Google ha anunciado el respaldo para lo que se denomina un *«V8 Sandbox»* en el navegador web Chrome con el propósito de abordar los problemas de corrupción de memoria.

El sandbox, según lo explicado por Samuel Groß, líder técnico de seguridad de V8, tiene como objetivo evitar que la *«corrupción de memoria en V8 se propague dentro del proceso principal».*

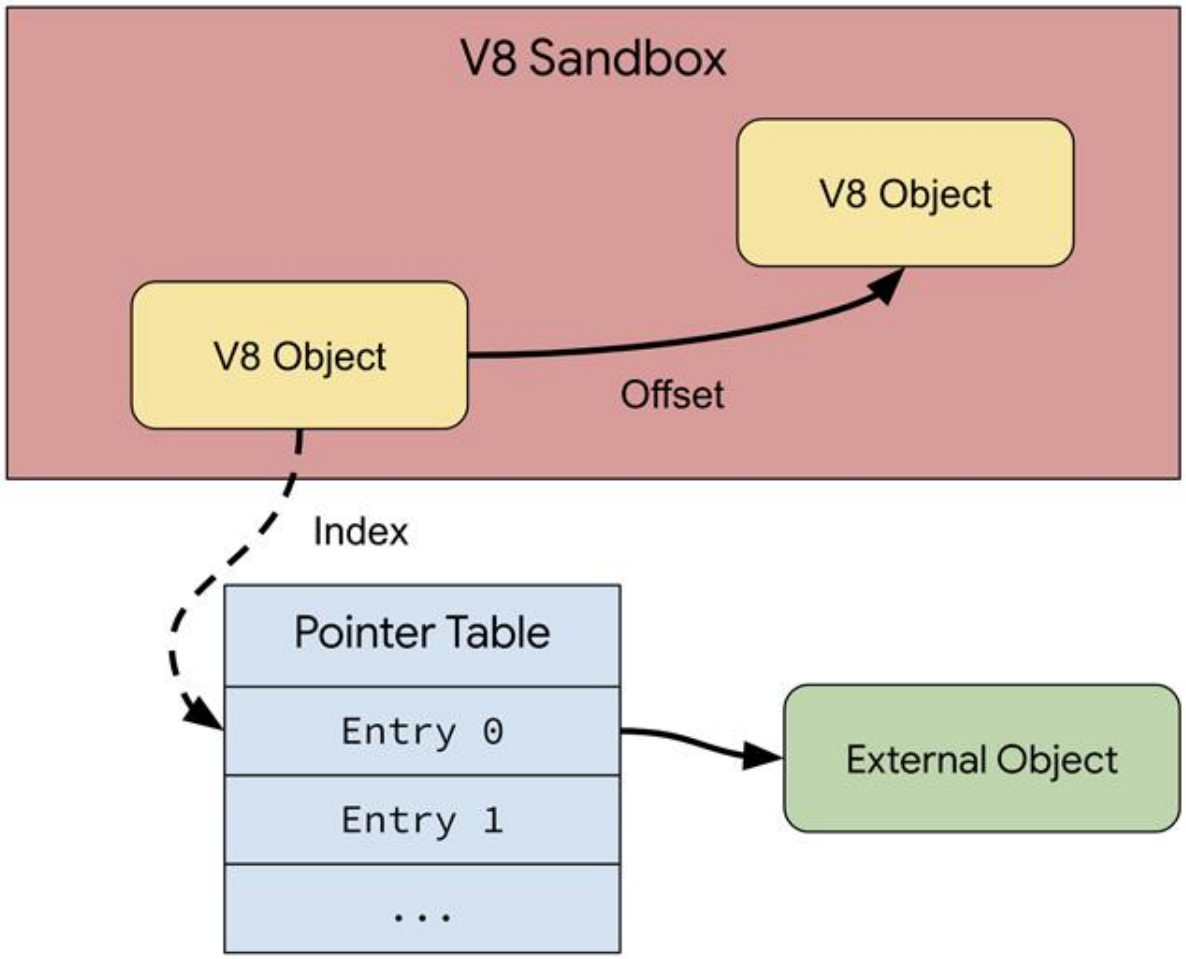
El gigante de las búsquedas ha caracterizado a V8 Sandbox como un sandbox ligero, interno, para el motor de JavaScript y WebAssembly, diseñado para mitigar vulnerabilidades comunes de V8.

La idea es reducir el impacto de las vulnerabilidades de V8 al limitar el código ejecutado por V8 a una parte del espacio de direcciones virtuales del proceso («el sandbox») y aislarlo del resto del proceso.

Las deficiencias que afectan a V8 han sido responsables de una parte importante de las vulnerabilidades de día cero que Google ha abordado entre 2021 y 2023, con hasta 16 fallos de seguridad descubiertos durante ese período de tiempo.

> *«El sandbox supone que un atacante puede modificar arbitrariamente y simultáneamente cualquier memoria dentro del espacio de direcciones del sandbox, ya que este aspecto se puede construir a partir de vulnerabilidades comunes de V8»*, mencionó el equipo de Chromium.
>
> *«Además, se supone que un atacante puede leer la memoria fuera del sandbox, por ejemplo, a través de canales laterales de hardware. El sandbox, entonces, tiene como objetivo proteger el resto del proceso de dicho atacante. Por lo tanto, cualquier corrupción de memoria fuera del espacio de direcciones del sandbox se considera una violación del sandbox.»*

Groß destacó los desafíos para abordar las vulnerabilidades de V8 mediante el cambio a un lenguaje seguro para la memoria como Rust o enfoques de seguridad de memoria de hardware, como el etiquetado de memoria, debido a la *«sutileza de los problemas lógicos»* que pueden ser explotados para corromper la memoria, en contraposición a errores de seguridad de memoria más convencionales, como el uso después de liberar, los accesos fuera de límites, y otros.



> *«Casi todas las vulnerabilidades encontradas y aprovechadas en V8 en la actualidad tienen un punto en común: la eventual corrupción de memoria ocurre dentro del*

> *heap de V8, ya que el compilador y el tiempo de ejecución (casi en su totalidad) operan en instancias de HeapObject de V8»*, señaló Groß.

Dado que estos problemas no pueden ser contrarrestados con las mismas técnicas utilizadas para las vulnerabilidades típicas de corrupción de memoria, el V8 Sandbox está concebido para aislar la memoria del heap de V8. De esta manera, si surge alguna corrupción de memoria, no podrá extenderse más allá de los límites de seguridad hacia otras áreas de la memoria del proceso.

Este objetivo se logra sustituyendo todos los tipos de datos que puedan acceder a memoria fuera del sandbox con alternativas *«compatibles con el sandbox»,* lo que efectivamente impide que un atacante acceda a otra memoria. La activación del sandbox se puede realizar estableciendo «v8_enable_sandbox» en true en los parámetros gn.

Los resultados de las pruebas de velocidad y rendimiento de Speedometer y JetStream muestran que la función de seguridad añade un incremento de aproximadamente el 1% en las cargas de trabajo típicas. Esto permite que se active por defecto a partir de la versión 123 de Chrome, cubriendo Android, ChromeOS, Linux, macOS y Windows.

> *«El V8 Sandbox requiere un sistema de 64 bits debido a que necesita reservar una gran cantidad de espacio de direcciones virtuales, actualmente un terabyte»,* explicó Groß.

> *«La creación del sandbox surge ante la constatación de que las tecnologías actuales de seguridad de memoria no son aplicables a los motores JavaScript optimizados. Aunque estas tecnologías no pueden evitar la corrupción de memoria en V8 en sí, sí pueden proteger la superficie de ataque del V8 Sandbox. Por lo tanto, el sandbox representa un paso necesario hacia la seguridad de la memoria».*

Estos avances se producen mientras Google destaca el papel del Kernel Address Sanitizer (KASan) en la detección de fallos de memoria en código nativo y en el fortalecimiento de la seguridad del firmware de Android. Agrega que utilizaron esta herramienta basada en el compilador para descubrir más de 40 errores.

> *«El uso de compilaciones habilitadas con KASan durante las pruebas y/o fuzzing puede ayudar a detectar vulnerabilidades de corrupción de memoria y problemas de estabilidad antes de que afecten a los dispositivos de usuario»,* señalaron Eugene Rodionov e Ivan Lozano del equipo de Android.