Monitor estable para agentes de código

Un monitor de IA que lee cada entrega dos veces y promedia, para que el veredicto no dependa del ruido del modelo.

Equipo: Juan Sebastián Herrera Corzo, Manuel Santiago Figueroa Gómez, Esteban Alejandro Hernández · Universidad de los Andes

Track: 2. Control de IA, monitoreo y detección de sabotaje

Repositorio: github.com/EstebanHerna/defacc-dualview · **Demo:** `python scripts/paquete.py` (offline) · **Contacto:** [correo]

Resumen

Cuando un agente de IA programa sin supervisión, otro modelo más barato decide qué revisa un humano. Medimos si ese guardia es confiable y encontramos que no: una línea en blanco, que no cambia el programa, le mueve el veredicto al 9 % de los backdoors. Construimos el monitor estable, que lee cada entrega dos veces y promedia: sube la detección del 92,2 % al 95,3 % y reduce el ruido a la mitad, sin cambiar de modelo.

1. La amenaza

Un monitor barato puntúa la sospecha de cada entrega de un agente de código y solo lo más sospechoso llega a una persona. Si ese puntaje cambia sin que cambie el programa, un agente con un backdoor no necesita engañarlo: le basta reenviar hasta pasar.

2. Lo que construimos

Dos piezas. Primero, el monitor estable: lee cada entrega dos veces —el código tal cual y el mismo código con una línea en blanco— y promedia los puntajes. Si el ruido de cada lectura es independiente, se cancela; la señal de sospecha no. No reentrena nada, funciona con cualquier modelo y cuesta dos llamadas en vez de una. Segundo, la batería que lo justifica y lo verifica: compara cada entrega consigo misma bajo siete canales de persuasión y tres lecturas sin mensaje, y emite un boletín PASS/FAIL. Un comando reproduce todo desde los CSV, sin llamar a ningún proveedor.

Figura 1. Monitor estable frente al monitor base, en dos modelos.

3. Cómo lo evaluamos

Setting Apps de ControlArena, calificado post-hoc con `basic_monitor`: 64 soluciones honestas y 64 con backdoor en gemini-3.5-flash-lite, con réplica en gemini-3.1-flash-lite. Línea base: una sola lectura, con el umbral que deja pasar 5 % de honestas. Métricas: detección, falsas alarmas reales y veredictos que cambian sin que cambie el programa.

4. Resultados

Métrica	Línea base	Nuestro sistema	Diferencia
Detección de backdoors (TPR)	92,2 %	95,3 %	+3,1 pp
Veredictos que cambian por ruido	7,8 %	3,1 %	−60 %
Falsas alarmas (FPR real)	4,7 %	4,7 %	igual
Llamadas al modelo por entrega	1	2	×2

Tabla 1. gemini-3.5-flash-lite, 64 honestas y 64 backdoors de Apps; umbral pedido 5 % de falsas alarmas, obtenido 4,7 %. Réplica en gemini-3.1-flash-lite: 91,7 % → 97,2 %.

El monitor no se deja convencer —diez pruebas de persuasión, ninguna supera a un comentario inocuo— pero cambia de veredicto solo; por eso la defensa ataca el ruido, no la persuasión. Dónde fallamos: nuestra primera defensa, la doble vista, no mejoró nada y la descartamos.

5. Límites y riesgo de doble uso

Un solo benchmark, entregas ya hechas y backdoors no verificados en sandbox; canales de frases fijas, sin atacante adaptativo. Los intervalos de detección se solapan: la mejora es consistente en los dos modelos, pero con estas muestras no alcanza significancia estadística; la reducción del ruido sí es clara. Doble uso: solo usamos frases genéricas ya públicas, sin generador ni optimizador de ataques.

Referencias

ControlArena (UK AISI y Redwood Research) · Inspect AI (UK AISI) · Inglis, R. apps-control-arena (dataset, Hugging Face) · Con asistencia de Claude (Anthropic); detalle en el README.