Hasta 207 sitios web fueron infectados con código malicioso diseñado para lanzar un minero de criptomonedas aprovechando WebAssembly (Wasm) en el navegador.

La empresa de seguridad web Sucuri, que publicó los detalles de la campaña, dijo que inició una investigación luego de que la computadora de uno de sus clientes se ralentizara de forma significativa cada vez que navegaba a su propio portal de WordPress.

Esto descubrió un compromiso de un archivo de tema para inyectar código JavaScript malicioso desde un servidor remoto, hxxps://wm.bmwebm[.]org/auto.js, que se carga cada vez que se accede a la página del sitio web.

> *«Una vez codificados, los contenidos de auto.js revelan inmediatamente la funcionalidad de un criptominero que comienza a minar cuando un visitante llega al sitio comprometido»*, dijo el investigador de malware de Sucuri, Cesar Anjos.

Además, el código auto.js desofuscado utiliza WebAssembly para ejecutar código binario de bajo nivel directamente en el navegador.

WebAssembly, que es compatible con todos los principales navegadores, es un formato de instrucciones binarias que ofrece mejoras de rendimiento sobre JavaScript, lo que permite que las aplicaciones escritas en lenguajes como C, C++ y Rust se compilen en un lenguaje similar a un ensamblador de bajo nivel que puede ser directamente ejecutado en el navegador.

> *«Cuando se usa en un navegador web, Wasm se ejecuta en su propio entorno de ejecución en espacio aislado. Como ya está compilado en un formato de ensamblaje, el navegador puede leer y ejecutar sus operaciones a una velocidad que JavaScript no puede igualar»,* dijo Anjos.

Se cree que el dominio controlado por el atacante, wm.bmwebm[.]org, se registró en enero de 2021, lo que implica que la infraestructura siguió activa durante más de un año y medio sin llamar la atención.

Además, el dominio también cuenta con la capacidad de generar automáticamente archivos JavaScript que se hacen pasar por archivo aparentemente inofensivos o servicios legítimos como el de Google Ads para ocultar su comportamiento malicioso.

> *«Esta funcionalidad también hace posible que el mal actor inyecte los scripts en múltiples ubicaciones en el sitio web comprometido y aún mantenga la apariencia de que las inyecciones 'pertenecen' al entorno»,* dijo Anjos.

Esta no es la primera vez que la capacidad de WebAssembly para ejecutar aplicaciones de alto rendimiento en páginas web ha generado posibles señales de alerta de seguridad.

Dejando de lado el hecho de que el formato binario de Wasm hace que la detección y el análisis por parte de los motores antivirus convencionales sean más desafiantes, la técnica podría abrir la puerta a ataques basados en navegadores más sofisticados, como el e-skimming, que puede pasar desapercibido durante largos períodos de tiempo.

Lo que complica todo es la falta de controles de integridad para los módulos Wasm, lo que hace imposible determinar si una aplicación ha sido manipulada.

Para ayudar a ilustrar las debilidades de seguridad de WebAssembly, un estudio de 2020 realizado por un grupo de académicos de la Universidad de Stuttgart y la Universidad Bundeswehr de Múnich descubrió problemas de seguridad que podrían usarse para escribir en memoria arbitraria, sobrescribir datos confidenciales y secuestrar el flujo de control.

Una investigación posterior publicada en noviembre de 2021 basada en una traducción de 4469 programas C con vulnerabilidades de desbordamiento de búfer conocidas a Wasm descubrió que *«compilar un programa C existente en WebAssembly sin precauciones*

adicionales puede obstaculizar su seguridad».

En una línea similar, una investigación de seguimiento que involucró la compilación de 17,802 programas en C que mostraban debilidades conocidas en x86 de 64 bits y en binarios WebAssembly descubrió que 4911 diferían en el resultado cuando se ejecutaba su WebAssembly y su binario x86, ya sea imprimiendo una salida distinta, o al diferir en su código de retorno.

> *«Compilar un programa C existente en WebAssembly para su distribución entre plataformas puede requerir adaptaciones del código fuente; de lo contrario, la seguridad de la aplicación WebAssembly puede estar en riesgo»,* dijeron los investigadores.

Para contrarrestar los escenarios en los que las fallas del código clásico se transfieren de los programas originales a los binarios de Wasm compilados de forma cruzada, académicos de la Universidad de Lisboa lanzaron un escáner de vulnerabilidad esática llamado Wasmati para identificar problemas en los binarios de Wasm.