Engineer

# integridad de subrecursos

Hash con el que el navegador comprueba un archivo

## Qué es

La integridad de subrecursos permite que una página declare de antemano qué bytes exactos debe contener un script o una hoja de estilo y hace que el navegador descarte el archivo si llega algo distinto. Suele abreviarse SRI. La página escribe un resumen criptográfico del archivo en el atributo `integrity` del elemento `script` o `link` que lo carga. Al recibir la respuesta, el navegador aplica la misma función hash a los bytes recibidos y compara el resultado con el resumen del elemento. Si coinciden, el archivo se ejecuta o aplica estilos a la página como de costumbre. Si difieren, el navegador trata la respuesta como un error de red: el script no se ejecuta y la hoja de estilo no se aplica.

El valor se compone del nombre de una función hash, un guion y el resumen en base64. Una orden sobre el archivo terminado lo produce, y ejecutarla dos veces muestra lo poco que hace falta para cambiar el resultado:

```
$ printf "alert('Hello, world.');" > hello.js
$ openssl dgst -sha384 -binary hello.js | openssl base64 -A
H8BRh8j48O9oYatfu5AZzq6A9RINhZO5H16dQZngK7T62em8MUt1FLm52t+eX6xO
$ echo >> hello.js
$ openssl dgst -sha384 -binary hello.js | openssl base64 -A
qu8P7hCLFcBpas8BOnGmaANCkfXJIpFttDEaxOj0Pq9H6Q8NYjsMb+PVqwuJygVe
```

Con `sha384-` delante, la primera salida es el valor del atributo para ese script de una línea. Añadir un salto de línea al final produce un resumen completamente distinto, y un navegador que conserve el primero rechazaría el segundo archivo.

La especificación reconoce tres funciones hash: SHA-256, SHA-384 y SHA-512, todas de la familia SHA-2, y considera SHA-384 un buen punto de partida en el momento de redactarla. Un atributo puede enumerar varios resúmenes

separados por espacios. El navegador elige la función más fuerte y acepta el archivo cuando coincide con cualquiera de los resúmenes calculados con ella. Eso permite pasar a un algoritmo más fuerte o aceptar dos versiones de una biblioteca durante una transición, sin un intervalo en el que la página solo contenga resúmenes incorrectos.

El atributo existe en `script` y `link`, y el mismo valor puede viajar en una cabecera de respuesta `Link` que precargue un archivo. El texto actual de la especificación es su segunda edición, un borrador de trabajo del W3C elaborado por el grupo de trabajo de seguridad de aplicaciones web. La primera edición se convirtió en Recomendación del W3C en 2016.

Utilice el concepto en una decisión

Aplique «¿Qué es la integridad de subrecursos?» a una decisión real. Podemos ayudarle a relacionar el concepto con el sistema que está construyendo o comprando. Cuentas, contraseñas y permisos organizados para que cada persona tenga exactamente el acceso que necesita y lo pierda el mismo día que deja la empresa.

**Explore nuestros servicios** Póngase en contacto →

Compruebe los supuestos detrás del término

Utilice la explicación y los trabajos enlazados para formular preguntas sobre su proyecto. Podemos examinar los supuestos con usted antes de elegir una herramienta. 1.000 reglas RBAC para servidores Linux — 1,000 reglas de acceso escritas para cerrar una brecha de seguridad

**1.000 reglas RBAC para servidores Linux** Póngase en contacto →

## Por qué importa

Una página que carga una biblioteca desde una red de distribución de contenido, o desde cualquier servidor que sus propietarios no administren, concede a ese servidor la capacidad de ejecutar código en la página. HTTPS protege el archivo durante su recorrido por la red, pero no dice nada sobre lo que el servidor decidió enviar. Si alguien logra acceso no autorizado al almacenamiento del CDN, toma el control de una cuenta que publica en él o sustituye discretamente el archivo de una dirección conocida, todas las páginas que lo incluyan ejecutan el nuevo código con toda la autoridad de la página. El propio código de la página no tiene forma de distinguirlo.

SRI devuelve esa decisión a la página. El propietario del sitio calcula el resumen sobre un archivo que ha leído o construido, y el resumen viaja dentro de la página desde el propio servidor del sitio. Si un servidor entrega algo distinto, su archivo se rechaza en lugar de ejecutarse. La página pierde una función hasta que alguien lo revisa, un fallo visible y con una causa clara, en vez de ejecutar discretamente código ajeno.

El resumen también fija una versión. Una dirección como `/lib/latest.js` puede empezar a entregar una versión nueva en cualquier momento, sin que cambie la página; el resumen no puede seguir ese cambio. Una actualización que el propietario nunca eligió se convierte en un rechazo de un elemento concreto, notificado mediante un evento de error.

Conviene precisar lo que SRI no hace. Comprueba un archivo frente a un resumen; no juzga si el archivo es seguro. El resumen de una biblioteca que ya contenga una línea maliciosa protege fielmente esa línea. No interviene en los scripts en línea, que corresponden a Content Security Policy. Y confía en la página que contiene el resumen, por lo que la propia página también tiene que llegar intacta.

## Cómo se aplica en la práctica

Este sitio lo utiliza en su propia hoja de estilo. La página de seguridad indica que la hoja de estilo utiliza integridad de subrecursos: durante la generación del sitio se calcula la huella y el hash del paquete, de modo que el navegador pueda verificar que llegó sin modificaciones. No se publica JavaScript ejecutable ni se hacen solicitudes a terceros. Si se cuentan los servidores en el panel de red del navegador, aparece uno: el del propio sitio.

La página de créditos explica cómo se produce ese mismo archivo: un único paquete CSS concatenado y con huella calculada durante la generación del sitio, que se sirve con un hash SRI. La huella introduce un hash del contenido en el nombre del archivo, por lo que una hoja de estilo modificada recibe una dirección nueva y el navegador no confunde una copia antigua en caché con la actual. El valor de integridad permite, a su vez, comprobar los bytes de esa dirección. El proceso de generación produce ambos a partir de los mismos bytes y en el mismo paso, de modo que cada versión entrega juntos la hoja de estilo y su resumen. Así queda muy poco margen para el fallo más común que se describe a continuación: que el archivo cambie después de calcular su hash.

## Dónde surgen los problemas

**Algo entre la generación del sitio y el navegador reescribe el archivo.** Un CDN que minifica scripts, un proxy que inserta contenido en la respuesta o un paso de despliegue que reescribe un archivo después de calcular su resumen cambia los bytes del contenido, y el navegador rechaza el resultado aunque no haya habido ningún ataque. Volver a comprimir con gzip o Brotli conservando el contenido no

provoca esa discrepancia: la comprobación usa el cuerpo descodificado de la respuesta, no los bytes comprimidos enviados por la red. La especificación asigna la responsabilidad a los intermediarios que modifican contenido: deben mantener el resumen acorde con lo que sirven o entregar el original intacto, y el servidor de origen debería enviar `Cache-Control: no-transform` para pedirles que no alteren el archivo. El orden fiable es calcular el resumen al final, sobre los bytes exactos del contenido descodificado que comprobará el navegador.

**Un archivo de otro origen llega sin CORS.** La especificación exige que la comprobación de integridad de un archivo de otro origen use CORS y considera un error lógico intentarlo sin él. Por ejemplo, el elemento puede usar `crossorigin="anonymous"`, y el servidor debe responder con una cabecera que permita compartir el archivo. `crossorigin="use-credentials"` también usa CORS, con cabeceras del servidor que permitan expresamente el origen solicitante y las credenciales. Sin una solicitud CORS y una política coincidente en el servidor, el navegador bloquea el archivo en lugar de cargarlo sin comprobarlo. La regla protege a otros sitios: si no existiera, un atacante podría cargar una respuesta privada con un resumen adivinado y deducir del resultado, por ejemplo, si la persona ha iniciado sesión allí.

**El resumen llega por HTTP sin cifrar.** SRI protege una página frente a un servidor de archivos comprometido, nunca frente a alguien situado en el trayecto de red. En una página servida sin HTTPS, quien pueda reescribir el archivo durante el tránsito también puede reescribir o borrar el resumen de la página. Por eso la especificación recomienda entregar los metadatos de integridad solo en un contexto seguro.

**El navegador omite la función hash elegida.** La especificación solo reconoce SHA-256, SHA-384 y SHA-512 y recomienda que los navegadores rechacen funciones conocidas por su debilidad, como MD5 y SHA-1. La sorpresa está en lo que ocurre con un token ajeno a esa lista: el navegador lo ignora, y un atributo formado únicamente por tokens ignorados cuenta como si no incluyera metadatos. Por tanto, el archivo se carga sin comprobarse. Una herramienta de generación como Hugo, capaz de calcular huellas con MD5, facilita que ese error llegue a publicarse por accidente.

**Un archivo rechazado deja inutilizable la página.** Cuando falla la comprobación, el navegador dispara un evento de error en el elemento y el script o la hoja de estilo simplemente falta. Para una biblioteca sin la que la página no pueda funcionar, la especificación sugiere capturar el evento y cargar una copia

alternativa desde una fuente fiable, aunque sea más lenta, como el propio servidor del sitio. Sin ella, la protección funciona tal como se diseñó y la persona sigue recibiendo una página rota.

**Solo se comprueban los elementos que llevan un resumen.** El atributo de integridad de un script no dice nada del siguiente que se añada sin él, y revisar los scripts que sí lo tienen nunca revelará el que carece de él. El borrador añade una cabecera de respuesta que aborda el problema desde el servidor. `Integrity-Policy` con `blocked-destinations=(script)` bloquea todos los scripts externos que llegan sin metadatos de integridad, e `Integrity-Policy-Report-Only` los enumera primero para poder corregirlos antes de bloquear nada.

## Lecturas adicionales

- Integridad de subrecursos — El borrador actual, con el algoritmo de verificación, las cabeceras `Integrity-Policy` y las notas sobre proxies, HTTP sin cifrar y filtraciones entre orígenes.
- Integridad de subrecursos, Recomendación del W3C de 2016 — La primera edición, la Recomendación estable de la que parte el borrador.
- Fetch Standard — Dónde se contrastan los metadatos de integridad de una solicitud con el cuerpo de la respuesta y el protocolo CORS del que depende la comprobación entre orígenes.
- Hugo: resources.Fingerprint — Cómo calcula un generador estático la huella de un archivo durante la generación del sitio y devuelve el valor de integridad correspondiente.
- FIPS 180-4, estándar de hash seguro — El estándar federal que define SHA-256, SHA-384 y SHA-512.

Glosario

Elija según sus restricciones reales

Describa la ruta de despliegue, los riesgos operativos y quiénes se encargan de las guardias. Podemos definir el alcance de un cambio con comprobaciones, una vía de recuperación y responsabilidades claras.

**Vea cuánto cuesta** Póngase en contacto →

# Haga práctica la terminología

Pregunte cómo se aplica «¿Qué es la integridad de subrecursos?» a su organización. Basta un ejemplo concreto de su sistema actual para iniciar la conversación.

Explore nuestros servicios

welcome@engineer.company +45 50 10 61 84 Copenhague, Dinamarca

Llévese esta página

Tarjeta de contacto

Creado con esmero y cariño por D.

CVR 45969703 · Tagensvej 105C, 3. th, 2200 København N

Derechos de autor © 2025 – actualidad · Engineer ApS · Todos los derechos reservados.

Impreso desde https://engineer.company/es/glossary/sri/