HERRAMIENTAS XZUTILS

SISTEMAS INFORMÁTICOS 24-25

Trabajo de Alejandro Sainz Sainz

Alejandro Sainz Sainz

INTRODUCCIÓN 3
CONTEXTO 4
COMIENZO DE LA HISTORIA 4
LIGERA EXPLICACIÓN DEL FUNCIONAMIENTO DEL CODIGO 5
EL HEROE INESPERADO (O LA JUSTICIA POÉTICA) 7
AFECTADOS 7
CONSECUENCIAS 8
POSIBLES MEDIOS DE PREVENCIÓN 9
CONCLUSIÓN Y REFLEXIÓN 9
BIBLIOGRAFIA Y ENLACES 11

TABLA DE FIGURAS

Ilustración 1 Explicación ligera y gráfica de SSH 5
Ilustración 2 funcionamiento de XZ 6
Ilustración 3 Imagen de un canal de github cerrado, que se consideraba responsable 7

INTRODUCCIÓN

Dado que, en el tema del que voy a hablar, considero ya que hablar de problemas de Microsoft se ha convertido en una especie de cliché, voy a relatar un caso muy grave sucedido para el sistema Linux y en parte para Mac.

Este suceso sucedió hace, más o menos, 2 años, pudiendo haberse convertido en uno de los peores ataques y de los peores problemas que hemos conocido en el entorno de sistemas operativos a lo largo de la historia, si no el peor.

Para ello voy a relatar lo sucedido con la herramienta XZUtils, y como un exceso de confianza, que acabó en la inserción de muchos sistemas de código malicioso, hizo que hubiese que plantearse muchas cuestiones de seguridad y de vigilancia, pues podía haber afectado a infinidad de servidores que ofrecen conexiones del tipo SSH, así como a equipos doméstico que realizan sus tareas utilizando este tipo de conexión.

Fallos tanto en la certificación del código, como en el monitoreo de procesos y subprocesos y de vigilancia pudieron tener consecuencias fatales. En mi caso, en esos momentos yo usaba una versión de Fedora, que era una de las distros afectadas, lo que hizo que cambiase de distro por un tiempo, aún a pesar de no usar este tipo de conexión para mis tareas diarias.

CONTEXTO

Como bien es sabido por gran parte de las personas que tienen contacto con el mundo informático, Linux es un sistema de código abierto, que puede ser auditado y que se nutre de muchas aportaciones de programadores y desarrolladores que se dedican a mejorar este sistema constantemente.

De la misma forma, Mac OS, es un sistema que, por decirlo de alguna manera, es pariente de Linux, ya que gran parte de su núcleo hereda gran cantidad de elementos de UNIX, el ancestro común de los dos, aunque este no fue afectado de forma tan severa.

Y en este punto aparecen las herramientas XZUtils y algunas de sus bibliotecas. Estas herramientas se usan principalmente porque realizan operaciones de compresión y descompresión (amén de alguna tarea más, pero esta es una buena forma de resumirla) que goza de gran popularidad, sobre todo dentro del ecosistema Linux. Desarrolladas en código abierto, por un solo programador de forma independiente, que gozaba de buena credibilidad por el trabajo realizado hasta la fecha y sus buenos resultados y que, llegado un determinado momento, comenzó a contar con ayuda de otros programadores.

COMIENZO DE LA HISTORIA

Para poder acceder al mantenimiento de esas herramientas y librerías, otro programador, cuyo origen es un tema bastante desconocido, y que no creo que tampoco yo vaya a comentar aquí, comenzó a hacer aportaciones al canal de github que mantenía dichas herramientas, ganando credibilidad y respeto.

Lo que nadie sabía era que el mismo creaba cuentas falsas, con las cual mandaba falsos reporte de fallos de las herramientas que necesitaban correcciones, mientras con su cuenta oficial ofrecía soluciones y mejoras al código y a las propias herramientas. Gracias a eso, ganó credibilidad y notoriedad, consiguiendo así permiso para mantener ese código y ayudar a mejorarlo. No siendo esto suficiente, continuó con la farsa hasta conseguir privilegios de commit y compilación en el código, lo que a posteriori le permitió insertar código malicioso bien escondido en el código original.

Para esconder este código, escondía sus scripts en las versiones tar del código, pero, evidentemente, no lo mostraba en la página principal del repositorio en github. Quiero recalcar

que sus scripts usaban bash como lenguaje, lo que nos da una idea de la potencia de ese lenguaje dentro del sistema.

LIGERA EXPLICACIÓN DEL FUNCIONAMIENTO DEL CODIGO

Ahora viene la parte complicada, que es aquella en la que intento explicar, con mis propias palabras, como funcionaba todo.

Cuando se descargaba el código de las herramientas y sus librerías, se ejecutaba este, y en algún momento se ejecutaba también el primero de los scripts, que realizaba una serie de operaciones de prueba, en las que ejecutaba un segundo script, que de nuevo ejecutaba ciertas operaciones temporales, dentro de las cuales se ejecutaba uno o más scripts, que eran los que inyectaban el código malicioso.

Tras todo esto, ese código lo que hacía era capturar todo lo referente a la conexión SSH del equipo. Todos conocemos la importancia de las conexiones ssh, aquellas que nos permiten acceso remoto a otros equipos o servidores.

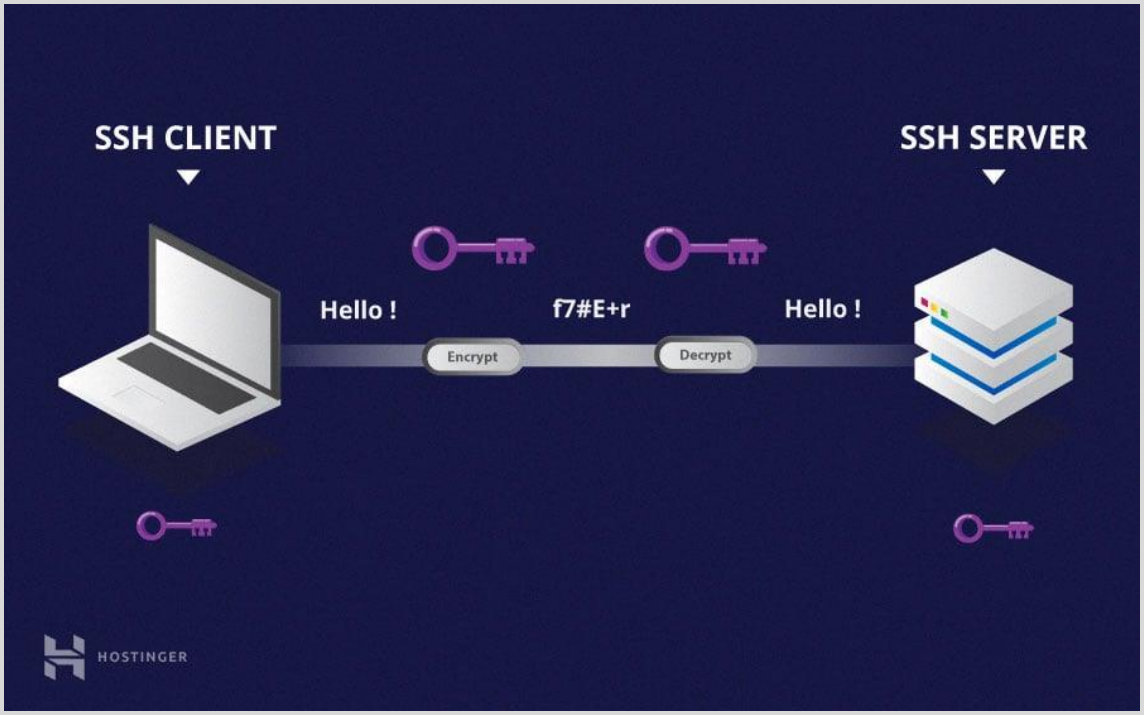

Ilustración 1 Explicación ligera y gráfica de SSH

Ahora, la segunda parte, que me es más complicada que la anterior, pero que voy a intentar explicar de la mejor manera posible.

Cuando el código ya está en nuestra máquina, según he leído en varias páginas, una de ellas lo explica bastante bien, existen una serie de condiciones que de ser cumplidas disparan las funciones de este código. En ese momento, si no he entendido mal, no es que el código engañe a las funciones del protocolo SSH, es que prácticamente se convierte en el Daemon del SSH. En el momento en el que hay una petición por parte del sistema, este código bloquea al símbolo del sistema y al Daemon SSH, capturándolo y ejecutando su código, lo que permite que, de estar usando una conexión SSH o una puerta SSH pública, esta quede abierta para el atacante, permitiéndole acceso, ya que su código inyectado hace que las credenciales del atacante sean validadas de forma automática, pasando a insertarse en el proceso system, lo que le permite ejecutar código dentro de la máquina atacada.

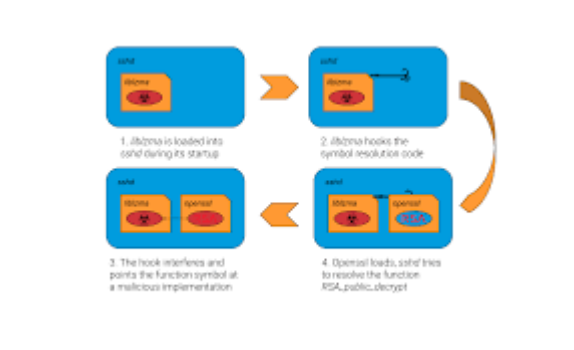

Ilustración 2 funcionamiento de XZ

A este punto hemos llegado gracias a la “buena labor” del atacante, su pericia y la falta de supervisión del código, ya sea por parte de la comunidad, del desarrollador principal, de los responsables de los diferentes sistemas operativos que usan esas herramientas… vaya usted a saber.

Durante dos o tres versiones de esas herramientas, no se sabe si han podido ser más, las conexiones ssh y el propio protocolo, estaban en peligro y, dado que gran parte de la infraestructura mundial de servidores funciona sobre Linux, la cantidad de equipos y conexiones vulnerables eran elevadísimas.

EL HEROE INESPERADO (O LA JUSTICIA POÉTICA)

Algo ya he comentado de Microsoft en la introducción, pero gracias a un hombre, se llevan mis respetos. Algo les tenía que salir bien.

Como ya se sabe, algunas de las herramientas de Microsoft, ya sea Azure, Xbox, Game Pass, etc. funcionan sobre Linux. Y he aquí la poesía, o la justicia, que un ingeniero alemán que trabajaba en Microsoft, Andres Freund, descubrió el fallo, antes de que definitivamente estas herramientas llegasen a la versión de producción de todos los sistemas operativos afectados. Que no quiere decir que no hubiese nadie afectado, mucha gente ya se había bajado esas versiones infectadas por su cuenta.

Así que aquí tenemos a nuestro héroe, solitario, del cual, si nos interesa su nombre, que en una actualización rutinaria, notó una anomalía, pues en una de las operaciones de actualización observó un pico de latencia de 500ms, y eso le llevó a investigar, a seguir los rastros y a realizar pruebas que hicieron posible la detección del error. Ni siquiera fue por usar herramientas de monitorización de procesos y subprocesos, o de monitorización de la red en general. Simplemente un pico de latencia y su curiosidad hicieron saltar todas las alarmas a nivel mundial. Y por una vez, en vez de esconder sus 0 days, como en otras ocasiones, Microsoft, rauda y veloz, comunicó el hallazgo a todo el mundo para que se pusiesen manos a la obra y arreglasen lo que se podría convertir en el mayor ataque a la seguridad de siempre, sobrepasando incluso el ataque Solarwinds, que tuvo como algunos afectados al Pentágono y algunas de las empresas más importantes del mundo.

AFECTADOS

Como afectados, todos aquellos servicios que usasen estas herramientas dentro de sus paquetes y realizasen operaciones mediante SSH.

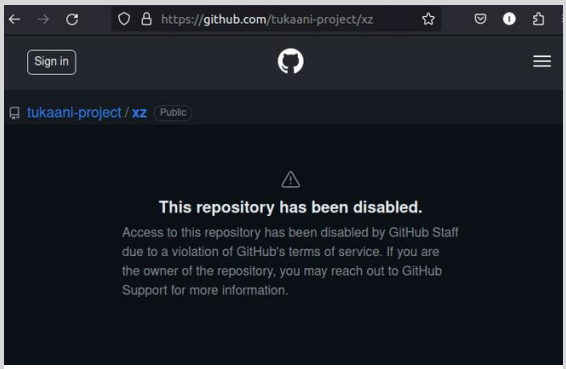

Ilustración 3 Imagen de un canal de github cerrado, que se consideraba responsable

Amén de aquellos que usasen estas herramientas y las conexiones SSH, muchas de las distribuciones de Linux se veían comprometidas. Entre ellas, gigantes como Fedora, Ubuntu, Debian estaban expuestas.

RHEL (Red Hat Enterprise Linux) lanzó rápidamente un comunicado indicando que ni ellos ni ninguna de sus branches se encontraban afectadas. Algunos de los sistemas Mac, más de lo mismo. Sin embargo, todos indicaban que se debía de volver a versiones anteriores de las herramientas XZUtils por seguridad, lo que podría entenderse como un indicio de que algo si les podía haber pasado.

Se lanzaron parches y medidas de seguridad para evitar posibles problemas, aunque, a día de hoy, no se conocen a ciencia cierta las consecuencias, y por ingeniería inversa todavía, parece ser, se siguen descubriendo diferentes condiciones por las que puede saltar vulnerabilidades en el ssh por el uso de esas herramientas, aunque ahora en entornos controlados.

CONSECUENCIAS

A ciencia cierta se desconoce el alcance que haya podido tener ya que no se detectaron fallos graves en infraestructuras importantes, pero tampoco se puede saber con exactitud si se han robado datos debido a esas intrusiones remotas y cuantos pueden ser los afectados domésticos que se hubiesen instalado esos paquetes por su mano mayor y hubiesen realizado operaciones mediante conexiones ssh. Y la pregunta que comenzaron a hacerse todos es si este tipo de cosas pudieron haber pasado con anterioridad y que nadie se hubiese dado cuenta.

Si bien es cierto que se cerraron cuentas de github y se buscaron más responsables en aquellos que tuvieron contacto con el programador encargado de crear estas herramientas, muchas de ellas han sido abiertas de nuevo a su funcionamiento normal al demostrarse que esos usuarios no tenían nada que ver, y muchas de ellas se han dedicado a colaborar realizando pruebas sobre estas herramientas y su código, para buscar código malicioso adicional que pudiese estar todavía oculto.

Dentro de la comunidad se empezó a cuestionar el sistema de aportaciones de programadores o grupos de desarrolladores independientes que lanzaban sus propias herramientas para estos sistemas, pero dado que son herramientas auditables por su condición de código abierto, quedó simplemente en la reacción inicial.

De todas formas, dada la magnitud del intento de ciberespionaje o ataque, o como quieran denominarle, se es mucho más prudente en la utilización de ciertas aplicaciones o en la credibilidad dada por sentado de ciertos programadores.

POSIBLES MEDIOS DE PREVENCIÓN

A raíz de estos hechos, muchas compañías, ya a toro pasado como suele ser habitual, hablan de una mayor utilización de monitoreo de procesos y subprocesos que ayuden a identificar estos comportamientos anómalos en sus sistemas.

Pero en este caso el problema es más profundo, ya que el ataque viene dado dentro de unas herramientas que gozaban de gran nombre y confianza dentro del sector, y de las formas en las que se actualiza su código y se difunde. En este caso, la capa de engaño que se generó para llegar a esta situación hace que sea más difícil de detectar, pero unas auditorias de código o unas pruebas más exhaustivas ayudarían en gran medida a prevenir estos casos.

También se pueden usar más ciertas herramientas de monitorización de la red, pues hay que recordar que fue una latencia inusual la que hizo saltar las alarmas.

Pero si repasamos el tema de nuevo, nos damos cuenta de que fueron el conocimiento y la curiosidad de un solo ingeniero las que ayudaron a prevenir un mayor desastre.

Otro de los medios, que se inició a raíz del caso SolaWinds, fue instar a las empresa a aplicar los principios de redes de confianza cero y los controles de acceso basados en roles para las aplicaciones y los servidores

CONCLUSIÓN Y REFLEXIÓN

Aunque se que el tema es mucho más profundo y aquí la explicación se puede considerar bastante genérica, ya que mis conocimientos del tema no dan para más, y su repercusión en daños al final no fue tan dramática vemos como un ataque bien dirigido a determinados sistemas o métodos de comunicación puede afectar a una cantidad ingente de usuarios.

Si consideramos que, queramos o no, la tecnología a hecho que lo remoto sea muy cercano, a golpe de click, una persona desde cualquier parte del mundo, sentada en una silla delante de un ordenador, puede afectar ampliamente a la seguridad y estabilidad de muchos de nuestros sistemas. Y dado que todo ahora mismo está conectado 24/7, la vigilancia de estos

sistemas, nuestros datos y nuestras credenciales deben de ser una prioridad para nosotros, pues como hemos visto a lo largo de estas páginas, un exceso de confianza o una falta de vigilancia pueden ser fatales.

En mi caso personal, pues como ya dije yo en esos momentos usaba Fedora, uno de los posibles afectados, implica el no estar con confianza ni siquiera con mi propio sistema, y eso que yo ni siquiera hacia conexiones en remoto, pero, sin embargo, la Shell de Linux, hasta donde yo sé, inicia un proceso SSH en el mismo momento en el que se inicia una sesión, lo que quiere decir que tampoco estaba completamente seguro. Eso hizo que no actualizase nada en meses, y tiempo después, cambiase de distro simplemente por si acaso. No me quiero imaginar la locura o la paranoia que se produjo en empresas mucho más grandes cuando les dijeron que podían estas afectadas por esta situación.

La otra pequeña conclusión, pero esta ya en tono más cómico, es que, aunque su sistema sea bloatware puro, sus actualizaciones hagan que muchas cosas salten por la ventana, impongan la instalación de aplicaciones que no quiere nadie, lo que hay que decir bien alto es:

Gracias, ingeniero alemán de Microsoft. De la que nos has librado.

BIBLIOGRAFIA Y ENLACES

https://procesia.com/andres-freund-ingeniero-evita-ciberataque-a-linux/

https://gist.github.com/thesamesam/223949d5a074ebc3dce9ee78baad9e27#faq-on-the-xz-utils-backdoor

https://www.akamai.com/es/blog/security-research/critical-linux-backdoor-xz-utils-discovered-what-to-know

https://www.zscaler.com/es/resources/security-terms-glossary/what-is-the-solarwinds-cyberattack