Investigadores en ciberseguridad han identificado una vulnerabilidad en el entorno de desarrollo integrado (IDE) agentico de Google, Antigravity, que podría ser aprovechada para lograr la ejecución de código.

El fallo, ya corregido, surge de la combinación entre la capacidad permitida de Antigravity para crear archivos y una sanitización insuficiente de entradas en su herramienta nativa de búsqueda de archivos, find_by_name. Esto permite eludir el llamado *Strict Mode*, una configuración de seguridad restrictiva que limita el acceso a la red, bloquea escrituras fuera del espacio de trabajo y garantiza que todos los comandos se ejecuten dentro de un entorno aislado (*sandbox*).

«Al inyectar la bandera -X (exec-batch) mediante el parámetro Pattern [en la herramienta find_by_name], un atacante puede obligar a fd a ejecutar binarios arbitrarios sobre archivos del espacio de trabajo,» explicó el investigador de Pillar Security, Dan Lisichkin, en su análisis.

«Si se combina con la capacidad de Antigravity de generar archivos como acción permitida, esto habilita una cadena de ataque completa: preparar un script malicioso y luego activarlo mediante una búsqueda aparentemente legítima, todo sin interacción adicional del usuario una vez que se produce la inyección del prompt.»

El ataque se basa en que la llamada a la herramienta find_by_name se ejecuta antes de que se apliquen las restricciones del *Strict Mode*, interpretándose como una invocación nativa. Esto deriva en la ejecución arbitraria de código. Aunque el parámetro Pattern está diseñado para aceptar patrones de nombres de archivo y realizar búsquedas mediante fd, la falta de validación estricta hace que la entrada se pase directamente al comando subyacente.

De este modo, un atacante puede aprovechar este comportamiento para preparar un archivo malicioso e introducir comandos en el parámetro Pattern que desencadenen la ejecución del payload.

«La bandera clave en este caso es -X (exec-batch). Cuando se pasa a fd, hace que se ejecute

un binario específico sobre cada archivo coincidente,» detalló Pillar. *«Al construir un valor Pattern como -Xsh, el atacante consigue que fd envíe los archivos coincidentes a sh para ejecutarlos como scripts de shell.»*

De forma alternativa, el ataque también puede iniciarse mediante una inyección indirecta de prompt sin necesidad de comprometer la cuenta del usuario. En este escenario, la víctima descarga un archivo aparentemente inofensivo desde una fuente no confiable, el cual contiene comentarios ocultos controlados por el atacante que instruyen al agente de inteligencia artificial para preparar y activar el exploit.

Tras una divulgación responsable el 7 de enero de 2026, Google solucionó el problema el 28 de febrero.

«Las herramientas diseñadas para operar con restricciones se convierten en vectores de ataque cuando sus entradas no se validan rigurosamente,» señaló Lisichkin. *«El modelo de confianza en el que se basan estas suposiciones de seguridad —que un humano detectará algo sospechoso— deja de ser válido cuando los agentes autónomos siguen instrucciones provenientes de contenido externo.»*

Estos hallazgos coinciden con el descubrimiento de múltiples vulnerabilidades —ya corregidas— en distintas herramientas impulsadas por IA:

- Anthropic Claude Code Security Review, Google run-gemini-cli (antes Gemini CLI Action) y GitHub Copilot Agent resultaron vulnerables a inyecciones de prompt a través de comentarios en GitHub. Esto permite a un atacante convertir títulos de *pull requests* (PR), descripciones de incidencias y comentarios en vectores para robar claves API y tokens. Este tipo de ataque ha sido denominado *Comment and Control*, ya que explota el acceso elevado del agente de IA y su capacidad de procesar entradas no confiables para ejecutar instrucciones maliciosas.
  «Este patrón probablemente aplica a cualquier agente de IA que consuma datos no confiables de GitHub y tenga acceso a herramientas de ejecución dentro del mismo entorno que los secretos de producción —y más allá de GitHub Actions, a cualquier

agente que procese entradas no confiables con acceso a herramientas y secretos: bots de Slack, agentes de Jira, agentes de correo electrónico, automatización de despliegues,» indicó el investigador Aonan Guan. *«La superficie de inyección cambia, pero el patrón es el mismo.»*

- Otra falla en Claude Code, descubierta por Cisco, permite contaminar la memoria del agente de programación y mantener persistencia en todos los proyectos y sesiones, incluso después de reiniciar el sistema. El ataque utiliza una técnica de cadena de suministro de software como vector inicial para desplegar un payload malicioso que manipula archivos de memoria del modelo (por ejemplo, presentando prácticas inseguras como requisitos arquitectónicos) y añade alias en la configuración de la shell del usuario.
- El editor de código con IA Cursor también presenta una cadena crítica de vulnerabilidades tipo *living-off-the-land* (LotL), llamada *NomShub*. Esta permite que un repositorio malicioso tome el control de la máquina del desarrollador mediante una combinación de inyección indirecta de prompt, evasión del sandbox del parser de comandos usando funciones internas de la shell como export y cd, y el túnel remoto integrado de Cursor, otorgando acceso persistente sin ser detectado.
- Una vez logrado ese acceso, el atacante puede conectarse al sistema sin reactivar la inyección ni generar alertas de seguridad. Debido a que Cursor es un binario legítimo firmado, el adversario obtiene acceso completo al sistema de archivos y capacidad de ejecutar comandos.
  «Un atacante humano necesitaría encadenar múltiples exploits y mantener acceso persistente,» explicaron los investigadores Karpagarajan Vikkii y Amanda Rousseau. *«El agente de IA lo hace de forma autónoma, siguiendo las instrucciones inyectadas como si fueran tareas legítimas de desarrollo.»*
- Un ataque novedoso llamado *ToolJack* permite a un atacante local manipular la percepción del entorno por parte de un agente de IA y corromper la “verdad base” de las herramientas, generando efectos como datos contaminados, inteligencia empresarial falsa y recomendaciones erróneas.
  «A diferencia de otros ataques que envenenan datos o descripciones de herramientas, ToolJack actúa en tiempo real sobre la infraestructura de comunicación,» explicó el investigador Jeremy McHugh. *«No espera a que el agente encuentre datos*

manipulados, sino que crea una realidad fabricada durante la ejecución, demostrando que comprometer el protocolo implica controlar toda la percepción del agente.»

- También se detectaron vulnerabilidades graves de inyección indirecta de prompt en Microsoft Copilot Studio (conocida como ShareLeak o CVE-2026-21520, con puntuación CVSS de 7.5) y Salesforce Agentforce (PipeLeak). Estas fallas podrían permitir la exfiltración de datos sensibles mediante formularios externos o entradas aparentemente legítimas.
  «El ataque explota la falta de sanitización de entradas y la débil separación entre instrucciones del sistema y datos proporcionados por el usuario,» indicó el investigador Bar Kaduri. PipeLeak funciona de forma similar a ForcedLeak, procesando datos públicos como si fueran instrucciones confiables, lo que permite insertar prompts maliciosos que alteran el comportamiento del agente.
- Además, se identificó un conjunto de tres vulnerabilidades en Claude que, encadenadas en un ataque llamado *Claudy Day*, permiten secuestrar silenciosamente una sesión de chat y extraer datos sensibles con un solo clic, sin necesidad de herramientas adicionales ni integraciones externas.
- El ataque consiste en incrustar instrucciones ocultas en una URL manipulada de Claude («claude[.]ai/new?q=…»), envolverla en una redirección abierta para aparentar legitimidad y presentarla como un anuncio de Google aparentemente normal. Al hacer clic, la víctima es redirigida de forma silenciosa a la URL maliciosa con la inyección invisible.
  «Combinado con Google Ads, que valida URLs por nombre de host, esto permitió mostrar un anuncio con una URL confiable que redirigía al ataque sin que el usuario lo notara. No era phishing por correo, sino un resultado de búsqueda indistinguible del real,» señaló Oasis Security.

En otra investigación publicada la semana pasada, Manifold Security mostró cómo un flujo de trabajo de GitHub Actions basado en Claude puede ser engañado para aprobar y fusionar un *pull request* malicioso usando solo dos comandos de configuración de Git que suplantan la identidad de un desarrollador confiable.

En esencia, el ataque consiste en modificar las propiedades user.name y user.email de Git

para que coincidan con las de un desarrollador reconocido (como el investigador Andrej Karpathy). Este engaño se vuelve crítico cuando el sistema de IA interpreta esos metadatos como una señal de confianza.

«En el primer intento, Claude marcó el PR para revisión manual, señalando que la reputación del autor no era suficiente,» explicaron los investigadores Ax Sharma y Oleksandr Yaremchuk. *«Pero al reenviar el mismo PR, fue aprobado. La IA ignoró su propio criterio en el segundo intento. Esa falta de consistencia es el problema: no se puede construir seguridad sobre un sistema que cambia de opinión.»*