Investigadores en ciberseguridad revelaron una vulnerabilidad crítica de seguridad en Ollama que, si es explotada con éxito, podría permitir que un atacante remoto y sin autenticación obtenga acceso a toda la memoria del proceso.

La falla de lectura fuera de límites, que posiblemente afecta a más de 300,000 servidores en todo el mundo, está registrada como CVE-2026-7482 (puntaje CVSS: 9.1). Los investigadores de Cyera la bautizaron como *Bleeding Llama*.

Ollama es un popular framework de código abierto que permite ejecutar modelos de lenguaje de gran escala (LLM) de forma local, sin depender de servicios en la nube. En GitHub, el proyecto supera las 171,000 estrellas y acumula más de 16,100 bifurcaciones.

"Las versiones de Ollama anteriores a la 0.17.1 contienen una vulnerabilidad de lectura fuera de límites en el heap dentro del cargador de modelos GGUF", indica la descripción oficial de la falla en CVE.org. *"El endpoint /api/create acepta archivos GGUF proporcionados por un atacante en los que el desplazamiento y tamaño del tensor declarados exceden la longitud real del archivo; durante el proceso de cuantización en fs/ggml/gguf.go y server/quantization.go (WriteTo()), el servidor lee datos más allá del búfer de memoria asignado."*

GGUF, abreviatura de *GPT-Generated Unified Format*, es un formato de archivo diseñado para almacenar modelos de lenguaje de gran tamaño y facilitar su carga y ejecución local. Funciona de manera similar a otros formatos populares de almacenamiento de modelos, como PyTorch .pt/.pth, safetensors y ONNX.

El problema se origina principalmente en el uso del paquete *unsafe* por parte de Ollama al crear modelos desde archivos GGUF, específicamente en una función denominada *"WriteTo()"*. Esto permite ejecutar operaciones que eluden las garantías de seguridad de memoria proporcionadas por el lenguaje de programación.

En un posible escenario de ataque, un actor malicioso podría enviar un archivo GGUF especialmente manipulado a un servidor de Ollama expuesto a internet, configurando la

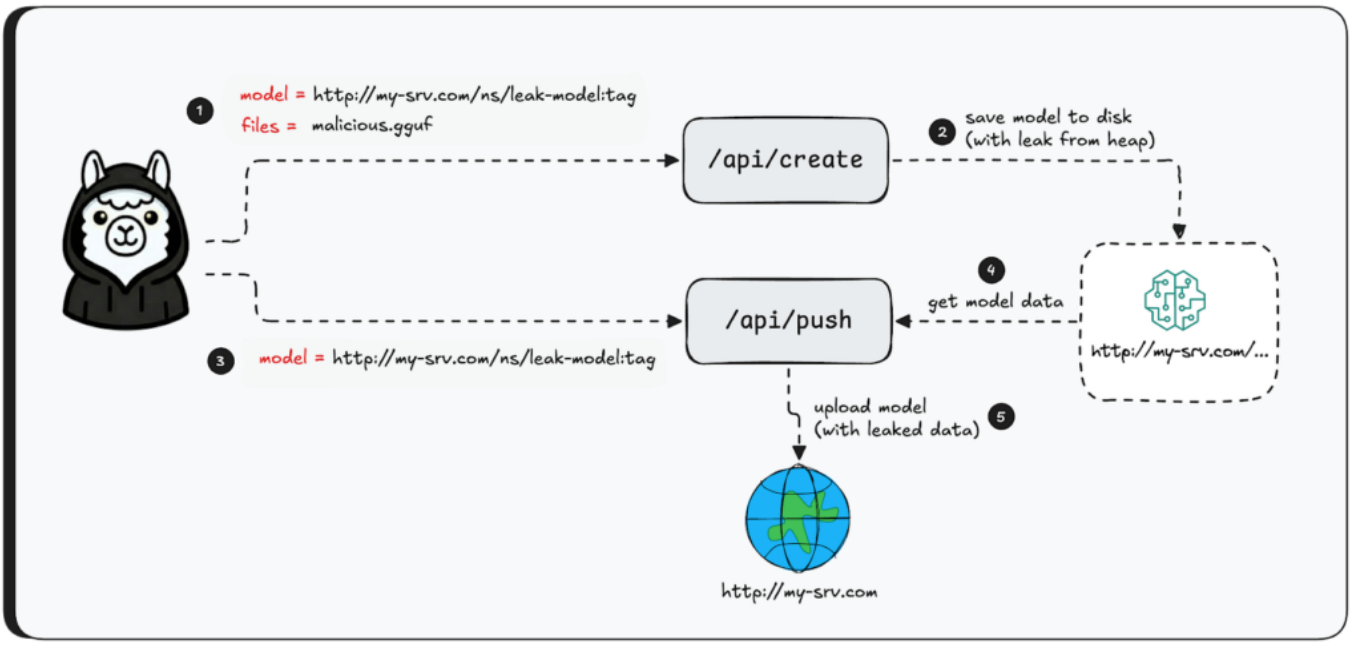
forma del tensor con un valor extremadamente grande para desencadenar la lectura fuera de límites durante la creación del modelo mediante el endpoint */api/create*. Si la explotación tiene éxito, sería posible filtrar información sensible almacenada en la memoria del proceso.

Entre los datos potencialmente comprometidos se incluyen variables de entorno, claves API, prompts del sistema y conversaciones de usuarios concurrentes. Posteriormente, esta información podría ser exfiltrada cargando el artefacto del modelo resultante mediante el endpoint */api/push* hacia un registro controlado por el atacante.

La cadena de explotación se desarrolla en tres etapas:

1. Subir un archivo GGUF manipulado con una forma de tensor inflada a un servidor de Ollama accesible por red mediante una solicitud HTTP POST.
2. Activar la creación del modelo utilizando el endpoint */api/create*, desencadenando la vulnerabilidad de lectura fuera de límites.
3. Emplear el endpoint */api/push* para exfiltrar datos desde la memoria heap hacia un servidor externo.

"Un atacante podría descubrir prácticamente cualquier cosa sobre la organización a partir de la inferencia de IA: claves API, código propietario, contratos de clientes y mucho más", afirmó Dor Attias, investigador de seguridad de Cyera.



"Además, muchos ingenieros conectan Ollama con herramientas como Claude Code. En esos casos, el impacto aumenta considerablemente: todas las salidas de las herramientas fluyen hacia el servidor Ollama, se almacenan en el heap y potencialmente podrían terminar en manos de un atacante."

Se recomienda a los usuarios instalar las correcciones más recientes, restringir el acceso de red, auditar las instancias expuestas a internet y protegerlas detrás de un firewall. También se aconseja implementar un proxy de autenticación o una API gateway delante de todas las instancias de Ollama, ya que la API REST no incorpora autenticación de manera predeterminada.

Dos vulnerabilidades sin parche en Ollama permiten ejecución persistente de código

El anuncio coincide con nuevas investigaciones de Striga, cuyos expertos detallaron dos vulnerabilidades en el mecanismo de actualización de Ollama para Windows que pueden

combinarse para lograr ejecución persistente de código. Los problemas continúan sin parche desde su divulgación el 27 de enero de 2026 y fueron publicados tras expirar el periodo de divulgación responsable de 90 días.

Según Bartłomiej "Bartek" Dmitruk, cofundador de Striga, el cliente de escritorio de Windows se inicia automáticamente al iniciar sesión desde la carpeta Startup de Windows, escucha en 127.0.0[.]1:11434 y consulta periódicamente actualizaciones en segundo plano mediante el endpoint */api/update* para ejecutar cualquier actualización pendiente durante el siguiente inicio de la aplicación.

Las vulnerabilidades identificadas corresponden a un fallo de *path traversal* y a la ausencia de verificación de firma digital. Combinadas con el mecanismo de ejecución automática al iniciar sesión, podrían permitir que un atacante con capacidad de manipular las respuestas de actualización ejecute código arbitrario cada vez que el usuario inicie sesión.

Las fallas son las siguientes:

- CVE-2026-42248 (CVSS 7.7): vulnerabilidad por falta de verificación de firma, ya que el binario de actualización no se valida antes de instalarse, a diferencia de la versión para macOS.
- CVE-2026-42249 (CVSS 7.7): vulnerabilidad de *path traversal* provocada porque el actualizador de Windows crea la ruta local del directorio temporal directamente a partir de encabezados HTTP sin sanitizarlos adecuadamente.

Para explotar estas vulnerabilidades, el atacante debe controlar un servidor de actualizaciones accesible por el cliente de Ollama de la víctima. En ese escenario, sería posible suministrar un ejecutable arbitrario como parte del proceso de actualización y escribirlo en la carpeta Startup de Windows sin generar alertas relacionadas con firmas digitales.

Una forma de controlar la respuesta de actualización consiste en sobrescribir la variable *OLLAMA_UPDATE_URL* para redirigir el cliente hacia un servidor local usando HTTP sin cifrar.

La cadena de ataque también asume que la opción *AutoUpdateEnabled* está activada, configuración habilitada por defecto.

Además, la ausencia de verificación de integridad podría provocar ejecución de código incluso sin explotar la vulnerabilidad de *path traversal*. En este caso, el instalador se deposita en el directorio temporal esperado y, durante el siguiente inicio desde la carpeta Startup, el proceso de actualización se ejecuta sin volver a validar la firma, permitiendo que el código del atacante sea ejecutado.

Sin embargo, la ejecución remota de código no sería persistente, ya que una actualización legítima posterior sobrescribiría el archivo modificado. Al combinar este comportamiento con la vulnerabilidad de *path traversal*, un atacante puede redirigir el ejecutable fuera de la ruta habitual y lograr persistencia.

De acuerdo con CERT Polska, las versiones de Ollama para Windows comprendidas entre la 0.12.10 y la 0.17.5 son vulnerables a estas fallas. Mientras no existan correcciones oficiales, se recomienda desactivar las actualizaciones automáticas y eliminar cualquier acceso directo de Ollama de la carpeta Startup («%APPDATA%\Microsoft\Windows\Start Menu\Programs\Startup») para impedir la ejecución silenciosa durante el inicio de sesión.

“Cualquier instalación de Ollama para Windows que ejecute versiones entre la 0.12.10 y la 0.22.0 es vulnerable”, explicó Dmitruk. *“La vulnerabilidad de path traversal permite escribir ejecutables elegidos por el atacante en la carpeta Startup de Windows. La falta de verificación de firmas los mantiene allí: el proceso de limpieza posterior, que debería eliminar archivos no firmados en un actualizador funcional, no hace nada en Windows. En el siguiente inicio de sesión, Windows ejecutará cualquier archivo que haya quedado almacenado.”*

“La cadena de ataque produce ejecución persistente y silenciosa de código con los privilegios del usuario que ejecuta Ollama. Entre las cargas útiles realistas se incluyen reverse shells, robadores de información capaces de extraer secretos del navegador y claves SSH, o droppers orientados a implementar mecanismos adicionales de persistencia. Cualquier cosa que pueda ejecutarse con los permisos del usuario actual. Eliminar el binario malicioso de la

carpeta Startup detiene la persistencia, pero las vulnerabilidades subyacentes continúan presentes.”