede pensarse que en muchos aspectos este análisis de la actividad humana alrededor de los sistemas de información pertenece más al dominio de la psicología social.

Una de las claves para afrontar con éxito el problema de la inclusión del factor humano en el análisis de seguridad de un sistema de información es la identificación de manera precisa de todos los elementos que intervienen en él, y el seguimiento de una metodología concreta. Si no se realiza de esta manera, es fácil adentrarse demasiado en el ámbito de la psicología social, en particular en el análisis de la confianza humana, o el de la persuasión, que aumenta la complejidad y hace más difícil el diseño de soluciones prácticas para la detección y prevención de ataques. Una parte importante del trabajo de investigación realizado sobre este tema no se encuentra en el dominio de informática, sino en el de las ciencias sociales.

Definiremos los conceptos previos que usaremos al respecto de las vulnerabilidades dentro de la ingeniería social

> "En el contexto de la seguridad informática, llamaremos ingeniería social a la secuencia de acciones que tienen como finalidad la obtención de información, el fraude o el acceso no autorizado a sistemas informáticos, y que ha implicado en algún momento la manipulación psicológica de las personas implicadas."

Un sistema vulnerable a ataques de ingeniería social será, por tanto, aquel susceptible a ser atacado mediante estas técnicas, por lo tanto existe una intervención humana en algún momento del proceso. El punto que hace diferentes a los ataques de ingeniería social de otros ataques es la manipulación psicológica de las personas que se realiza en algún momento. En el siguiente subapartado se verá de qué manera se puede realizar esta manipulación.

1.5.1. Aspectos explotables

La base de la ingeniería social es la aplicación de técnicas de psicología para conseguir, por medio del engaño o manipulación, que las personas realicen ciertas acciones o desvelen la información deseada. Encontramos la base teórica de estas técnicas en la psicología social. Hay tres aspectos de ésta que serán importantes para los propósitos de la ingeniería social: las técnicas para la persuasión y la influencia, las actitudes y creencias que afectan a la interacción social, y la falsa confianza.

Técnicas de persuasión e influencia

En la psicología social se identifican dos modos que se pueden utilizar para persuadir a las personas. Por un lado, utilizando argumentos sistémicos y lógicos para estimular una respuesta favorable, induciendo a la persona a pensar detenidamente y dar un

consentimiento. Por otro, de una manera más lateral, basada en indicaciones periféricas y atajos mentales para eludir el argumento lógico y la contraargumentación para intentar provocar la aceptación, sin pensar en profundidad sobre el tema. Un modo como se puede hacer a una persona más susceptible a esta persuasión periférica es a través de mensajes o acciones en el inicio de la interacción que provoquen reacciones emocionales, como excitación o miedo. Estas reacciones, así como otras formas de distracción, sirven para interferir en la capacidad de pensamiento lógico de la víctima y permitir explotar esta persuasión periférica para realizar ataques de ingeniería social.

Una gran parte de la bibliografía de la psicología social reconoce al menos siete factores que se basan en la persuasión periférica que son efectivas en la persuasión e influencia de personas (Cialdini, 2008):

1) **Autoridad**. La mayoría de las personas son muy receptivas a la autoridad. En las condiciones adecuadas, la autoridad se cuestiona poco, y la tendencia a obedecer instrucciones de alguien que dice tenerla es muy alta, incluso cuando se reciben de manera indirecta (por ejemplo, a través del teléfono). Un ejemplo de recurso a la autoridad es pedir que abran una puerta haciéndose pasar por un agente de policía.

2) **Parquedad**. Cuando hay escasez de un bien o servicio en el que se podría estar interesado, o su disponibilidad es solo por un período de tiempo limitado, las personas tienden a quererlo más. Saber que esta disponibilidad limitada puede crear competencia entre otras personas para su adquisición incrementa más aún el deseo de adquirirlo. Un ejemplo de recurso a la parquedad es anunciar que solo quedan tres unidades de un pendrive de gran capacidad para las primeras personas que contesten una encuesta.

3) **Similitud**. Existe una innegable tendencia humana a que nos guste aquello que es similar a nosotros mismos. Tener elementos en común, como por ejemplo aficiones, gustos musicales o artísticos, o incluso compartir los mismos problemas, crean un fuerte incentivo para tratar a alguien de una manera especial, más favorable. Un ejemplo de recurso a la similitud es comentar casualmente que hemos nacido en la misma ciudad para pedir posteriormente una información no pública.

4) **Reciprocidad**. Cuando alguien da algo, o promete que lo hará, las personas sentimos una fuerte tendencia a devolver algo a cambio, incluso si lo que se ha recibido nunca fue solicitado. Esta regla básica de la interacción humana es innata a las personas y funciona aun cuando el coste de lo dado y recibido es muy diferente. Un ejemplo de recurso a la reciprocidad es desconectar inadvertidamente a alguien el cable de acceso a la red de un ordenador simulando una avería para solucionar el problema después y pedir un favor a cambio.

5) **Compromiso**. Si se incumplen las promesas, las personas tienen la certeza de que serán consideradas de poca confianza por otras personas. Esto provoca que las personas realicen un gran esfuerzo para cumplir con los compromisos adquiridos. Un ejemplo de compromiso: Si se convence a una persona de que hizo una promesa, cuando en realidad no la hizo, esta se esforzará por cumplirla.

6) **Consistencia**. Las personas tienen tendencia a actuar de manera consistente con sus acciones pasadas, aunque en la situación actual mantener esta coherencia ya no tenga sentido, ya que creen que al no hacerlo serán consideradas de poca confianza (Cacioppo y otros, 1986). Ejemplo de consistencia: Se puede predecir la reacción de una persona a partir de sus acciones pasadas.

7) **Prueba social**. Ante la duda sobre si realizar o no una acción, o cuál es la más apropiada, las personas deciden según el comportamiento de otras personas cercanas. Esto puede llevar a realizar acciones que van contra los intereses de las personas sin que se lleguen a pensar detenidamente. Ejemplo de prueba social: Si tres personas compinchadas muestran el DNI a un falso portero, la siguiente persona también lo mostrará.

Actitud y creencias

Otro aspecto de la psicología social importante en la ingeniería social son las creencias y actitudes. Las personas generalmente piensan que los otros comparten sus mismos sentimientos e ideas, lo que se denomina el efecto del falso consenso. El ingeniero social puede utilizar estas creencias para manipular a la víctima y conseguir que realice alguna acción. Experimentos de psicología social muestran que incluso algunas de las personas que analizan detenidamente los mensajes persuasivos dejan de hacerlo cuando perciben que el origen es más honesto. Así, algunas víctimas de ingeniería social tienden a confiar más en sus creencias o impresiones personales sobre la honestidad que les ofrece alguien, que en el análisis objetivo del contenido del propio mensaje.

Falsa confianza

Por defecto, las personas tienen una tendencia natural a confiar en sus congéneres, aunque no haya un histórico de interacciones que avale dicha confianza. Es más, ser desconfiado de entrada suele ser mal visto socialmente y supone una percepción negativa de las personas. Cuando no existe una percepción elevada de riesgo, se concede de entrada el beneficio de la duda en contra de lo que aconsejaría un pensamiento racional detenido, hasta ciertos límites (no se suelen dejar la llaves de casa al primer desconocido que nos las pide, por ejemplo). Incluso si de entrada una persona decide no confiar, suele ser sencillo hacerle cambiar de idea simplemente haciéndole ver que está desconfiando.

Además de la confianza natural directa, resulta fácil realizar acciones que se perciben como indicadores que aumentarán la confianza que se deposita en alguien. Sin llegar a la persuasión, el simple contacto continuado en el tiempo aumenta automáticamente la percepción de confianza. Por ejemplo, confiaremos más en alguien que hemos visto cada mañana en la biblioteca durante el último mes que en alguien que acabamos de conocer, aunque racionalmente no haya ninguna razón para ello.

El pretexto es una de las técnicas más conocidas para ganar confianza. Establecer contacto con una persona a través de una historia falsa suele ser suficiente para que esa persona deposite confianza sobre el atacante y reduzca su reluctancia a ofrecer información privada. Esta percepción de confianza es uno de los puntos explotables de las personas más utilizados en la ingeniería social y que suele ser más efectivo.

Otros aspectos explotables

A parte de los aspectos ya vistos, hay otras cualidades humanas que son explotables desde el punto de vista de la ingeniería social. A continuación encontramos algunos ejemplos:

Curiosidad	Ignorancia	Cortesía	Avaricia	Apatía
Paranoia	Lujuria	Orgullo	Inseguridad	Amabilidad
Recelo	Solidaridad	Ira	Envidia	Empatía
Consuelo	Caridad	Compasión	Amor	Indulgencia

Estas cualidades humanas, no siempre presentes en todas las personas, representan puntos débiles que facilitan el engaño y la manipulación. La curiosidad, por poner un caso, es una cualidad humana muy común. ¿Quién puede resistirse, por ejemplo, a ver el contenido de un pendrive encontrado accidentalmente en la calle? Muchos ataques de ingeniería social utilizan este rasgo humano de una manera u otra, numerosas infecciones de sistemas aislados de Internet se producen por esta técnica.

Otras se perciben como cualidades deseables en una buena persona, y son incluso inculcadas en algunas religiones, como la caridad, la compasión o la amabilidad. Muchas personas deberían hacer un gran esfuerzo para resistir el impulso de ofrecer ayuda al necesitado o de aliviar el sufrimiento del que padece. ¿No se saltaría alguna norma de la política de seguridad de la empresa para cubrir el error de alguien con una familia a cargo, parar sus lloros, y quitarle el miedo del despido?

1.5.2. El proceso de la ingeniería social

Generalmente, los ataques de ingeniería social se basan en la realización de acciones de este tipo para conseguir resultados parciales, que podrán combinarse para poder realizar nuevas acciones, y así reiteradamente hasta conseguir el objetivo final. Es lo que denominamos el proceso de la ingeniería social. La representación gráfica de este proceso para casos concretos nos permitirá estudiar sus características y realizar un análisis para determinar cuáles son los puntos críticos del sistema o diseñar estrategias de prevención.

Cada una de las acciones particulares que se realizan en estos ataques vendrá caracterizada a tres niveles:

1. **Estrategia**. Determina el tipo de la acción y los objetivos que se persiguen. Básicamente existen tres posibles estrategias: recogida de información, forzar una acción y ataque directo. Todas las acciones pertenecerán a una u otra de estas categorías.

2. **Técnica**. Para cada estrategia existen varias técnicas concretas para conseguir sus objetivos. Estas técnicas pueden ser muy variadas y pueden ir apareciendo nuevas a medida que se desarrollen nuevas tecnologías.

3. **Vía**. La vía identifica exactamente el medio por el cual se utiliza una cierta técnica en una acción. Una misma técnica puede aplicarse a través de vías diferentes.

En la representación gráfica de los ataques de ingeniería social, cada acción se identifica mediante la estrategia a la que pertenece, la técnica utilizada y la vía de aplicación, y está representada en un nodo.

Dentro del proceso de la ingeniería social, las acciones se realizarán secuencialmente hasta conseguir la finalidad del ataque. En algunos casos serán necesarias varias acciones previas para realizar otras, o existirán acciones alternativas para conseguir un mismo resultado parcial. Los diagramas DAIS (diagramas de ataques de ingeniería social) nos permitirán representar gráficamente estas relaciones de secuencialidad entre las acciones.

1.6. Estrategias y técnicas

Como hemos visto, el proceso de la ingeniería social comprende un conjunto de estrategias, técnicas y vías para alcanzar un objetivo final. De acuerdo con ello, en este apartado presentamos una taxonomía de dichas estrategias según tres categorías principales:

1. recogida de información,
2. forzar una acción y
3. ataque directo.

Describamos cada una de estas categorías de manera más detallada, mostrando también algunos ejemplos de técnicas y vías asociadas a cada una. Es importante remarcar que, a diferencia de la categorización que se hace para las estrategias, que es cerrada, tanto las técnicas como las vías asociadas que se exponen aquí no son únicas. Es decir, las técnicas y vías aquí presentadas deben ser consideradas como ejemplos particulares. De hecho, tanto las técnicas como las vías son suficientemente abiertas como para que no sea posible realizar una clasificación e incluso, en un futuro, es posible que aparezcan nuevas que desconocemos. Por lo tanto, es posible encontrar otras técnicas asociadas a cada una de las estrategias diferentes a las aquí expuestas.

1.6.1. Recogida de información

> "La estrategia de recogida de información tiene como finalidad captar información útil para el atacante, y para la cual no debería hipotéticamente tener acceso."

Se trata de la estrategia más sencilla que un atacante puede emplear. La razón de esto está en el hecho de que en la mayoría de los contextos en los que el ingeniero social puede actuar, siempre suele existir una cierta fuga de información, que puede obedecer a dos patrones posibles. En primer lugar, la información puede ser obtenida al ser expuesta de manera inconsciente por los usuarios. Aquí, el concepto de inconsciencia lo expresamos en relación con las consecuencias que puede tener desde un punto de vista de la ingeniería social. Cabe destacar el hecho de que este primer patrón siempre implica la existencia de algún componente social o humano. Por tanto, la utilización de, por ejemplo, un *sniffer* de red de manera aislada no es considerado como parte de la estrategia de recogida de información. En segundo lugar, el siguiente patrón se sustenta en obtener la información mediante la interacción

del ingeniero social con otras personas. Cada uno de estos patrones los denominaremos como fuga de información sin interacción humana y fuga de información por interacción humana, respectivamente.

Patrón de fuga de información sin interacción humana

Dentro de la fuga de información sin interacción humana encontramos varias técnicas. En concreto, existen tres posibles:

1. búsqueda en Internet,
2. el agenciamiento físico de información y
3. la observación.

En relación con la técnica de búsqueda en Internet, un usuario malintencionado puede encontrar una gran cantidad de información que le puede ser de utilidad simplemente explorando por la Red de redes. Para esto, el atacante puede utilizar algún buscador web empleando filtros diseñados específicamente, o explorando directamente webs conocidas de compartición de información. Algunos tipos de vías para esta estrategia pueden ser las redes sociales, los foros o las páginas web corporativas. En estos entornos la cantidad de información que puede conseguirse es considerable y muy variada. Algunos ejemplos de tal información pueden ser: relaciones entre personas, información sobre la red o los sistemas de una empresa, personas asociadas a cargos, o números de teléfono, entre otros.

La técnica de agenciamiento físico de información se basa en la apropiación no autorizada de elementos físicos que contienen información, y de los cuales el ingeniero social extraerá los datos de su interés posteriormente. Algunos ejemplos de este tipo de información pueden ser nombres, direcciones, números de teléfono, cuentas bancarias, cuentas de correo, etc. Esta categoría podemos subdividirla en dos subcategorías según si el tipo de elemento físico es desechable o no. En el primer caso, la apropiación se realizará en sistemas destinados al almacenamiento de desechos para su posterior recogida y tratamiento. Por ejemplo papeleras, contenedores de basura, contenedores de papel para su reciclado, máquinas destructoras de documentos, etc.

Aunque esto pueda parecer inverosímil, la realidad muestra que existe una cierta despreocupación por parte de organizaciones y empresas por la destrucción efectiva de elementos que contienen información. Ya no solo de información en papel, sino también en dispositivos tales como discos duros o pendrives. En este contexto, existen incluso empresas especializadas en la destrucción de componentes que contienen información. En cuanto a la segunda subcategoría, los elementos físicos no serán desechables, y la apropiación puede realizarse en cualquier lugar frecuentado por usuarios. Esta subcategoría incluye vías como pueden ser agendas, documentos, cartas, teléfonos móviles, portátiles, pendrives, etc.

Por último, la técnica de observación se centra en la obtención de información basándose exclusivamente en examinar el comportamiento de personas, hechos u objetos. Asimismo, sobre esta observación se pueden aplicar razonamientos lógicos basados en relaciones entre elementos visualizados para extraer conclusiones. Alguna de las vías posibles que puede utilizar un ingeniero social son las cámaras ocultas, las relaciones o los comportamientos sociales, el seguimiento de personas, etc. A esta categoría también pertenece la vía que denominados como *shoulder surfing*. Esta se basa en observar detenidamente la pulsación de las teclas realizadas por un usuario

en el proceso de entrada a un sistema. Durante este proceso, el usuario malicioso captará visualmente las pulsaciones tanto para el nombre de usuario como para su contraseña.

Patrón de fuga de información por interacción humana

Este patrón incluye dos tipos de técnicas. La primera la denominamos interacción directa y la segunda, interacción mediante un medio de comunicación.

> "La interacción directa es aquella técnica en la que la acción recíproca de comunicación entre el ingeniero social y la víctima se realiza de manera personal y directa, es decir, sin el uso de ningún medio de comunicación entre ambas personas."

Esta técnica puede afectar a tantas personas como puedan existir en una corporación u organismo. Algunos ejemplos pueden ser vigilantes, repartidores, recepcionistas, etc. En este caso, la vía utilizada es la misma conversación que se establece entre el ingeniero social y las personas implicadas.

La técnica de interacción mediante un medio de comunicación es la homóloga a la interacción directa, con la diferencia de que existe un medio de comunicación entre el ingeniero social y la víctima. En este caso, la vía vinculada tiene multitud de formas, tales como teléfono, carta, correo electrónico, fax, aplicaciones de mensajería instantánea, software de VoIP, etc.

1.6.2. Forzar una acción

> "La estrategia de forzar una acción es aquella que emplea alguno de los aspectos explotables presentados anteriormente, y cuya finalidad es la de conseguir que alguien realice una acción en beneficio del ingeniero social."

Así, una técnica que explotase la autoridad podría ser la impersonación de alguien que ocupase un cargo superior a la víctima, y a la que persuadiría para que modificase la regla de un cortafuegos amenazándole con un posible despido en el caso de no cooperar. En este caso, la vía empleada podría ser, por ejemplo, el teléfono.

Keylogger:
Un ejemplo diferente de técnica podría ser la captura de la contraseña de un usuario utilizando un *keylogger*. Asociada a esta técnica se podría emplear un *pendrive* preparado como vía que contendría el *keylogger* camuflado y que sería puesto en un lugar cercano a la víctima. La víctima, al ver el *pendrive* y desconocer su propietario, podría tomarlo e introducirlo en su máquina influenciada por la curiosidad de conocer su contenido. A partir de aquí, el software malicioso podría instalarse de manera automática empleando algún tipo de vulnerabilidad (Larimer, 2011) para, posteriormente, enviar las pulsaciones de teclas a través de la red.

Explotar la empatía
Otro ejemplo de técnica estaría basado en explotar la empatía. En particular, el atacante podría "dar pena" a una víctima para que esta proporcionase ayuda al ingeniero social y le facilitase, por ejemplo,

información sobre la arquitectura de una red con la excusa de que le han amenazado en despedirlo si no soluciona un problema. Aquí la vía empleada sería la misma conversación que mantendrían el atacante y la víctima.

1.6.3. Ataque directo

La estrategia de ataque directo comprende aquellos ataques de carácter técnico que forman parte de un proceso de ingeniería social. Como ya hemos visto, estos tipos de ataque pueden ser finales o intermedios respecto al proceso de ingeniería social. Estos ataques explotan alguna de las vulnerabilidades que hemos ido viendo a lo largo de esta asignatura. Algunos ejemplos particulares de técnica empleada en este tipo de estrategia podrían ser un ataque de denegación de servicio vía un SYN flooding o una escalada de privilegios vía un desbordamiento de un buffer.

1.7. Casos prácticos

Vamos a ver algunos ejemplos prácticos ficticios de ataques de ingeniería social. Cada uno de los casos que se exponen han sido estructurados de la misma manera. En primer lugar, se expondrá cuál es el objetivo final que persigue el usuario malicioso. Y seguidamente, se relatará cuál es el proceso del ataque enumerando los distintos pasos que se realizan hasta que se alcanza el objetivo final.

1.7.1. Caso 1: Robo a un banco electrónico

Objetivo

Realizar un robo en un banco que opera por Internet. Es decir, realizar una transferencia monetaria a una cuenta remota.

Escenario

En la sala de juntas de *e-bank*, un banco que opera por Internet, se respira un aire tenso. La directora general está explicando que durante la ausencia del jefe de seguridad ha habido un robo importante (de hecho, una transferencia irrevocable no autorizada de muchos Pesos a una cuenta de las Islas Caimán). Las fuertes medidas de seguridad informática no parecen haber sido efectivas, pero todavía no saben qué ha fallado. De repente, Santiago, que escuchaba con atención los hechos, abre mucho los ojos y empieza a sonrojarse.

Hace unos días, Santiago recibía una llamada. Era de un técnico de los servicios de informática de la propia empresa que lo llamaba en relación con un cambio de software que se había de producir en breve. Carlos ya sabía de este cambio por una circular interna (recordaba haberla tirado a la papelera). El técnico le explicó que había hablado con el jefe de seguridad, Juan David Rodríguez, antes de que marchara y habían acordado que le pasarían el código de seguridad para la administración de

cuentas. Eso era, le explicó, para comprobar que todo funcionara correctamente en la nueva versión antes de ponerla en producción. La clave era diferente cada día, y por ello no se la había dado el propio Roque antes de irse. A Santiago no le hacía mucha gracia dar aquella clave, pero lo hizo finalmente considerando racionalmente la situación: el técnico sabía dónde estaba el jefe, sabía que él estaba a cargo ahora, y, como le hizo ver, no dársela supondría retrasar el cambio del software por culpa suya y, muy probablemente, una reprimenda de Juan David. Además, al conectarse desde su navegador a la URL que le dio el técnico pudo validar la veracidad de lo que le decía: bajo el logotipo corporativo estaba la hoja de seguimiento del proceso de cambio, y en la tarea 16 decía que él mismo debía proporcionar el código al recibir esta llamada.

Unas semanas antes, alguien "buceaba" la basura en los contenedores de papel de e-bank. Algunos empleados que lo vieron pensaron que era algún indigente buscando cartón para venderlo. Nada más lejos de la realidad. Este personaje estaba muy contento porque había encontrado una circular donde se avisaba de un cambio de software, un resguardo de una reserva de un billete de avión a nombre de Juan David Rodríguez, y algunas copias de facturas de proveedores de servicios. No le costó mucho establecer cierta confianza con la recepcionista de la empresa: después de muchas llamadas haciendo ver que era de una de las empresas proveedoras, ¡ya eran casi amigos! Sin mucho esfuerzo le sustrajo qué posición ocupaba en la empresa Roque ("por cierto, ¿no conocerás a un tal Juan David Rodríguez que trabaja allí, que es primo de mi cuñada?"), y quién lo sustituía…

En este caso tenemos un único nodo de relación de Conjunto, que define dos conjuntos de acciones que realizar: por un lado, la obtención de la información sobre quién es el sustituto del responsable de seguridad; y, por otro, la preparación de un sitio web que será necesario en la siguiente parte del ataque. A partir de aquí vendrá el punto más difícil, conseguir las claves de acceso a través de la persuasión y el engaño, para finalmente culminar el ataque con la sustracción económica a través del desvío a una cuenta extranjera.

1.7.2. Caso 2: modificación de la página web de un portal

Objetivo

Modificar la página web principal de un portal de venta de libros como medida para desacreditarlo.

Escenario

Partimos de la tienda online e-Books, una tienda que vende libros a través de Internet, sabe lo importante que es tener una buena imagen desde el punto de vista de la seguridad i la 'marca'. Esta es consciente de que noticias acerca de ataques en los que se compromete la seguridad de portales que ofrecen venta de productos perjudica seriamente a las ventas, ya que este tipo de situaciones genera desconfianza hacia los usuarios compradores.

Desde hace unos meses, las ventas de e-Books han descendido considerablemente tras la aparición de una nueva web de venta de libros, CheapBooks, que realiza ofertas muy competitivas y contra las que e-Books no puede competir, no puede bajar los precios. Tras este período de tiempo de crisis para e-Books, y en vista de la inminente quiebra de la empresa, el directivo de la empresa decide poner en práctica una campaña de desacreditación de CheapBooks. Para ello, decide contratar a

'BlackMan' —un experimentado individuo en comprometer la seguridad de sistemas— con la finalidad de modificar la web principal del portal de la competencia.

Tras realizar BlackMan un análisis de la seguridad del portal de CheapBooks, concluye que aparentemente no existe ninguna brecha remota que explotar de manera sencilla y que le permita comprometer el sistema. Asimismo, determina que la máquina remota es un sistema GNU/Linux corriendo un servidor web Apache. También descubre que el mismo portal contiene un *frontend* de administración para el que se requiere un nombre de usuario y contraseña.

En esta situación, y dado que BlackMan sabe que en muchas ocasiones el eslabón más débil en seguridad son las personas, decide emplear técnicas de ingeniería social. Después de buscar por la web de CheapBooks, encuentra el nombre y el teléfono tanto del responsable IT, como del administrador de la red, así como el correo electrónico de este último, al que dirigirse en el caso de haber algún problema con la página web. (Recordemos que en algunos casos en la consulta Whois de los dominios ya tenemos esa información….)

En primer lugar, BlackMan decide intentar llamar por teléfono al administrador de la red para conseguir la contraseña del *frontend* que le dé acceso a la parte restringida del portal. Para conseguir esto, se intenta hacer pasar por el responsable de IT y le pide que le proporcione la contraseña para modificar una configuración urgentemente. A pesar de ello, el administrador de la red, con una dilatada experiencia en seguridad, reconoce rápidamente el intento de ataque al no coincidir el timbre de voz con la de su inmediato superior. Seguidamente el administrador de la red informa a su jefe del intento de ingeniería social.

Tras este fracaso, BlackMan decide dejar pasar un tiempo como medida preventiva para no levantar sospechas de que alguien está intentando comprometer la seguridad de CheapBooks. Transcurrido un mes, decide emplear una estrategia de ingeniería social más elaborada. En esta ocasión llama por teléfono al responsable de IT haciéndose pasar por un proveedor de hardware. En esta conversación, BlackMan convence al responsable de IT para mantener una reunión y poder presentarle así supuestas ofertas de interés para la empresa. Antes de mantener la reunión, BlackMan prepara un pendrive con un *keylogger* programado por él mismo, y que no es detectable por ningún software antivirus actual al implementar distintos mecanismos de ofuscación. Este software malicioso es almacenado en el pendrive con apariencia de ser un documento de nóminas, cuando en realidad se trata de un ejecutable. En concreto, se denomina "Nominas.doc_____.exe". Durante la reunión, realizada en el despacho del responsable de IT, BlackMan deja caer el pendrive sin que nadie se percate. Durante el tiempo que dura la presentación de productos, BlackMan intenta ser lo más convincente posible presentando datos reales que ha conseguido previamente. Al despedirse, le pide el correo electrónico al responsable de IT con el pretexto de que le enviará más información.

Al cabo de unos días, el servicio de limpieza encuentra el pendrive en el despacho del responsable de IT y, pensando que pertenecería a él, se lo dejan encima de la mesa. Al llegar ese día al despacho el responsable de IT y ver el pendrive tiene la curiosidad de saber qué contiene y lo conecta a su portátil. Al inspeccionar el contenido ve un supuesto documento con el nombre "Nominas", sin darse cuenta de que en realidad es un ejecutable. Tras esto, decide abrirlo sin ser consciente del peligro que supone. Procede entonces a hacer un "doble clic" en el archivo y, seguidamente, el mismo ejecutable malicioso muestra una ventana diciendo que el documento está corrupto. Tras esto, el responsable de IT piensa que por algún motivo aquel documento está incompleto o es erróneo y no presta mayor atención a lo ocurrido. Después de esto, el

portátil del responsable queda infectado por el *keylogger* que preparó BlackMan y, a partir de entonces, todas las pulsaciones de teclas son enviadas por la red a BlackMan. Gracias a esto, no lleva más de una hora a BlackMan tener en su poder las credenciales de acceso como administrador a una máquina de la red interna denominada neptuno.

Al percatarse BlackMan de que la máquina neptuno forma parte de la red interna de CheapBooks y de que, por tanto, tiene una IP de un rango privado, es decir, que no puede acceder directamente a ella desde el exterior, decide proceder de la siguiente manera. Al haber obtenido el correo electrónico del responsable de IT durante la reunión, y conocer el del administrador de la red, envía a este último un correo falseando el remitente y haciéndose pasar por su superior. Esto es posible, ya que el sistema de correo de electrónico de CheapBooks no implementa ninguna tecnología como podría ser OpenSPF o DKIM. El contenido de dicho correo es el siguiente:

```
Subject: Habilitar NAT hacia neptuno!
From: Juan Gómez Sánchez <JGomez@cheapbooks.com>
To: webadmin@cheapbooks.com
Date: 01-06-11 13:50
Hola José!
Este próximo fin de semana necesitaría poder acceder a
la máquina neptuno desde casa. Por favor, añade una regla
en el firewall para hacer NAT redirigiendo el puerto 2222
hacia el 22 de la máquina neptuno.
¡Buen fin de semana!
```

Al recibir el responsable de la red el correo procede a realizar la petición. Durante ese fin de semana BlackMan se conecta a la máquina neptuno desde el exterior, gracias a la redirección NAT que ha podido conseguir, y se da cuenta que desde ella no puede acceder directamente al servidor web para modificar su contenido. Sin embargo, en el archivo `/etc/hosts` de la maquina neptuno descubre que la IP del servidor web es 10.0.0.10. Después de un rápido análisis del servidor web desde la red interna, concluye que la única manera de acceso a este es físicamente o a través del *frontend* que ya conocía de hacía tiempo. Entonces, decide esperar al lunes, momento en el que lanza un ataque de tipo *Man-in-the-Middle* empleando *ARP-Spoofing* desde dentro de la misma red, que le permite capturar las credenciales del administrador de la red que le autentican en el *frontend*. En ese momento, BlackMan, con pleno acceso al servidor web, modifica la página web principal. En esta deja un mensaje donde se proclama que la seguridad de CheapBooks es deficiente y que los clientes no deberían fiarse de las compras que realizan.

1.7.3. Caso 3: Modificación de notas en una universidad

Objetivo

Alterar la nota de una asignatura en el expediente electrónico de un alumno en una universidad.

Escenario

No podía ser que le suspendieran Criptografía sólo por haber tenido un mal día cuando hicieron el examen final, se consideraba buen estudiante. Tenía que actuar… se modificaría el 4 que le había puesto el profesor, por un 9, una nota más acorde a su nivel, creía que esa era la nota justa.

Después de averiguar más o menos cómo funcionaba el proceso de introducción de calificaciones finales en el expediente electrónico y cierre de actas, por medio de los manuales de ayuda para profesores publicados en la página web de la universidad, ya podía empezar a urdir su plan de ataque basado, cómo no, en la ingeniería social. Después de todo, estudiar el criptosistema utilizado para encontrar vulnerabilidades tampoco acababa de ser lo suyo.

Intentar conseguir directamente la contraseña del profesor estaba descartado. Era un profesor de seguridad, así que el riesgo era grande. El primer objetivo, por tanto, sería saber la máquina donde estaba el servidor central de LDAP. Si comprometía esa máquina, podría conocer, probablemente, la contraseña de su profesor para entonces entrar utilizando su identidad y realizar el cambio de nota. Después de algunas llamadas para intentar averiguar qué ordenador albergaba el servidor, decidió cambiar de estrategia. El personal de la universidad se mostraba muy receloso a revelar detalles tan técnicos. Pensó en una alternativa ingeniosa, y en unas pocas llamadas telefónicas averiguó pronto dónde enviaban los equipos obsoletos cuando eran sustituidos por otros más nuevos. No le fue difícil agenciarse, en la planta de reciclaje donde enviaban los ordenadores viejos, de algunos discos duros que venían de la universidad. Aunque el contenido de los discos había sido borrado, con la ayuda de software especializado consiguió recuperar mucha de la información que contenían. En el segundo disco encontró lo que buscaba: un fichero de configuración con la dirección IP del servidor de LDAP y su contraseña de autenticación.

No sería sencillo entrar en el servidor de LDAP desde fuera, así que debería arreglárselas para realizar una intrusión en uno de los ordenadores internos. Después de preparar un programa troyano, lo envió por correo electrónico a varias personas de la administración de la universidad. El correo parecía un mensaje típico que se envían entre amigos, con una multitud de destinatarios, y haciendo referencia a un vídeo que se incluía y parecía ser muy gracioso. Una de las víctimas seleccionó el vídeo para visualizarlo, lo que, además, instaló un pequeño programa que daría acceso remoto al ordenador. Una vez dentro, consiguió entrar en el servidor de LDAP y tener acceso al hash SHA1 de la contraseña del profesor. Utilizando un programa de romper contraseñas basado en palabras de diccionario encontró después de unas horas la palabra de paso que buscaba. ¡Bingo!. El profesor tenía una contraseña débil y el programa la encontró en unas horas.

Con la contraseña de acceso ya podía entrar en el portal web de introducción de calificaciones suplantando a su profesor, pero aún necesitaba otra contraseña, la de traspaso de calificaciones al acta de la asignatura. Teniendo ya el *password* de acceso y los datos básicos del profesor (número de identificación en la universidad, nombre completo, DNI, etc.) consiguió cambiar esta segunda contraseña por teléfono alegando que la había olvidado. Ahora ya tenía todo, solo faltaba la estocada final. Desde un navegador de la biblioteca, accedió al portal de notas, se cambió el 4 a un 9, y transfirió las notas al acta. Ya lo había conseguido; ¡se había hecho justicia! La alegría, sin embargo, le duró poco. Después de unos días, el profesor detectó el cambio cuando estaba repasando los sobresalientes para decidir quién recibiría matrículas de honor. Aunque durante todo el ataque no dejó ninguna traza que la inculpara, estaba claro quién era la única interesada en modificar esa nota en concreto.

1.8. Casos especiales de ingeniería social

Podemos encontrar multitud de ataques con características diferentes. Estos ataques vienen definidos como un proceso compuesto por un conjunto de acciones relacionadas secuencialmente. Entre todas las diferentes posibilidades, existen una serie de ataques de ingeniería social que tienen una especial relevancia por su cotidianidad. Trataremos brevemente cinco casos de ingeniería social que son familiares para la mayoría de los usuarios, y cuyo denominador común es que, a excepción del *SMiShing*, se producen en el contexto de Internet. Estos casos especiales de ingeniería social son el *phishing*, el *vishing*, el *SMiShing*, el *scareware* y los *hoaxes*.

1.8.1. Phishing

> "El *phishing* es un ataque de ingeniería social cuyo objetivo final es el de obtener datos bancarios de una víctima. Para conseguirlo, se suplanta la identidad de una entidad financiera y se convence a un usuario para realizar una acción que le permite captar dichos datos."

Para realizar el *phishing*, el atacante utiliza el correo electrónico como vía. En ocasiones este tipo de envío se realiza de manera masiva a cuentas de correo que han podido ser obtenidas a través de distintos medios. El envío indiscriminado de correos se realiza con el objetivo de maximizar las probabilidades de éxito del ataque.

Fig. 5 ejemplo de Phishing de Banco de Santander

En este correo se solicita a la víctima que visite la web de la entidad, donde posteriormente se le pide que introduzca sus datos personales junto a sus credenciales. El engaño en este proceso reside en el mismo correo, donde se suele incluir un enlace a la supuesta web legítima; sin embargo, dicho enlace apunta en realidad a un servidor especialmente preparado por el atacante, donde es capaz de recoger los datos de su interés. Dependiendo de la habilidad del atacante en preparar un correo electrónico convincente, y de lo creíble que le resulte a la víctima, el ataque tendrá éxito o no.

1.8.2. SMiShing

El *SMiShing* es una variante del phishing, con la diferencia de que este se realiza utilizando mensajes de telefonía móvil de tipo SMS (*short message service*) o a través de los mensajes de aplicaciones como Whatsapp, Telegram, etc.

Fig. 6 Ejemplo de SMiShing

Aquí, el usuario recibe un mensaje procedente de una supuesta entidad bancaria o institución y en el que se le fuerza, mediante ingeniería social, a realizar una llamada a un número particular, o a enviar una respuesta al mensaje con datos solicitados, tal y como se puede ver en la figura 6.

1.8.3. Vishing

El *vishing* es también una variante del *phishing*. En este caso, la diferencia radica en que es realizado por voz utilizando el sistema de telefonía tradicional, o en algunas ocasiones el atacante suele usar los servicios ofrecidos por la telefonía sobre IP (VoIP). El ataque se basa en utilizar un sistema automático que realiza llamadas a números de teléfono. Cuando detecta que en el otro extremo hay una persona, se le comunica que hay algún tipo de problema con su tarjeta de crédito y se le convence para que proporcione ciertos datos, entre los que figura el número de su tarjeta, la fecha de caducidad y el código de seguridad.

1.8.4. Scareware

El *scareware* es una forma de software malicioso para obtener beneficios empleando ingeniería social. En concreto, explota el engaño, la persuasión, la coacción o el miedo mediante mensajes de alarma o de amenaza para forzar a la víctima a realizar un

pago. El ejemplo más común es el de supuestas soluciones de seguridad que se instalan y nos advierten de que el sistema está infectado por algún tipo de código malicioso. A partir de aquí, el *scareware* brinda la posibilidad al usuario de eliminar el hipotético malware realizando un pago, ya sea mediante una tarjeta de crédito o incluso vía SMS.

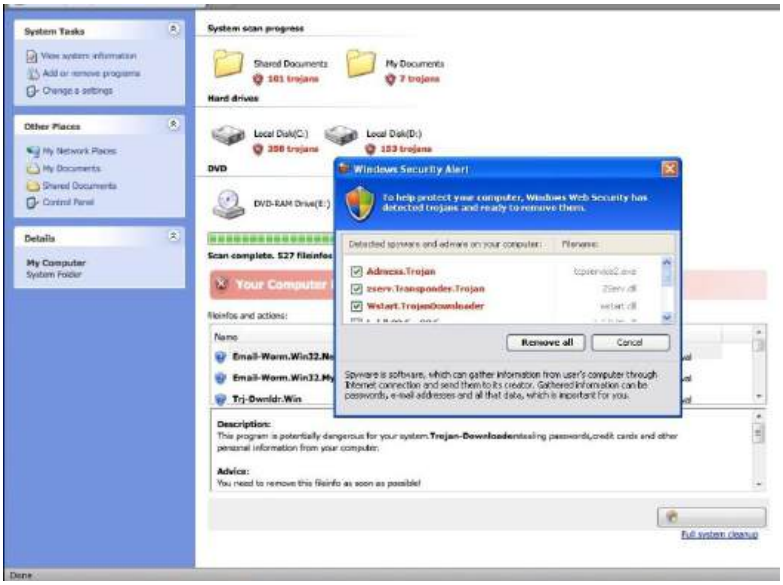


Fig. 7 Ejemplo de alerta falsa

Una alternativa en cuanto al método utilizado es el malware identificado por algunos antivirus comoW32/CardPay.A,Win32/DotTorrent.A o FraudTool.W32/Fakecopyright, entre otros. Se basa en explotar el sentimiento de culpa y el miedo que pueden sentir usuarios al descargar software ilegal. En particular, una vez instalado muestra mensajes de advertencia indicando que se están violando las leyes del copyright y que se tiene identificada la IP del usuario. Seguidamente, propone evitar un juicio realizando un pago como modo de solventar la situación.

1.8.5. Hoaxes

Un *hoax* es un tipo de ataque de ingeniería social que se da en el ámbito del correo electrónico. Se basa en crear una noticia o un rumor totalmente falso en forma de correo electrónico, en el que se fuerza al destinatario, utilizando algunos aspectos explotables mezclados vistos, al reenvío de este para que se propague y distribuya en cadena. Detrás de los *hoaxes* pueden existir distintos intereses, tales como causar alarma social, confundir o modificar la opinión pública, desprestigiar una empresa o recogida de correos electrónicos para posteriormente utilizarlos como destinatarios de *spam*. De ahí la necesidad de explotar aspectos como son el engaño, la parquedad, la prueba social, la amabilidad o la empatía entre otros. Asimismo, la temática de estos suele ser muy variada, como por ejemplo: virus informáticos, leyendas urbanas, cadenas de solidaridad, etc. Sirva de ejemplo el siguiente correo considerado como un *hoax*:

```
¡ATENCIÓN ESTE MENSAJE ES VERÍDICO!
¿Sabes que la leche en cartón que no se vende dentro del
plazo de caducidad regresa a la fábrica para ser
repasteurizada y vuelve al supermercado de nuevo?.
Increíble ¿verdad?. Pues la ley permite a las centrales
lecheras repetir este ciclo hasta 5 veces, lo que termina
dejando la leche casi sin sabor y con una significativa
reducción de su calidad y valor nutricional.
Cuando la leche llega al supermercado para la venta al
consumidor final, el cartón debe exhibir un pequeño número
que está marcado en su parte inferior. Ese número varía del
1 al 5. Lo más que se debe tolerar es comprar leche hasta
el número 3, es decir, leche que ha sido repasteurizada 2
veces, recomendándose no comprar cartones de leche cuyo
número sea 4 o 5, ya que ello significa que la calidad de
la lecha estará degradada. Si compras una caja cerrada,
basta verificar el número de la caja, ya que todos los
cartones en su interior tendrán la misma numeración.
Por ejemplo, si un cartón tiene el número 1, significa que
es la primera vez que sale de la fábrica y llega al
supermercado para su venta, pero si tiene el número 4,
significa que caducó 3 veces y que fue repasteurizada 3
veces volviendo al supermercado para tratar de ser vendida
y así sucesivamente...
Así es que, ya sabes, cuando compres leche, mira el fondo
del cartón y no compres cajas que tengan los números 4 o 5,
y para los más escrupulosos, ni siquiera el 3.
En el archivo adjunto podrán ver el número en cuestión. Id
al super, tomad una caja de leche y comprobad el número,
dudo que encuentréis el 1 o el 2.
SI TIENES CONCIENCIA CIUDADANA, ¡DIVULGA ESTE MENSAJE!!
```

1.9. Prevención y reflexiones

El proceso de la ingeniería social es bien conocido, y sus bases han sido utilizadas durante siglos para realizar acciones fraudulentas de diversa índole. Con la aparición de los sistemas computacionales de información, la ingeniería social ha cobrado una especial relevancia al no aplicarse ya únicamente en el mundo físico, sino en el mundo digital. Las soluciones para evitar ataques de ingeniería social son un tema complicado y, pese a ser un aspecto fundamental en la seguridad de los sistemas de información, lo encontramos casi marginado en la bibliografía y excluido en muchos tratados de seguridad. El motivo es la complejidad asociada a este problema. Aunque existen algunas medidas para minimizar la probabilidad de éxito de los ataques de ingeniería social, veremos en este apartado que no existe una manera de erradicar completamente estas graves vulnerabilidades.

Las maneras de afrontar el problema se basan en la formación y sensibilización, así como en la definición y aplicación de políticas de seguridad, cosa que hace muy importante el tratamiento de esta problemática desde el departamento de tecnologías de la información de cualquier empresa mediana o grande.

1.9.1. Formación y sensibilización

La formación del personal en una organización sobre la ingeniería social y su problemática es un elemento básico para la lucha contra esta. Sin embargo, no está claro quiénes han de ser los receptores de dicha formación, ya que todo el personal relacionado con la organización de un modo u otro es susceptible de ser víctima de la

ingeniería social. Esto incluye desde el personal de limpieza y mantenimiento, hasta los altos cargos de gerencia. Esto complica mucho la gestión y favorece por tanto que se puedan realizar ataques de ingeniería social tan fácilmente.

Para ser efectiva, el aprovechamiento de la formación debe validarse frecuentemente, normalmente mediante auditorías especializadas. Asimismo, una medida que suele ayudar a mejorar los resultados de este proceso de formación es la utilización de esquemas de incentivo/penalización, que recompense al personal que utilice de manera adecuada los conocimientos adquiridos y que penalice, económicamente o de otro modo, al que demuestre no haber aprovechado esta formación. De esta manera se incentiva al personal a considerar seriamente los problemas de seguridad asociados a la ingeniería social. La concienciación en este campo es muy importante, por lo que se debe asegurar de que los empleados tengan incentivos para seguirla y sobre todo para aplicarla posteriormente, hay que mostrar los perjuicios que puede tener la organización en caso de sufrir un ataque.

El problema más grave del proceso de formación es su elevado coste, ya que debe estar dirigido a un colectivo muy numeroso, se trata de una docencia muy especializada y requiere su constante validación generalmente por medio de auditorías. Éstas no solo han de tener en cuenta los aspectos más tradicionales como revisar versiones y protocolos, sino que una buena auditoria de seguridad también debe de tener en cuenta esta parte de ingeniería social.

Una alternativa mucho más económica son las campañas de sensibilización. El objetivo de estas campañas es concienciar a los miembros de una organización sobre los métodos utilizados por la ingeniería social para poder identificarlos cuando se presentan y poder así evitarlos. La sensibilización puede ser complementaria a la formación, reservando esta última para los colectivos más sensibles. Estas campañas de sensibilización se deberían de ir repitiendo en el tiempo para hacer recordatorios de las problemáticas que pueden haber, así los empleados irá cogiendo consciencia de forma constante.

Fig. 8 anuncio de concienciación

1.9.2. Ingeniería social, psicología y humanidad

La mayoría de las vulnerabilidades explotadas por la ingeniería social, tal como se ha visto en este apartado, vienen derivadas de cualidades humanas como la facilidad de establecer cierta confianza entre individuos tras un período de interacción. Algunas de estas cualidades, como la caridad o la cortesía, son generalmente considerados aspectos positivos en nuestra sociedad. Otras, como la empatía o el deseo de ayudar, responden a reacciones más instintivas. Incluso los aspectos más racionalmente controlables, como la reacción ante la autoridad o la avaricia, pueden usarse para manipular psicológicamente al individuo y ejercer una influencia para la realización de acciones inducidas.

Si consideramos a un ordenador inmune a los ataques de ingeniería social es precisamente porque carece de todas estas cualidades que nos acaban haciendo, precisamente, humanos. Si una persona se comporta de tal manera que no sea posible manipularla de algún modo utilizando las técnicas descritas, será muy difícil que esté integrada en la sociedad, ya que la percepción que se tendrá de ella es muy negativa. Lo podemos ver claramente con un ejemplo. Si al tercer intento de introducir una contraseña en un ordenador que utilizamos habitualmente, este nos bloquea el acceso hasta que un supervisor lo vuelva a habilitar, nos conformaremos resignadamente. Si en cambio una persona nos pide la identificación para entrar al trabajo y un día la hemos olvidado, no entenderemos por qué nos bloquea el acceso, y le exigiremos flexibilidad apelando, precisamente, a su humanidad. Es difícil comprender en el momento que esa misma flexibilidad podría permitir el acceso a un atacante.

Evitar el problema, por lo tanto, es muy difícil, si no imposible. Las estrategias de formación y concienciación son claramente útiles y necesarias, pero no eluden totalmente los ataques de ingeniería social. Para evitar el ataque hay que focalizarse en las amenazas directas, para bloquear los ataques asumiendo que el ingeniero social tiene a su disposición toda la información. Cuanto menos escondamos de nuestra organización mejor lo habremos de configurar y por tanto menos expuestos estaremos. Si confiamos en la ofuscación, malos resultados tendremos.

2. Herramientas de hacking ético

En capítulo se pretende dar una visión global de la complejidad y diversidad de las vulnerabilidades en red. Para ello, se hará un repaso de algunas vulnerabilidades presentes en distintos niveles de redes. Se comentaran algunas vulnerabilidades importantes de protocolos de red, ya sea por su importancia desde el punto de vista histórico, didáctico, o actual. El capítulo se centra principalmente en redes TCP/IP, y en concreto en la versión IPv4, que es la más extendida y utilizada hoy en día. Por último, también se presenta los escáneres de vulnerabilidades, herramientas básicas para mejorar la seguridad de sistemas informáticos. Los escáneres de vulnerabilidades permiten detectar vulnerabilidades que pueden derivar en problemas de seguridad. Es decir, que aplicaremos las herramientas de hacking ético para intentar atacara nuestra entidad para saber si podemos ser atacados por terceras personas.

2.1. Conceptos básicos

Repasaremos algunas vulnerabilidades de las redes informáticas. Para ello, nos centraremos en dar una visión global del tipo de vulnerabilidades que nos podemos encontrar en la redes telemáticas. Antes de entrar en detalle, repasamos algunos conceptos básicos.

Las redes informáticas se organizan en una pila o *stack* de protocolos. El modelo OSI de interconexión de redes define 6 capas que van desde el medio físico de transmisión de señal, hasta las aplicaciones de alto nivel que hacen uso de la red. Esta separación de protocolos por capas permite definir y aislar claramente la funcionalidad de cada protocolo y aporta un diseño muy modular.

Sin embargo, actualmente y desde un punto de vista más práctico, se ha impuesto el modelo TCP/IP, que es el que define la suite de protocolos que dieron luz a Internet tal como se conoce en la actualidad. Éste tiene 4 capas de protocolos que van también desde la capa de red hasta las de aplicaciones.

Nos centraremos pues en el modelo TCP/IP. A continuación detallamos la principal funcionalidad de cada capa:

- **Red**: también llamada *link layer* o *network access layer*, engloba las conexiones de red local.
- **Internet**: la capa de interconexión de red (en inglés internet) permite el envío de datos entre redes locales. Para ello, proporciona un sistema de direccionamiento global (direcciones IP) y el encaminamiento de paquetes de datos.
- **Transporte**: se encarga de la transferencia de datos extremo a extremo con independencia de la red local. Puede incluir funcionalidad para el control de errores, segmentación, control de flujo, control de congestión, y el direccionamiento a la capa de aplicación mediante el uso de puertos.
- **Aplicación**: incluye protocolos de alto nivel utilizados directamente por las aplicaciones, como por ejemplo HTTP (*hypertext trasfer protocol*), FTP (*file transfer protocol*) o SMTP (*simple mail trasport protocol*).

Para revisar las vulnerabilidades de red nos centraremos en tres grandes bloques: protocolos locales, interconexión de redes y protocolos de extremo a extremo. Estos

bloques se corresponden ligeramente con las capas TCP/IP de red, internet y transporte, respectivamente. La correspondencia no es exacta, ya que nos centramos, por motivos de simplicidad, en la funcionalidad propia de cada bloque con cierta independencia de si los protocolos tratados corresponden estrictamente a una u otra capa.

2.2. Protocolos locales

Existen muchos protocolos de red de área local o LAN (*local area network*). Sin duda, las redes locales cableadas más utilizadas son las basadas en Ethernet, pero existen otras, muy extendidas años atrás. La familia de tecnologías Ethernet permite hoy en día la creación de redes locales relativamente extensas y capaces de alcanzar gran velocidad de transmisión.

Ethernet utiliza direcciones físicas o MAC (*media access control*), de 48 bits únicas globalmente y asignadas por el fabricante. Utiliza paquetes denominados *frames* de 1.518 bytes con un espacio para 1.500 bytes de datos. Cuando una red Ethernet se conecta a otra red (o a Internet) mediante TCP/IP es necesario "traducir" direcciones físicas a direcciones IP. Esta traducción se lleva a cabo mediante un protocolo de bajo nivel denominado ARP (*address resolution protocol*).

Aunque Ethernet ha evolucionado mucho desde sus orígenes, sigue presentando problemas de seguridad y existen vulnerabilidades importantes en este tipo de redes. A continuación veremos algunas de las más representativas.

2.2.1. *Sniffers* de Ethernet

Uno de los principales problemas o vulnerabilidad que presentaba inicialmente Ethernet era la facilidad de escuchar el tráfico de la red local. Ethernet utilizaba una topología de bus donde todos los paquetes se enviaban al bus y solo el host con la dirección destino del paquete recogía dicho paquete. Cualquier dispositivo con acceso a la red podía escuchar todo el contenido que por ella circulaba.

En las tarjetas de red Ethernet existe un modo de funcionamiento promiscuo (*promiscuous mode*) que permite recoger todos los paquetes que pasan por el bus. Este modo de funcionamiento está pensado para monitorizar la red en la detección de problemas y permite el uso de la tarjeta de red como puente (*bridge*) para la virtualización de hardware. Pero se puede hacer un uso malicioso de este modo para escuchar todo el tráfico local de la red. Típicos sniffers de red como *tcpdump*, *ettercap* o *Wireshark* pueden esnifar paquetes Ethernet desde una tarjeta en modo promiscuo.

> "La mayoría de los sistemas operativos requieren privilegios administrativos (de superusuario o *root*) para poder operar una tarjeta en modo promiscuo."

La tecnología Ethernet ha evolucionado mucho. De la clásica topología de bus se pasó a topologías de estrella utilizando unos dispositivos de red denominados concentradores o *hub,* que simulan la funcionalidad de un bus, y por tanto presentan los mismos problemas que un bus. Actualmente, el concentrador suele ser reemplazado por un conmutador, o *switch*. A diferencia de un *hub*, un *switch* Ethernet no simula el funcionamiento del bus, sino que tiene la capacidad de aprender la topología de la red a partir del tráfico que recibe. Es decir, basándose en las

direcciones MAC de los paquetes que reciben/envían los dispositivos, sabe dónde está conectado cada uno y solo envía los paquetes destinados a dicho dispositivo por su cable correspondiente.

De esta manera, no solo se consigue minimizar el tráfico de toda la red, ya que solo circula por cada puerto o cable del *switch* el tráfico propio, sino que además imposibilita que un dispositivo en modo promiscuo pueda recibir todos los paquetes que circulan por la red, solo verá los que vayan para él.

Existen vulnerabilidades inherentes a los actuales conmutadores Ethernet que permiten convertir un conmutador en un concentrador y consecuentemente esnifar todo el tráfico de la red desde un host. Esta vulnerabilidad se puede explotar con un ataque de *MAC flooding* que vamos a ver ahora.

MAC flooding

Un conmutador, o *switch*, Ethernet mantiene una tabla llamada CAM (*content addressable memory*) donde establece un vínculo entre direcciones MAC y puertos físicos del propio conmutador. Esta tabla le permite al conmutador enviar los paquetes únicamente a su destinatario por el puerto físico correspondiente. El conmutador establece la tabla CAM observando el tráfico generado y destinado a cada host conectado a este.

El problema es que la tabla CAM de un conmutador tiene una memoria limitada y, por tanto, un atacante puede saturar dicha tabla con el propósito de dejarla inutilizada. Para ello, bombardea el conmutador con paquetes Ethernet con direcciones MAC de orígenes diferentes, lo que provoca que el conmutador las añada a la tabla CAM hasta que esta se agota. En ese momento, dado que el conmutador no puede añadir más entradas, pasa a un modo de funcionamiento conocido como *failopen*, en el que empieza a actuar como un *hub*. En este modo el conmutador envía los paquetes a todos los dispositivos de la red, ya que ahora no conoce en que puerto está conectado.

Actualmente se puede intentar mitigar esta vulnerabilidad utilizando sistemas de monitorización de red. Muchos fabricantes de conmutadores permiten limitar el número máximo de direcciones MAC para cada puerto físico (técnica conocida como *port security*), por ejemplo si se sabe que solo hay un dispositivo al final del cable, o hay otro *switch*. También existen mecanismos para requerir la autenticación con servidores de autenticación y autorización.

2.2.2. Modificación de direcciones MAC

La dirección física de Ethernet de cada una de las tarjetas de red (NIC) es asignada por el fabricante y tradicionalmente estaba inequívocamente asociada a la tarjeta de red. Es decir, dicha dirección es única y no se puede modificar. Esto hizo que se desarrollasen mecanismos de control de acceso basados en direcciones MAC. En este caso, un dispositivo de control de red puede permitir el acceso solo a unas direcciones MAC concretas de los otros dispositivos, las tarjetas que no estén dentro del listado de direcciones MAC permitidas no se les dará acceso a la red.
Un ejemplo muy común hoy en día son las redes (generalmente inalámbricas) que permiten una conexión gratuita a Internet durante un período concreto de tiempo al día

(típicas en los Aeropuertos por ejemplo); en ellas, el control sobre el tiempo de conexión de cada usuario suele hacerse mediante la dirección MAC.

> "Hoy en día la dirección física Ethernet puede ser fácilmente modificada. Esta funcionalidad es actualmente tan común que muchos sistemas operativos incluyen la opción de modificar la dirección MAC con las herramientas propias de gestión de red, sin necesidad de programas externos."

Por ejemplo, utilizando el comando `ifconfig` o las utilidades `iproute2` de Linux podemos modificar la dirección física de la interfaz de red eth0:

```
# ifconfig eth0 hw ether 00:11:22:33:44:55
# ip link set dev eth0 address 00:11:22:33:44:55
```

En Windows se puede modificar desde las propiedades del adaptador red , como muestra la figura 9.

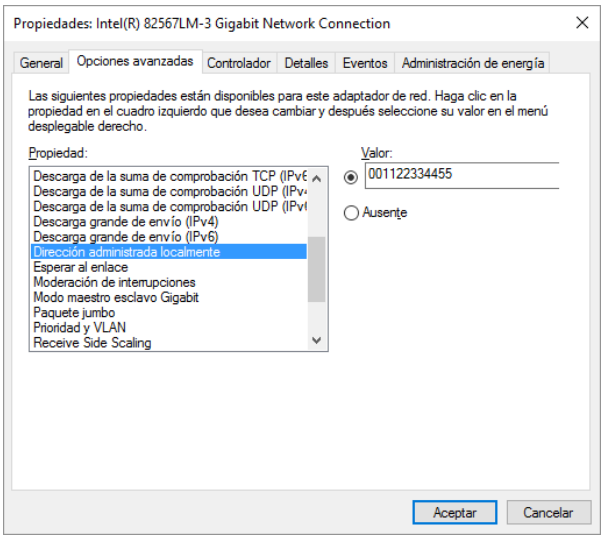


Fig. 9 Modificación de MAC en Windows 10

Con la facilidad que hay hoy en día para poder poner una tarjeta de red en modo promiscuo y ver las diferentes direcciones MAC que hay en la red, sobre todo en las inalámbricas, supone una vulnerabilidad importante en sistemas de control de acceso o autenticación de red basados en dichas direcciones.
Hay que implementar sistemas de autenticación más robustos y no simplemente la verificación de la dirección MAC de la tarjeta de red a la que permitimos el acceso al medio.

2.2.3. Vulnerabilidades en el protocolo ARP

El protocolo *address resolution protocol* (ARP) permite traducir direcciones MAC en direcciones IP. En redes locales que soportan *broadcast* (envío a todos los sistemas de la red) como Ethernet, el funcionamiento de ARP es el siguiente. Supongamos que el sistema A quiere conocer la dirección IP de B (IP_B) pero desconoce su dirección física F_B, que necesita para establecer la comunicación posteriormente entre los dos sistemas.

1) El sistema A envía en *broadcast* una petición ARP preguntando quién tiene la dirección IP_B.
2) B contesta con una respuesta ARP a A diciendo que él tiene la dirección IP_B (es decir, que IP_B le corresponde a la dirección física F_B).

Para evitar estar haciendo peticiones ARP continuamente, cada host mantiene una tabla con las correspondencias entre dirección física y dirección IP llamada cache ARP. Las entradas tienen una caducidad de aproximadamente 20 minutos. Un punto importante es que cualquier petición ARP es utilizada por el resto de los hosts para actualizar la entrada correspondiente al emisor.

Este protocolo actualmente presenta vulnerabilidades difíciles de solucionar. La mayoría de los problemas de seguridad de ARP se basan en que el atacante envía mensajes ARP falsos para "envenenar" las cachés ARP, es decir, para introducir información falsa. En inglés, esta técnica recibe el nombre de ARP *poisoning*. Dependiendo de cómo se use, ARP *poisoning* permite varios ataques diferentes, como:

- **Denegación de servicio**: se puede conseguir que un host no reciba ningún paquete al difundir una dirección física inexistente asociada a su dirección IP real. Si la víctima de la denegación de servicio es el encaminador o puerta de enlace de la red local, se consigue aislar la red local del exterior.
- **Man in the middle** (MITM): un atacante se puede hacer pasar por otro host (víctima) y recibir así todos los paquetes destinados a dicha víctima.

Todo y los problemas que puede ocasionar, ARP *poisoning* también puede tener usos legítimos, por ejemplo para direccionar accesos de red a un portal de autenticación (típicamente usado en hoteles, redes universitarias o redes de acceso público), o en sistemas de redundancia para que un servidor pueda tomar el lugar de otro en caso de que este sufra algún percance.

Actualmente solo existen soluciones limitadas para mitigar estas vulnerabilidades. Algunas soluciones actuales son incluir entradas fijas en la caché ARP, generalmente las correspondientes a hosts críticos como encaminadores, el uso de *port security* en los conmutadores, o la monitorización de la red en busca de comportamiento inusual.

2.3. Interconexión de redes

Dentro de la interconexión de redes nos centramos en el protocolo IP y los servicios asociados. Dada la complejidad de Internet y todas las tecnologías asociadas a la interconexión de redes, el número de vulnerabilidades potenciales es muy grande. En este apartado enumeramos algunas de ellas a modo ilustrativo y con fines didácticos.

2.3.1. Vulnerabilidades en IP

Vemos a continuación algunas de las vulnerabilidades del protocolo IP especialmente relevantes partir de los siguientes ataques:

- **IP spoofing**: consiste en crear paquetes IP con la dirección de origen falsa e introducirlos en la red. Esta vulnerabilidad se suele explotar con el objetivo de hacer ataques de denegación de servicio o suplantar a un host concreto, ya que las respuestas irán todas a la dirección falsa.

- **Packet-of-death**: IP ha sufrido algunas vulnerabilidades en la implementación. El envío de paquetes IP deliberadamente erróneos puede causar problemas importantes en algunas implementaciones. Un ejemplo es el uso de la misma dirección IP como origen y destino (*land attack*), provocando bucles incoherentes.

- **Vulnerabilidades en la fragmentación**: IP puede realizar fragmentación de paquetes para adaptarse a los tamaños de paquetes de redes locales. El envío de fragmentos erróneos donde se solapan los campos de datos ha dado problemas en algunas implementaciones. Un caso conocido es el *teardrop attack*, que explotaba una vulnerabilidad en la implementación de SMBv2 de Windows Vista.

- **IP source routing**: IP incluye un par de opciones que permiten especificar la ruta (parcial) de retorno que debe seguir el paquete de respuesta. Esto permite que un atacante utilizando IP spoofing con una dirección de origen de un host de confianza de la víctima pueda recibir el paquete de respuesta. Actualmente estas opciones no se suelen utilizar, y muchos dispositivos de red bloquean el paso de paquetes con estas opciones.

2.3.2. Vulnerabilidades en ICMP

El protocolo ICMP (*Internet control message protocol*) es un protocolo de control y notificación de errores del protocolo IP que ha presentado algunas vulnerabilidades importantes, generalmente asociadas a ataques de denegación de servicio. A continuación, detallamos algunos ataques que explotan vulnerabilidades de ICMP:

Ping flooding: ataque clásico de denegación de servicio utilizando mensajes *echo request* (*ping*) de ICMP.

Ping of death: ataque de interés histórico que afectó a casi todas las implementaciones de ICMP hasta finales de los años noventa. Consiste en enviar un paquete ICMP de mayor tamaño que el máximo permitido por IP fragmentado. Esto provocaba un *buffer overflow* en el host de destino.

Smurf attack: ataque de denegación de servicio en el que el atacante envía mensajes de *ping* en *broadcast* con la dirección de origen de la víctima. Esto provoca que todos los hosts que reciben el paquete envíen la respuesta del *ping* a la víctima. En este caso la vulnerabilidad se encuentra en el uso de mensajes ICMP en *broadcast*. Actualmente, los hosts no contestan a *pings* enviados en *broadcast*, lo que hace ya inútil este ataque. Asimismo, se modificó el estándar para requerir que por defecto los encaminadores bloqueen mensajes enviados en *broadcast*.

2.3.3. Vulnerabilidades en DNS

DNS (*domain name system*) permite la resolución de nombres de dominio mediante servidores organizados jerárquicamente a partir de 13 servidores raíz (9 de ellos distribuidos geográficamente utilizando *anycast*). DNS presenta muchas ventajas. Es un sistema distribuido, eficiente en la resolución de nombres y tolerante a fallos. Existen sin embargo algunas vulnerabilidades en DNS que pueden dar lugar a ataques importantes:

- **DNS *spoofing***: consiste en retornar información errónea de manera deliberada sobre la correspondencia de dirección IP a nombre de dominio. El objetivo puede ser por ejemplo asociar una IP "falsa" a un nombre de dominio conocido (con lo que se puede redirigir el tráfico a dicho dominio). Existen distintos modos de realizar estos ataques. El más sencillo es poner un servidor DNS que emita respuesta falsas o que pueda suplantar un servidor conocido (por ejemplo, mediante IP *spofing*), también se puede interceptar la petición de DNS y responder antes de lo que lo haría el servidor legítimo, o mediante la técnica de *Man in The Midle*. Es importante tener en cuenta que para que una respuesta sea aceptada como legítima debe cumplir los siguientes puntos:

  - Volver a la misma dirección IP que emitió la petición.
  - Volver por el mismo puerto desde donde se envió la petición.
  - Que la respuesta corresponda a la petición.
  - Que el número de transacción coincida con la petición. Este número es teóricamente aleatorio y permite vincular respuesta a petición.

  En muchos casos la facilidad para predecir el número de transacción ha sido una vulnerabilidad importante, similar a la predicción de números de secuencia en TCP. Un atacante puede falsear una respuesta DNS sin necesidad de ver la petición para saber el número de transacción. El hecho de que DNS funcione sobre UDP también facilita este tipo de ataques. Es importante remarcar también que, dada la organización jerárquica de DNS, estos ataques se pueden hacer directamente sobre el cliente o a algún servidor intermedio.

- **DNS *cache poisoning***: para mejorar la eficiencia de DNS cada servidor mantiene una caché con las últimas resoluciones obtenidas (respuestas de DNS) para poder contestar a futuras peticiones de manera rápida. Al igual que en el caso de ARP, se puede forzar la entrada de relaciones de nombre de dominio a dirección IP falsa en dicha caché. Muchos ataques de DNS *spofing* buscan precisamente "envenenar" la caché de servidores intermedios (por ejemplo, el DNS de un ISP) con el objetivo de que todos sus clientes queden afectados.

- **DNS *amplification attacks***: los ataques de amplificación de DNS son unos ataques de denegación de servicio que explotan el hecho de que las peticiones de DNS se resuelven recursivamente y que una petición de tamaño pequeño (60 bytes) puede llegar a generar respuestas más grandes (512 bytes). De manera similar a los ataques *smurf*, se envían muchas peticiones con la dirección IP de origen de la víctima que recibirá todas las respuestas. La denegación de servicio se agrava por el gran tamaño que pueden alcanzar dichas respuestas (amplificación). En octubre del 2002 se realizó un gran ataque de amplificación en el que las víctimas eran los servidores raíz de DNS

que consiguió comprometer a alguno de ellos. El hecho de que no todos los servidores fuesen comprometidos es visto como una ventaja de la redundancia de DNS. Un ataque similar fue repetido en octubre de 2016 y consiguió afectar a los servidores de la empresa DynDNS que controla las resoluciones de muchas empresas de Estados Unidos, dejando sin acceso a ellas. En este último caso se hizo a través de una red de dispositivos en el Internet of Things que se tenían controlados, cámaras y videograbadores.

El DNS fue diseñado sin tener en cuenta la seguridad, y sus vulnerabilidades han sido importantes e incluso detalladas en el RFC 3833. Actualmente existe DNSSEC (*domain name system security extensions*), un conjunto de especificaciones de la IETF que busca solucionar los problemas de seguridad de DNS. DNSSEC proporciona principalmente autenticación e integridad.

2.4. Protocolos extremo a extremo

Dentro de los protocolos extremo a extremo utilizados por TCP/IP cabe destacar por su extenso uso: TCP (*trasmission control protocol*), y UDP (*user datagram protocol*).

Estos protocolos introducen el uso de puertos que permiten direccionar datos de la capa inferior IP a aplicaciones concretas. Esto es considerado por algunos autores como una vulnerabilidad, ya que posibilita el uso de escáneres de puertos para obtener información sobre qué servicios está ofreciendo un host, incluso información adicional, como el sistema operativo, la versión, etc. Sin embargo, otros autores no lo consideran una vulnerabilidad, ya que no consideran que la información obtenida por estos sistemas sea crítica desde el punto de vista de la seguridad. No es un problema que sea público los puertos utilizados, sino las aplicaciones que hay detrás y que sí pueden tener vulnerabilidades.

2.4.1. Vulnerabilidades de TCP

El protocolo TCP proporciona un servicio genérico de transmisión fiable de datos extremo a extremo, entre dos puntos de comunicación. Se encarga de controlar errores en la transmisión, el orden de los paquetes, la detección de duplicados, el control de velocidad de transmisión, etc.

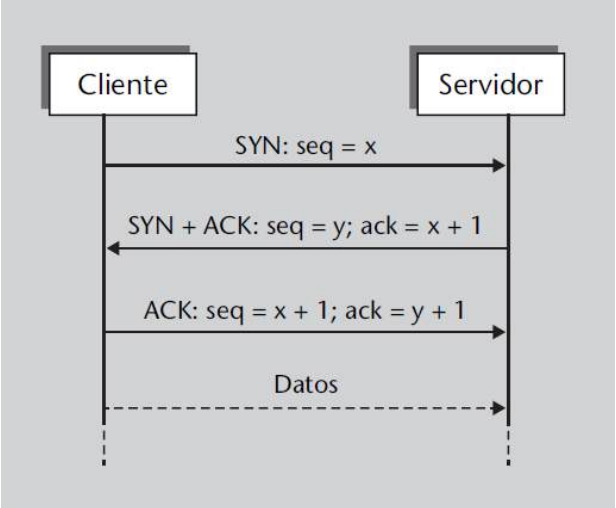


Fig. 10 *3-way handshake*

TCP es un protocolo orientado a conexión. Se establece una conexión entre los dos extremos que se mantiene durante la transmisión de datos. Esta conexión se crea mediante el denominado *3-way handshake*.

Como vemos en la figura 10, la conexión se establece con tres mensajes: un mensaje tipo SYN (de sincronización), al que el servidor contesta con un SYN y ACK (sincronización y reconocimiento) y finalmente el cliente envía un ACK (reconocimiento). Una vez establecida la conexión, el cliente y el servidor se pueden enviar datos que serán reconocidos cada cierto tiempo (mediante mensajes ACK) por el receptor.

Como vemos, una de las tareas del establecimiento de conexión es fijar los números de secuencia de cliente y servidor (en la figura 10, *seq* y *ack*). Estos números se utilizan para identificar los bytes de datos enviados y permiten realizar una gestión del flujo de datos: control del orden de envío de segmentos, pérdida, duplicados, etc. El número de *ack* confirma la correcta recepción de todos los mensajes con número de secuencia igual o inferior.

A continuación veremos algunos ataques y vulnerabilidades de TCP.

SYN flooding

En el establecimiento de sesión, cuando el cliente envía el mensaje SYN, el servidor contesta (SYN+ACK) y se queda esperando el ACK del cliente. ¿Qué sucede si el cliente no envía ese ACK?

Como es de esperar, el servidor espera un tiempo prudencial y si no recibe el ACK, da la conexión por perdida. Esta funcionalidad presenta una vulnerabilidad importante de TCP que puede ser explotada para realizar ataques de denegación de servicio. La idea es bombardear un servidor con peticiones de conexión y no hacer el último ACK. De esta manera, el servidor se queda con conexiones medio establecidas que consumen, durante un tiempo determinado, recursos del servidor (memoria principalmente). Si hay suficientes intentos simultáneos, se puede llegar a saturar el servidor y agotar sus recursos.

Este ataque se conoce como *SYN flooding*, y fue muy importante cuando se descubrió. Existe una variante basada en *IP spoofing*, en la que el primer mensaje SYN del cliente lleva una dirección IP de origen falsa, por lo que le resulta imposible al servidor enviar el SYN+ACK.

Hoy en día esta vulnerabilidad no suele suponer un riesgo importante, ya que existen varios mecanismos que previenen frente a ataques de *SYN flooding.*

Predicción de números de secuencia

La facilidad a la hora de predecir los números de secuencia de una conexión TCP resultó ser una vulnerabilidad de seguridad importante. Para verlo, explicaremos brevemente uno de los ataques informáticos más conocidos y documentados en la historia de la seguridad informática.

El día de Navidad de 1994, un *hacker* llamado Kevin Mitnick realizó un ataque al ordenador de Tsutomu Shimomura situado en la Universidad de California, en San Diego. El ataque pretendía obtener el código fuente de un modelo de teléfono móvil (el

que tenía Mitnick) que estaba almacenado en el ordenador de Shimomura. Mitnick pretendía modificar el software del teléfono y así intentar evitar los sistemas de seguimiento y localización de este.

Sin entrar en muchos detalles, el ataque consistió en los siguientes pasos que se detallan a continuación (podéis ver también la figura 11):

1) El ataque se inicia desde un servidor externo E, al que el atacante ha podido acceder con anterioridad.

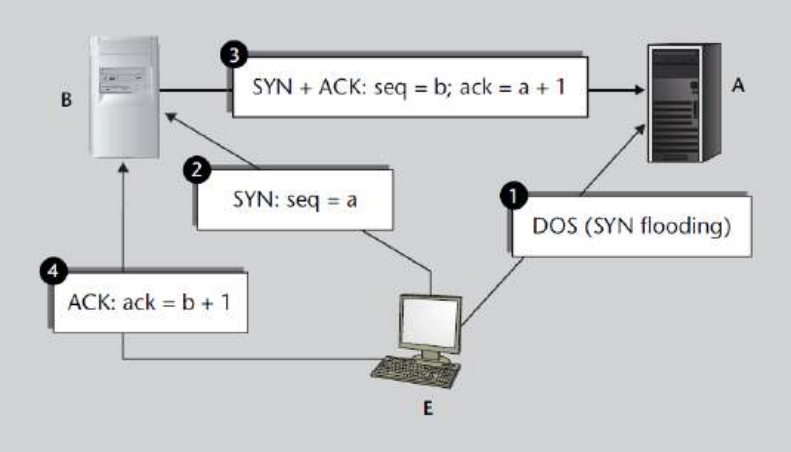


Fig. 11 ataque de Mitnick

2) Desde la máquina externa E se recopila información del objetivo y se descubre que entre dos servidores A y B existe una relación de confianza. Esta permitía realizar conexiones de uno a otro a partir de su dirección IP. Es decir, el servidor A acepta peticiones de conexión del servidor B.

3) Desde E se realiza un ataque de *SYN-flooding* a A para evitar que dicho servidor pueda responder a cualquier mensaje. De esta manera, se quiere silenciar el servidor A, con el objetivo de que el atacante se pueda hacer pasar por dicho servidor, e iniciar así una conexión a B. Para suplantar al servidor A, el atacante realiza un ataque de IP *spoofing* que le permita suplantar la dirección IP de A. En este punto el atacante puede enviar mensajes a B haciéndose pasar por A pero no podrá ver la respuesta que genera, ya que no se encuentra en la misma red local.

4) Para poder establecer una conexión con B el atacante necesita predecir cómo va a contestar A al intento de conexión de B (dado que no puede ver dicha contestación). Esto es, predecir el número de secuencia TCP de los mensajes que genera B. La conexión TCP consta de 3 tres pasos, como se detalla en la figura 10, en la que los números de secuencia seq y ack deben coincidir.

5) Una vez se puede predecir el número de secuencia, el atacante puede realizar una conexión al servidor B para forzar a que B pase a aceptar conexiones de cualquier dirección IP.

6) El atacante puede ahora acceder a B desde cualquier sitio.

Una vez Mitnick obtuvo acceso al servidor de Shimomura pudo copiar todo el código fuente que buscaba. Este ataque dio lugar a uno de los sucesos más sonados de la

seguridad informática. Finalmente, el FBI, con la ayuda de Shimomura, capturó a Mitnick, quien acabó pasando 5 años en prisión.

La posibilidad de predicción de números de secuencia resultó ser una vulnerabilidad importante de TCP; hoy en día las implementaciones de TCP ponen mucho interés en generar los números de secuencia de la manera más aleatoria posible para evitar este tipo de problemas.

2.4.2. Vulnerabilidades en UDP

User datagram protocol (UDP) es un protocolo de transmisión extremo a extremo que ofrece una funcionalidad mínima. No proporciona ninguno de los mecanismos de control de flujo de TCP. Es decr, que no hay control de si llegan o no los datos al otro extremo.

La mayoría de las vulnerabilidades de UDP son propias de errores de implementaciones concretas y no del protocolo. De esta manera, nos encontramos con un tipo de ataque conocido como UDP *Bomb*, que explota vulnerabilidades presentes en implementaciones que fallan al recibir un datagrama UDP erróneo o mal construido. Un error típico es especificar en la cabecera del datagrama un tamaño que no se corresponde con el tamaño real del datagrama. Esto puede producir un *buffer overflow* en la implementación del protocolo.

Existen otros ataques, como el *fraggle attack*, que explotan el uso (y encaminamiento) de direcciones de *broadcast*. Este es un ataque idéntico al ataque *smurf* de ICMP pero con paquetes UDP, en este caso dirigidos a los servicios UDP *echo* y *chargen* (puertos UDP 7 y 19 respectivamente).

El hecho de utilizar UDP en ocasiones facilita la explotación de otras vulnerabilidades debido a la falta de control en comparación con TCP. Por ejemplo, realizar un ataque de MITM usando IP *spoofing* es mucho más sencillo sobre UDP que sobre TCP, ya que no es necesario hacer el secuestro de sesión descrito en el apartado anterior “Predicción de números de secuencia”.

Aquí vemos un caso claro de la relación entre la complejidad de un sistema y la presencia de vulnerabilidades. Un sistema complejo suele presentar más vulnerabilidades que uno más sencillo. De esta manera, el número de vulnerabilidades que afectan a TCP es mucho mayor que las asociadas a UDP.

2.5. Escáneres de vulnerabilidades

Entre las herramientas y los mecanismos que nos permiten mejorar la seguridad de los sistemas informáticos se encuentran las que están enfocadas a la detección de anomalías que pueden derivar en problemas para la seguridad del sistema. En este sentido, podemos realizar una distinción entre las herramientas que permiten detectar una vulnerabilidad y aquellas que permiten detectar un ataque. Si bien la distinción entre vulnerabilidad y ataque a menudo puede quedar diluida, en cuanto a su detección se considera que las herramientas que permiten la detección de vulnerabilidades quedan englobadas dentro de lo que se conoce como escáneres de vulnerabilidades, mientras que aquellas herramientas que se utilizan para la detección de los ataques se incluyen dentro de la familia de sistemas de detección de intrusos.

Sin embargo, también puede suceder que un escáner de vulnerabilidades detecte un ataque, ya que, por ejemplo, un ataque puede poner al descubierto o generar una nueva vulnerabilidad.

En este apartado nos centraremos solo en los escáneres de vulnerabilidades que permiten detectar las vulnerabilidades de un sistema.

2.5.1. Características generales de los escáneres

Los escáneres de vulnerabilidades son un conjunto de herramientas que nos permiten detectar las vulnerabilidades de un sistema, ya sea por medio de simulaciones de ataques, ya sea porque se detecta una configuración que implica una deficiencia de seguridad.

El funcionamiento de un escáner de vulnerabilidades se podría dividir en tres fases:

1) Durante la primera etapa se realiza una extracción de muestras del conjunto de atributos del sistema para poder almacenarlas posteriormente en un contenedor de datos seguro.
2) En la segunda etapa, estos resultados son organizados y comparados con unas bases de datos de reglas y firmas que permiten identificar configuraciones inseguras.
3) Finalmente, se generará un informe con las diferencias entre ambos conjuntos de datos.

La principal ventaja de los escáneres de vulnerabilidades es que permiten la detección y solución de la vulnerabilidad antes de que esta pueda ser explotada para realizar un ataque. No obstante, hay que tener en cuenta que la mayoría de las vulnerabilidades detectadas por un escáner no pueden ser reparadas por el propio escáner. Este se limita a proporcionar una serie de información e informes que, en la mayoría de los casos, requieren una intervención manual del administrador.

El mecanismo de trabajo de los escáneres de vulnerabilidades se asemeja al de los antivirus, por su dependencia a una base de datos en la que se incluyen las reglas que tendrá en cuenta el escáner. Una de las limitaciones básicas de los escáneres de vulnerabilidades es que únicamente permiten identificar las vulnerabilidades que están ya tipificadas en su base de datos. Esto implica que, en general, solamente se puedan detectar vulnerabilidades en software estándar, de modo que, por ejemplo, aplicaciones web personalizadas no pueden ser escaneadas con escáneres de vulnerabilidades de propósito general y precisan analizadores específicos de aplicaciones web. Por otro lado, dada la dependencia de la base de datos de vulnerabilidades con la que trabaja el escáner, la frecuencia de actualización de esta es un punto muy importante que se debe tener en cuenta para la selección de un escáner de vulnerabilidades, dado que será preciso una actualización constante de la base de datos de referencia para que el escáner pueda detectar las últimas vulnerabilidades publicadas.

Dada la importancia de la base de datos de vulnerabilidades en los escáneres, existe un procedimiento para determinar la compatibilidad del producto respecto al estándar CVE (*common vulnerabilities and exposures*) que permite etiquetar las vulnerabilidades. Este procedimiento, establecido por *The Mitre Corporation*, especifica los requisitos que un escáner de vulnerabilidades debe poseer para que sea

compatible con CVE. En concreto, para que un producto sea compatible con CVE debe cumplir:

- **Búsqueda por CVE**: el producto certificado debe permitir la búsqueda de vulnerabilidades en su base de datos utilizando el identificador CVE.
- **Salida CVE**: la información de la vulnerabilidad que ofrece el producto debe incluir el identificador CVE.
- **Identificación**: el producto debe proporcionar información suficiente de cómo identifica la vulnerabilidad de su base de datos con la versión específica de CVE y, a su vez, debe intentar que esta identificación sea tan precisa como sea posible.
- **Documentación**: La documentación estándar del producto debe incluir una descripción de CVE, la compatibilidad CVE, y los detalles de cómo sus clientes pueden utilizar la funcionalidad relacionada con CVE de su producto o servicio.

El cumplimiento de estas condiciones por parte de un escáner es relevante puesto que permite a los usuarios complementar las informaciones que el escáner proporciona en sus informes utilizando fuentes externas, como los CERT, dado que la identificación de la vulnerabilidad será unívoca. Por otro lado, la identificación unívoca de las vulnerabilidades mediante su código CVE también permite comprobar el grado de actualización de la base de datos del escáner.

Software para el escaneo de vulnerabilidades

Existen diferentes empresas que comercializan software para el escaneo de vulnerabilidades. Las principales diferencias entre los distintos productos se encuentran en sus características, como, por ejemplo, la ratio de falsos positivos que genera la detección, la variedad de los posibles sistemas operativos que permiten escanear, los tipos de dispositivos que pueden escanear (servidores, encaminadores, impresoras de red, etc.), el número diferente de aplicaciones que pueden escanear (bases de datos, servidores de aplicaciones PHP, Java, .NET, etc.), la frecuencia de actualización de la bases de datos o la calidad de la información que reportan para que los administradores puedan arreglar o eliminar la vulnerabilidad encontrada.

2.5.2. Clasificación de los escáneres

Los escáneres de vulnerabilidades admiten diferentes tipos de clasificación. Los escáneres se pueden clasificar en aquellos basados en máquina (*host-based scanners*) y los escáneres basados en red (*network-based scanners*). Los primeros son escáneres situados en los propios dispositivos que se pretende escanear, mientras que los segundos se sitúan en servidores de la red y permiten realizar análisis de otras máquinas.

La sofisticación de los escáneres basados en red, así como la aparición de nuevas técnicas para el escaneo de vulnerabilidades, permiten una nueva clasificación más precisa. De este modo, podemos clasificar los escáneres en función de sus habilidades de escaneo:

- Escaneo interno y activo de un dispositivo.
- Escaneo externo y activo de un dispositivo.
- Escaneo externo y pasivo de un dispositivo.

Todos estos escáneres no son excluyentes (en el sentido de que la utilización de un tipo de escáner no invalida a los demás), ya que hay vulnerabilidades que se pueden detectar con un tipo de escáner pero no con otro. Por tanto, un buen administrador de sistemas utilizará cada uno de ellos para detectar diferentes tipos de vulnerabilidad.

Escaneo interno y activo de un dispositivo

El escaneo interno y activo de un dispositivo se refiere a la posibilidad de ejecutar el propio escáner dentro de la máquina que se pretende escanear.

Esta característica permite un escaneo de datos de bajo nivel, como pueden ser servicios específicos de la máquina, detalles de su configuración, el propio sistema de ficheros, así como información específica del software y sistema operativo que utiliza. Permite analizar si las cuentas creadas en la máquina escaneada tienen contraseñas por defecto, o incluso si no tienen contraseñas. También permite verificar si el sistema ya ha sido atacado, analizando la existencia de ficheros sospechosos o programas en ejecución con privilegios inadecuados.

Este tipo de escaneos se pueden realizar mediante escáneres basados en máquina y también utilizando escáneres basados en red con credenciales. Los primeros fueron los primeros en utilizarse para la evaluación de vulnerabilidades. Se basan en la obtención de la información mediante consultas al sistema o a través de la revisión de distintos atributos de este. Alternativamente, también se puede realizar un escaneo interno y activo mediante un escáner de red, lo que se conoce como escáner basado en red con credenciales. De esta manera, el escáner, accediendo al sistema con las credenciales, normalmente a través de conexiones SSH, puede ejecutar los mismos controles que tradicionalmente se realizaban salvo con los escáneres basados en máquina.

Escaneo externo y activo de un dispositivo

El escaneo externo y activo de un dispositivo es un tipo de escaneo que se realiza mediante las herramientas conocidas como escáneres basados en red. En este caso, dicho escaneo se puede categorizar como un escaneo sin credenciales, en el sentido de que el actor que escanea un dispositivo no tiene acceso a él. En esta situación, la información obtenida del proceso de escaneo es una información semejante a la que puede ver un atacante.

Este tipo de escáneres de vulnerabilidades se instala en una máquina que será la encargada de escanear distintos dispositivos de la red. A través de la red el escáner obtiene la información necesaria, mediante las conexiones que establece con el objetivo que hay que analizar. Esta característica facilita la instalación de los escáneres basados en red, dado que al instalarse en máquinas distintas a las que escanea no es preciso instalar ningún software concreto en los dispositivos que se pretenden escanear. Este tipo de escáneres permite la detección de cortafuegos mal configurados, servidores web vulnerables, riesgos asociados a software utilizado en los servidores, así como los riesgos asociados a una mala administración tanto de los servidores como de la red. Cabe destacar que, a diferencia de los escaneos internos, el escaneo externo no permite detectar ciertas vulnerabilidades porque no tienen acceso al propio dispositivo.

Si se sitúa el escáner detrás de un cortafuegos esto implica que los resultados del escaneo se vean filtrados por las propias reglas de éste. Si realizamos un escaneo de la red interna desde fuera del cortafuegos se estará analizando solamente las vulnerabilidades que se pueden explotar desde fuera de la red, pero no se tendrá información de las posibles vulnerabilidades que se encuentran dentro de ella, una vez que un posible atacante haya conseguido burlar la seguridad del cortafuegos. Por este motivo, es muy importante tener en cuenta la topología de la red.

Es muy importante tener en cuenta la topología de la red para el posicionamiento del escáner, el proceso de los diferentes escáneres situados en distintos puntos de la red nos permita tener una idea clara de las vulnerabilidades de nuestro sistema dependiendo desde dónde se accede, antes o detrás del cortafuegos.

Dentro de los escáneres basados en red podemos encontrar distintos tipos, como por ejemplo escáneres de propósito general, escáneres de puertos, escáneres de servidores web o escáneres de aplicaciones web. Es muy importante tener en cuenta que el tráfico que puede generar el escaneo exhaustivo de distintos dispositivos puede provocar un aumento sustancial en el volumen de tráfico en la red que provoque su saturación, pudiendo llegar a provocar una denegación de servicio de la red.

Nessus es un escáner de vulnerabilidades activo de propósito general que puede trabajar tanto con credenciales (escaneo interno) como sin ellas (escaneo externo). Su gran popularidad se debe a que hasta su versión 3 se distribuía bajo licencia GPL (*general public license*) de GNU, pero actualmente su distribución es comercial mediante la compañía TENABLE Network Security. Sin embargo, sigue siendo el escáner de vulnerabilidades más utilizado.

Otro aspecto importante que cabe tener en cuenta en la utilización de escáneres externos es la protección de la información obtenida del escaneo. Cuando se ejecuta un escaneo externo y activo se está generando un conjunto de información referente al dispositivo o dispositivos que se están escaneando, que puede ser interesante para un atacante, puesto que puede identificar posibles vulnerabilidades sin la necesidad de ejecutar análisis que puedan resultar sospechosos. Por este motivo, la información generada por el escaneo que circule por la red debe intentar protegerse, en la medida de lo posible, utilizando técnicas de cifrado.

Desde el punto de vista del funcionamiento, dos de las técnicas más utilizadas para la evaluación de vulnerabilidades basadas en red son las siguientes:

- **Prueba por explotación**. Esta técnica consiste en lanzar ataques reales contra el objetivo. Estos ataques están programados normalmente mediante guiones de comandos. En lugar de aprovechar la vulnerabilidad para acceder al sistema, se devuelve un indicador que muestra si se ha tenido éxito o no. Obviamente, este tipo de técnica es bastante agresiva, sobre todo cuando se prueban ataques de denegación de servicio.

- **Métodos de inferencia**. El sistema no explota vulnerabilidades, sino que busca indicios que indiquen posibilidades de ataque, tratando de detectar posibles deficiencias de seguridad en el objetivo. Este método es menos agresivo que el anterior, aunque los resultados obtenidos son menos exactos.

Escaneo externo y pasivo de un dispositivo

El escaneo externo y pasivo de dispositivos es una técnica que combina las capacidades de escucha de los *sniffers* con las capacidades de análisis de los escáneres de vulnerabilidades activos para detectar vulnerabilidades en los sistemas.

Un escáner pasivo de vulnerabilidades se coloca en la red en una posición en la que se puede controlar el tráfico que viene de varios segmentos, de manera similar a lo que se haría con un sistema de detección de intrusos. El escáner pasivo escucha el tráfico en tiempo real y lo analiza mediante la comparación con un conjunto de reglas, como un escáner de vulnerabilidades activo, de modo que si se incumplen las reglas establecidas, se alerta al administrador de la red. Estas características permiten detectar vulnerabilidades de manera más continua que los escáneres activos. Si bien un escáner de vulnerabilidades pasivo puede parecer lo mismo que un sistema de detección de intrusos, es importante destacar que las tareas de análisis de tráfico que realizan ambos son distintas.

Por ejemplo, si suponemos las miles de conexiones que se pueden realizar a un servidor web, un sistema de detección de intrusos deberá analizar todas y cada una de ellas para identificar un ataque, mientras que el análisis que lleva a cabo un escáner pasivo se puede llevar a cabo únicamente con el análisis (tan exhaustivo como se requiera) de una única conexión que tiene como destino el servidor que se pretende escanear.

Una de las principales ventajas de los escáneres pasivos es la poca incidencia que tienen sobre los sistemas que analizan. Dado que se trata de un análisis de la información que viaja por la red, los escáneres pasivos son muy poco intrusivos y no afectan al rendimiento del sistema que se está escaneando, hecho que puede ocurrir con los escáneres activos. Esta característica les permite ser utilizados en sistemas críticos en los que no se puede permitir una disminución del rendimiento o la eventual parada del sistema, que podría llegar a provocar un escaneo activo.

Una de las principales ventajas de los escáneres pasivos es su capacidad de análisis continuo, característica que no presentan los escaneos activos. El modo de análisis del escaneo activo le confiere una visualización instantánea del estado del sistema que se analiza, que se asemeja a una fotografía de la situación de los sistemas en el instante preciso que se realiza el escaneo. Esto implica que una modificación del sistema analizado con posterioridad al escaneo activo (por ejemplo, por una actualización o la instalación de nuevo software) puede dar lugar a una vulnerabilidad sin que el proceso de escaneo lo detecte. Por el contrario, los escáneres pasivos analizan constantemente el tráfico alertando de posibles vulnerabilidades cuando estas se detectan.

Esta idea de continuidad en el análisis nos lleva hasta una diferencia temporal entre un escáner activo y uno pasivo. Los escáneres pasivos precisan un tiempo para la realización del análisis. Por ejemplo, hasta que el usuario no se comunica con el servidor, el escáner no puede analizar si el puerto por el cual el servidor se comunica tiene algún servicio con una vulnerabilidad. Sin embargo, en un escaneo activo, el propio escáner es quien inicia la comunicación con el servidor y por tanto, en cualquier momento, puede determinar si existe la vulnerabilidad en dicho puerto.

Es importante destacar que un servidor no siempre puede responder cuando el escáner activo le interrogue. Además, en esta situación, se asume que el escáner activo tiene identificadas las máquinas que debe escanear. Esta suposición, si bien puede parecer adecuada, en ocasiones no es real, puesto que los administradores desconocen la existencia de máquinas o servicios que requieren su escaneo.

Ejemplo

Un usuario podría iniciar un servidor FTP en una máquina que no estuviera autorizada a ello y esto podría pasar inadvertido al administrador. Un escáner activo no podría detectar una posible vulnerabilidad en la versión de servidor FTP iniciada puesto que el escáner no analizaría dicha máquina, dado que supuestamente no tendría que estar albergando un servidor FTP. Sin embargo, el análisis de tráfico que realiza un escáner pasivo podría detectar la utilización del servicio de FTP en dicha máquina y analizar la posible existencia de vulnerabilidades.

Otra habilidad que proporciona el escaneo pasivo es la optimización del proceso de escaneo, no escaneará todo, sino únicamente las máquinas que realmente sean necesarias por albergar algún tipo de servicio.

Por último, otra de las ventajas de un escaneo pasivo es la posibilidad de análisis de vulnerabilidades del cliente en un entorno cliente-servidor. Dado que los escáneres pasivos analizan el tráfico de la red, estos pueden detectar vulnerabilidades en la parte de la comunicación del cliente. Esta es otra ventaja de los escáneres pasivos, dado que los escáneres activos se focalizan en el análisis de vulnerabilidades de la parte del servidor, descuidando la parte del cliente.

Si bien hemos visto distintas ventajas de los escáneres pasivos, existen también algunas limitaciones en su uso. Una de las principales desventajas de los escáneres pasivos es la dificultad de fijar su correcto emplazamiento. Al igual que los sistemas de detección de intrusos, la determinación del emplazamiento de los analizadores de red es de vital importancia para la efectividad del escáner, puesto que el tráfico que circule por el segmento de red donde se sitúa el analizador será el que proporcionará la información para el análisis.

Otra de las limitaciones que presentan los escáneres pasivos es la dependencia que tienen de los datos que analizan. Si el escáner se limita a analizar las cabeceras de los paquetes que circulan por la red para determinar, por ejemplo, el tipo de sistema operativo que se encuentra en una máquina, es posible que un atacante pueda manipular la información de los paquetes para que el escáner pasivo proporcione información incorrecta o genere tanta información que el escáner no pueda analizar. Es decir, el nivel de análisis que el escáner realiza de los datos que obtiene determinará la calidad de las alertas que genere.

3. Malware (troyanos/virus)

A pesar de que la idea del código malicioso pueda parecer novedosa, la realidad es que si queremos buscar sus orígenes debemos remontarnos al año 1949. Cuando el matemático John von Neumann presentó varias conferencias en la Universidad de Illinois bajo el nombre de *Theory and Organization of Complicated Automata*, en las que englobaba la teoría sobre autómatas complejos. En estas conferencias, Neumann estableció la idea de programa almacenado y teorizó por primera vez la posibilidad de que un programa pudiera replicarse por sí mismo. Sin duda alguna, esto supuso un resultado plausible para la teoría de la computación. Posteriormente, su investigación fue publicada en el año 1966 en el libro titulado *Theory of self-reproducing automata*.

Durante los años setenta aparecieron los primeros programas capaces de autorreplicarse según las teorías que ya había postulado John von Neumann tiempo atrás. Fue más tarde, en el año 1983, cuando Frederick B. Cohen acuñó el término virus para referirse a un programa capaz de autorreplicarse. Un año más tarde, y según las sugerencias de su mentor Leonard M. Adleman, Cohen empleó el término *virus informático*.

> Cohen define un virus informático (*computer virus*, N. del T.) como un "programa que puede infectar a otros incluyendo una copia posiblemente evolucionada de sí mismo".
>
> Cohen (1984)

Cohen profundizó en este campo, investigando sobre otras propiedades de los virus informáticos, y formalizó la definición de virus basándose en el modelo de la máquina de Turing en su tesis doctoral (Cohen, 1986).

En la década de los ochenta, y en paralelo a los trabajos de Cohen, los ordenadores personales se popularizaron y supuso un nuevo nicho de oportunidad para los virus informáticos. Esta fue la época de explosión de los virus, los cuales empezaron a incluir en su código rutinas con finalidades maliciosas. Sin duda, era el nacimiento del *malware*.

La evolución del código malicioso ha sido constante y especialmente de carácter empírico. Cada día, nuevos tipos de código malicioso más complejos aparecen, hasta el extremo de que se han tenido que acuñar nuevos términos para referirnos a cada una de las nuevas variantes. En paralelo, se ha trabajado en establecer definiciones formales válidas para cualquier forma de malware, como la presentada por Kramer y Bradfield (2010). Sin lugar a dudas, estos avances han sido posibles gracias a los trabajos previos de investigadores como Neumann, Cohen o Adleman, cuyas contribuciones han permitido establecer las bases para la lucha contra el *malware* actual.

El término *malware* proviene de la contracción de las palabras inglesas *malicious* y *software*. En español utilizamos los términos software malicioso o código malicioso. El *malware* es un tipo de software intrusivo y hostil que tiene como objetivo infiltrarse o dañar un sistema de información sin la aprobación ni el conocimiento de su propietario.

3.1. Taxonomía del malware

A lo largo de los años, la proliferación, la diversidad y la sofisticación del software malicioso han crecido de forma espectacular. Hoy en día resulta complejo establecer una taxonomía completa para todos los tipos, de manera que cada categoría que la compone permita clasificar el malware de forma excluyente. Es decir, dado un espécimen de malware particular, es posible que este tenga características híbridas que pertenezcan a más de una de las categorías en el que podemos clasificarlo. Así, es habitual, por ejemplo, encontrar código malicioso que pueda ser considerado como de propagación y oculto al mismo tiempo. Por tanto, la taxonomía del software malicioso que presentamos debe ser entendida como una categorización genérica, en la que un código malicioso dado puede pertenecer a más de una clase de forma simultánea.

En particular, consideramos que hay tres grandes categorías de software malicioso. Estas son, *malware de propagación automática*, *malware oculto* y *malware lucrativo*. Seguidamente definiremos cada una de ellas.

3.1.1. Malware de propagación automática

El *malware* de propagación es aquel cuya principal finalidad es la de extenderse de forma automática infectando nuevos sistemas de información.

Dependiendo de la forma que emplea para propagarse distinguimos dos subcategorías:

- **Malware de propagación por infección vírica**. Es aquel en el que el código malicioso se replica a sí mismo al añadirse a archivos ejecutables, o con la capacidad de ejecutar algún tipo de código. Asimismo, este modifica el código del programa original para que, en algún instante, el flujo de ejecución sea redirigido a las instrucciones del malware. Cada vez que el código malicioso toma el control, busca nuevos archivos no contaminados con la intención de infectarlos y, tras esto, devuelve el control al programa original. A esta misma categoría, y de forma análoga al caso anterior, pertenece el *malware* que infecta el *master boot record* (MBR) de los discos duros, lo que le permite ejecutarse tras las pertinentes operaciones de la BIOS, y tomar el control antes de que se cargue el sistema operativo. Al *malware* que emplea este tipo de propagación se le conoce tradicionalmente como virus, y su nombre se debe a la analogía que existe con los virus biológicos, y a cómo estos infectan y se multiplican dentro de las células de otros organismos. En ocasiones, el término virus se utiliza de forma errónea para referirse a otros tipos de *malware*, tales como el *spyware* o los *troyanos*, que no deben ser confundidos dado que estos no infectan a otros archivos.

- **Malware de propagación como gusano**. Es aquel que se replica a sí mismo empleando la red a la que está conectado el sistema infectado, enviando copias de sí mismo a otros sistemas de la red que no están contaminados. Este envío se realiza sin la intervención del usuario, y puede emplear varias estrategias para propagarse, tales como la explotación remota de un desbordamiento de buffer conocido, el envío indiscriminado de correos electrónicos infectados con el malware, las redes P2P, o los clientes de mensajería instantánea entre otros. Al malware que emplea este tipo de propagación se le conoce tradicionalmente como gusanos.

3.1.2. *Malware* oculto

El *Malware* oculto es un tipo de software malicioso que se caracteriza por intentar permanecer desapercibido para el usuario dentro del sistema infectado.

Dependiendo de la forma en la que intenta pasar desapercibido, y del propósito del *malware*, distinguimos entre tres categorías:

- ***Rootkits***. Este tipo de código malicioso tiene sus orígenes en los sistemas Unix, y su denominación se utilizaba para hacer referencia al conjunto de herramientas que permitían a un atacante obtener privilegios de administrador (*root*). Hoy en día, el término *rootkit* se emplea de forma más generalizada para designar al conjunto de técnicas que permiten eludir la detección y la eliminación de cualquier *malware*. Estas técnicas se basan en la modificación del sistema operativo de la máquina infectada a bajo nivel, permitiendo la ocultación de procesos, archivos o conexiones de red utilizadas por el software malicioso

- **Troyanos**. El software malicioso que pertenece a esta categoría se caracteriza por estar enmascarado detrás de un supuesto programa legítimo. La víctima, previo a la instalación de dicho software, lo percibe como un programa que proporciona unas funcionalidades de su interés. Sin embargo, tras la instalación, y sin el consentimiento del usuario y sin que este sea consciente, el malware actúa escondiéndose detrás de un software aparentemente lícito. La actividad oculta que lleva a cabo el código malicioso puede ser de distinta índole, aunque normalmente suele estar enfocada al control remoto de la máquina infectada, o bien al robo de información.

- **Puertas traseras (*backdoors*)**. Esta categoría engloba a todo software malicioso que es instalado en un sistema ya comprometido, y que permite eludir los mecanismos de autenticación a la vez que permanece oculto a los administradores del sistema. De esta manera, una puerta trasera en una máquina infectada permite a un atacante garantizar el acceso al sistema en un futuro de una manera sencilla y rápida. A pesar de que la existencia de las puertas traseras no es una idea nueva, la proliferación de este tipo de código malicioso ha tenido especial relevancia desde la explosión de Internet. En ocasiones las puertas traseras pueden adoptar forma de troyano e incluso puede incluir técnicas de *rootkits*.

3.1.3. *Malware lucrativo*

El software malicioso que pertenece al *malware* lucrativo se caracteriza, como sugiere su nombre, por proporcionar algún tipo de beneficio al atacante. Aunque no siempre es así, en la mayoría de ocasiones el tipo de beneficio es de carácter económico. Dependiendo de la finalidad que persigue y de cómo actúa, distinguimos entre las siguientes subcategorías:

- ***Spyware***. Es un software malicioso que registra información sensible de usuarios sin su consentimiento, violando la privacidad de estos. La información recogida por este tipo de aplicaciones puede ser de distinta índole, como por ejemplo, datos personales, números de tarjeta de crédito, hábitos de

navegación web, contraseñas, pulsaciones de teclas, o incluso capturas de pantalla.

- ***Ransomware***. Es un tipo de malware que extorsiona a los usuarios propietarios de una máquina infectada, exigiendo algún tipo de pago tras cifrar archivos, o bien desactivar o bloquear partes del sistema. Si el usuario realiza el pago –usualmente vía transferencia bancaria mediante *bitcoins* usualmente o SMS con cargo adicional–, el atacante proporciona algún mecanismo para eliminar el perjuicio causado por el mismo código malicioso. En el caso particular del malware que hace uso de la criptografía para cifrar archivos se le conoce como *criptovirus*. Estos suelen basarse en esquemas criptográficos asimétricos o híbridos, lo que impide el acceso al contenido de los archivos cifrados al no disponer la víctima de la clave privada.

- ***Scareware***. Es un código malintencionado que, basándose en estrategias de ingeniería social, puede proporcionar beneficios económicos al atacante. En concreto, explota el engaño, la persuasión, la coacción o el miedo a través de mensajes de alarma o de amenaza para forzar a la víctima a realizar un pago. El ejemplo más común es el de programas que detrás de la apariencia de software para la detección de malware esconden este tipo de código malicioso. Una vez instalado, el programa reporta la existencia de una cantidad elevada de software malicioso en el sistema, cuando la realidad no es así. Entonces, el *scareware* brinda la posibilidad al usuario de eliminar las amenazas detectadas pagando por una versión diferente del software de detección. A este tipo de *scareware* se le conoce con el nombre de *rogueware*. Otras formas de *scareware* se basan en explotar el sentimiento de culpa y el miedo que pueden sentir algunos usuarios al descargar software ilegal. En particular, una vez instalado, este tipo de *scareware* muestra mensajes de advertencia indicando que se están violando las leyes del copyright y que se tiene identificada la IP del usuario. Seguidamente, propone evitar un juicio realizando un pago como forma de solventar la situación.

- ***Bot***. Es un código malicioso que permite a un atacante controlar de forma remota la máquina que lo ejecuta. Al conjunto de máquinas distribuidas e infectadas por *bots*, y controladas por un atacante, se le conoce con el nombre de *botnet*. La suma de recursos proporcionados por cada máquina infectada permiten realizar actividades fraudulentas, como son ataques de denegación de servicio distribuidos o el envío masivo de SPAM.

- ***Adware***. Es un tipo de malware que de forma automática muestra publicidad no consentida al usuario, con la finalidad de que realice algún tipo de compra. Esta publicidad suele aparecer en los navegadores web como ventanas emergentes, siendo un comportamiento molesto e indeseable para el usuario.

3.2. Vectores de infección

Uno de los aspectos importantes a tener presente contra la lucha del *malware* es la identificación de los posibles vectores de infección. Conocer estas vías de infección nos permite centrar nuestros esfuerzos en diseñar e incorporar mecanismos de seguridad adecuados.

En términos generales, podemos distinguir dos estrategias posibles para la infección de un sistema: los procesos de infección iniciados por el usuario víctima, y los

procesos de autoinfección iniciados a través de vulnerabilidades existentes en los sistemas. En el primer caso, el *malware* suele venir camuflado en programas supuestamente legítimos y que, sin el consentimiento del usuario, se instala junto a este software. Así, programas de tipo P2P, complementos para los navegadores web, software descargado de forma ilegal, o *cracks*, son ejemplos de programas que pueden esconder *malware*. En otras ocasiones, la vía de infección se basa en el acceso a una determinada web con componentes *ActiveX* o *Applets* Java especialmente preparados, que, tras la autorización de su ejecución por parte del usuario, conllevan la instalación del código malintencionado. En cualquier caso, el proceso de infección requiere una aprobación de ejecución tácita por parte del usuario. La utilización de la ingeniería social suele estar presente en esta metodología. De esta manera, los atacantes utilizan estrategias para persuadir a los usuarios a descargar determinado software, o realizar determinadas acciones que conlleven la instalación del *malware*.

En el segundo caso, los mecanismos de infección del código malicioso se basan en la explotación de vulnerabilidades en el software. Así, la visita a una determinada web preparada haciendo uso de un navegador vulnerable, o la apertura de un archivo especialmente preparado mediante software que incluya deficiencias de seguridad, pueden provocar la ejecución de código arbitrario que conduzca a la instalación del malware. En otras ocasiones, un servicio vulnerable ofrecido en una red a través de un puerto TCP/IP puede ser explotado por el malware para la infección. En cualquier caso, en este proceso de infección, la instalación del software malicioso suele pasar totalmente desapercibida para el usuario afectado, sin requerirse ningún tipo de acción a realizar que pueda considerarse motivo de sospecha.

3.3. Mecanismos de prevención

Podemos considerar cuatro métodos principales para prevenir nuestros sistemas de las infecciones de malware:

Fomento de buenas prácticas. En primer lugar, fomentar las buenas prácticas por parte de los usuarios, manteniendo actualizado tanto el sistema operativo como las aplicaciones, impedir las descargas de software de fuentes no fiables o ignorar los correos y contenidos adjuntos de remitentes desconocidos.

Diseño de patrones de protección. Otras formas de prevención más técnicas se basan en el diseño de patrones de protección. El objetivo de estos patrones es impedir la infección de un sistema o bien reducir el daño de la infección. Podemos agrupar dentro de esta categoría la utilización de anillos de protección. Dichos anillos establecen una estructura de confianza por capas en el sistema operativo y se complementan con hardware específico para poder realizar una separación efectiva entre procesos de confianza y procesos sospechosos. A través de esta solución, se pueden ofrecer distintos niveles de acceso a los recursos del sistema. Los anillos se organizan de forma jerárquica, estructurando aquellos dominios más privilegiados y confiables hasta los de menores privilegios y nivel de confiabilidad. De este modo, se reduce el riesgo de que procesos de tipo *malware* ataquen al núcleo del sistema operativo. Estos métodos han demostrado que, aunque reducen las consecuencias de una infección, no son totalmente efectivos.

Uso de firmas digitales. El tercer mecanismo genérico consiste en verificar la autenticidad del código que se está ejecutando mediante la utilización de firmas digitales. Estas firmas se asociarán al código y se verificarán antes de su ejecución. Las dificultades surgen aquí para aquel código que no haya sido firmado. En este

caso, los usuarios deberán decidir entre no usar sus funcionalidades, o bien exponerse a ciertos niveles de riesgo si dicho programa es lanzado. La práctica ha demostrado que, ante situaciones de este tipo, un porcentaje elevado de usuarios prefieren ejecutar el software antes que verificar su legítima procedencia o validez.

Uso de aplicaciones automáticas. Las tendencias actuales se centran en la investigación de aplicaciones automáticas capaces de detectar y aislar código de tipo malicioso.

3.4. *Botnets*

Una *botnet* (también conocida como red de equipos robot o red de *zombies*) es entendida, hoy en día, como un conjunto de equipos informáticos de todo tipo conectados a Internet, cuyos recursos (tales como memoria, ejecución de procesos, sistema de ficheros y conexiones de red) son controlados, a distancia, sin que sus usuarios y/o propietarios sean conscientes (figura 12). Los operadores de una botnet (a menudo conocidos bajo el sobrenombre de *botmasters* o *botherders*) crean la red mediante la propagación de código malicioso, que infecta los recursos de los futuros equipos de la *botnet* y garantiza el control permanente de los estos.

El término *botnet* proviene de la superposición de dos palabras inglesas: roBOT y NETwork. Una traducción al español sería, por lo tanto, red de robots.

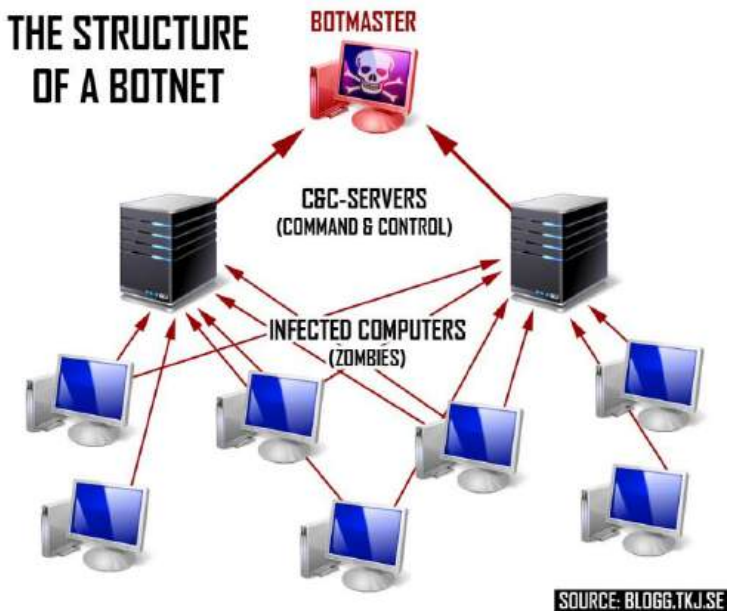


Fig. 12 conexión de botnet

Una vez infectados, los equipos de la *botnet* son vistos por sus operadores como un conjunto de robots, o *zombies*, software al servicio de los clientes de la *botnet*. Estos clientes podrán alquilar los equipos para realizar actividades tales como campañas de *spam*, denegaciones de servicio distribuidas, almacenamiento de contenidos multimedia ilícitos, etc. La mayoría de los equipos infectados, y futuros robots de la

botnet, suelen ser ordenadores domésticos, a menudo activamente conectados a Internet durante largos períodos de tiempo (horas, días, semanas) y con unos niveles de protección (en cuanto a seguridad se refiere) bastante bajos. Aunque ya se han detectado sistemas de los llamados *Internet of Thing* que están dentro de estas redes de *botnets*.

El operador de la *botnet* enviará periódicamente mensajes de control vía protocolos tradicionales como, por ejemplo, IRC (*Internet relay chat*) o HTTP (*hypertext transfer protocol*). Los robots de la *botnet* pueden incluso ser reprogramados por terceras partes con el objetivo de ampliar sus dominios o retomar los recursos de *botnets* ya existentes.

Aunque no siempre han sido consideradas maliciosas, las botnets aparecen a finales de la década de los ochenta. Aparece relacionada con la invención del protocolo IRC (*Internet relay chat*) y con la versión en red de *Hunt the Wumpus* a través de canales IRC. De hecho, y como sucede con muchas otras tecnologías fraudulentas asociadas a Internet, el origen de esta primera *botnet*, totalmente lícita e inofensiva, fue concebida para la automatización de tareas de gestión virtuales de canales IRC. Los equipos de esta primera botnet tenían por objetivo la construcción y el mantenimiento de procesos asociados a juegos para usuarios IRC. Los robots de la plataforma debían estar disponibles las 24 horas del día para ofrecer a los usuarios la posibilidad de jugar con ellos. Rápidamente, y de manera espontánea, redes similares fueron desplegadas para dar soporte a operadores de otros servicios.

Un aspecto importante en el desarrollo de estos precursores de las botnets actuales es la capacidad de crear un canal de comunicación entre los operadores y los robots, así como la inclusión de mecanismos de control de acceso para evitar que terceras partes pudieran tomar el control de los equipos de la red de gestión. Por ello, encontramos en esta época una arquitectura basada en canales de control, a través de los cuales el operador podrá comunicar instrucciones de gestión, tales como inicialización de servicios, reanudación de tareas, operaciones de actualización y mantenimiento de versiones. Estos robots fueron, así pues, evolucionando desde simples programas autónomos capaces de jugar y entretener a internautas, hacia gestores automatizados de tareas y lanzamiento de nuevas aplicaciones al servicio de terceras partes. Es común encontrar en el código fuente de estos robots de finales de los noventa la posibilidad de creación de cuentas de usuarios con privilegios estratificados, así como la inclusión de consolas de comando y la posibilidad de ejecución de macros y scripts por parte de usuarios con suficientes privilegios.

Hasta finales de la década de los noventa no podemos encontrar la aparición de las primeras *botnets* con connotaciones fraudulentas o malvadas. Uno de los primeros casos relevantes que debemos destacar es el despliegue de robots basado en la infección a gran escala del gusano *IRC/Jobbo* y la posterior instalación en las máquinas infectadas de la herramienta *SubSeven*. El gusano *IRC/Jobbo* tuvo como vector de transmisión la explotación remota de errores de programación en clientes de IRC de la época (mayoritariamente, el cliente mIRC para sistemas MS Windows). Mediante la explotación de vulnerabilidades, y la posterior escalada de privilegios, el resultado fue la construcción de una red de equipos controlados mediante la inyección en las víctimas de *malware* de tipo troyano. La herramienta instalada en dichos equipos, *Subseven*, ofrecerá al operador de la botnet un control de administración total sobre cada máquina infectada.

4. Detecciones

4.1. Detección de *malware*

Sin duda alguna, la detección del software malicioso ha sido una de las medidas más extendidas como mecanismo de prevención. Analizaremos las técnicas más comunes empleadas por software especializado en la detección de *malware*. Como veremos, la complejidad de estas estrategias varía entre ellas. Esta divergencia de complejidades obedece a la evolución que ha experimentado el código malicioso a lo largo del tiempo, cuya finalidad siempre ha sido evadir los sistemas de detección. Así, software malicioso que ha incorporado mecanismos para dificultar su detección ha requerido de nuevas formas más sofisticadas de análisis.

A pesar de los progresos que se han hecho en el campo del software para la detección de *malware* en los últimos años, es importante remarcar que, tal y como ya postuló Cohen (1987) en sus trabajos sobre los virus, el problema de la detección perfecta de software malicioso es un problema indecidible. No existe un programa capaz de detectar la totalidad de variantes de código malicioso que pueden llegar a existir. Por lo tanto, no siempre será posible realizar una detección proactiva, lo que conduce, en alguna ocasión, a una inevitable infección de los sistemas.

Sin embargo, esto no significa que no podamos diseñar mecanismos que nos permitan detectar y luchar contra un subconjunto de la totalidad del espectro del *malware*. En la actualidad, múltiples estrategias y modelos han sido propuestos para la detección del *malware* a través de software especializado. Todos los modelos de detección de *malware* pueden ser clasificados en dos categorías en función del tipo de detección que realizan. En particular, tenemos los modelos imprecisos y los exactos. En el caso de los imprecisos, el motor de detección sólo es capaz de determinar si un determinado objeto se trata de *malware* o no, sin llegar a precisar los detalles de la versión o variante de código malicioso de que se trata. En contraposición, los modelos exactos son capaces de realizar una detección concisa, proporcionando información particular acerca de la versión de software malicioso detectado. Evidentemente, el uso de un modelo u otro tiene consecuencias en el procedimiento de eliminación del código malicioso en un sistema infectado, siendo en el exacto un proceso más directo al conocerse los detalles de la infección de antemano.

Entre las diferentes tecnologías para la detección nos centraremos en el estudio de dos concretas por su amplio uso en la actualidad. En primer lugar, analizaremos la detección sintáctica basada en firmas, un modelo exacto que, por su naturaleza, nos impide luchar contra mecanismos de ofuscación que pueden incorporar el malware. En segundo lugar, veremos cómo la detección semántica se presenta como una manera de superar las deficiencias de la detección basada en firmas, sin embargo, éste se basa en un modelo impreciso con las consecuencias que esto supone.

4.1.1. Detección sintáctica basada en firmas

Desde un punto de vista histórico, la detección sintáctica basada en firmas fue la primera técnica empleada para la identificación de software malicioso. Actualmente,

esta tecnología continúa siendo parte del núcleo de los motores de detección de malware, y se caracteriza por su ratio baja de falsos positivos.

La detección sintáctica basada en firmas es una estrategia que se basa en localizar dentro de objetos potencialmente maliciosos (comúnmente ficheros o procesos) algún patrón que identifique a un determinado malware conocido. Estos patrones –también conocidos como firmas sintácticas– vienen expresados como una secuencia de bytes, y representan cadenas de texto o instrucciones de bajo nivel en forma de *opcodes*. Existen varios tipos de firmas con características diferentes.

Los motores de detección sintáctica disponen de una base de datos de estas firmas que definen el conjunto de software malicioso reconocible. Si el motor encuentra alguna de las firmas de la base de datos en un objeto, este se identifica de forma fehaciente como un malware específico. Para conseguir localizar firmas dentro de un objeto, diversos algoritmos de búsqueda y concordancia pueden ser empleados. Estos algoritmos están optimizados y tienen una complejidad acotada en función del tipo de firma que se utiliza. Asimismo, estos motores pueden mejorar el rendimiento al limitar la búsqueda a partes estratégicas de los objetos, y no realizar una búsqueda exhaustiva en todo su contenido.

La tecnología de detección de software malicioso basado en firmas sintácticas presenta diversas deficiencias, lo que lo convierte en un método ineficaz bajo ciertas circunstancias:

1) Las firmas de una base de datos siempre están asociadas a software malintencionado conocido, no permitiendo la detección de nuevas formas de código malicioso. Así, si un nuevo malware es liberado y su firma no está presente en la base de datos de un sistema, este podría ser infectado al pasar desapercibido para el motor de detección. De hecho, algo tan simple como la modificación de un código malicioso conocido a nivel de la firma que lo detecta permite evadir su detección. En este sentido, el software malicioso ha ido incorporando a lo largo del tiempo estrategias de evasión que se benefician de esta debilidad. Dichas estrategias se basan en crear, de forma automática, mutaciones del código malicioso en cada nueva infección. De esta manera no es posible establecer patrones de detección únicos, ya que una firma válida para un malware en concreto no será útil para la identificación de su versión mutada.

2) La generación de las firmas puede conllevar un proceso largo y tedioso de análisis, lo que supondría dejar expuestos a una infección a los sistemas durante un tiempo elevado. En particular, cuando se localiza un objeto que se presume como malicioso, es habitual aplicar ingeniería inversa a su código para determinar si se trata o no de malware. Tras este análisis, y una vez que el objeto puede ser catalogado como código malicioso, se buscan características que lo identifiquen de forma única. A partir de estas características de unicidad se genera la firma, que posteriormente se distribuye e incorpora a las bases de datos. Este proceso, que requiere de intervención humana, no siempre es trivial, y el tiempo necesario se ve influenciado por las técnicas incluidas en el malware para dificultar su análisis. Por tanto, desde que el software malintencionado se libera, hasta que las firmas son creadas e incorporadas a las bases de datos, existe un periodo de tiempo crítico durante el cual no es posible la detección por parte de los motores sintácticos.

3) Y por último, a pesar de que la distribución de las firmas en los mejores casos se realiza de forma automática, existen motores en los que se requiere la

intervención del usuario para lanzar el proceso de actualización de la base de datos. Si este proceso manual no se efectúa a tiempo, un sistema puede verse comprometido por el software malintencionado al no poderse detectar.

Tipos de firmas

Hemos podido ver cómo las firmas sintácticas son el pilar de la metodología de detección que estamos tratando. Con el objetivo de mejorar la precisión en la identificación de malware se utilizan cuatro tipos de firmas sintácticas:

1) **Firmas como cadenas estáticas**. Las firmas como cadenas estáticas son la forma más básica entre las distintas posibles. Estas vienen definidas como una secuencia de bytes consecutivos de longitud arbitraria. Si esta secuencia se localiza en un objeto analizado, entonces es identificado como código malicioso.

2) **Firmas con expresiones regulares**. Este tipo de firmas soporta expresiones regulares en su definición. De esta manera es posible localizar concordancias de bytes según repeticiones, rangos, combinaciones, etc. Debido a esto, el algoritmo de búsqueda y concordancia tiene una mayor complejidad que el de las cadenas estáticas y, por tanto, menor velocidad en el proceso de identificación de *malware*. Sin embargo, el soporte de expresiones regulares le dota de mayor flexibilidad para detectar software malicioso más complejo, o variantes respecto a una versión de *malware* ya conocida. Así, a menudo, una única firma que utiliza expresiones regulares permite detectar toda una familia de variantes de un determinado código malicioso. Como es lógico pensar, estas firmas genéricas son útiles si, al liberarse una variante, esta comparte el mismo patrón identificativo con versiones anteriores.

3) **Firmas basadas en funciones hash criptográficas**. Esta variante de firmas sintácticas se sustenta en el uso de funciones hash criptográficas. Estas funciones matemáticas permiten guardar las firmas como el cálculo de una función hash sobre un objeto (o parte de este) correspondiente a código malicioso. Las ventajas del uso de este tipo de firmas es doble. En primer lugar, pueden reducir el espacio necesario para almacenar una firma, siempre y cuando el tamaño resultante de calcular la función hash sea inferior al tamaño de otro tipo de firma alternativa. En segundo lugar, una firma sintáctica basada en funciones criptográficas puede ser computacionalmente menos costosa de localizar que los algoritmos de búsqueda de otros tipos de firmas.

4) **Combinaciones lógicas de firmas**. Este tipo de firmas combina múltiples subfirmas sintácticas relacionándolas mediante operadores lógicos, permitiendo definir patrones más flexibles y precisos. Por tanto, la búsqueda de la nueva firma para la identificación de un determinado malware pasará por verificar que se cumple la expresión lógica compuesta por todas las subfirmas. Esta estrategia puede ser utilizada para, por ejemplo, detectar una familia de variantes de un código malicioso concreto. Así, se podría definir una firma sintáctica genérica para todas las variantes, y luego una específica para cada variante particular. Para este escenario, combinaríamos con el operador lógico AND la firma genérica con la particular para la detección de una variante concreta.

Cada una de estas cuatro firmas tiene asociado un algoritmo de búsqueda y concordancia para la identificación de código malintencionado, y cuya complejidad algorítmica es diferente para cada uno. La utilización de un tipo u otro de firma varía

en función del software malicioso que debe identificar y, por tanto, no se puede afirmar que ninguna de ellas sea mejor que otra. Dependiendo del caso particular de software malintencionado, la elección de una u otra mejorará la precisión y el tiempo necesario en la identificación. Es habitual encontrar motores de detección sintáctica que soportan varios tipos de firmas de forma simultánea.

4.2. Mecanismos de evasión del *malware*

Sin duda alguna, el software especializado en la detección de código malicioso ha sido una de las vías más empleadas contra la lucha del *malware*. Si se es capaz de detectar un código malicioso a tiempo, es posible evitar la infección. Cuando en los inicios los autores de *malware* vieron que la proliferación de sus creaciones era contrarrestada mediante este tipo de software, tuvieron que idear nuevas formas de supervivencia. Estas estrategias se centran en conseguir precisamente que el software malicioso permanezca oculto a los motores de detección. Cuanto más tiempo un código malintencionado pase desapercibido, más tiempo dispondrá para propagarse, y la infección afectará a más sistemas. Adicionalmente, estos mecanismos de evasión pretenden hacer más complejo su análisis, lo que requiere de un mayor esfuerzo por parte de los expertos y, por consiguiente, más tiempo para la generación de las firmas.

Debido al amplio uso de las estrategias de evasión por parte del *malware*, en la actualidad nos vemos en la necesidad de entender y analizar cuáles son los métodos empleados. Por tanto, presentamos en este apartado las técnicas más comunes utilizadas por el software malicioso con la finalidad de no ser detectado.

Podemos distinguir básicamente tres tipos de tecnologías de evasión:

1) las técnicas de ofuscación,
2) los métodos de ocultación y autoprotección y
3) los mecanismos *antidebugging*.

A pesar de la categorización que aquí exponemos, es importante remarcar que estas no son excluyentes. Es decir, podemos –y de hecho es muy habitual– encontrar software malicioso que combine varias técnicas de las tres categorías de forma simultánea. A continuación se describen cada una de las citadas categorías y los diversos métodos que podemos encontrar en cada una de estas.

4.2.1. Técnicas de ofuscación

Las técnicas de ofuscación pueden verse como una transformación del código de un programa con el objetivo de hacer su comprensión más difícil, mientras que al mismo tiempo se preserva su funcionalidad.

Estas técnicas han sido aplicadas tanto a la protección del software en general como a la del *malware*. Desde la perspectiva de la protección del software, la ofuscación del código permite proteger los programas de ataques que atenten contra la propiedad intelectual. En concreto, estas técnicas dificultan aplicar ingeniería inversa, lo que dificulta dichos ataques. Por otro lado, desde el punto de vista del *malware*, la finalidad de los mecanismos de ofuscación es doble. En primer lugar, incrementar la dificultad en el proceso de análisis así como el tiempo necesario para realizar este y, como consecuencia, retrasar la generación de las firmas. En segundo lugar, las estrategias

de ofuscación tienen como finalidad la de mutar el código y, por tanto, dificultar o incluso hacer inviable el uso de técnicas de detección basadas en firmas sintácticas

Compresión de ejecutables

El origen de la compresión de los ejecutables se remonta a los años ochenta, cuando la capacidad de los sistemas de almacenamiento de información era menor que en la actualidad y su coste más elevado. Por aquel entonces, y con el objetivo de obtener un mejor aprovechamiento del espacio, se utilizaron algoritmos de compresión sobre los ejecutables. Si bien es cierto que esta idea puede continuar siendo aplicada en la actualidad de forma legítima, esta estrategia también puede ser utilizada por software malicioso como mecanismo de evasión. El motivo de esto radica en la siguiente idea. Si un código malicioso se modifica ligeramente y se libera como una nueva versión, es probable que comparta porciones de código respecto a su predecesor y que, por lo tanto, la existencia de una firma sintáctica que detectaba la generación previa probablemente también detecte la nueva. Sin embargo, si se utiliza un algoritmo de compresión sobre la nueva versión, aunque esta incorpore un simple cambio, resultará en un ejecutable radicalmente diferente. De esta manera, la firma sintáctica que detectaba la versión anterior no es útil para variantes futuras.

Ofuscación por virtualización

La ofuscación por virtualización es un mecanismo de ofuscación que implementa en el mismo código del *malware* un entorno de ejecución junto a un intérprete, el cual es capaz de ejecutar programas escritos en un lenguaje específico en forma de *bytecodes*. El intérprete –altamente ofuscado– acepta un lenguaje que se elige aleatoriamente en cada infección. Dicho de otro modo, podemos considerar que el software malicioso que incluye esta estrategia implementa un procesador virtual que acepta un repertorio de instrucciones particular.

El modo de operar de este tipo de malware se basa en seleccionar aleatoriamente, previa una nueva infección, un lenguaje aceptado por el intérprete, y recodificar el cuerpo del malware sobre la base de ese nuevo lenguaje. Evidentemente, el intérprete que contendrá la nueva variante se genera de acuerdo al nuevo lenguaje elegido. De esta manera, tanto el intérprete como el conjunto de *bytecodes* se modificarán sintácticamente, mientras que se preserva el comportamiento del código malicioso.

4.2.2. Técnicas de ocultación y autoprotección

Las técnicas de ocultación y autoprotección son mecanismos que implementa el *malware* con el objetivo de pasar desapercibido tanto a administradores como a software de detección, o bien para protegerse y dificultar su erradicación en un sistema infectado. Comúnmente, al conjunto de técnicas que permiten conseguir esto se las denomina mecanismos *rootkit*. Para conseguir ocultarse o protegerse, el código malicioso modifica partes del sistema operativo.

4.2.3. Mecanismos antidebugging

Los mecanismos *antidebugging* son un conjunto de estrategias que el *malware* puede incorporar en su propio código, cuya misión es la de dificultar cualquier proceso de ingeniería inversa que se intente aplicar sobre él. Su objetivo principal es el de

detectar si un *debugger* está supervisando la ejecución del propio código malicioso. Si este es el caso, el mismo *malware* modificará su comportamiento.

En este sentido, puede adoptar varias posturas:

- ejecutar código complejo sin ninguna finalidad para desalentar al analista,
- exhibir un comportamiento legítimo en vez de malicioso, o
- finalizar su ejecución.

Las técnicas más recientes incluso emplean algún método para detectar si la ejecución se está realizando en una máquina virtual, dado que su uso es muy habitual en los procesos de análisis de software malintencionado. Es preciso destacar que la mayoría de los mecanismos empleados son altamente dependientes de los sistemas operativos y de las arquitecturas de los sistemas.

4.3. Detección de intrusiones

Otro tipo de intrusiones son las que acceden a los sistemas mediante las vulnerabilidades de redes, ya descritas anteriormente. Aquí podemos utilizar los sistemas de detección de intrusos, también llamados IDS.

Un IDs es un sistema, normalmente compuesto de un software especializado y un hardware dedicado únicamente a este menester, que controla en todo momento el contenido de la red interna, ya sea hacia fuera de la organización o hacia adentro.

El IDs busca anomalías en las comunicaciones, descargas de ficheros ejecutables, filtración de datos internos hacia Internet, etc. Analiza el tráfico de la red interna comparando con firmas ya preestablecidas de ataques conocidos, escaneo de puertos sobre los equipos, paquetes de TCP/IP malformados que puedan crear problemas en la red, etc. Aunque además analiza también el contenido de ese tráfico.

Los IDs pueden trabajar por firmas o por heurística, en el primer caso, disponen de una base de datos con las firmas de los ataques ya conocidos, lo que permiten al IDs diferenciar entre el tráfico normal de la red y el tráfico malicioso.

Se puede compaginar la detección del IDs con un firewall, lo que crea un conjunto de actuación muy potente. Por ejemplo, el IDs puede detectar el escaneo masivo de puertos legítimos por parte de direcciones de internet fraudulentas y, a partir del aviso del IDs, el firewall puede cerrar todo acceso a la red desde esas direcciones. El uso compaginado entre el sistema IDs y el firewall es muy extendido y útil.

En sistemas muy críticos se acostumbra a instalar un IDs externo, entre internet y el firewall de entrada y otro después del firewall, interno. Esto hace que en el firewall exterior se detecten muchos ataques, que el firewall detectará y bloqueará, por lo que tendremos muchos falsos positivos en el IDs que serán detectados después. En cambio, en el IDs interno solo tendremos las comunicaciones que hayan pasado a través del firewall, por lo que las detecciones de intrusiones en la mayoría de los casos serán falsos positivos, es decir, detectará comunicaciones legítimas en algunos casos como ataques. Con esta configuración se podrá ver si el firewall está bien configurado o no, o si la detección y bloqueo de los ataques exteriores se realiza correctamente mediante el paso de reglas desde el IDs al firewall. En los casos más normales, tanto el IDs como el firewall estarán en la misma máquina, lo que facilitará la comunicación entre ellos de las reglas a aplicar.

Un ejemplo de IDs *open source* es SNORT, ampliamente utilizado como herramienta básica por las empresas y por los desarrolladores de soluciones de software de IDs, que basan muchos de sus productos en subversiones adaptadas de este programa. Aunque también existen algunos otros no tan usados pero muy conocidos también, como puede ser el software *Suricata* que también es un IDs muy potente. Ambos poseen una base de datos que la comunidad afín a esta iniciática va ampliando con nuevas firmas de ataques.

4.4. Amenazar actuales (cryptolockers)

Desde hace ya algún tiempo los ataques han cambiado de ser un acto para demostrar la valía de las personas que lo hacían a, directamente, una fuente de dinero fácil. Se calcula que actualmente la ciberdelincuencia en general ya mueve más dinero que el narcotráfico a nivel mundial. Lo que hace que todo lo relativo a la seguridad informática se haya de tener muy en cuenta.

Los sistemas de información de todas las entidades, ya sean pequeñas o grandes, públicas o privadas, han dejado de ser una simple herramienta para realizar el trabajo, ahora ya son objeto del negocio, son parte de él y sin estos sistemas el negocio no puede seguir adelante.

Por ejemplo, para una empresa que vende sus productos propios por internet, un ataque de denegación de servicio a sus servidores públicos donde tiene la tienda virtual, hará que el negocio caiga en declive. Ya no solo perderá las ventas de productos durante el ataque en el que la web ha estado inaccesible, sino que va más allá, la reputación de la empresa se verá entredicha.

Lo mismo puede pasar para un organismo público, en el que se guardan infinidad de documentos y datos privados de los ciudadanos de esa región o país. Si recibiera un ataque en el que hubieran vulnerado la seguridad y extraído toda la información personal allí custodiada, quedaría muy gravemente afectada la reputación de la entidad por parte de los ciudadanos, que verían como sus datos se pueden hacer públicos por otros medios.

En este sentido, las amenazas o ataques actuales, van mucho más allá e intentan sacar un beneficio económico, por lo tanto la cantidad de virus que surgen actualmente ha disminuido considerablemente. Recordemos que los virus cambian el comportamiento de los sistemas informáticos, pero no van más allá.

Los troyanos por el contrario, sí han visto un aumento considerable de las nuevas muestras que se programan cada día. Cada semana aparecen miles de ellos, en su mayoría son troyanos bancarios o los que secuestran el sistema y piden dinero a cambio de descifrar los documentos que tenemos almacenado en ellos

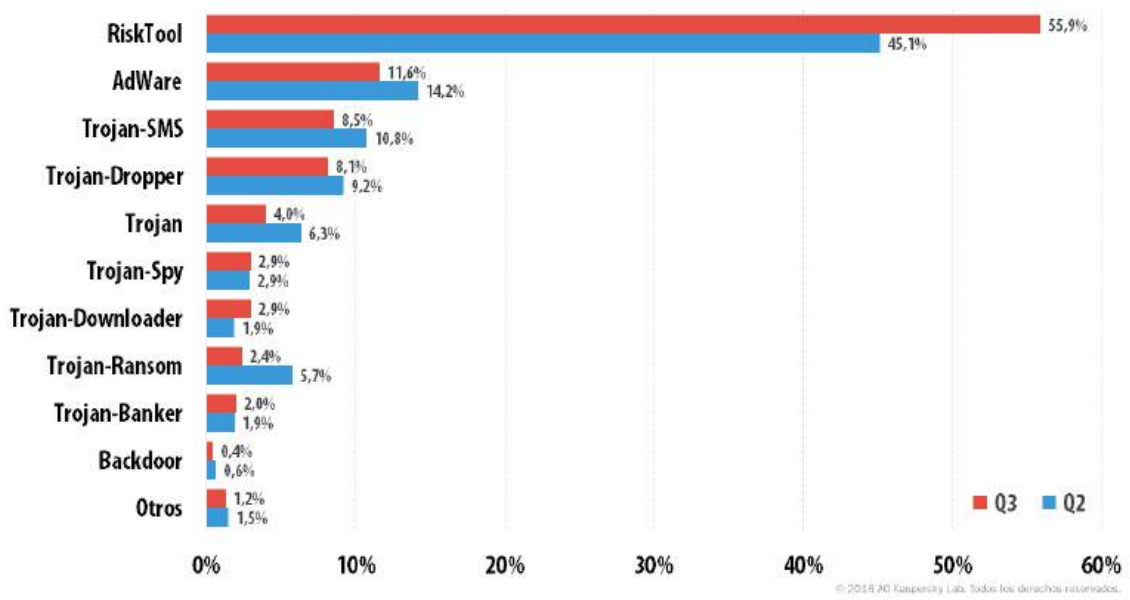


Fig. 13 Distribución de troyanos en Q2 y Q3 de 2016

La figura 13 muestra la distribución de los nuevos troyanos detectados por la empresa Kaspersky durante el segundo y tercer cuatrimestre del año 2016. Como podemos ver, a parte de las detecciones de las herramientas de riesgo, que normalmente son legales pero las detecta como un potencial peligro para el sistema, y los AdWare típicos de los navegadores, en los que se instalan por defecto si no se navega con sentido común, el resto son ya troyanos.

Mirando los resultados de esta empresa de seguridad, se puede observar como ya para este año 2016 la cantidad de nuevos troyanos o malware para ordenadores de sobremesa o servidores se ha reducido considerablemente. Los nuevos programas maliciosos están ahora atacando a los dispositivos móviles, sobretodo a los celulares Android, y también a los nuevos dispositivos del Internet of Things que se están popularizando durante estos últimos meses.

Existen ya troyanos para los celulares que bloquean y piden una recompensa por liberar el terminal. Ya han saltado de los ordenadores de sobremesa a los dispositivos que casi todas las personas tenemos en el bolsillo y que no deja de ser un sistema informático en el que guardamos toda nuestra vida, tanto la real como la virtual.

4.4.1. Desbloqueo de terminales.

Existen ya diversas herramientas libres para poder descifrar los archivos secuestrados de los dispositivos, empresas como Kaspersky ya dispone de una muy potente para poder revertir esos cifrados. Aunque no en todos los casos podrán servir, ya que en el momento en que una clave de cifrado se publica para poder descifrar, los ciberdelincuentes emplean nuevos sistemas de cifrado para seguir con su *negocio.*

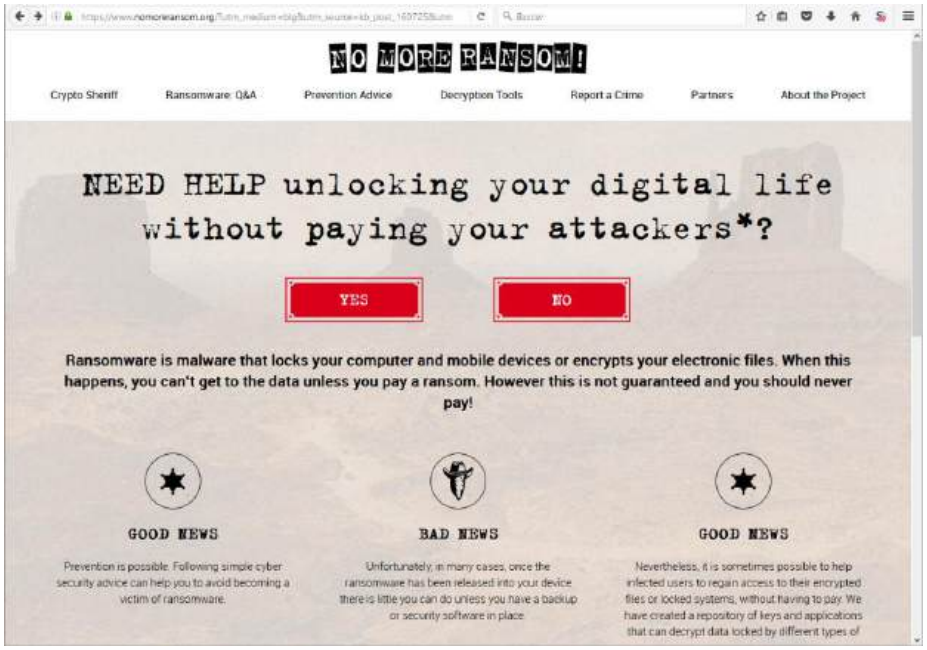


Fig. 14 Web de descifrado de ficheros

En esta página web (figura 14), producto de la unión de la empresa Kaspersky, la policía holandesa, Europol y Intel Security a partir de dos ficheros cifrados por un ransomware es capaz de detectar el tipo de malware utilizado y en el caso de tener sistema de descifrado se describe como utilizarlo.

5. Buenas prácticas en seguridad.

5.1. Copias de seguridad

Hay un principio básico que, como administradores o jefes de seguridad, debemos tener siempre muy presente, y es que los discos fallan. Hoy en día, es difícil encontrar un administrador de sistemas que no se haya visto involucrado en la pérdida de un disco. Para ello, solo hay una solución posible: hacer una copia de seguridad.

La mayoría de veces en las que hay una pérdida de información guardada en un disco se debe al hecho de que no se ha planificado nunca que estos datos se tenían que tener guardados, copiados en otro sistema para recuperarlos luego. La decisión de si se tiene que hacer una copia de seguridad o no es bastante importante para no tomársela a la ligera. Por lo tanto, en cierto modo, cuando planificamos estamos decidiendo de qué datos se hará la copia de seguridad y de cuáles no: estamos asumiendo qué datos estamos dispuestos a perder, ya que no los tendremos en caso de pérdida de información.

Esto no solo pasa a escala empresarial; en un estudio llevado a cabo en los Estados Unidos en mayo del 2012 y encargado por la empresa de discos duros Seagate, el 54% de los encuestados reconocieron haber perdido datos, ya sean documentos o fotografías digitales, que ya no pueden recuperar. Pero solo el 11% de las personas a las que se les planteó la encuesta reconocen tener un plan de copias de seguridad de los datos en general en su casa.

En la planificación de los datos de los que se hará una copia de seguridad tenemos que prever no solo los datos de los usuarios, sino también los archivos de configuración, las bases de datos, las bibliotecas importantes y toda la información que sea útil para la empresa.

Cuando sepamos cuáles son los datos que queremos guardar en una copia de seguridad, tenemos que decidir cuándo se tiene que hacer esta copia. Hay datos (los de usuarios, por ejemplo) que cada día tienen cambios; hay otros que cambian mucho más despacio y, finalmente, hay bibliotecas y archivos de configuración que puede ser que no cambien durante toda la vida útil de la máquina, de forma que tenemos que determinar la frecuencia de copia de seguridad de cada uno de estos datos. Cuando tengamos toda esa información, podemos empezar a planificar las copias de seguridad. Para hacerlo, primero debemos saber los diferentes niveles de copias de seguridad que hay.

Nivel 0. Este nivel se conoce como copia de seguridad total. Hace copia de todos los datos marcados para guardar, lo que hay dentro de las particiones, o de los discos, donde se hace la copia de seguridad. Este tipo de copia suele tener, dependiendo de la cantidad de datos que haya, un coste temporal muy alto. Es decir, puede tardar bastantes días en hacerse.

Nivel 1-9. La mayoría de las herramientas comerciales designan estos niveles con un único nombre: incremental. Estos niveles consisten en guardar solo los datos que se han modificado desde la última copia de seguridad total o de nivel inferior (por ejemplo, un incremental de nivel 2 hace una copia de los datos que se han modificado respecto a la última copia de nivel 0 y a la última de nivel 1). Esta manera de anidar la información la veremos mucho más clara con un ejemplo:

> Imaginemos que el sábado por la noche se hace una copia de seguridad total de todas las carpetas y archivos del servidor y el lunes, siguiente día laborable, se hace por la noche una copia incremental de nivel 1, es decir, se copia todo lo que se ha modificado durante el día. ¿Qué se puede hacer el martes por la noche? Si hacemos un incremental de nivel 1, copiamos todos los datos que se han modificado el martes y el lunes, con independencia de que estos últimos datos se hayan modificado el martes o no, es decir, se vuelven a copiar todos los datos que se han modificado desde el sábado. Si por el contrario se hace un incremental de nivel 2, solo se hace una copia de los datos modificados el martes y ocupan menos espacio en el disco duro, ya que los datos modificados el lunes ya están guardados el día anterior, pero en el caso de tener que recuperar todo el sistema de archivos, se tendrá que pasar por todas las copias diarias, ya que no existirá una única donde esté todo lo modificado desde el sábado. El proceso de recuperación de los datos es más lento, sobre todo si no se sabe de qué día puede ser la última modificación del archivo que se quiere recuperar.

En la mayoría de las herramientas comerciales de copia de seguridad encontramos el término diferencial, con el que se designa un tipo de copia de seguridad incremental, de manera genérica, que hace copias solo de los datos que han cambiado desde el último diferencial. Se trata de un incremental sin tener en cuenta los niveles.

En este tipo de herramientas comerciales, solo se designan tres tipos de copias: total, incremental y diferencial. La recomendación es una total a la semana, una diferencial todas las noches –en la que se reflejarán los cambios que se han producido a lo largo de aquel día– y una incremental solo en los casos en los que nos interesen los cambios que ha habido desde la última total.

Otro factor que debemos tener en cuenta para planificar la copia de seguridad es el tiempo que se tarda en realizar estas copias. En las totales se hace una copia de todos los datos, tanto si se han modificado la última semana como si no. Por lo tanto, si el disco es muy grande podemos tardar muchas horas antes de que se acabe. Por ejemplo, una copia del disco de usuarios de 500 GB puede tardar horas. Hagamos unos pequeños cálculos:

> Si tenemos un disco de datos con todo aquello de lo que se tiene que hacer copias de seguridad, por ejemplo los discos de usuarios, la contabilidad, la página web, los logs y configuraciones del sistema con unos 500 GB de información, y disponemos de un disco duro SATA2, que tienen una transferencia de datos de 300MB/s, se tardará 27 minutos en leer estos datos y otros 27 minutos en guardarlos en otro disco duro de las mismas características. En el supuesto de que se guardaran en una cinta magnética DLT(Digital Line Tape) a 60MB/s tardaría 2,3 horas en grabarse e idéntico tamaño para los dispositivos USB 2.0. En cambio, si la copia se ejecuta sobre otro dispositivo que está en la red y, por lo tanto, los datos se tienen que transmitir por una red Ethernet a 100Mb/s tardaríamos once horas en transmitirlos, lo que colapsaría la red si no se hace en horas no laborales. En el caso de tener una red de comunicaciones que funcione a gigabit, estas once horas se convierten en una hora, lo que hace que la inversión valga la pena para aligerar la red.

Por lo tanto, es aconsejable hacer estos cálculos antes de configurar cuándo se hacen las copias y cómo se hacen, ya que podemos saturar el servidor o la red en el momento de hacer las copias. Si se ve que se tienen que hacer las copias sobre otro servidor, se debe tener en cuenta que por la red los datos tienen que ir cifrados si salen de nuestra red, y esto comporta un tiempo añadido de cómputo y más tiempo en la transmisión por Internet, ya que no es lo mismo que hemos visto hasta ahora.

Una vez tenemos todos estos datos es cuando hemos de planificar las copias de seguridad. A continuación, mostramos algunas de las planificaciones más usuales que se hacen.

- **Total cada día**. Es una planificación que solo hacemos si el volumen de información del que queremos hacer la copia es bastante pequeño para que se haga por la noche. La recuperación de los datos será muy rápida.

- **Total semanal e incremental (nivel 1) diario**. La principal ventaja de este método es que solo se necesitan dos volúmenes para recuperar toda la información. Esto se debe al hecho de que todos los días se hace una copia de seguridad de las modificaciones que se han producido a lo largo de la semana. Este tipo de política es la más recomendable si usamos las utilidades simples de copia de seguridad (como `cpio`, `tar`, `dump`, `amanda` o `rdiff-backup` en GNU/Linux, y Windows Server Backup en Windows). Sin embargo, el inconveniente es que hacemos una copia de seguridad de datos innecesarios. Imaginemos que un archivo se modifica solo el lunes: a lo largo de toda la semana haremos copias de seguridad de manera innecesaria de ese archivo.

- **Total semanal, diferencial diario**. La principal ventaja de este método es que las copias de seguridad diarias son muy pequeñas y, por lo tanto, más rápidas. El inconveniente principal es que si queremos recuperar algo el viernes necesitamos seis volúmenes. Si usamos herramientas comerciales, esta desventaja se minimiza, ya que la mayoría de estas herramientas incorporan un sistema de gestión de los volúmenes.

- Hay también otras filosofías de copias de seguridad incrementales que trabajan con las progresiones matemáticas (sobre todo la llamada torre de Hanoi). Mediante estas progresiones se hacen todos los días copias de seguridad incrementales de diferentes niveles (por ejemplo 0, 3, 2, 5, 4, 7, 8, 9 y 1); con esto, conseguimos optimizar el número de cintas en uso, optimizar el tiempo de la copia de seguridad e incluso llegar a hacer una única copia total al mes.

Las políticas de copias de seguridad no son solo las que hemos comentado. Se puede diseñar una que se acople más a las necesidades específicas de cada caso, según el volumen de información y de la frecuencia de cambios que tengamos en la empresa o organización. Ahora bien, hay que tener presente que no habrá nunca suficientes copias de seguridad para satisfacer a los usuarios. Por ejemplo, si falla el sistema un miércoles por la tarde, todos los datos modificados a lo largo de la mañana no están reflejados en ninguna copia de seguridad. Una posible solución a este problema es el uso de discos redundantes (RAID). Si el fallo se ha producido en el ordenador del usuario, allí seguramente no tendremos

los discos redundados; por lo tanto, es importante ofrecer a los usuarios unidades de trabajo remotas (unidades de red), situadas en el servidor, donde puedan guardar los documentos.

Dentro de las políticas de copias de seguridad hay otro factor que se debe tener en cuenta: ¿durante cuánto tiempo estamos dispuestos a guardar la copia de seguridad? Si solo utilizamos una única cinta (CD o dispositivo físico donde se hacen las copias) para cada día, quiere decir que si un usuario no se da cuenta de que ha perdido algo antes de una semana ya no la podremos recuperar. Si guardamos las cintas (o CD o DVD) durante más tiempo, necesitaremos muchas más. Por ejemplo, necesitaremos un juego de siete cintas para cada semana de más que decidamos guardar.

Otro factor que se debe tener en cuenta es el etiquetado de los dispositivos de copia. Imaginemos que queremos guardar las copias de seguridad durante un mes. Esto hace que tengamos cuatro juegos completos de cintas, uno para cada semana. Debemos tener muy claro qué cinta se tiene que poner cada día de la semana y no confundir cintas del mismo día de la semana pero de semanas diferentes. Por lo tanto, el etiquetado de los soportes físicos es muy importante para no perder información (en caso de poner una cinta que no toca) o para recuperar rápidamente la información que necesitemos.

Una vez tengamos definida la política de copias de seguridad y hayamos hecho la primera, es hora de verificar que la copia de seguridad funciona correctamente. No sirve de nada hacer copias de seguridad si no hemos comprobado que podemos recuperar la información. Para hacerlo, una vez hecha la primera copia de seguridad intentaremos recuperar información. Para no interferir en el funcionamiento del servidor utilizamos la funcionalidad que nos ofrecen las herramientas de copia de seguridad que nos permiten recuperar datos en un directorio diferente del que se hizo la copia. Por ejemplo, los datos de `/home/user` se pueden recuperar en `/tmp/home/user` en el caso de GNU/Linux. De este modo, se comprueba que la copia de seguridad funciona sin modificar los datos que hay en el directorio raíz.

Tras comprobar que las copias de seguridad funcionan correctamente, es decir, que se pueden recuperar tanto archivos individuales como los discos enteros, ya podemos empezar a hacer las copias según la política que hemos definido.

5.2. Decálogos de Seguridad

Des de los Estudios de Informática, Multimedia y Telecomunicación de la (UOC), se ha elaborado cuatro decálogos sobre seguridad destinados al grueso de los usuarios digitales debido a que cada vez hay más ciudadanos preocupados por la falta de seguridad de sus datos digitales, y el debate sobre la vulnerabilidad y la falta de privacidad que surge del uso de aparatos tecnológicos y de las redes sociales cada vez está más presente.

5.2.1. Seguridad en general

1. La seguridad informática 100 % no existe: sin convertirse en un paranoico, hay que tomar en consideración el nivel de seguridad que esperamos de nuestros datos y de las máquinas que acceden a ellos.

2. No debemos asumir nunca que los datos que damos (p. ej. redes sociales) están circunscritos a un círculo conocido de personas: es muy posible que el comportamiento normal de la gente haga que se divulgue fácilmente en otros círculos.

3. Hay que tener en cuenta que la información que ponemos en las redes sociales tiene una validez espacial (el lugar donde aparece) y temporal (periodo de validez), que se puede alargar mucho más de lo que era esperable, tanto en sitios web donde estará presente como en duración temporal. Aquella información nuestra puesta en un lugar en un momento dado, puede no hacernos la misma gracia pasados los años, o según los espectadores que la vean.

4. Debemos tener una política de contraseñas, que permita sin dificultad disponer de contraseñas adecuadas para los diferentes lugares y motivos, y revisarlas de manera periódica. Es recomendable que tengan una cierta longitud (mayor que 8 caracteres) y que tengan información alfanumérica (dígitos y letras, y caracteres especiales). No se tienen que basar nunca únicamente en información personal (DNI, nombres de amigos/conocidos/familiares/mascotas, etc.), o patrones muy conocidos (12345). O cambiar las vocales por números en palabras conocidas.

5. Si somos usuarios de comercio electrónico, tenemos que basarnos en el prestigio social del vendedor, y en experiencias previas de sus clientes. Los sitios web tienen que ser seguros (usar https://) y a la vez pasarelas de pago electrónico (ya sean tradicionales como Visa/Mastercard, Paypal, o equivalentes). Y a su vez, hay que hacer un seguimiento de nuestra cuenta corriente, para comprobar los cargos de las tarjetas, o incluso avisar al banco que lo compruebe o nos avise ante determinados cargos.

6. En el caso del correo electrónico, no debemos abrir correos que claramente no iban destinados a nosotros, especialmente si incorporan adjuntos, o que aunque van dirigidos directamente a nosotros, nos hacen ir a páginas para corregir nuestros datos personales o de servicios diversos (especialmente los bancarios o administrativos).

7. En el caso de los teléfonos móviles, en especial los teléfonos inteligentes, tenemos que tener en cuenta que tenemos nuestra vida entera (comunicaciones sociales, contraseñas, acceso bancario o a servicios) a disposición de quien tenga acceso a nuestro teléfono. Hay que proteger bien, por palabras de paso o patrones, el acceso indebido a la información. Y cuando tengamos que estar por un periodo largo sin el aparato (p. ej. préstamo a terceros, o reparación en servicio técnico), es adecuado restaurarlo a los valores de fábrica (*reset* de software y datos). Igualmente, hagamos periódicamente copias de seguridad de nuestro dispositivo, para minimizar la pérdida de datos.

8. En el acceso a redes, especialmente de líneas Wi-Fi abiertas o compartidas, hay que ser conscientes de que estamos en un medio abierto, visible a todo el mundo, y por lo tanto hay que minimizar o evitar todo uso de servicios que comporten identificación personal o acceso a información personal. Sin disponer de mecanismos de cifrado, o uso de VPN (redes virtuales), hay que evitar el uso en estas redes abiertas, salvo que sea una simple navegación sin acceso a datos personales.

9. Hay que proteger las redes de casa, los *routers* ADSL+Wi-Fi modernos soportan diferentes niveles de seguridad: hay que cambiar las contraseñas por defecto del aparato y de sus servicios, y elevar el nivel de seguridad Wi-Fi (por ejemplo al protocolo WPA2). En caso de incidencias o bajadas de rendimiento inexplicables, hay que tener un control de los dispositivos conectados a nuestra red para detectar intrusiones. Hay que cambiar contraseñas y niveles de seguridad en estos casos. Especialmente no tenemos que olvidar aparatos más sensibles que se han incorporado a internet últimamente: TV, videoconsolas, aparatos de *streaming* de vídeo/audio, videocámaras…), los cuales pueden tener problemas de seguridad: tenemos que consultar a los fabricantes para las actualizaciones necesarias.

10. Hay que actualizar el software básico de nuestra máquina/móvil/tableta, disponer de las últimas versiones de las actualizaciones del sistema operativo y paquetes ofimáticos, y de software de uso general: navegadores y conectores (*plugins*), lectores PDF (como Acrobat), Java. En general, todos los softwares que sean de un uso habitual, aunque sea indirecto y no seamos tan conscientes de ello (por ejemplo PDF, Java, *flashplayer*, etc.).

5.2.2. Comercio electrónico

1. Utilizar solo el ordenador personal. Es preferible no hacer compras por internet poniendo datos personales o de pago en ordenadores ajenos, que el usuario no controle, como puede ser en el trabajo o en cibercafés.

2. No autoguardar la contraseña para acceder a los datos bancarios. A menudo hay que registrarse para hacer uso de una tienda en línea. Es mejor forzar para que siempre que se quiera acceder a estos datos, haya que escribir forzosamente de nuevo la contraseña.

3. Hay que comprobar y contrastar las referencias del comercio. Siempre es posible usar un buscador en internet para localizar referencias o impresiones de otros compradores y evaluar la reputación. No hay que quedarse con la primera que se encuentra: conviene contrastar varias.

4. Siempre se tiene que buscar información adicional de contacto sobre la tienda en línea. Por ejemplo, una dirección física o un teléfono.

5. Es mejor usar tarjeta de crédito o un sistema de pago por internet (*Paypal*, *Wallet*). Estos sistemas prevén la posibilidad de pedir un retorno del cargo. Con transferencias o tarjetas de débito, será muy difícil recuperar el dinero en caso de que el producto no sea lo que se esperaba. Otra opción es hacer un pedido contra reembolso.

6. No enviar ningún dato sensible si la conexión no es segura. Para garantizar que los datos que se envían con el navegador son seguros, habrá que ver si en la barra del navegador aparece un candado, o si la dirección web empieza por https en vez de http.

7. No evaluéis la fiabilidad de un comercio por una fachada bonita. Es muy fácil replicar la fachada de una tienda en línea con la intención de suplantarla. Independientemente de cómo se ha llegado allí (seguramente a partir de un buscador), hay que asegurarse de que la dirección del navegador que se usa es la correcta para la tienda.

8. No hay que fiarse solo de la imagen del producto. Hay que leer muy atentamente su descripción. Una estafa muy arraigada en subastas en internet es que la imagen del producto no tenga nada que ver con lo que realmente se ofrece (que queda, eso sí, perfectamente explicado en texto muy pequeño para evitar reclamaciones). Por ejemplo, mostrar en la imagen la caja de una consola, cuando en la descripción dicen que realmente lo que te venden es solo la caja.

9. Hay que comprobar los cargos periódicos. Muchas entidades bancarias suelen tener un sistema de alertas al móvil en caso de pagos. De este modo es posible controlar si se hacen cargos. En cualquier caso, también vale la pena ir controlando los extractos bancarios para asegurarse de que los cargos que se han hecho son los correctos.

10. En el peor caso, nada es fútil, siempre puede servir como mal ejemplo. Si se tiene una mala experiencia, se puede compartir con otras personas para que lo tengan presente de ahora en adelante.

11. No hacer caso de cualquier correo de publicidad o de supuestas compras que no se han hecho. El correo electrónico todavía es la vía predilecta para hacer que alguien entre a una tienda falsa, mediante enlaces falsos.

5.2.3. Contraseñas

1. No utilizar palabras del diccionario ni relacionadas con los datos personales (nombre, fecha de nacimiento, calle donde se vive, etc.), muy fáciles de romper.

2. No utilizar la misma contraseña en diferentes sitios web. Si alguien roba las contraseñas de un servidor puede entrar inmediatamente a las otras cuentas de un usuario.

3. No guardar ni compartir las contraseñas con herramientas que no están pensadas para este propósito, como cuentas de correo electrónico o gestores documentales. El proveedor de servicio podría tener acceso a las contraseñas.

4. No guardar contraseñas en la memoria cache del navegador (principalmente si el ordenador no es del usuario).

5. Utilizar contraseñas con un mínimo de ocho caracteres y que contengan algún tipo de símbolo diferente además de las tradicionales letras minúsculas: letras

mayúsculas, números o signos de puntuación. Para sitios web que gestionan datos sensibles, la longitud mínima recomendada es doce.

6. Hay que crearse una política para generar contraseñas y seguir siempre los mismos pasos. De este modo es más fácil recordar las contraseñas.

7. Una de las maneras de crear contraseñas es:

    - Crear una secuencia de palabras y/o letras que parezca aleatoria. Para crearla, hay que tomar una cita, frase hecha, estrofa, etc. que guste al usuario y recordarla fácilmente. Ejemplo: primer verso del «Autorretrato» de Antonio Machado («Mi infancia son recuerdos de un patio de Sevilla»). Cogemos la primera letra de cada palabra y añadimos el autor en mayúsculas para crear más complejidad: «misrdupdsAM».

    - Añadamos signos de puntuación y/o números para hacerlo más seguro. Ejemplo: Utilizar el número 1 para sustituir un en la frase original: «misrd1pdsAM».

8. Para evitar tener que gestionar decenas de contraseñas diferentes, usando los pasos descritos en el punto anterior podemos crear un conjunto de contraseñas con una misma frase base, añadiendo prefijos y/o sufijos. Generalmente se intentará que estos prefijos y sufijos se relacionen de alguna manera con el web al cual se quiere acceder para que sean más fáciles de recordar. Ejemplo: Para el web de Facebook, podemos añadir un prefijo cl (traducción literal «cara libro») de forma que quede «clmisrd1pdsAM».

9. Si se tiene problemas para recordar todas las contraseñas, se puede:

    - Escribir en una libreta a la que solo el usuario tenga acceso aquellas ideas o referencias que pueden ayudar a recordar la contraseña. Cuanto más personales y más difíciles de deducir por terceras personas sean, mejor.

    - Escribir las ideas o referencias para recordar contraseñas en una hoja de cálculo o documento de texto en el ordenador. Si es así, este documento tendrá que ir protegido con contraseña (tanto Microsoft Office como el LibreOffice permiten hacerlo). No nombrar este fichero «contraseñas».

    - Utilizar un gestor de contraseñas. Los gestores pueden ser útiles y muy portables (algunos están en la nube) pero el usuario se tiene que asegurar de que el que se selecciona tiene buenas referencias.

10. Algunos sitios web ofrecen sistemas de recuperación de contraseñas por medio de preguntas personales para restablecer tu cuenta. Hay que tener en cuenta que si son preguntas muy generalistas las puede intentar responder un atacante. Por lo tanto, hay que iniciar el sistema con respuestas no triviales (por ejemplo, si se pide el nombre del padre del usuario hace falta no indicar el nombre correcto sino un apodo utilizado en la intimidad) o deshabilitar este sistema de recuperación de la cuenta.

5.2.4. Uso del WhatsApp

1. No dejar el móvil descuidado y sin vigilancia fuera de casa, especialmente en ámbitos con muchas personas, como el trabajo.

2. No conectar el teléfono inteligente a ordenadores ajenos, y especialmente en el caso de los *iPhones*, no los sincronicéis a un ordenador que no sea el personal.

3. Utilizar un sistema de desbloqueo del móvil que requiera algún tipo de autenticación, ya sea un PIN, un patrón, una huella dactilar, etc.

4. No poner datos que no se puedan considerar públicos como mensaje de estado.

5. No poner fotos comprometidas como imagen de perfil.

6. Actualizar la aplicación de WhatsApp siempre que exista una actualización disponible.

7. No utilizar versiones no oficiales de la aplicación, es decir, versiones que no se han descargado de la tienda oficial del teléfono inteligente (Google Play Store, AppStore, Windows Phone Store…) o de la página web oficial de WhatsApp.

8. Hay que tener presente que haciendo un uso normal de la aplicación ya se está dando bastante información a cualquier persona que se añada el número del usuario a su agenda, independientemente de si está en nuestra agenda o no. Concretamente puede saber si estamos utilizando el WhatsApp en ese momento, cuándo fue la última vez que se utilizó, nuestro estado y nuestra fotografía de perfil.

9. No os obsesionéis con el doble signo de verificación al mandar un mensaje, puesto que según si el móvil de destino es un Android o un iPhone quieren decir dos cosas diferentes. Es un problema recurrente que ha provocado muchas discusiones.