Reporte de Muestra
AIHACKERS

Evaluación externa de superficie · Documento de ejemplo, cliente ficticio

Cliente · **Grupo Ejemplo, S.A. de C.V.**

Versión · **1.0 · Documento de muestra**

Clasificación · **Material comercial, distribución libre**

Cliente: Grupo Ejemplo, S.A. de C.V. **(empresa ficticia)**
Objetivo: `grupo-ejemplo.example` **(dominio reservado, no existe)**
Perfil: 240 empleados · 34 nombres de host públicos · 4 proveedores de nube
Versión: 1.0 · Documento de muestra
Clasificación: Material comercial, distribución libre

Este documento es una muestra

No corresponde a ninguna evaluación real ni a ningún cliente real. La empresa, el dominio, los nombres de host y todas las direcciones IP son ficticios: el dominio usa el TLD `.example`, reservado por el RFC 2606 y que no puede resolver, y las direcciones provienen de los rangos de documentación del RFC 5737. Ningún sistema descrito aquí existe.

Lo que sí es real es **la forma**: la estructura, la evidencia citada textualmente, la disciplina con que se califica cada hallazgo, la remediación al nivel de línea de configuración, y la manera de declarar lo que no se probó. Los hallazgos reproducen los patrones que esta práctica encuentra con más frecuencia, para que una organización de perfil similar reconozca el suyo.

Cómo leer este reporte

Cuatro cosas, para que no tenga que leerlo completo:

1. **§3 es el resumen ejecutivo.** Una página: causa raíz, hallazgo principal y qué hacer primero.
2. **Cada hallazgo tiene dos cajas.** La de color arriba dice el riesgo en una línea. La verde abajo dice qué hacer. Si solo lee esas dos por hallazgo, tiene el 80 por ciento.
3. **§9 dice lo que NO se probó**, y por qué. Una evaluación sin límites declarados no es verificable.
4. **§8 documenta un hallazgo que se retiró.** Está a propósito.

---

1. Metadatos

CAMPO	VALOR
Cliente	Grupo Ejemplo, S.A. de C.V. (ficticio)
Objetivo	`grupo-ejemplo.example` y los subdominios que resuelven bajo él
Superficie	34 nombres de host, 19 direcciones IP, 4 proveedores
Tipo	Caja negra externa, autorizada, **no destructiva**
Autorización	Alcance cerrado por escrito, firmado por el cliente
Metodología	iNBest CVP
Estándares	OWASP WSTG v4.2 · OWASP API Security Top 10 2023 · PTES · NIST SP 800-115
Evidencia	58 archivos, índice en §10
Elaborado por	AIHACKERS · iNBest CVP

2. Cómo se ejecutó, y qué cubrió

La evaluación corre una lista de verificación fija. No es discrecional: cada paso deja un archivo de evidencia, y una puerta de calidad impide entregar el reporte si un paso no produjo evidencia que sustente su resultado.

#	PASO	ESTADO	EVIDENCIA
1	Enumeración de subdominios, 5 fuentes independientes	Ejecutado	`all_hostnames_merged.txt`
2	Inventario de servicios desde buscadores de dispositivos	Ejecutado	`shodan_inventory.json`
3	DNS pasivo histórico	Ejecutado	`virustotal.json`
4	Escaneo de puertos y banners	Ejecutado	`nmap_live_ips.txt`
5	Autenticación de correo y DNS	Ejecutado	`email_auth.json`
6	Postura TLS y certificados	Ejecutado	`cert_audit.json`
7	Encabezados de seguridad	Ejecutado	`headers_audit.json`
8	Apropiación de subdominios y registros colgantes	Ejecutado	`takeover.json`
9	Software fuera de soporte	Ejecutado	`stack_eol.json`
10	Revisión de paquetes y mapas de código	Ejecutado	`sourcemap.json`
11	Credenciales en corpus de filtraciones	Ejecutado	`breach_corpus.xlsx`
12	Pastes y repositorios públicos	Ejecutado	`paste_recon.json`
13	Documentación de API expuesta	Ejecutado	`api_docs.json`
14	Matriz de autenticación de endpoints	**Parcial**	Ver §9
15	Escaneo por plantillas de vulnerabilidad	Ejecutado	`nuclei.jsonl`
16	Puerta de calidad de evidencia	**Aprobada**	salida cero

Sobre el paso 1. Se usan cinco fuentes y no una, porque ninguna es completa. En esta evaluación los registros de Certificate Transparency vieron **11 de los 34** nombres, ya que buena parte del patrimonio se sirve bajo certificados comodín. El barrido de resolutor aportó 9 nombres que ninguna otra fuente tenía, y **el mapa de código expuesto aportó uno que ninguna de las cinco encontró** (CRIT-A).

3. Resumen ejecutivo

Riesgo global: Alto.

Causa raíz. Los cuatro hallazgos de mayor severidad comparten un patrón: **artefactos que llegaron a internet sin que nadie decidiera publicarlos.** Un mapa de código dentro de la compilación de producción, cuatro entornos no productivos con nombre público, una consola administrativa en su ruta original, y una redirección que degrada el cifrado. Ninguno es una vulnerabilidad de software. Los cuatro son decisiones de despliegue que nadie revisó.

Hay un segundo patrón, más sutil: **controles que existen pero no aplican.** El DMARC está publicado en modo monitoreo, así que no bloquea nada. La política de contenido es extensa pero permite el envío de formularios a cualquier destino. Ambos leen como protección en una auditoría de configuración y no detienen nada en la práctica.

El hallazgo principal. El portal de socios se publicó con su mapa de código fuente. Ese archivo entregó el código original de 28 archivos, el contrato completo de la API interna, y el nombre de un servidor que **ninguna de las cinco fuentes de enumeración había encontrado**. Un atacante que enumere subdominios de la forma habitual no lo ve; cualquiera que descargue el mapa, sí.

Postura.

SEVERIDAD	CANTIDAD
Crítico	1
Alto	3
Medio	5
Bajo	2
Informativo	1

Qué hacer primero. Los dos primeros son cambios de una línea:

1. Borrar dos archivos del almacenamiento y desactivar su generación en la compilación (CRIT-A).
2. Cambiar el destino de una redirección y publicar un encabezado (HIGH-C).
3. Restablecer 19 identidades y revocar sus sesiones (HIGH-B).

Lo que está bien hecho. La transferencia de zona DNS está correctamente rechazada en los cuatro servidores de nombres. El panel administrativo de la tienda usa un nombre no predeterminado, lo que es buena práctica aplicada deliberadamente. No hay dominios susceptibles de apropiación. Se dice porque un reporte que solo enumera defectos no es una evaluación, es una lista de quejas.

---

4. Cómo se califica la severidad

No se usa CVSS. CVSS puntúa la vulnerabilidad en abstracto; esta rúbrica puntúa **la exposición real del cliente**. Un hallazgo no sube de categoría porque costó trabajo encontrarlo.

Pisos, nunca se califica por debajo:

SITUACIÓN	PISO
Código fuente o secreto legible (mapa de código, `.env` publicado, token embebido)	**Alto**, aunque nunca se use el secreto
Acceso no autenticado a datos personales o de clientes	**Alto**
Software con más de 4 años fuera de soporte, o versión con CVE pre-autenticación	**Alto**

Techos, nunca se califica por encima:

SITUACIÓN	TECHO
Divulgación de versión sin versión vulnerable conocida	**Informativo**
Vulnerabilidad cuya precondición es un compromiso que no se demostró	**Medio**

Ese último techo es el que mantiene honesto el reporte. Si un hallazgo necesita una inyección de script que no se encontró, se califica la exposición real y no la cadena hipotética.

5. Inventario de superficie

HOST	FUNCIÓN	PROVEEDOR	OBSERVADO
`www`	Sitio corporativo	CDN	301 a `http://` (HIGH-C)
`tienda`	Venta en línea	Nube pública	Plataforma de comercio con pagos integrados
`portal`	Portal de socios	Almacenamiento estático + CDN	**Mapa de código público (CRIT-A)**
`api-datos`	API interna	Nube pública	**No enumerable; revelado por el mapa**
`erp`	Integración con ERP	Centro de datos propio	Consola en ruta predeterminada (MED-C)
`soporte`	Portal de soporte	Nube pública	Certificado vencido (MED-D)
`qa`, `dev`, `stage`, `uat`	No productivos	Nube pública	Públicos (MED-A)
`vpn`, `correo`	Acceso remoto y correo	Centro de datos propio	

Cobertura por fuente:

```
FUENTE                              NOMBRES   UNICOS DE ESA FUENTE
Certificate Transparency            11 de 34                     3
Buscador de dispositivos            22 de 34                     6
DNS pasivo historico                18 de 34                     4
Corpus de terceros                  14 de 34                     1
Barrido de resolutor, 557 etiquetas 21 de 34                     9
Mapa de codigo expuesto              1 de 34                     1   <-- CRIT-A
```

6. Hallazgos detallados

Si solo lee dos cosas por hallazgo, lea la caja de color superior y la caja verde inferior.

CRIT-A Mapa de código en producción expone el portal de socios y un servidor no enumerable

ACTIVO `portal.grupo-ejemplo.example` **SEVERIDAD** Crítico **ESTADO** Abierto

EL RIESGO, EN UNA LÍNEA

El código del portal de socios es público, y entrega el mapa completo de una API interna que no aparece en ningún inventario de subdominios.

Qué encontramos. El portal es una aplicación de página única publicada en almacenamiento estático detrás de una red de distribución. Se desplegó con su mapa de código fuente, y el mapa contiene `sourcesContent`, es decir el código original completo y no solo la lista de nombres de archivo.

REPRODUCCIÓN · SOLO LECTURA, SIN AUTENTICACIÓN

```
# 1. El paquete compilado declara su propio mapa en la ultima linea
curl -s https://portal.grupo-ejemplo.example/static/js/main.<hash>.js | tail -c 120

# 2. El mapa responde
curl -sI https://portal.grupo-ejemplo.example/static/js/main.<hash>.js.map
```

RESPUESTA DEL SERVIDOR · LO QUE ENTREGA A CUALQUIERA

```
//# sourceMappingURL=main.<hash>.js.map

HTTP/1.1 200 OK
Content-Type: application/json
Content-Length: 4518621
Cache-Control: public, max-age=31536000

{"version":3,"sources":[...612 rutas...],"sourcesContent":[...codigo original...]}
```

612 fuentes, de las cuales **28 son archivos propios de la aplicación** y el resto son librerías.
Los 28 se recuperan textualmente:

ARCHIVOS PROPIOS RECUPERADOS DEL MAPA

```
services/Service.jsx                components/auth/SignIn.jsx
utils/Constants.js                  components/auth/ForgotPassword.jsx
components/pages/ClientesView.jsx   components/pages/RegistrosView.jsx
components/pages/DetalleView.jsx    components/pages/Uploader.jsx
```

Leer dos de ellos entrega el contrato completo de la API:

CONTRATO DE API, RECUPERADO DE `UTILS/CONSTANTS.JS`

```
export const API_URL = process.env.REACT_APP_API_URL;

export const SIGN_IN_URL       = `${API_URL}/api/v1/sign-in`;
export const RESET_OTP_URL     = `${API_URL}/api/v1/resetPasswordWithOtp`;
export const GET_CLIENTES_URL  = `${API_URL}/api/v1/getClientes`;
export const GET_REGISTROS_URL = `${API_URL}/api/v1/getRegistros`;
export const GET_REPORTE_CSV   = `${API_URL}/api/v1/getReporteCSV`;
export const UPLOAD_URL        = `${API_URL}/api/v1/cargarCatalogo`;
```

`API_URL` es una variable de entorno en el código, pero la compilación de producción la escribe literalmente dentro del paquete:

DIRECCIÓN DEL SERVIDOR, INLINEADA EN EL PAQUETE COMPILADO

```
const bt="https://api-datos.grupo-ejemplo.example",
      vt="".concat(bt,"/api/v1/sign-in")
```

Por qué importa. `api-datos` no apareció en ninguna de las cinco fuentes de enumeración: ni Certificate Transparency, ni el buscador de dispositivos, ni DNS pasivo, ni el corpus de terceros, ni un barrido de diccionario de 557 etiquetas. **El mapa de código es la única razón por la que está**

en este reporte. Un atacante que enumere de la forma habitual no lo encuentra; uno que descargue el mapa, sí.

Los endpoints recuperados incluyen una exportación masiva a CSV y una consulta que recibe el identificador de cliente como parámetro del navegador (ver HIGH-A).

Qué verificamos y qué no. Confirmamos que el mapa responde, que contiene el código original, y que el paquete nombra ese servidor. **No se encontró ninguna credencial activa** en el mapa ni en el paquete: las cadenas que la herramienta marcó como posibles secretos resultaron ser constantes de librerías de terceros al revisarlas una por una. Se dice porque sobrevender un hallazgo no le sirve a nadie. La severidad descansa en la divulgación del código y en el servidor que quedó al descubierto, no en una llave filtrada.

REMEDIACIÓN

1. **Eliminar los archivos `.map` del almacenamiento e invalidar la caché de la red de distribución.**
   Mientras la caché no se invalide, el archivo se sigue sirviendo aunque ya no exista en el origen.
2. Fijar la variable en la compilación de producción, para que no reaparezca en el siguiente despliegue:
   `GENERATE_SOURCEMAP=false`
3. Conservar los mapas donde sí sirven: como artefacto de integración continua, o cargados al servicio de seguimiento de errores, que los consume de forma privada.
4. Tratar `api-datos` como un nombre ya conocido por terceros. Retirarlo del DNS no lo protege: seguiría siendo alcanzable por dirección IP.

HIGH-A La consulta de registros acepta el identificador suministrado por el navegador

ACTIVO `api-datos.grupo-ejemplo.example` **SEVERIDAD** Alto **ESTADO** Abierto

EL RIESGO, EN UNA LÍNEA

Si el servidor no valida a quién pertenece el registro consultado, un socio podría leer los datos de los clientes de otro.

Qué encontramos. El código recuperado en CRIT-A muestra cómo el portal construye la consulta y cómo transporta la sesión.

FRAGMENTO DE `SERVICES/SERVICE.JSX`, RECUPERADO DEL MAPA

```
export async function getRegistros(page, limit, clienteId) {
    return fetch(
        `${GET_REGISTROS_URL}?clienteId=${clienteId}&page=${page}&limit=${limit}`,
        { method: "GET",
          headers: { Authorization: localStorage.getItem("sessionToken") } }
    ).then((r) => r.json());
}
```

Dos debilidades visibles en el código mismo:

- `clienteId` viaja como parámetro del navegador. Que el parámetro venga del cliente no es en sí el defecto; el defecto sería que el servidor no verifique la pertenencia.
- El token de sesión vive en `localStorage`, legible por cualquier script en el origen, y se envía sin prefijo de esquema. Una inyección de script en el portal se convierte en robo completo de sesión en lugar de un defecto contenido.

Por qué importa. Corresponde a **API1:2023, Broken Object Level Authorization** del OWASP API Security Top 10, que encabeza esa lista por frecuencia y por impacto.

Esto es una precondición, no una vulnerabilidad confirmada. Determinar si el servidor valida la pertenencia requiere autenticarse con dos cuentas de socio distintas y verificar si una alcanza los registros de la otra. Eso es prueba autenticada y quedó fuera del alcance acordado. Se documenta porque la divulgación de CRIT-A vuelve públicos el endpoint y el nombre del parámetro, de modo que la precondición pasó de teórica a conocida.

REMEDIACIÓN

1. **Derivar el alcance del socio en el servidor** a partir de la sesión autenticada y verificarlo contra el `clienteId` solicitado en cada llamada. No confiar en el parámetro del navegador.
2. Mover el token a una cookie con los tres atributos, y enviarlo con esquema: `Set-Cookie: session=<token>; HttpOnly; Secure; SameSite=Lax; Path=/ Authorization: Bearer <token>`
3. **Autorizar una prueba autenticada entre dos cuentas** para cerrar la pregunta. Es la verificación pendiente de mayor valor en este reporte, y toma menos de una hora.

HIGH-B Credenciales corporativas en texto plano en corpus públicos

ACTIVO Dominio corporativo **SEVERIDAD** Alto **ESTADO** Abierto

EL RIESGO, EN UNA LÍNEA

19 contraseñas corporativas circulan públicamente en texto plano, y son reutilizables contra el acceso remoto y el correo.

Qué encontramos. 84 registros con correo corporativo en corpus públicos de credenciales. La composición importa más que el total:

COMPOSICIÓN DEL CORPUS, POR ORIGEN

ORIGEN	REGISTROS	EN TEXTO PLANO
Listas de combinacion derivadas de malware	23	19
Broker de datos empresariales	41	0
Filtraciones historicas de sitios de terceros	20	0
TOTAL	84	19

Por qué el número que cuenta es 19. Los 41 registros del broker son información de contacto profesional: nombre, correo, puesto. No contienen credencial. Los 20 de filtraciones históricas traen hash, no contraseña utilizable.

El alcance real de rotación son 19 identidades, no 84. La distinción no es cosmética: pedirle a un equipo de TI que restablezca 84 cuentas cuando 65 no tienen credencial comprometida gasta su tiempo y la credibilidad de quien lo pidió. La siguiente vez que se reporte algo, lo van a pesar distinto.

Sobre la antigüedad. Los corpus de origen mezclan listas antiguas con registros de malware que tampoco son recientes. La afirmación defendible es que **existen credenciales en texto plano en circulación pública**, no que hayan sido robadas recientemente. La antigüedad no las neutraliza: una contraseña sin rotar sigue siendo válida indefinidamente, y la reutilización entre servicios es la norma.

Las contraseñas no se reproducen en este documento ni se usaron contra ningún sistema.

REMEDIACIÓN

1. **Restablecer las 19 identidades con contraseña en claro Y revocar sus sesiones y tokens de actualización.** El restablecimiento por sí solo no invalida una cookie de sesión ya robada: el atacante sigue dentro hasta que la sesión expire por sí misma.
2. Confirmar que el segundo factor está aplicado en todo el tenant, preferentemente resistente a phishing (FIDO2 o certificado) y no OTP, que los registros de malware también capturan.
3. Tratar las 65 restantes como lista de objetivos de phishing conocida: se atiende con concientización dirigida, no con rotación de contraseñas.
4. Establecer monitoreo continuo de estos corpus. Se actualizan de forma permanente.

HIGH-C

Toda visita al sitio transita una vez por HTTP en claro, y no hay HSTS

ACTIVO `www` y los 3 hosts web restantes **SEVERIDAD** Alto **ESTADO** Abierto

EL RIESGO, EN UNA LÍNEA

La redirección canónica baja de HTTPS a HTTP antes de recuperarse, y ningún host indica al navegador que rechace esa degradación, en un dominio que procesa pagos.

REPRODUCCIÓN

```
curl -sI https://www.grupo-ejemplo.example/ | grep -i '^HTTP\|^location'

for h in www tienda portal soporte; do
  echo -n "$h: "
  curl -sI https://$h.grupo-ejemplo.example/ | grep -ci strict-transport-security
done
```

RESPUESTA DEL SERVIDOR

```
HTTP/1.1 301 Moved Permanently
location: http://grupo-ejemplo.example/        <-- degradacion de esquema

www: 0
tienda: 0
portal: 0
soporte: 0
```

La cadena completa es `https://www` → **`http://apex`** → `https://apex`. El navegador recupera el cifrado en el segundo salto, pero la petición ya viajó una vez en claro.

Por qué importa. Un atacante en la ruta de red (una red inalámbrica pública, un salto de proveedor comprometido, un enrutador doméstico) observa ese salto y puede retirar el cifrado del resto de la sesión. HSTS es el control diseñado exactamente para que el navegador se niegue, y no está publicado en ninguno de los cuatro hosts.

REMEDIACIÓN

1. **Corregir el destino de la redirección para que conserve el esquema.** Una línea:
   `nginx return 301 https://grupo-ejemplo.example$request_uri;`
2. Publicar el encabezado en el ápice y en `www`:
   `nginx add_header Strict-Transport-Security "max-age=31536000; includeSubDomains" always;`
   Desplegar `includeSubDomains` **solo después** de confirmar que todos los subdominios sirven HTTPS correctamente, incluidos los no productivos de MED-A.
3. Una vez estable durante algunas semanas, considerar la inclusión en la lista de precarga de HSTS.

MED-A Cuatro entornos no productivos con nombre público

ACTIVO `qa` , `dev` , `stage` , `uat` **SEVERIDAD** Medio **ESTADO** Abierto

EL RIESGO, EN UNA LÍNEA

Los entornos no productivos ejecutan versiones más antiguas, muestran errores detallados y suelen contener copias de datos reales.

ESTADO OBSERVADO DE CADA ENTORNO

HOST	HTTP	SERVIDOR	OBSERVACION
stage	200	<servidor>/1.18.0	version anterior a produccion
dev	200	<servidor>/1.24.0	traza de error de la aplicacion visible
qa	200	<servidor>/1.24.0	sin autenticacion de borde
uat	503	balanceador	sin destinos activos

Tres de los cuatro fueron invisibles a Certificate Transparency y aparecieron solo en el barrido de resolutor. `uat` apunta a un balanceador sin destinos: es costo sin función, y se convierte en riesgo de apropiación el día que ese balanceador se elimine y el registro DNS quede apuntando al vacío.

Por qué importa. `stage` corre una versión de servidor anterior a la de producción, lo que indica que el ciclo de parches no alcanza a los entornos no productivos. Y `dev` devuelve la traza de la aplicación, que revela rutas internas, versiones de librerías y en ocasiones fragmentos de consulta.

REMEDIACIÓN

1. **Colocar los cuatro detrás de VPN o de una lista de direcciones permitidas.** Ninguno necesita ser alcanzable desde internet abierto.
2. Desactivar la visualización de errores de desarrollo en cualquier entorno con DNS público.
3. Incorporar los entornos no productivos al mismo ciclo de parches que producción, o darlos de baja.
4. Eliminar el registro DNS de `uat` y añadir la baja de registros al procedimiento de decomiso.
5. Emitir certificados comodín para no productivo, de modo que los nombres individuales dejen de publicarse en Certificate Transparency.

MED-B La suplantación del dominio en correo no está bloqueada

ACTIVO Dominio corporativo **SEVERIDAD** Medio **ESTADO** Abierto

EL RIESGO, EN UNA LÍNEA

Un correo suplantado en nombre de la empresa se entrega en lugar de rechazarse, frente a socios y clientes que esperan cotizaciones y facturas.

REGISTROS DNS OBSERVADOS

```
$ dig +short TXT grupo-ejemplo.example
"v=spf1 include:_spf.<proveedor-correo> ip4:198.51.100.0/24 ~all"
                                                        ^^^^  softfail

$ dig +short TXT _dmarc.grupo-ejemplo.example
"v=DMARC1; p=none;"
               ^^^^ solo monitoreo, y sin rua= no se recopila nada

DKIM    sin selector encontrado entre los nombres comunes
CAA     ausente
DNSSEC  ausente (sin DNSKEY, sin DS)
```

Por qué importa. Los tres controles fallan de formas distintas y complementarias:

- **SPF termina en `~all`.** Un mensaje de un emisor no listado se marca como sospechoso, no se rechaza. Se sigue entregando.
- **DMARC en `p=none`** indica al receptor que **no tome ninguna acción** sobre los mensajes que fallan. Y sin `rua=` tampoco se recopilan los reportes agregados, que son lo que mostraría quién está suplantando el dominio hoy.
- **Sin DKIM**, buena parte del correo legítimo no tiene firma criptográfica contra la cual alinear.

Sin CAA, cualquier autoridad certificadora pública puede emitir un certificado para el dominio. Es el control más barato del reporte y no está puesto.

Para una empresa cuyo negocio corre sobre cotizaciones, órdenes de compra y facturas, este es el habilitador directo del fraude por correo en su propio nombre, contra sus propios clientes. Es el hallazgo con la ruta más corta a una pérdida económica.

REMEDIACIÓN

1. **Publicar DKIM** para el proveedor de correo y confirmar que alinea con el dominio visible.
2. **Añadir dirección de reportes al DMARC**, leer los agregados durante algunas semanas, y avanzar:
   `_dmarc TXT "v=DMARC1; p=none; rua=mailto:dmarc@grupo-ejemplo.example; pct=100" -> "v=DMARC1; p=quarantine; rua=...; pct=25" (luego 50, luego 100) -> "v=DMARC1; p=reject; rua=..."`
3. **Cambiar SPF a** `-all` una vez confirmado que el inventario de emisores está completo.
4. **Publicar CAA** restringido a las autoridades en uso:
   `@ CAA 0 issue "<ca-en-uso>" @ CAA 0 iodef "mailto:seguridad@grupo-ejemplo.example"`

MED-C Consola administrativa accesible en su ruta predeterminada

ACTIVO `erp.grupo-ejemplo.example` **SEVERIDAD** Medio **ESTADO** Abierto

EL RIESGO, EN UNA LÍNEA

La interfaz de administración de la integración con el ERP responde desde internet abierto, sin restricción por origen.

Qué encontramos. La consola responde y presenta una pantalla de autenticación.

RESPUESTA OBSERVADA

```
HTTP/1.1 200 OK
Content-Type: text/html; charset=UTF-8
Set-Cookie: <sesion>; path=/; HttpOnly
X-Frame-Options: SAMEORIGIN

<title>Panel de administracion</title>
```

Lo que se confirma y lo que no. Se confirma que la consola es alcanzable desde cualquier dirección de internet. **No se intentó autenticar**, ni con las credenciales de HIGH-B ni por ningún otro medio: eso es prueba autenticada y quedó fuera del alcance. No se afirma nada sobre la fortaleza de la autenticación, porque no se probó.

Contraste que vale registrar: el panel administrativo de la tienda **sí** usa una ruta no predeterminada, y devuelve 404 en la ruta original. La práctica correcta ya existe dentro de la organización; solo no se aplicó aquí.

REMEDIACIÓN

1. **Restringir la consola por dirección IP o VPN.** Es el control que cierra el hallazgo: `nginx location /admin { allow 198.51.100.0/24; deny all; }`
2. Cambiar la ruta predeterminada, replicando lo que ya se hizo en la tienda.
3. Exigir segundo factor resistente a phishing sobre esa interfaz.

MED-D

Certificado vencido en el portal de soporte

ACTIVO `soporte.grupo-ejemplo.example` SEVERIDAD Medio ESTADO Abierto

EL RIESGO, EN UNA LÍNEA

El portal que usa el personal de soporte presenta un certificado vencido, lo que entrena al equipo a ignorar advertencias del navegador.

ESTADO DEL CERTIFICADO

```
subject   CN=soporte.grupo-ejemplo.example
issuer    <autoridad certificadora>
notAfter  hace ~4 meses            <-- VENCIDO
cadena    valida, nombre coincide
```

Por qué importa, y no es criptográfico. El cifrado sigue funcionando. El daño es conductual: un equipo acostumbrado a hacer clic en "continuar de todos modos" varias veces al día pierde la capacidad de distinguir esa advertencia de la que genera un ataque real de interceptación. El control de seguridad más caro de una organización es la atención de su gente, y esto la gasta.

REMEDIACIÓN

1. Renovar el certificado y **automatizar la renovación**. Automatizarla es lo que evita la reincidencia; renovarlo a mano garantiza que vuelva a ocurrir.
2. Añadir monitoreo de expiración con alerta a 30 días sobre todos los certificados del patrimonio, no solo este.

MED-E Política de contenido permisiva en la ruta de pago

ACTIVO `tienda.grupo-ejemplo.example` SEVERIDAD Medio ESTADO Abierto

EL RIESGO, EN UNA LÍNEA

La política permite el envío de formularios a cualquier destino, que es justo la directiva que impediría exfiltrar datos de tarjeta.

DIRECTIVAS RELEVANTES DE LA POLÍTICA PUBLICADA

```
script-src       ... 'unsafe-inline' 'unsafe-eval'
default-src      ... 'self' 'unsafe-inline' 'unsafe-eval'
form-action      ... *                        <-- comodin
frame-ancestors  *.<proveedor-extension> 'self'
```

Por qué importa. La política ocupa varios kilobytes y lee como un control robusto. Tres directivas la vacían:

- `'unsafe-inline'` con `'unsafe-eval'` en `script-src` significa que la política **no puede detener código inyectado**, que es la razón principal por la que existe una política de contenido.
- `form-action` **con comodín** permite que un formulario de la página envíe datos a cualquier destino. En una ruta de pago, eso elimina la única directiva que bloquearía a un robador de datos de tarjeta del lado del cliente.
- `frame-ancestors` autoriza a un proveedor externo de extensiones a incrustar la tienda en un marco.

REMEDIACIÓN

1. **Retirar el comodín de** `form-action` y fijarlo a `'self'` más las pasarelas de pago nombradas. Es el cambio de mayor efecto y el más barato de los tres.
2. Retirar el proveedor externo de `frame-ancestors` salvo que una integración activa lo requiera.
3. Trabajar hacia la eliminación de `'unsafe-inline'` mediante nonces por petición. Es el más laborioso, así que conviene secuenciarlo después de los dos anteriores.

LOW-A Divulgación de versiones e infraestructura interna

ACTIVO 4 hosts web SEVERIDAD Bajo ESTADO Abierto

EL RIESGO, EN UNA LÍNEA

Individualmente menor; en conjunto entrega un perfil de objetivo preciso y gratuito.

DIVULGACIÓN OBSERVADA

```
Server: <servidor>/1.24.0 (<distribucion>)     version exacta, 4 hosts
X-Host: <nombre-interno-produccion>            nombre interno del servidor
/version                        HTTP 200       endpoint de version accesible
Content-Security-Policy                        enumera todos los proveedores integrados

AUSENTES en los 4 hosts: Referrer-Policy, Permissions-Policy
```

La política de contenido, además, enumera la totalidad de proveedores externos integrados. No es un defecto en sí, pero un atacante obtiene de un solo encabezado la lista completa de terceros contra los cuales buscar avisos de seguridad.

REMEDIACIÓN

1. Desactivar la publicación de versión y retirar el encabezado interno: `nginx server_tokens off; proxy_hide_header X-Host;`
2. Devolver 404 en el endpoint de versión.
3. Añadir `Referrer-Policy: strict-origin-when-cross-origin` y una `Permissions-Policy` mínima.
4. Depurar la política de contenido a los orígenes efectivamente en uso.

LOW-B Registros DNS que apuntan a hosts que ya no responden

ACTIVO Zona del dominio **SEVERIDAD** Bajo **ESTADO** Abierto

EL RIESGO, EN UNA LÍNEA

Once registros apuntan a direcciones que ya no contestan. Ninguno es apropiable hoy, pero lo sería si la dirección se libera.

Se verificó cada uno de los once contra el registro regional de direcciones. **Cero candidatos a apropiación** al momento de la evaluación: las direcciones siguen asignadas a la organización o a su proveedor.

El riesgo es latente y depende del tiempo. El día que un proveedor libere una de esas direcciones y la reasigne a otro cliente, el registro DNS de la organización apuntará a infraestructura de un tercero.

REMEDIACIÓN

1. Eliminar los once registros.
2. **Incorporar la baja del registro DNS al procedimiento de decomiso de servicios.** Es la medida que evita que se vuelva a acumular.

INFO-A Resultados limpios, registrados

ACTIVO Patrimonio completo **SEVERIDAD** Informativo **ESTADO** Abierto

EL RIESGO, EN UNA LÍNEA

Ninguno. Se documenta para que una evaluación posterior no repita el trabajo, y para dejar constancia de lo que sí está bien.

- **Transferencia de zona DNS rechazada** en los cuatro servidores de nombres. Se probó contra los cuatro, no contra uno.
- **Sin candidatos a apropiación de subdominios**, incluidas las delegaciones a plataformas externas de marketing, que son las que suelen fallar.
- **Panel administrativo de la tienda con nombre no predeterminado**, y 404 en la ruta original.
- **Sin registro de paquetes ni almacén de artefactos expuesto**: 23 nombres candidatos probados, ninguno resuelve.
- **Rutas sensibles de la plataforma correctamente ausentes**: archivos de configuración, directorio de control de versiones y registros de error devuelven 404.
- **Sin detecciones** en los servicios de reputación consultados, y ninguna muestra de malware observada comunicándose con el dominio.

REMEDIACIÓN

1. Sin acción requerida. Mantener, y verificar en la siguiente evaluación que sigue así.

7. Narrativa: cómo un archivo llevó a un servidor invisible

Vale exponer la cadena completa, porque explica por qué el hallazgo principal no lo produce un escáner.

Paso 1. El barrido de resolutor encuentra `portal`, un nombre que Certificate Transparency no tenía porque el patrimonio usa certificados comodín.

Paso 2. El portal resulta ser una aplicación de página única servida desde almacenamiento estático. Ese patrón es la señal: las compilaciones de ese tipo de aplicación generan un mapa de código, y con frecuencia se publica por omisión.

Paso 3. Se solicita el mapa. Responde 200, con 4.5 MB.

Paso 4. El mapa incluye `sourcesContent`. No es una lista de nombres, es el código original.

Paso 5. Dos archivos del código entregan el contrato completo de la API. La compilación, además, escribe literalmente la dirección del servidor de respaldo.

Paso 6. Ese servidor no estaba en ninguna de las cinco fuentes de enumeración.

Ningún paso requirió una técnica ofensiva. Se pidió un archivo y se leyó. Lo que cambia el resultado no es la sofisticación, es haber sabido que había que pedirlo.

---

8. Un hallazgo retirado, y por qué se documenta

La primera pasada reportó los servidores de nombres como resolutores abiertos a recursión, un hallazgo Medio con riesgo de amplificación.

Antes de publicarlo se ejecutó la prueba de control obligatoria: la misma consulta contra una dirección **fuera del alcance y que no pertenece al cliente**.

PRUEBA DE CONTROL QUE INVALIDÓ EL HALLAZGO

```
$ dig @<servidor-del-cliente>  ejemplo-externo.test  +short
  respuesta recursiva   -> aparente resolutor abierto

$ dig @192.0.2.1              ejemplo-externo.test  +short   # direccion de control,
  respuesta recursiva   -> MISMO RESULTADO                   # no pertenece al cliente
```

Una dirección de documentación que no aloja ningún servidor no puede resolver nada. Que devolviera el mismo resultado demuestra que la medición estaba contaminada en el punto de origen, no que el cliente tuviera un resolutor abierto.

El hallazgo se retiró por completo, y esta sección es la constancia.

Se incluye en una muestra comercial a propósito. Un proveedor que nunca retira nada no está siendo riguroso, está siendo conveniente. Y la disciplina que obliga a borrar este es la misma que hace creíbles los otros doce.

---

9. Lo que esta evaluación no cubrió

Se declara porque una evaluación sin límites declarados no es verificable.

Fuera de alcance por acuerdo:

- **Pruebas autenticadas.** Nada detrás de un inicio de sesión. Es lo que HIGH-A necesita para cerrarse, y lo que impidió evaluar la fortaleza de la autenticación en MED-C.
- **Validación de explotación.** Ningún hallazgo se llevó más allá de confirmar que la condición existe. No se accedió a datos ni se escalaron privilegios.
- **Acciones que modifiquen estado**, pruebas de carga y denegación de servicio.
- **Ingeniería social** y phishing contra el personal.
- **Revisión de código fuente**, más allá de lo que el mapa expuesto entregó por sí solo.
- **Activos de terceros**, incluidos los proveedores que la política de contenido nombra. Sus vulnerabilidades se reportan al cliente; corregirlas corresponde al proveedor.

Límites de la recolección, registrados como desconocidos y no como limpios:

- El paso 14, matriz de autenticación de endpoints, se ejecutó **parcialmente**: cubre los endpoints alcanzables sin credencial. La parte que requiere credencial quedó pendiente.
- El corpus de credenciales entrega un subconjunto del total disponible. **Las cifras son un piso.**
- Certificate Transparency vio 11 de 34 nombres. Las otras cuatro fuentes cubrieron la diferencia, y por eso se usan cinco y no una.

---

10. Índice de evidencia

58 archivos respaldan este reporte. Cada hallazgo cita el suyo.

ARCHIVO	CONTENIDO
`all_hostnames_merged.txt`	34 nombres consolidados de 5 fuentes
`shodan_inventory.json`	Inventario de servicios por dirección
`nmap_live_ips.txt`	Puertos y banners de las 19 direcciones
`headers_audit.json`	Encabezados de seguridad por host
`cert_audit.json`	Vigencia, cadena y coincidencia de nombre
`email_auth.json`	Matriz SPF, DKIM, DMARC, DNSSEC, CAA
`sourcemap.json`	Mapas alcanzables y fuentes reconstruibles
`main.<hash>.js.map`	El mapa expuesto, 4.5 MB
`breach_corpus.xlsx`	Corpus de credenciales, desglosado por origen
`takeover.json`	Verificación de los 11 registros colgantes
`nuclei.jsonl`	Salida del escaneo por plantillas
`dns_control_test.txt`	La prueba que retiró el hallazgo de §8

About aihackers.inbest.cloud

This assessment service is delivered by **AIHACKERS**, an AI-driven offensive security practice, through **aihackers.inbest.cloud**, the iNBest CVP (Customer Vulnerability Program) platform. CVP runs AI-assisted, Claude-powered red-team engagements scoped to each customer's authorized surfaces and delivers reproducible, severity-ranked findings with verbatim evidence and concrete remediation.

AIHACKERS · Seguridad ofensiva impulsada por IA · Metodología CVP · Desarrollado con Claude (Anthropic)
aihackers.inbest.cloud · cybersecurity@inbest.cloud