Aproximadamente el 70% de todos los errores de seguridad graves en la base de código de Chrome, son errores de seguridad y administración de memoria, según confirmaron los ingenieros de Google esta semana.

La mitad del 70% son vulnerabilidades sin uso, un tipo de problema de seguridad que surge de la gestión incorrecta de los punteros de memoria (direcciones), dejando puertas abiertas para que los atacantes comprometan los componentes internos de Chrome.

El porcentaje fue compilado después de que los ingenieros de Google analizaron 912 errores de seguridad corregidos en la rama estable de Chrome desde 2015, errores que tenían una calificación de gravedad alta o crítica.

Este número es similar a las estadísticas compartidas por Microsoft. En una conferencia de seguridad de febrero de 2019, los ingenieros de Microsoft afirmaron que durante los últimos 12 años, alrededor del 70% de todas las actualizaciones de seguridad para productos Microsoft abordaron las vulnerabilidades de seguridad de memoria.

Ambas compañías están lidiando con el mismo problema, llegando a la conclusión de que los dos lenguajes predominantes en sus bases de código, C y C++, son lenguajes inseguros.

Se trata de herramientas viejas de programación creadas hace décadas cuando la explotación de seguridad y los ataques cibernéticos no eran un modelo de amenaza relevante y estaban muy lejos de la mente de la mayoría de los primeros desarrolladores de software.

Esto da como resultado que C y C++ permiten a los programadores tener un control total sobre cómo administran los punteros de memoria (direcciones) de una aplicación y no vienen con restricciones o advertencias para prevenir o alertar a los desarrolladores cuando están cometiendo errores básicos de administración de memoria.

Estos primeros errores de codificación provocan la introducción de vulnerabilidades de administración de memoria en las aplicaciones, incluyendo vulnerabilidades como use-after-

free, desbordamiento de búfer, condiciones de carrera, doble libre, punteros salvajes, entre otros.

Estas vulnerabilidades de administración de memoria son los errores más buscados que los atacantes intentan encontrar y explotar, ya que les otorgan la capacidad de plantar código dentro de la memoria de un dispositivo y que la aplicación de la víctima (navegador, servidor, sistema operativo, etc.), lo ejecute.

En un ranking publicado a inicios del año, la Corporación MITRE, la organización responsable de la administración de la base de datos de vulnerabilidades del gobierno de Estados Unidos, clasificó el desbordamiento de búfer como la vulnerabilidad más peligrosa, y otros dos problemas relacionados con la administración de memoria también fueron catalogados entre los 10 primeros.

Conforme la ingeniería de software ha avanzado en los últimos años, los desarrolladores van mejorando en cuanto a evitar la mayoría de los fallos de seguridad y adicionar protecciones correspondientes.

Google asegura que desde marzo de 2019, 125 de las 130 vulnerabilidades de Chrome con una calificación de gravedad *«crítica»* eran problemas relacionados con corrupción de memoria, lo que demuestra que a pesar de los avances en la corrección de otras clases de errores, la gestión de memoria sigue siendo un problema.

Debido a este problema de administración de memoria, los ingenieros de Chrome ahora tienen que seguir la «regla de 2». Según esta regla, cada vez que los ingenieros escriben una nueva función de Chrome, su código no debe romper más de dos de las siguientes condiciones:

- El código maneja entradas no confiables
- El código se ejecuta sin sandbox
- El código está escrito en un lenguaje de programación inseguro (C/C++)

Aunque las compañías de software han intentado solucionar antes los problemas de administración de memoria de C y C++, Mozilla ha sido un gran avance al patrocinar, promover y adoptar en gran medida el lenguaje de programación Rust en Firefox.

Ahora, Rust es considerado como uno de los lenguajes de programación más seguros y un reemplazo ideal para C y C++.

Por otro lado, Microsoft también está invirtiendo en la exploración de alternativas a C y C++. Desde su primer proyecto Checked C, la compañía está experimentando con Rust, y también está desarrollando su propio lenguaje de programación seguro similar a Rust (parte del proyecto secreto Verona).

En la conferencia virtual Build de esta semana, Microsoft dijo que estos dos esfuerzos han tenido éxito y que la compañía se volvió a dedicar a adoptar un lenguaje de programación seguro en el futuro.

Esta semana, Google también anunció algo parecido. La compañía dijo que también planea *«abordar el problema de seguridad de la memoria»* para Chrome, el navegador web más popular hoy en día, utilizado por el 70% de los usuarios de Internet.

Hasta hoy, los ingenieros de Google han sido muy partidarios del enfoque sandbox en Chrome. Aislaron muchos procesos en su propia sandbox y recientemente implementaron el aislamiento del sitio, una característica que también pone los recursos de cada sitio en su propio proceso de sandbox.

Sin embargo, los ingenieros de Google aseguran que su enfoque para el sandboxing de los componentes de Chrome ha alcanzado sus máximos beneficios al tener en cuenta el rendimiento, y que la compañía ahora debe buscar nuevos enfoques.

En el futuro, Google asegura que planea estudiar el desarrollo de bibliotecas C++ personalizadas para usar con la base de código de Chrome, bibliotecas que tienen mejores

protecciones contra errores relacionados con la memoria.

Además, la compañía también está explorando el proyecto MiraclePtr, que tiene como objetivo *«convertir errores aprovechables sin uso en fallas no relacionadas con la seguridad con un rendimiento aceptable, memoria, tamaño binario y un impacto mínimo en la estabilidad».*

Finalmente, Google dijo que planea explorar el uso de lenguajes seguros cuando sea posible, entre estos, detalla Rust, Swift, JavaScript, Kotlin y Java.