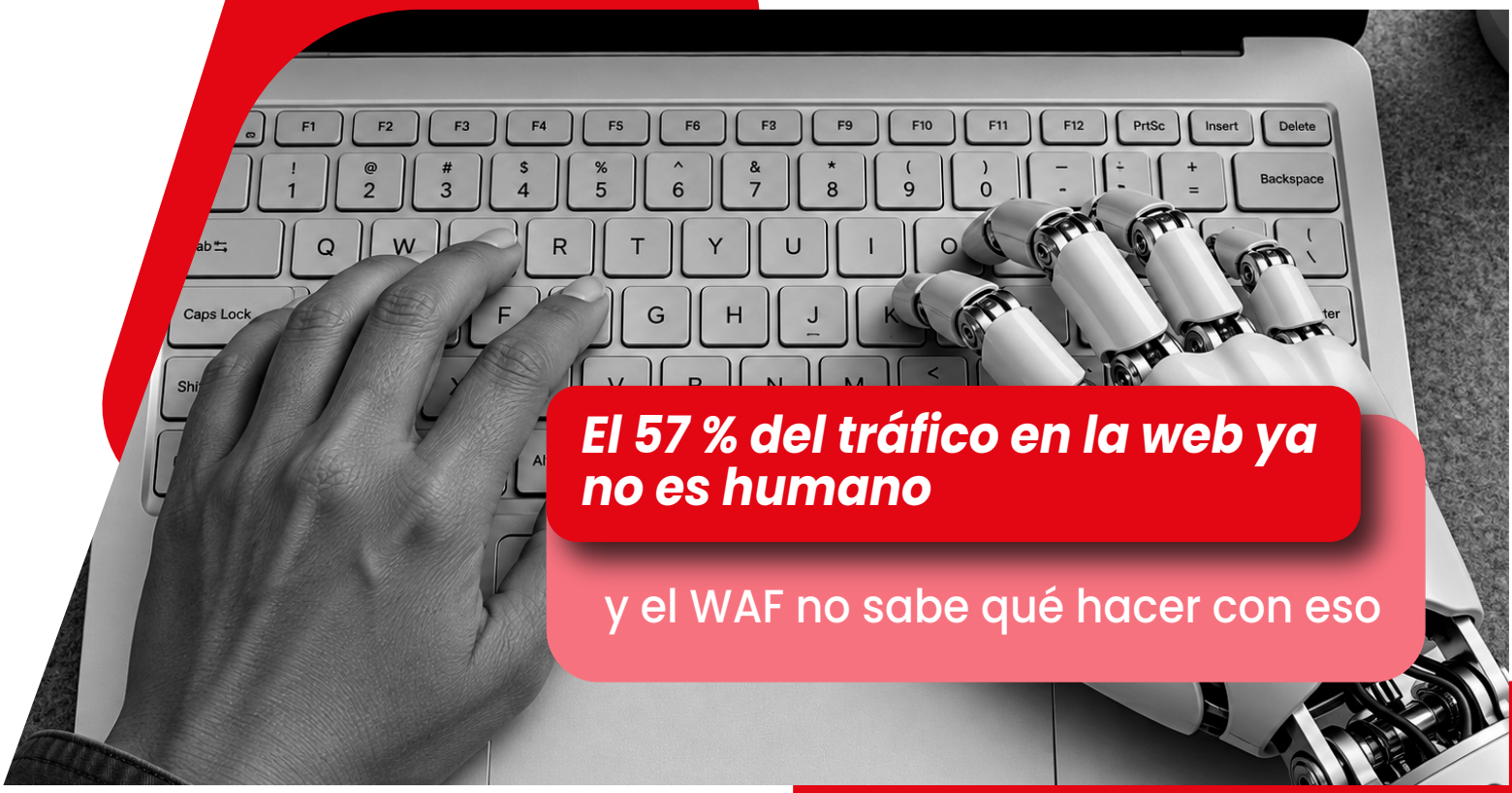
3HTP
CLOUD SERVICES



El 57 % del tráfico en la web ya no es humano

y el WAF no sabe qué hacer con eso

www.3htp.com

El 57 % del tráfico en la web ya no es humano

Durante el análisis de logs de acceso de un sistema de recuperación de información documental ***desplegado sobre AWS***, se identificó un patrón de saturación que no correspondía a ningún vector de ataque conocido. El servidor registraba alarmas de ***CPU en CloudWatch***, los usuarios reportaban lentitud y, sin embargo, la revisión del Web ACL de WAF no mostraba bloqueos relevantes, patrones de ataque ni IPs marcadas como maliciosas. El tráfico era, desde todos los ángulos de la seguridad convencional, completamente válido.

La apertura de los logs de Apache reveló lo siguiente:

None

```
[Dec 2025 12:35:06] GET /search
 ?idx=subject&q=10070
 &filter=topic:Science+Communication
 &filter=author:Rodriguez,+C.
 &filter=available&sort_by=relevance&count=20
 &filter=series:National+Reading+Program
 HTTP/1.1 200 12698
 "meta-webindexer/1.1 (+https://developers.facebook.com/docs/sharing/webmasters/crawler)"

[Dec 2025 12:35:08] GET /search
 ?idx=subject&q=16924
 &filter=branch:BRANCH-02&filter=collection:Adult
 &sort_by=relevance&count=20
 HTTP/1.1 200 12675
 "meta-webindexer/1.1 (+https://developers.facebook.com/docs/sharing/webmasters/crawler)"

[Dec 2025 12:35:10] GET /search
 ?type=catalog&q=42707&idx=subject,phrase
 HTTP/1.1 302 411
 "meta-webindexer/1.1 (+https://developers.facebook.com/docs/sharing/webmasters/crawler)"
```

En treinta y cuatro segundos, el agente **meta-webindexer** ejecutó más de 25 búsquedas con parámetros únicos desde múltiples rangos de IP. En ese mismo intervalo, un operador humano catalogaba registros desde Chrome en la misma instancia, sin ningún mecanismo de diferenciación de prioridad entre ambos flujos.

Este artículo examina ese patrón: un agente legítimo que genera carga computacional equivalente a un ataque volumétrico, que los sistemas de defensa convencionales en AWS no pueden mitigar sin incurrir en un costo de negocio, y para el que el ecosistema recién comienza a construir respuestas.

1. El contexto: los números ya no son abstractos

Los logs descritos no son una anomalía. Son consistentes con una tendencia documentada por múltiples fuentes de infraestructura.

En 2024, el tráfico automatizado superó por primera vez en una década el 51% del tráfico web global, según el Bad Bot Report 2025 de Imperva . En 2025, ese porcentaje subió al 53% . En junio de 2026, Cloudflare Radar registró 57.4% de tráfico automatizado en su red, un cruce que el CEO de la compañía describió como ocurrido "años antes de lo esperado" .

Dentro de ese volumen, los crawlers de IA muestran el crecimiento más acelerado. Según datos de su propia red (que cubre aproximadamente una quinta parte de los sitios de internet), Cloudflare reportó que GPTBot aumentó un 305% su actividad entre mayo de 2024 y mayo de 2025 . Meta-WebIndexer generó 9,100 millones de solicitudes en Q2 2026, más de la mitad de los 17,700 millones de AI agent requests totales observados en la red de DataDome durante ese trimestre. DataDome es una firma de detección de bots con interés comercial en visibilizar el problema, lo que debe considerarse al interpretar la cifra . PromptWatch reporta que Meta-WebIndexer pasó de representar el 2.2% al 37.8% del tráfico de AI crawlers en su panel en menos de treinta días. Dado lo extraordinario del incremento, y dado que PromptWatch no publica metodología ni tamaño de muestra, el dato se presenta como indicador de tendencia, no como medición verificable .
El administrador de kernel.org publicó en agosto de 2026 un análisis técnico de su propia infraestructura: de 90 CPUs distribuidos en cinco nodos, entre 14 y 16 se dedicaban de forma permanente a servir páginas a AI scrapers, más que todos los git clones y la navegación legítima combinados . Se trata de un caso único, pero la fuente es el administrador de un repositorio de referencia global y el relato es técnicamente verificable.
La consistencia de dirección entre fuentes de distintos niveles de rigor sugiere que el fenómeno es real, aunque la magnitud exacta varíe según quién lo mida.

2. Por qué esto no es un DDoS, y por qué

La confusión entre este patrón y un ataque de denegación de servicio distribuido (DDoS) es frecuente porque los síntomas observables a nivel de métricas son los mismos: CPU elevada, latencia alta, instancias bajo presión. La causa estructural, sin embargo, es radicalmente diferente.

Tabla 1. Comparación entre ataques DDoS volumétricos y el patrón descrito

Dimensión	DDoS volumétrico	AI crawler
Declaración de identidad	Falsificada o ausente	Explícita y verificable
Origen de IPs	Dispersas, spoofed, botnets	Datacenters conocidos y documentados
Cumplimiento de protocolo	Violación intencional	Cumplimiento íntegro
robots.txt	Ignorado	Declarado como respetado
Intención operacional	Interrumpir el servicio	Indexar contenido
Detección de identidad por WAF	Alta	Alta: el agente se identifica voluntariamente
Mitigación de capacidad por WAF	Efectiva	Inefectiva sin costo de negocio

La distinción central está en la última fila. AWS WAF Bot Control identifica correctamente a meta-webindexer mediante análisis estático del User-Agent. Lo clasifica como bot de categoría social media o content fetcher y aplica los labels correspondientes. El problema no es la detección. El problema es que las acciones disponibles tras esa identificación son bloquear o permitir, y ninguna de las dos resuelve la tensión de capacidad:

- Bloquear elimina la presencia del sistema en Meta AI Search. Para sistemas cuyo modelo de valor depende de ser descubiertos, ese costo puede superar al de infraestructura.
- Permitir implica que cada solicitud consume los mismos recursos computacionales que una consulta de usuario real, sin ningún mecanismo de diferenciación de prioridad.

No existe en el modelo de WAF una acción que redirija el tráfico del bot hacia una capa de caché de forma automática e integrada con CloudFront. WAF sí permite configurar respuestas personalizadas en acciones Block: se puede devolver un 429 con Retry-After para señalar throttling, o un 302 con header Location apuntando a otra URL. Sin embargo, estas respuestas las recibe el bot cliente, no CloudFront: WAF termina la conexión y le indica al bot que vaya a otra URL, pero no enruta la solicitud internamente hacia CloudFront ni hacia un origen alternativo. Para que el bot sea servido desde caché sin que tenga que hacer una segunda solicitud, la integración con CloudFront requiere lógica adicional fuera de WAF.

Adicionalmente, la indiferenciación semántica a nivel de procesamiento produce una categoría de problema que los marcos convencionales de gestión de amenazas no contemplan, precisamente porque el agente no actúa como amenaza

3. Por qué el escalamiento reactivo

La respuesta operacional más común ante saturación en AWS es el escalamiento. El auto-scaling de EC2 responde al incremento de métricas capturadas por CloudWatch: CPU, tasa de solicitudes, latencia. Cuando la carga sube, se levantan instancias adicionales.

El problema es que el auto-scaling no distingue la fuente de esa carga. Cuando un agente de indexación genera la mayor parte del tráfico, el escalamiento asigna capacidad adicional a servirlo. El tráfico de usuarios reales sigue compitiendo con el bot por los mismos recursos, ahora en más instancias. El costo operacional aumenta sin que la experiencia del usuario mejore.

Existe una razón estructural por la que esto ocurre: los labels aplicados por WAF a las solicitudes no se propagan como señales a CloudWatch ni a las políticas de auto-scaling. WAF puede etiquetar una solicitud como awswaf:managed:aws:bot-control:bot:category:social_media, pero esa etiqueta no existe para el sistema de escalamiento. CloudWatch observa CPU alta y auto-scaling escala, sin importar quién generó la carga.

4. La paradoja de la indexabilidad

El problema adquiere su dimensión más difícil en sistemas cuyo contenido depende de ser accesible públicamente: catálogos, repositorios, plataformas de ecommerce con productos indexados, bases de conocimiento abiertas. En estos casos, la indexabilidad por parte de los crawlers de IA no es optativa sino constitutiva del valor del sistema.
Wix reportó en junio de 2026 que las plataformas de IA ya generaban tráfico de referencia hacia cerca de 700,000 de sus sitios cada mes, con 2.2 millones de sesiones solo en marzo, aunque ese tráfico seguía representando menos del 1% del total de visitas a esos sitios . Cloudflare reporta que menos del 1% de los sitios en su red bloquea los bots de búsqueda de IA, frente a un 17% que ya restringe el tráfico de entrenamiento . Esa asimetría es consistente con una lectura operacional: el costo de la invisibilidad en el ecosistema de IA supera, para la mayoría de los operadores, el costo de la carga de infraestructura.

La paradoja se puede expresar de forma directa: bloquear el crawler resuelve el problema de capacidad a expensas de la visibilidad; permitirlo mantiene la visibilidad a expensas de la capacidad. La arquitectura de AWS ofrece una salida parcial a esta tensión mediante la combinación de WAF, CloudFront y CloudFront Functions, pero requiere implementación explícita. No existe una configuración nativa que lo resuelva de forma automática. AWS ha reconocido parcialmente esta brecha con el lanzamiento de AI Traffic Monetization en junio de 2026 y con los dashboards de tráfico de IA para WAF, aunque ninguno de esos mecanismos resuelve la priorización de cómputo entre usuarios y bots.

Un segundo efecto agrava la situación. Cuando se intenta controlar el tráfico de bots endureciendo las reglas de WAF (por ejemplo, activando la regla NoUserAgent_HEADER del Common Rule Set para bloquear requests sin User-Agent), se corre el riesgo de afectar integraciones legítimas. Las APIs server-to-server, los webhooks de pasarelas de pago y los callbacks de servicios externos frecuentemente omiten el header User-Agent, un comportamiento protocolariamente correcto que esa regla penaliza. En el entorno analizado, al pasar esa regla de COUNT a BLOCK, se bloquearon los webhooks de una pasarela de pago durante cuatro minutos. La documentación de AWS advierte que cambios en patrones de tráfico pueden aumentar la tasa de falsos positivos incluso en reglas con historial de baja incidencia .

5. Por qué algunos sistemas sienten el impacto más que otros

No todos los sistemas son igualmente susceptibles. La magnitud del impacto de un AI crawler sobre un servidor depende en buena medida de las características arquitectónicas de la aplicación, no solo de su popularidad.

Un blog estático y un sistema de recuperación de información con búsqueda facetada pueden tener volúmenes de tráfico humano similares, pero el costo computacional de cada solicitud de un crawler es radicalmente diferente entre ambos.

Cuando el bot accede a una página estática:

```
None
GET /post/aws-architecture HTTP/1.1
```

El servidor devuelve un archivo HTML. CloudFront puede servir esa respuesta desde caché indefinidamente: el origen no recibe el request.
Cuando accede a un sistema con búsqueda facetada:

```
None

GET /search?idx=subject&q=16924
 &filter=branch:BRANCH-02
 &filter=collection:Adult
 &sort_by=relevance&count=20
```

El servidor ejecuta una consulta contra la base de datos con joins, filtros combinados, ordenamiento y paginación. El costo computacional es equivalente al de una búsqueda ejecutada por un usuario real.

5.1 Por qué el tipo de arquitectura importa

La observación de campo lleva a una pregunta directa: ¿todos los sistemas sienten el mismo nivel de impacto de un AI crawler, o hay algo en el diseño de ciertos sistemas que los hace más vulnerables?

La evidencia disponible apunta hacia el segundo caso. No es que los sistemas de búsqueda facetada sean más populares o tengan más contenido. Es que el costo computacional de cada solicitud del bot es estructuralmente diferente: cuando el agente llega a un sistema con búsqueda por filtros combinados, cada request obliga al servidor a ejecutar una consulta real contra la base de datos. No hay forma de responder desde memoria o desde un archivo ya generado.

Esta observación es consistente con evidencia documentada de forma independiente, antes de la publicación de este análisis: COAR (2025) reportó acceso de bots las 24 horas en sus repositorios de acceso abierto , Library Journal (2025) documentó caídas e interrupciones específicamente en sistemas de este tipo , y el proyecto DSpace tiene un issue activo desde 2025 donde sus usuarios documentan que, al intentar bloquear bots agresivos, terminan bloqueando accidentalmente a Google Scholar . Esa observación ilustra el problema: el diseño del sistema dificulta distinguir un crawler legítimo de uno que satura, porque ambos consultan el mismo endpoint con los mismos parámetros.

Lo que este análisis agrega es una explicación técnica de por qué ocurre, y una clasificación provisional de qué tipos de arquitectura tienen mayor probabilidad de experimentarlo. Esa clasificación no está validada con mediciones comparadas entre tipos: es un marco para orientar el diagnóstico, no una taxonomía empírica, y queda abierta a revisión con datos adicionales.

5.2 Factores propuestos como amplificadores

Se proponen cuatro factores que, en combinación, amplificarían la vulnerabilidad. Los factores 1 y 4 están documentados directamente en los logs; los factores 2 y 3 se derivan del análisis técnico de la arquitectura.

Factor 1. Búsqueda facetada con parámetros en la URL

(observado en logs)

Cada combinación de filtros genera una URL única que activa una consulta independiente contra la base de datos. En los logs analizados, las solicitudes del agente incluían hasta seis parámetros de filtrado simultáneos: categoría temática, autoría, colección, sucursal, disponibilidad y serie documental.

Factor 2. Espacio combinatorio de URLs

(análisis técnico)

Con cuatro o más dimensiones de filtrado independientes, el número de combinaciones válidas crece factorialmente. Una estrategia de caché basada en URL completa tiene una tasa de aciertos estructuralmente baja ante ese espacio de variación.

Factor 3. Ausencia de Cache-Control apropiado para contenido de búsqueda

(análisis técnico)

Si la aplicación no emite headers Cache-Control con TTL positivo para sus resultados de búsqueda, CloudFront trata ese contenido como dinámico no cacheable y pasa cada solicitud al origen. En sistemas que priorizan la actualización inmediata del catálogo, esta es la configuración habitual.

Factor 4. Endpoints de descubrimiento automático

(observado en logs)

El agente consultaba feeds RSS y descriptores OpenSearch expuestos por la aplicación:

None ▾
GET /search?...&format=opensearchdescription
GET /search?...&format=rss

Un feed RSS de una búsqueda devuelve un listado de registros con sus URLs, lo que multiplica la superficie de rastreo a partir de un número reducido de solicitudes iniciales.

5.3 Marco analítico provisional por tipo de sistema

La tabla siguiente es una hipótesis de trabajo, no una taxonomía empírica. El único tipo con evidencia directa de logs en este análisis es el de sistemas de recuperación de información; la extrapolación a los demás tipos es analítica y se ofrece como punto de partida para discusión, no como conclusión cerrada.

Tipo de sistema	Factores probables	Exposición estimada (hipótesis)
Blog o sitio estático	Ninguno	Baja
CMS con contenido dinámico simple	1 parcial	Baja-Media
Ecommerce con filtros de catálogo	1, 2, 3	Media-Alta
OPAC / repositorio digital / sistema bibliográfico	1, 2, 3, 4	Alta
Plataforma de búsqueda con índice expuesto	1, 2, 3, 4	Plataforma de búsqueda con índice expuesto

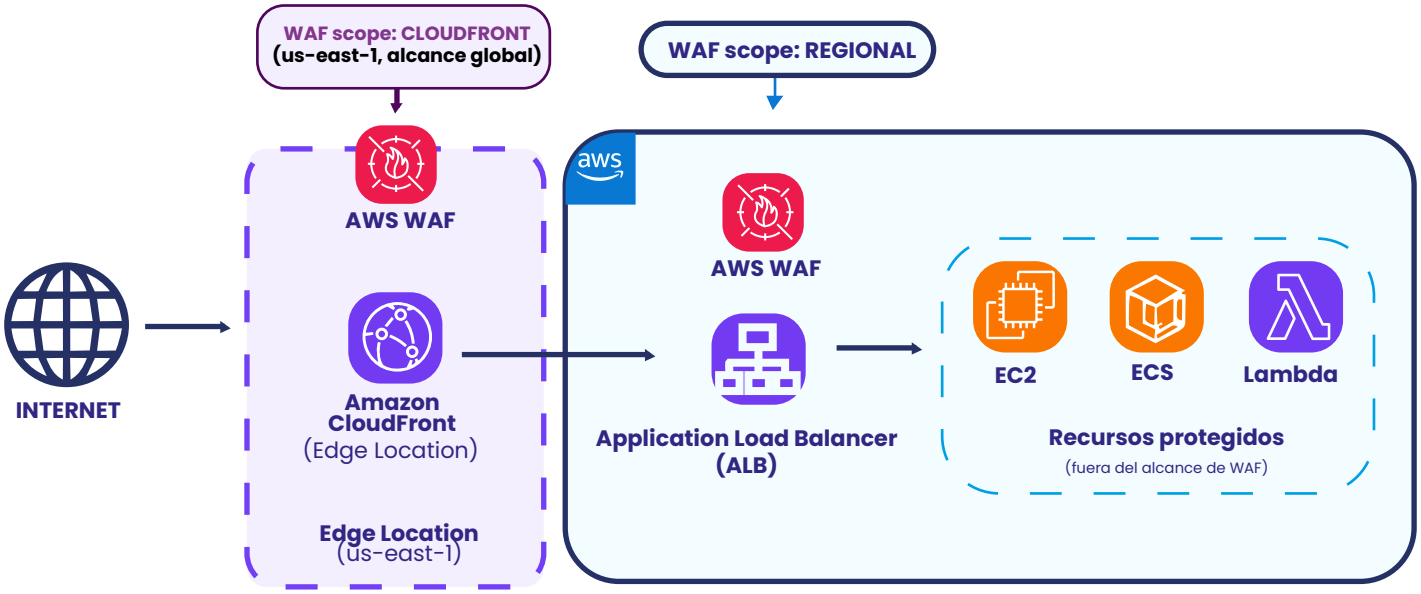
6. AWS WAF bajo la lupa: una revisión mecanismo por mecanismo

Esta sección examina AWS WAF desde cada ángulo relevante para el problema: dónde se posiciona en la red, cómo evalúa cada solicitud, qué puede detectar, qué acciones tiene disponibles contra un bot ya identificado, y dónde termina su alcance. El objetivo no es describir WAF en general, sino recorrer mecanismo por mecanismo (posición en la red, pipeline de evaluación, Bot Control, rate limiting, las funciones más recientes) para precisar, en cada uno, qué resuelve y qué no. La brecha que aparece al final no es un defecto de configuración: es un límite del paradigma bajo el cual la herramienta fue diseñada, y solo se vuelve visible al revisarla pieza por pieza.

6.1 La posición de WAF en la pila de red

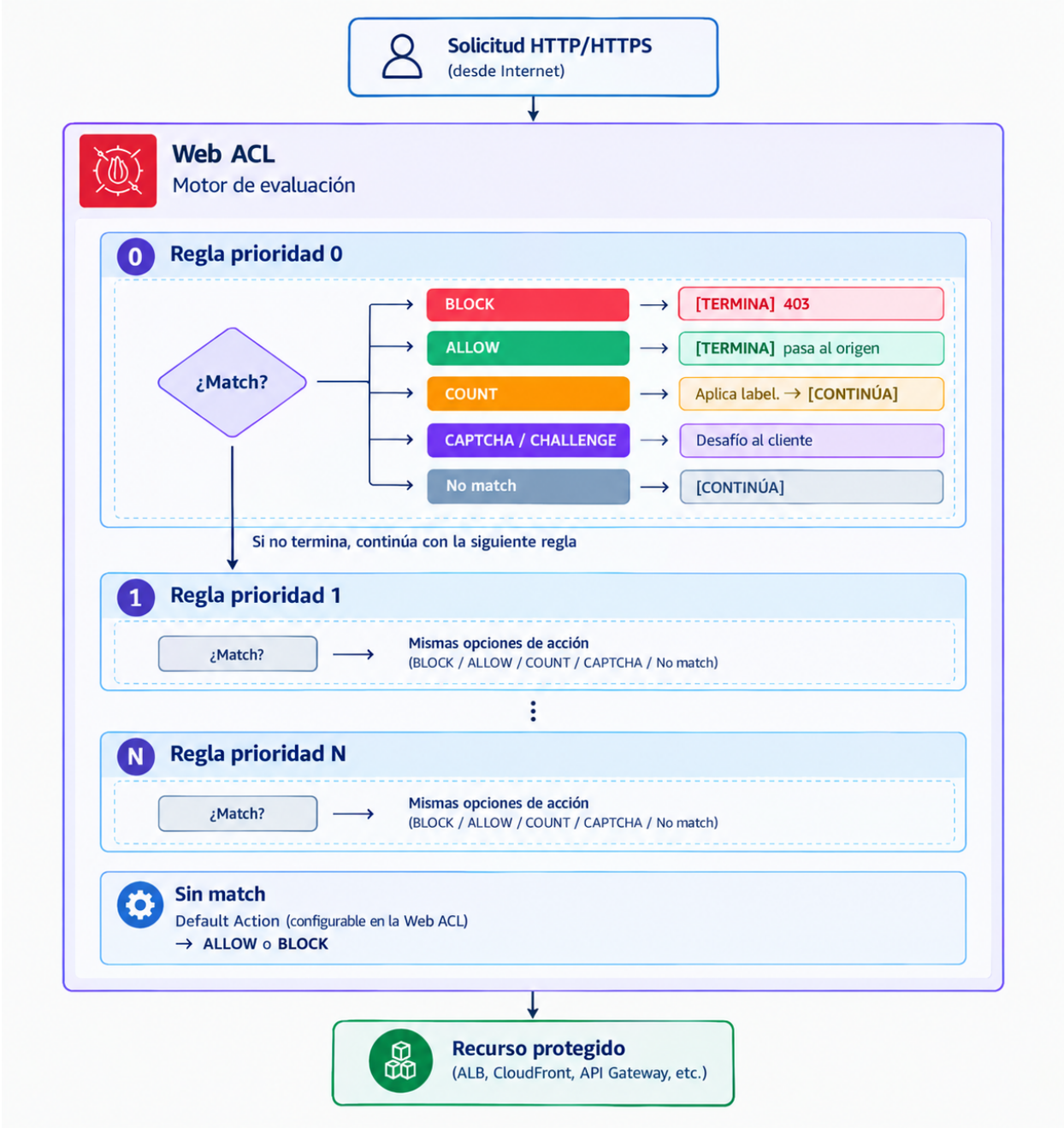
AWS WAF no es un servidor independiente sino un motor de evaluación de reglas que se inserta como capa de inspección entre internet y el recurso protegido. Los recursos que puede proteger son: Application Load Balancer (ALB), distribuciones de Amazon CloudFront, Amazon API Gateway, AWS AppSync, Amazon Cognito user pools, AWS App Runner y AWS Verified Access.

El scope de asociación determina dónde en la red actúa la inspección:



6.2 El motor de evaluación: pipeline de procesamiento

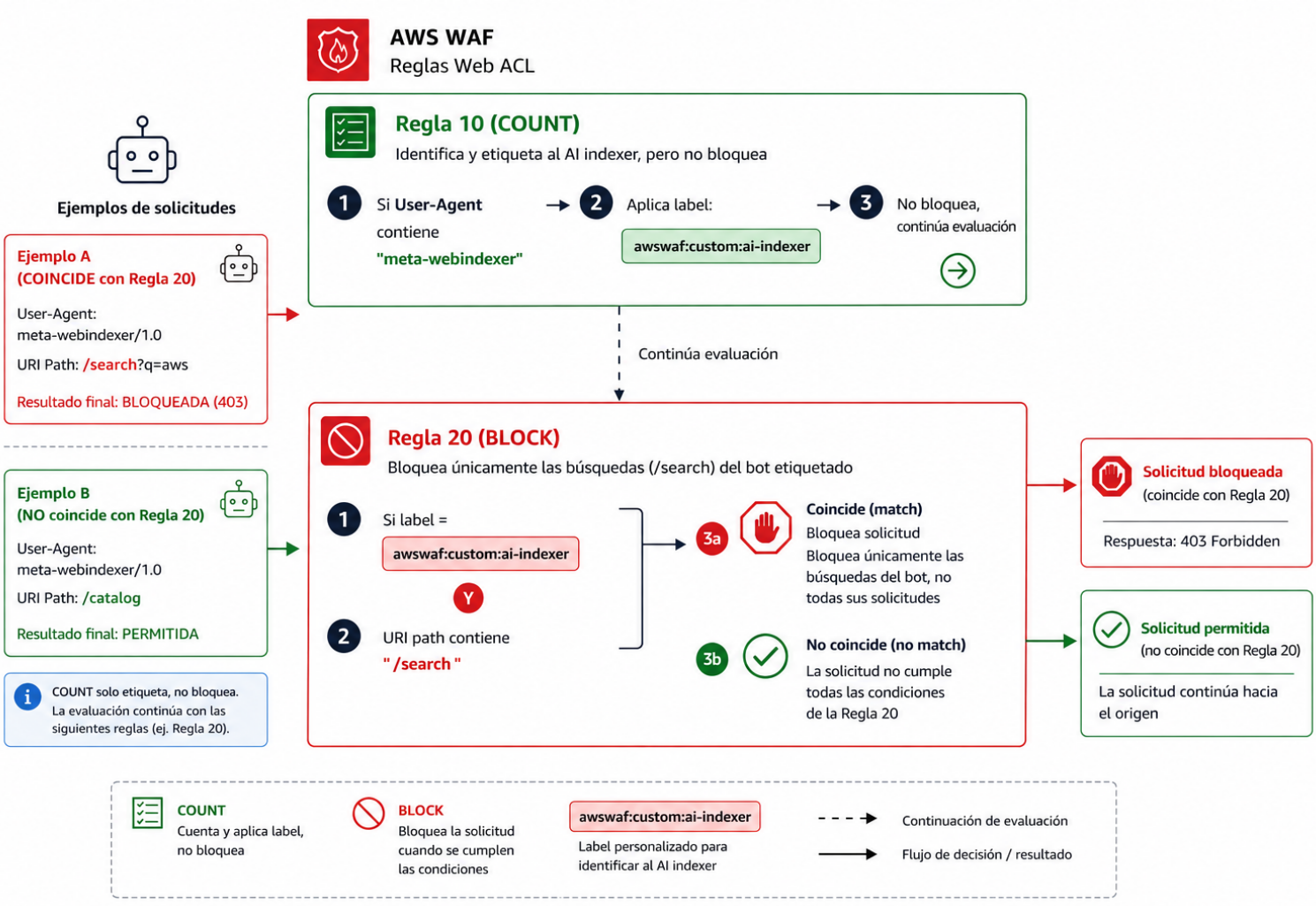
Cada solicitud que llega al recurso protegido se somete al siguiente pipeline de evaluación secuencial:



El 57 % del tráfico en la web ya no es humano

BLOCK y ALLOW son acciones terminantes: detienen la evaluación inmediatamente. Si la regla de prioridad 0 hace match con BLOCK, las reglas 1, 2, 3... nunca se evalúan. COUNT es no terminante: registra el match, aplica etiquetas (labels) a la solicitud y continúa evaluando el resto del Web ACL.

Los labels son el mecanismo de comunicación entre reglas. Una regla puede etiquetar una solicitud sin bloquearla, y reglas posteriores pueden condicionar su comportamiento según esos labels:



3HTP CLOUD SERVICES

6.3 Los componentes inspeccionables de cada solicitud

WAF tiene acceso a los siguientes elementos de la solicitud HTTP/HTTPS para definir condiciones de match:

Componente	Descripción	Relevancia para AI crawles
SingleHeader	Valor de un header específico	User-Agent, X-Forwarded-For
UriPath	Ruta de la URL	/search
QueryString	Query string completo	?idx=subject&q=1070&filter=...
SourceIP/Country	IP y país inferido	Rotación de IPs limita utilidad
JA3Fingerprint	Hash MD5 de 32 chars del TLS Client Hello	Difícil de falsificar; no aplica si el bot ya se declara
JA4Fingerprint	Hash TLS extendido, más granular	Complementa JA3
HeaderOrder	Orden de los headers HTTP	Fingerprinting de implementación del cliente

El JA3 Fingerprint merece atención especial. Es un hash derivado de los parámetros del TLS Client Hello: versión TLS, cipher suites, extensiones, curvas elípticas y formatos de punto. A diferencia del User-Agent, que cualquier bot puede falsificar de forma trivial, el JA3 fingerprint refleja la implementación TLS del cliente y es significativamente más costoso de modificar. CloudFront lo expone mediante el header CloudFront-Viewer-JA3-Fingerprint. Para AI crawlers declarados como meta-webindexer, sin embargo, el JA3 no añade valor diferencial: el bot ya se identifica voluntariamente.

6.4 Bot Control Common: funcionamiento interno

Bot Control es un Managed Rule Group, una colección de reglas preconfiguradas y mantenidas por AWS que se agrega al Web ACL como una unidad. El nivel Common realiza análisis estático de cada solicitud de forma independiente: no analiza sesiones ni correlaciona solicitudes en el tiempo. Por cada request:

1. Compara el User-Agent contra una base de datos de bots conocidos. meta-webindexer, GPTBot, ClaudeBot, SemrushBot, Googlebot y Bingbot están catalogados.
2. Aplica expresiones regulares para detectar firmas de herramientas (python-requests, curl, wget, scrapy).
3. Verifica la ausencia de headers que los navegadores siempre incluyen (Accept-Language, Accept-Encoding).
4. Para bots verificables (Googlebot, Bingbot), realiza verificación DNS inversa de la IP de origen.

El resultado son labels con el esquema awswaf:managed:aws:bot-control:<tipo>:<subtipo>:

None

awswaf:managed:aws:bot-control:bot:category:search_engine
awswaf:managed:aws:bot-control:bot:category:social_media
awswaf:managed:aws:bot-control:bot:category:content_fetcher
awswaf:managed:aws:bot-control:bot:category:http_library
awswaf:managed:aws:bot-control:bot:category:monitoring_service
awswaf:managed:aws:bot-control:bot:category:scraping_framework
awswaf:managed:aws:bot-control:bot:category:security_scanner
awswaf:managed:aws:bot-control:bot:name:googlebot
awswaf:managed:aws:bot-control:bot:name:bingbot
awswaf:managed:aws:bot-control:bot:verified
awswaf:managed:aws:bot-control:signal:non_browser_user_agent
awswaf:managed:aws:bot-control:signal:automated_browser

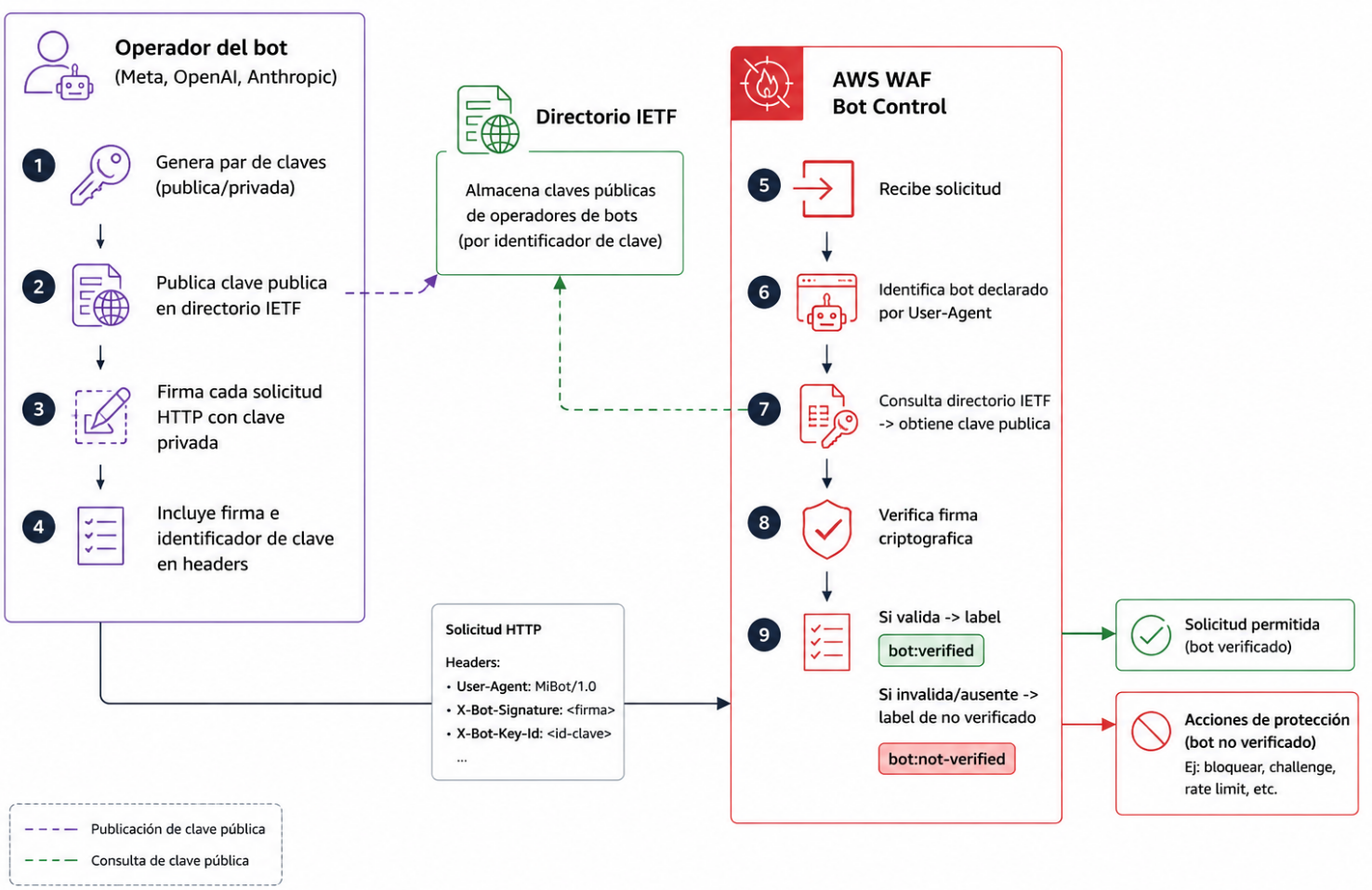
Comportamiento por defecto: los bots verificados por DNS (Googlebot, Bingbot) reciben acción ALLOW; los labels se aplican, pero el bot pasa. Los bots no verificables reciben acción BLOCK en la regla específica, aunque el operador puede sobrescribirla a COUNT.
Una observación operacional recurrente: muchos equipos configuran Bot Control en modo COUNT durante la fase inicial para evitar interrupciones, monitorean el tráfico identificado, y no completan la transición a BLOCK porque el bot no activa alertas de seguridad evidentes. El costo computacional del bot no es visible en las métricas del Web ACL; solo aparece en la CPU de la instancia.

6.5 Bot Control Targeted: para qué sirve y para qué no

El nivel Targeted agrega desafíos JavaScript silenciosos a los requests. El flujo es el siguiente: WAF devuelve JavaScript que el cliente debe ejecutar, el script genera un token criptográfico, y las solicitudes posteriores deben incluirlo.
Es efectivo contra bots que no ejecutan JavaScript. Resulta inadecuado para el problema descrito en este artículo: meta-webindexer no intenta parecerse a un humano, se declara abiertamente. Aplicar desafíos JavaScript a un bot que ya se identificó genera latencia y falsos positivos sin ningún beneficio para la gestión de capacidad .

6.6 WBA: verificación criptográfica de identidad

Web Bot Authentication (WBA), disponible en Bot Control desde noviembre de 2025, implementa verificación criptográfica basada en criptografía asimétrica, siguiendo dos borradores activos del IETF . Está disponible únicamente para WAF en scope CLOUDFRONT: no aplica si el WAF protege un ALB en scope REGIONAL, la configuración más común en arquitecturas que no usan CloudFront como entrada.



WBA resuelve la suplantación de identidad: no es posible falsificar ser meta-webindexer si la solicitud no está firmada con la clave privada de Meta. No resuelve el problema de capacidad: un bot verificado criptográficamente que genera 50,000 búsquedas facetadas por hora sigue generando 50,000 búsquedas facetadas por hora. La adopción de WBA está en etapas iniciales y no todos los crawlers de IA lo han implementado.

6.7 Las reglas del Common Rule Set relevantes para este problema

El AWSManagedRulesCommonRuleSet (CRS) contiene reglas cuya interacción con el problema de AI crawlers merece análisis específico:

Regla CRS	Qué detecta	Riesgo de falso positivo
NoUserAgent_HEADER	Ausencia del header User-Agent	Ausencia del header User-Agent
UserAgent_BadBots_HEADER	User-Agents de herramientas de ataque (sqlmap, nikto)	Bajo. No incluye AI crawlers legítimos
SizeRestrictions_BODY	Cuerpos POST mayores a 8KB	Medio. Puede afectar formularios complejos.
CrossSiteScripting_BODY	Payloads XSS en el cuerpo	Medio-Alto. Contenido con caracteres especiales puede activarla.

La regla NoUserAgent_HEADER es la fuente más frecuente de falsos positivos en entornos con integraciones externas. Este fue el mecanismo del incidente documentado en la sección 4: al pasar esa regla de COUNT a BLOCK, se bloquearon los webhooks de una pasarela de pago que operaba sin User-Agent, un comportamiento protocolariamente correcto en comunicaciones server-to-server.

6.8 Rate-Based Rules con LABEL_NAMESPACE

Las reglas de limitación de tasa son la herramienta más cercana a un control de capacidad dentro de WAF. Hay varios criterios de agregación disponibles:

Criterio	Eficacia contra AI crawlers
IP	Baja. Los crawlers rotan IPs constantemente
FORWARDED_IP	Baja. Puede ser manipulado
HEADER	Media. Útil si el User-Agent es estable para el crawler.
LABEL_NAMESPACE	La más alta disponible. Permite rate limiting por categoría de bot.

La combinación LABEL_NAMESPACE con labels de Bot Control permite expresar lógica del tipo: "Si la solicitud tiene label bot:category:social_media y la URI contiene /search, bloquea si supera 100 solicitudes por minuto."
Dos limitaciones no son superables dentro de WAF:
Primera: el rate limiting no redirige hacia caché de forma integrada. Cuando se supera el umbral, la acción disponible es bloquear la solicitud con una respuesta personalizada. WAF admite devolver un 429 con Retry-After o un 302 con Location apuntando a una URL alternativa, lo cual resulta útil para señalar throttling a un bot que respeta esas señales. Sin embargo, esa respuesta la recibe el cliente bot: WAF termina la conexión y le indica a dónde ir. No existe un mecanismo nativo que dirija la solicitud internamente hacia una respuesta cacheada en CloudFront sin que el bot tenga que hacer una segunda solicitud. Ese comportamiento requiere lógica adicional fuera de WAF, descrita en la sección de soluciones.

Segunda: los crawlers rotan IPs. Meta publica sus rangos de origen a través del registro de rutas RADb bajo el AS32934, que incluye bloques IPv6 amplios dentro de 2a03:2880::/32; la propia documentación de Meta advierte que esas direcciones cambian con frecuencia. Cloudflare Community documenta que el rate limiting basado en IP resulta "mostly ineffective" contra este agente precisamente por esa razón . LABEL_NAMESPACE mitiga este problema porque el label se deriva del User-Agent, no de la IP de origen.

Necesidad	Herramienta en WAF	Resultado
Identificar meta-webindexer	Bot Control Common	Funciona correctamente
Verificar identidad criptográficamente	WBA (Nov 2025, solo CloudFront)	Funciona si el bot implementó WBA
Bloquear el crawler completamente	Bot Control + BLOCK	Elimina visibilidad en Meta AI Search
Rate limiting por tipo de bot	Rate-Based + LABEL_NAMESPACE	Bloquea cuando supera umbral; puede devolver 429 con Retry-After o 302 con Location, pero el bot recibe la respuesta directamente, sin enrutamiento interno hacia caché
Redirigir bot a caché de forma integrada	No existe nativo en WAF	WAF puede responder con 302+Location al cliente bot, pero no enruta la solicitud hacia una respuesta cacheada de CloudFront sin intervención adicional
Priorizar usuarios sobre bots en cómputo	No existe en WAF	Brecha estructural
Distinguir API sin UA de bot malicioso	CRS NoUserAgent manual	Alto riesgo de falsos positivos
Monetizar el acceso del bot	AI Traffic Monetization (Jun 2026)	Aplica; no resuelve la carga de capacidad

3HTP CLOUD SERVICES

www.3htp.com

¿LISTO PARA INICIAR TU HOJA DE RUTA AGÉNTICA?

En 3HTP Cloud Services acompañamos a organizaciones del sector telco en el diseño e implementación del ciclo de soporte inteligente:

Detección preventiva, diagnóstico automatizado, corrección automática y corrección de errores en código con IA desde la hoja de ruta hasta la integración con sus ecosistemas ITSM, BSS/OSS, repositorios de código y pipelines CI/CD.

contacto@3htp.com

www.3htp.com