3

Cómo funciona el mundo del software

A estas alturas del libro ya hemos mencionado el término «mundo digital» unas cuantas veces. Este mundo no es otro que el formado por software de diferentes fabricantes que se ejecuta en servidores, en tu ordenador personal o en tu *smartphone*. Este software constituye el núcleo de lo que usamos a diario en el mundo digital y que conocemos como aplicaciones, plataformas o servicios online.

Al igual que cuando se fabrica un coche o un cuchillo de cocina, el software se construye a partir de materias primas y procesos avanzados de producción. En su caso, la materia prima son unos ficheros de texto conocidos como *código fuente* donde los programadores escriben instrucciones que se traducirán posteriormente a un lenguaje entendible por las máquinas. Así, si la calidad final de un coche depende enormemente de los materiales utilizados y de los profesionales que lo fabriquen, la calidad del software dependerá igualmente de la experiencia y maestría de los programadores que escriben el código fuente, así como de lo refinado que esté el proceso de desarrollo[24].

Debido a que llevamos relativamente poco construyendo aplicaciones, **los niveles de profesionalización de la industria del software no son comparables a los de oficios más clásicos** y con muchos más años de experiencia como la arquitectura o la aeronáutica. Los romanos ya construían puentes hace más de 2000 años. Por contra, la primera calculadora programable no se construyó hasta hace menos de 100 años[25]. Como llevamos relativamente poco tiempo desarrollando software, es obvio que todavía nos queda mucho por mejorar. A pesar de ello, nos las apañamos a diario para producir

aplicaciones increíbles y capaces de hacernos sentir que vivimos en el futuro. Pero debemos admitir que la fiabilidad y resiliencia de estos productos digitales está lejos de lo que nos gustaría.

«Se ha caído WhatsApp»

Piensa por un momento en la última vez que un servicio que utilizas estuviera no disponible durante un rato. Redes sociales, aplicaciones de mensajería o incluso nuestra banca online se caen de vez en cuando y dejan de funcionar sin previo aviso. El principal requisito de un puente o de un edificio es que no se caiga cuando lo estás usando y sin embargo los servicios online, no importa cómo de bien construidos o mantenidos, ocasionalmente apagan las luces cuando menos te lo esperas.

Estos apagones pasajeros se producen normalmente porque, a diferencia de un puente o un edificio, el software sufre modificaciones estructurales a lo largo de su vida útil[26]. Al igual que nuestro ordenador personal instala parches y se reinicia de vez en cuando, el software que cimienta los servicios online también recibe mantenimientos periódicos para arreglar goteras y añadir nuevas funcionalidades. Las tareas de aplicar parches a estas gigantescas construcciones hechas de diferentes trozos de software no siempre salen bien y a pesar de que los responsables hagan pruebas, pequeños cambios aparentemente inocuos y rutinarios suelen ser los causantes de grandes apagones inoportunos.

En ocasiones además otro tipo de cambios menos estructurales pueden también causar problemas. En el año 2012, un cambio rutinario de contraseña en la wifi de invitados de las oficinas de Google generó una serie de fallos en cadena que dejó fuera de servicio una aplicación interna que almacenaba contraseñas. Sin esta aplicación, algunos empleados no podían acceder a determinados servicios y no podían trabajar. El proceso de recuperación de la aplicación nunca se había probado y no funcionó a la primera. Esto obligó a sacar a gente de la cama en la otra punta del mundo e incluso fue necesario perforar con

un taladro la cerradura de una caja fuerte y recuperar de su interior una clave maestra necesaria para volver a dejar la aplicación operativa[27]. Este ejemplo ilustra a la perfección la imprevisibilidad del impacto de cambios aparentemente inocuos y la exposición a sufrir incidentes incluso por empresas con procesos de gestión maduros y bien probados. Ocasionalmente el caos encuentra un ángulo imprevisto por donde colarse incluso en las mejores casas.

La resiliencia y la seguridad del software son requisitos difíciles de cumplir de manera aislada pero mucho más de manera simultánea. Un software particularmente seguro puede hacer difíciles las tareas de recuperación en caso de incidencia, por ejemplo, obligando a los equipos de respuesta a obtener las autorizaciones pertinentes antes de poder actuar (o literalmente obligar a taladrar una caja fuerte), ralentizando su agilidad e incrementando la duración de los apagones. Por otro lado, tanto la resiliencia como la seguridad son requisitos particularmente invisibles ya que cuando están presentes nadie los nota. Y, sin embargo, para cualquier usuario resulta intolerable tanto una caída como una brecha de seguridad.

Quizá el método más fiable para entregar software estable y razonablemente seguro sea someterlo a la prueba del tiempo. Algunos sistemas de misión crítica, donde un usuario no puede permitirse un cuelgue repentino o un comportamiento errático, utilizan versiones de software que serían consideradas obsoletas para los estándares domésticos. Si visitamos una central nuclear o el puente de mando de un barco de guerra es probable que encontremos ordenadores con sistemas operativos varias versiones por detrás de la última que podemos encontrar en las tiendas. Esto se debe a que el software utilizado en esos entornos tan críticos debe pasar procesos muy estrictos de pruebas y certificación que necesitan tiempo hasta poder dar por buena una versión específica. A ojos de una persona enamorada de utilizar siempre lo último de lo último, parecerán sistemas antiguos y obsoletos, cuando la realidad es que su resiliencia, estabilidad y seguridad seguramente supere con creces algunos de los productos más modernos que utilizamos. El software de última generación en

ocasiones se libera bajo la urgencia de las necesidades de generar negocio y, por tanto, adolece de menor estabilidad y más probabilidades de tener vulnerabilidades pendientes aún de ser descubiertas.

Libertad o seguridad

Pero a pesar de que se invierta tiempo en probar cuidadosamente el software, es poco probable conseguir eliminar todas las vulnerabilidades. Hay una asimetría clara en el mundo del software: una aplicación puede declararse insegura tras observar una vulnerabilidad, pero, aunque no encontremos fallos, no podemos declarar con confianza que no existen. No es posible encontrar pruebas que certifiquen que una aplicación es segura[28]. Existen múltiples ejemplos de vulnerabilidades que han sobrevivido en la sombra durante años hasta ser descubiertas y aprovechadas[29] [30].

Sin embargo, los usuarios demandan que sus aplicaciones sean seguras y su tolerancia a fallos es muy baja. La publicación de una brecha de seguridad en un producto suele venir acompañada de indignación y de una sensación de que el responsable ha cometido una negligencia grave. Las empresas lo saben. En el capítulo 2 comentábamos la necesidad de muchas empresas de perpetuar modelos de negocio basados en que las personas continúen visitando y usando sus servicios cada vez con mayor frecuencia. Para estas empresas resulta primordial que los usuarios se sientan seguros compartiendo cada vez más datos, creando contenido e incrementando su dependencia con los servicios. Debido a esto, empresas como Apple, Google, Microsoft o Amazon invierten ingentes cantidades de dinero en disponer de los mejores equipos de ciberseguridad del mundo, pero también en campañas de marketing para presumir de que sus productos son más seguros que los de la competencia.

A nadie le gusta tener que lanzar un comunicado para avisar a sus clientes de una brecha de datos. Esta necesidad de transmitir confianza y evitar protagonizar titulares negativos ha obligado a las empresas a tomar medidas drásticas y a lanzar ecosistemas de software cada vez

más cerrados. El ejemplo más claro es Apple y su *App Store*. Mediante la implantación de un proceso de revisión de todo software antes de recibir permiso para poder ser instalado en un iPhone o iPad, Apple ha eliminado casi por completo cualquier posibilidad de que los usuarios pudieran infectarse con software malicioso[31]. El efecto colateral es que Apple impide también la instalación de software *non-grato* que podría entrar en competencia directa con su modelo de negocio (e.g. un App Store de otro fabricante) [32]. La mayoría de los usuarios de un iPhone no son conscientes de que han sacrificado parte de su libertad para conseguir una mejora sustancial en seguridad. En ciberseguridad nada sale gratis.

En el capítulo 8 hablaremos más sobre este compromiso y sobre otras características de los dispositivos basados en iOS (el sistema operativo de los iPhone e iPad) y de cómo otros ecosistemas han copiado parcial o totalmente este modelo en aras de mejorar la seguridad percibida por los usuarios.

La piedra anti-osos

Los fabricantes de software hacen grandes esfuerzos por conseguir que sus usuarios se sientan seguros utilizando sus productos. Esta sensación con frecuencia nace más de la influencia del marketing e imagen de marca y no tanto de evidencias o hechos concretos.

Al comienzo del episodio de Los Simpson titulado *Mucho Apu y Pocas Nueces* [33] un oso aparece en el barrio provocando el pánico entre los vecinos. Estos no tardan en demandar al alcalde soluciones urgentes y drásticas para que algo así no vuelva a suceder. El alcalde Quimby da respuesta a la demanda popular creando una desproporcionada patrulla anti-osos compuesta por vehículos de vigilancia e incluso aviones de combate. Esta patrulla es poco después elogiada por Homer por su eficacia utilizando como argumento el hecho de que no ha vuelto a ver un oso por el barrio. Lisa, de manera muy inteligente, le explica que la ausencia de osos no puede atribuirse a la presencia de la patrulla y que

ella misma podría atribuir el mérito de la desaparición de los osos a una simple piedra que sostiene en su mano.

Esta conversación entre Lisa y Homer sobre las causas por las que uno se encuentra seguro tiene su reflejo en algunas de las técnicas que utilizan las empresas para vender software. Además de las propias campañas sobre lo prioritario que es para ellas mantener seguros a sus usuarios, existe toda una industria de software específico dedicada a vender a los usuarios soluciones concretas para mejorar su seguridad. Ejemplos son los Antivirus, Firewalls personales, Redes Privadas Virtuales (VPNs), Limpiadores de Cookies...

Estas promesas en primer lugar buscan validación por argumento de autoridad. Es decir, su usuario objetivo no tendrá la capacidad o el conocimiento para verificar estas promesas por lo que debe fiarse de la empresa que está detrás y de su prestigio histórico. Un logo bien colocado puede hablar más sobre la calidad percibida de un producto que sobre su efectividad real. Esta práctica de explotar una imagen de marca para transmitir calidad es muy utilizada en todos los sectores. Pero, a diferencia de la industria textil o de la alimentaria, en el mundo del software es mucho más difícil para un usuario examinar la calidad real del producto que tiene entre manos.

Más allá del marketing podemos decir que detrás de cada producto en casi todos los casos existe un interés genuino por mejorar la postura de seguridad de sus usuarios. Exceptuando algunos casos concretos donde la ganancia en seguridad es marginal o donde los productos rozan la categoría de estafa [34], podemos decir que un antivirus es objetivamente útil para protegerse frente a algunas amenazas comunes. También podemos asumir que empresas como Google o Apple tienen un interés honesto en proteger a sus usuarios de algunas amenazas. Al menos de aquellas en las que no son directamente los causantes y que no resulten instrumentales para conseguir sus objetivos de negocio.

Por otro lado, es importante señalar los efectos secundarios del uso de servicios anunciados como seguros o de productos tales como antivirus, VPNs y demás software de seguridad. Un exceso de confianza en esta seguridad percibida puede hacer que nos relajemos y bajemos

la guardia. **No debemos echarnos a dormir pensando que un proveedor de servicios es invulnerable o que estamos protegidos por usar una VPN o un antivirus**. Descargar y ejecutar archivos con excesiva ligereza confiando en la cobertura de este tipo de software o dejarse llevar por las promesas de inmunidad que nos han vendido puede conducirnos a alguna que otra sorpresa desagradable. En el capítulo 10 hablaremos más sobre las limitaciones que tiene software específico como las VPNs y las implicaciones que tienen en términos de seguridad y privacidad. Por ahora, basta con recordar que las piedras anti-osos, desgraciadamente, no existen.

Arreglando goteras

La única estrategia sólida para mantener el software seguro y resistente es la del mantenimiento regular. Los fabricantes de software deben publicar con cierta regularidad actualizaciones que corrigen fallos, mejoran el rendimiento o eliminan vulnerabilidades. Sin actualizaciones el software se marchita poco a poco. Para una persona que busque una buena postura de seguridad (como la que está leyendo este libro) resulta de vital importancia valorar la calidad del mantenimiento de los productos y durante cuánto tiempo es esperable que se mantenga ese soporte.

La mayoría de las fabricantes ofrecen versiones con soporte a largo plazo. Algunas veces, también se publican paralelamente versiones del mismo software con funcionalidades más nuevas, pero con periodos de mantenimiento que terminan antes[35]. En ambos casos, debemos prestar atención al tiempo durante el cual el fabricante se compromete a seguir publicando dichos parches. Una vez este mantenimiento cese, no podemos confiar en que sea seguro continuar utilizando ese software.

Además, es relevante también considerar la cantidad de esfuerzo que supone para el usuario mantener su software actualizado. Existen varias formas de diseñar esta tarea que oscilan de la más rudimentaria hasta la más automatizada. Por lo general un usuario interesado en aplicar estas actualizaciones de manera disciplinada deberá favorecer

el uso de software que se actualice automáticamente en segundo plano, sin requerir acciones por su parte y con una experiencia de uso casi invisible. Este por ejemplo es el caso de la mayoría de los navegadores modernos o de aplicaciones descargadas en dispositivos móviles (Android o iOS). En estos casos, una vez instalado el software rara vez se requiere del usuario alguna acción adicional para la aplicación de parches periódicos.

En el otro extremo es conveniente huir del software que requiere que el usuario descargue parches de manera manual o que se limite a recordar al usuario que existe una actualización pendiente para ser instalada. En ocasiones será inevitable tener que pasar por el ocasional reinicio, pero debemos cuidarnos de la posibilidad de procrastinar indefinidamente o de poder ser engañados al tratar de descargar un parche de un sitio fraudulento.

En lo que respecta a servicios online el mantenimiento del software que los hace funcionar es obviamente responsabilidad del proveedor del servicio. El usuario final poco puede supervisar aquí y le será difícil identificar con facilidad qué servicios están pobremente mantenidos. Deberá por tanto limitarse a usar proveedores con aparente reputación[36] que previsiblemente puedan contar con los recursos necesarios para mantener su software actualizado.

Hay varios factores que juegan un papel relevante en la capacidad que tiene una empresa de servir software seguro y resiliente a sus clientes. Entre ellos, el tamaño de la empresa y su complejidad son probablemente los factores más relevantes. Lamentablemente, como veremos a continuación, algunas compañías deben lidiar con ambos factores en diferentes momentos de su vida.

Moverse rápido, romper cosas

Aunque desde hace algunos años todas las compañías afirman que la ciberseguridad de sus productos es una de sus prioridades, algunas tienen retos importantes para cumplir esta promesa. Algunos factores limitan sus oportunidades para realizar tareas rutinarias como parchear

software con la debida diligencia o probar con detenimiento y mimo los productos antes de ponerlos a disposición de sus clientes.

En primer lugar, la urgencia por dar beneficios o por empezar a vender un producto no juega a favor de la seguridad o resiliencia del software. Las *startups* o empresas cuyos inversores generan presiones por acortar los ciclos de desarrollo al máximo para crecer y generar ingresos rápidamente no son el mejor lugar para crear software seguro. Como hemos comentado antes, entregar software seguro requiere de un diseño cuidadoso, de una revisión por parte de muchos ojos y de unas pruebas pormenorizadas que tienden a dilatar los ciclos de desarrollo bastante. Esta filosofía va en contra de la mentalidad de muchas startups de iterar sus productos rápidamente hasta encontrar su encaje en el mercado (o *market fit*). Durante sus primeros 10 años de operación, Facebook hacía gala del eslogan *Muévete rápido y rompe cosas*[37]. No fue hasta 2014 cuando Mark Zuckerberg anunció su cambio por la frase *Muévete rápido, pero con una infraestructura estable*[38] ilustrando así el cambio de mentalidad necesario para continuar asegurando la fiabilidad de sus productos y no perder la confianza de sus usuarios.

Por otro lado, cuando este cambio de mentalidad sucede y las empresas reducen la velocidad con la que entregan nuevos productos, en algunos casos las dificultades para mantener su software seguro permanecen. Al principio de este capítulo hablábamos sobre cómo las técnicas para desarrollar software son todavía inmaduras y están en constante evolución. Esto hace que ciertos productos puedan quedarse obsoletos en menos de 10 años, creando auténticas «deudas» tecnológicas que son difíciles de sustituir una vez se ponen en manos de los clientes. Este tipo de software que todavía funciona, pero está obsoleto y es difícil de sustituir en algunos ámbitos se conoce como software *legacy*. Los proveedores de servicios más maduros necesitan continuar manteniendo este legacy, construyendo por encima y alrededor los nuevos desarrollos que les demanda el mercado. No es eficiente ni operativo comenzar de cero con cada nuevo

desarrollo por lo que el software antiguo y moderno están condenados a convivir.

Esta convivencia del software legacy con productos más modernos obliga a las compañías a realizar malabarismos que no favorecen la entrega de productos seguros y resilientes. Además, para el caso de empresas particularmente grandes las complejidades organizativas pueden dar lugar a la creación de software con diseños innecesariamente fragmentados o de entornos donde las responsabilidades pueden no estar suficientemente claras entre distintos equipos [39]. En estas compañías esta complejidad se suma a las presiones externas por parte de accionistas y entidades supervisoras para garantizar precisamente que los servicios críticos estén siempre disponibles[40]. El miedo a causar esos temidos apagones que comentábamos previamente genera dificultades adicionales para parchear y reiniciar servicios o aplicar mantenimientos con la debida diligencia.

Vulnerabilidades Zero Day

Hemos introducido la necesidad de que el software disponga de mantenimiento periódico para corregir vulnerabilidades. Sin embargo, existe un tipo de vulnerabilidades para las que este mantenimiento regular no sirve de nada y para las que no existe solución: son las llamadas vulnerabilidades *Zero Day*.

Un *Zero Day* es un tipo de vulnerabilidad desconocida por el fabricante del software y que puede estar siendo utilizada activamente por un atacante. Debido a esto, el fabricante tiene «cero días» de margen para poder publicar un parche antes de que los malos empiecen a utilizarla. Por el contrario, una vulnerabilidad normal típicamente es conocida por el fabricante antes de que los malos la descubran y aprovechen. En estos casos, la vulnerabilidad puede mantenerse en secreto hasta que se publique el parche correspondiente, dando más margen a los usuarios para protegerse y demorando su descubrimiento todo lo posible.

De un modo general podemos resumir las características principales de una vulnerabilidad Zero Day como:

1. **Peligrosidad**: Un Zero Day otorga a su conocedor la capacidad de atacar sin que sus víctimas puedan defenderse. Si bien hay Zero Days de diferente gravedad, normalmente el término se refiere a aquellos que proporcionan los privilegios necesarios para manipular el software a su antojo.

2. **Desconocimiento**: Al ser una vulnerabilidad totalmente desconocida no es posible arreglarla o defenderse de ella hasta que o bien se utiliza o se comparte con el fabricante para su resolución. Es difícil anticipar cuántos Zero Days se habrán descubierto para un software específico y si alguien con el interés suficiente tendrá alguno a su disposición sin usar todavía.

3. **Difícil respuesta**: Incluso una vez utilizada por primera vez [41] la vida útil de la vulnerabilidad hasta ser arreglada por el fabricante puede ser corta o larga dependiendo de muchos factores. El impacto que genere su uso o la notoriedad de la víctima pueden hacer el suficiente ruido como para atraer la atención de los medios, de investigadores y del propio fabricante. Por contra, un uso más discreto puede alargar su invisibilidad, potencialmente afectando a más víctimas a largo plazo. Por lo anterior, es frecuente que los fabricantes saquen parches lo más rápido posible una vez obtienen conocimiento de la vulnerabilidad. Sin embargo, debido a las dinámicas mencionadas en secciones previas, la aplicación del parche por parte de todas las potenciales víctimas podría demorarse días o semanas. En caso de que el parche requiera acciones manuales de los usuarios, la ventana de exposición puede ser todavía mayor.

Los Zero Days constituyen las armas más poderosas y terroríficas del mundo digital. Cuanto más popular es el software afectado más alto es su valor debido al volumen de potenciales

víctimas. Existen organizaciones en el mundo que se dedican tanto a mercadear con Zero Days[42], como a almacenarlos como si fueran cabezas nucleares esperando cumplir su función, sea esta la mera disuasión de sus enemigos o la completa devastación. En algunas de las aplicaciones más utilizadas del mundo las recompensas de fabricantes como Google por reportar Zero Days de los más graves pueden llegar al millón de dólares[43].

La persona célebre que esté leyendo el libro no tardará en preguntarse qué opciones tiene para protegerse frente a tan poderosa amenaza. La respuesta corta es: ninguna. Afortunadamente, el altísimo valor que tienen este tipo de armas digitales sumado al hecho de que tras usarlas por primera vez su valor disminuye drásticamente, reducen la probabilidad de que alguien «gaste» una de estas vulnerabilidades con nosotros. Sin embargo, a pesar de la dificultad para protegerse de un Zero Day no todo está perdido. En el capítulo 23 exploraremos algunas de las técnicas más drásticas para personas significativamente expuestas a ser víctimas de ataques así de avanzados.

Malware y Spyware

Una de las amenazas más frecuentes que afectan al software son los programas maliciosos o malware. Este tipo de software no siempre necesita de vulnerabilidades para colarse en un sistema, sino que aprovecha también descuidos o ignorancia por parte de los usuarios para conseguir que lo ejecuten e instalen. En ocasiones, incluso los usuarios otorgan permisos adicionales al malware cuando éste se lo solicita.

Tropezarse con malware es relativamente fácil. Basta con acudir a la carpeta de correo no deseado y observar (de lejos) los adjuntos que típicamente nos llegan en esos mensajes. Su aspecto es variado, pero siempre aparentemente inofensivo: un archivo que parece un documento en formato PDF, pero en realidad no lo es, un fichero comprimido ZIP que incluye dentro un ejecutable o un enlace a una web que nos solicita instalar algo adicional para visualizar el contenido.

Cada día se inventa un nuevo engaño para convencer a los usuarios de ejecutar el malware en sus dispositivos. En el capítulo 19 hablaremos sobre algunos de estos ataques y veremos consejos para evitar caer en estos engaños.

Dependiendo de lo dirigido que sea el malware, sus objetivos y su impacto variarán mucho. Es posible en los casos más comunes que el malware no sepa quién eres y simplemente esté buscando hacerse con el control de tu PC para poder utilizarlo de manera remota. Lo más frecuente es que tu dispositivo se acabe usando para cometer otras acciones delictivas (e.g. usar tu PC como trampolín para atacar a otros usuarios [44]) o simplemente bloquear tus archivos para pedirte un rescate monetario por ellos (lo que se conoce como *ransomware*).

Otro tipo de malware conocido como *spyware*, suele estar más dirigido a víctimas mejor seleccionadas. Su propósito es el de acceder a la totalidad de la actividad del usuario en un dispositivo concreto. En los últimos tiempos ha cobrado notoriedad este tipo de malware gracias a *Pegasus*, un tipo de spyware del que podrían haber sido víctimas varias personalidades de la política y autoridades de algunos gobiernos [45]. Las capacidades de un software de estas características una vez se instala y cuenta con los permisos necesarios son prácticamente ilimitadas. Desde rastrear la localización del usuario y registrar todo lo que teclea hasta compartir con los administradores de la herramienta cualquier información que salga o entre a tu teléfono: mensajes directos, emails y llamadas de voz. Este software puede incluso potencialmente activar el micrófono del terminal en cualquier momento para grabar las conversaciones que sucedan a su alrededor [46].

En el capítulo 21 hablaremos más sobre este tipo de software, de cómo los dispositivos móviles han aumentado sus oportunidades para espiar a usuarios célebres y en los capítulos 22 y 23 de algunas estrategias avanzadas para protegerse.

Un Superman en cada aplicación

Hasta ahora hemos hablado fundamentalmente de vulnerabilidades. Es decir, de fallos en la construcción del software que utilizamos que permiten a los malos manipularlo para su beneficio. También hemos hablado de software específicamente diseñado para hacer el mal. Sin embargo, no todo aquello que puede ir mal en términos de seguridad y resiliencia del software se debe a este tipo de fallos o al malware. **En muchos casos es el abuso del funcionamiento normal del software el causante de los incidentes.**

Casi todo el software utiliza de una u otra forma el concepto de cuenta de usuario. Es frecuente además que distintos usuarios puedan hacer uso del mismo producto simultáneamente o por turnos. Algunos de ellos inevitablemente necesitarán contar con privilegios especiales para poder configurar el software y potencialmente podrán afectar a otros usuarios con menos privilegios. Estos usuarios privilegiados, conocidos normalmente como *administradores* tienen la capacidad de configurar el software a su antojo y gobernar las cuentas y datos del resto de usuarios. A medida que el software crece en complejidad y funcionalidades, puede hacerse necesario establecer diferentes niveles de privilegios para distintos usuarios e incluso establecer una separación de funciones que evite abusos de poder. Así, por ejemplo, en un sistema corporativo para realizar pagos de alto importe puede hacerse necesario evitar que la misma persona que solicita una transferencia pueda también ejecutarla. Mediante la introducción de un segundo usuario que valide la solicitud se hace más complejo cometer fraude haciendo necesaria la colusión de al menos dos personas y garantizando la trazabilidad del proceso.

Desgraciadamente no importa lo bien montado que esté un sistema de privilegios y la exquisitez con la que se diseñe una segregación de funciones entre personas. Al final del día, siempre existirá al menos un usuario con privilegios máximos capaz de configurar los permisos y separación de funciones del resto de usuarios. No importa de qué tipo

de software se trate, siempre hará falta confiar en una o varias personas todopoderosas. Es inevitable.

La mera existencia de este tipo de superusuarios expone al software a ser totalmente comprometido y manipulado si un atacante consigue las credenciales de una de estas cuentas privilegiadas. En ese caso, dará igual cómo de bien diseñado esté el software o que no tenga vulnerabilidades. Todo se reduce a la capacidad de un atacante para conseguir hacer login como administrador. En los capítulos 13 y 14 hablaremos más sobre la gestión de credenciales y los procesos de autenticación de cuentas de usuario. Para el caso de plataformas usadas por miles o millones de usuarios, el acceso a este tipo de cuentas privilegiadas debe gestionarse con particular recelo ya que un potencial compromiso dejará a todos los usuarios de la aplicación totalmente desprotegidos. Por otro lado, en el caso de que el compromiso de una cuenta concreta sea premio suficiente por pertenecer a una persona célebre podremos decir que la propia cuenta del usuario será equivalente a la de un administrador.

No son pocos los incidentes que suceden como consecuencia del robo de cuentas de usuario privilegiadas. La facilidad e impunidad para lanzar ataques contra personas concretas de manera masiva hace que, de vez en cuando, los malos se hagan con las credenciales para utilizar una de estas cuentas. Lo habitual cuando se ataca a una persona en lugar de a una máquina es realizar lo que se conoce como *Ingeniería Social*. Este tipo de ataques tratan de engañar a la víctima haciéndose pasar por alguien de su confianza o confundiéndolo para conseguir que realice una acción en beneficio del atacante. Como ejemplo, algunos incidentes recientes es posible que todavía resuenen en la memoria del lector:

- En 2014, fotos íntimas de diversas celebridades (principalmente actrices) fueron robadas y filtradas online. Estas imágenes incluían fotos de desnudos y otros contenidos íntimos. La causa raíz de este incidente fue el acceso no autorizado a las cuentas de *iCloud* de las celebridades. Los atacantes utilizaron técnicas de ingeniería social y *phishing* para robar la contraseña

y acceder a las cuentas [47].

- En 2020, Twitter recibió un ataque donde varias cuentas pertenecientes a personalidades prominentes como Elon Musk, Barack Obama y Bill Gates o a empresas como Apple y Uber fueron comprometidas para promover un esquema fraudulento de Bitcoin. Los hackers publicaron mensajes que instaban a los seguidores a enviar Bitcoin a una cuenta específica mientras prometían al remitente recibir el doble de la cantidad que enviara. Entre los factores identificados como la causa raíz del incidente estaba el uso de la ingeniería social para engañar a determinados empleados de Twitter y obtener acceso a los sistemas internos de la empresa [48]

Si bien algunos autores citan el hecho de que casi todas las compañías no tienen más remedio que confiar en un puñado de empleados [49] no todo está perdido. Algunos servicios modernos se diseñan de tal forma que ni siquiera un administrador del propio servicio tiene la posibilidad de comprometer los datos de sus usuarios. Este tipo de software se diseña bajo la premisa de que el software podría ser vulnerado por alguien no autorizado, por un empleado disgustado o incluso por un agente de la ley con una orden judicial en la mano. En el capítulo 16 hablaremos sobre algunas plataformas modernas que ofrecen capacidades como el cifrado de extremo a extremo que impiden que un administrador pueda leer lo que los usuarios envían. No obstante, a pesar de la existencia de este tipo de servicios que presumen de ser a prueba de hackers o del FBI, debemos mantener una desconfianza natural ante este tipo de afirmaciones. Montar plataformas que mantengan la confidencialidad y privacidad de sus usuarios frente a una orden judicial no es una misión sencilla.