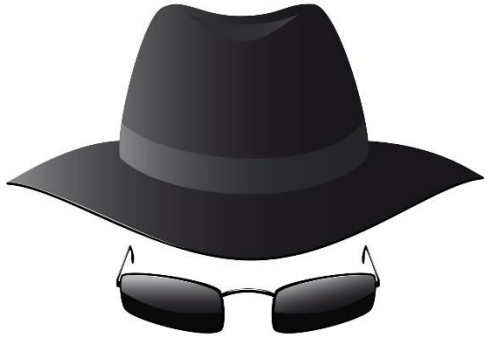


HACKING 101

Técnicas básicas de ataque y Defensa

INVERSIONES INTELECTUALES

¿ Quien ?

Jorge Alejandro Cajas Mota

jac.mota@gmail.com
@cajasmota
https://github.com/Motojo/Slides

¿ Quien ?

INVERSIONES INTELECTUALES

DROIDS4DROIDS

Developers

PE

GT

Google I/O Extended Guatemala

Java day17

COECYS 2017

ORACLE CODE

Contenido

Temas

- ¿Hacker?
  - Definición
  - Tipos
- ¿Por donde empiezo?
  - Herramientas
  - Reconocimiento
  - Vulnerabilidades
  - Pruebas
  - ¡ Lotería !
- Vulnerabilidades comunes
  - XSS
  - SQL Injection
  - Web Services
  - Frontend validations
- Bonus

¿Hacker?

Definición

- Es un Experto en una Ciencia
- Le apasiona el conocimiento
- Se divierte aprendiendo cosas nuevas
- Le gustan los retos difíciles.
- Es revolucionario, comparte su conocimiento

- NO es un pirata informático
- NO es un criminal
- NO es un intruso en sistemas ajenos

¿Hacker?

Tipos

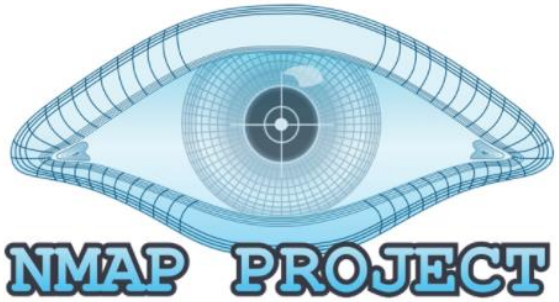
¿Hacker?

Tipos

¿Por donde empiezo?

Herramientas

Webscarab



¿Por donde empiezo?

Herramientas

Google

¿Por donde empiezo?

Reconocimiento

- ¿Quién es la Víctima?
- ¿Qué protocolos de comunicación utiliza?
- ¿Están usando algún framework?
- ¿Tienen alguna vulnerabilidad común?
- ¿Qué espero obtener?

ANONIMATO

¿Por donde empiezo?

Vulnerabilidades

- OWASP Vulnerabilities List
- Exploit DB
- Mala Configuración
- Malas practicas de programación
- Ingeniería Social
- Exposición de datos sensibles

¿Por donde empiezo?

Pruebas

- No dejemos un rastro, puede ser problemático.
- No violemos la ley, también es problemático.
- Usemos las herramientas adecuadas
- Muchas pruebas se pueden hacer 1 sola vez

ANONIMATO

¿Por donde empiezo?

¡ Lotería !

Vulnerabilidades Comunes

XSS

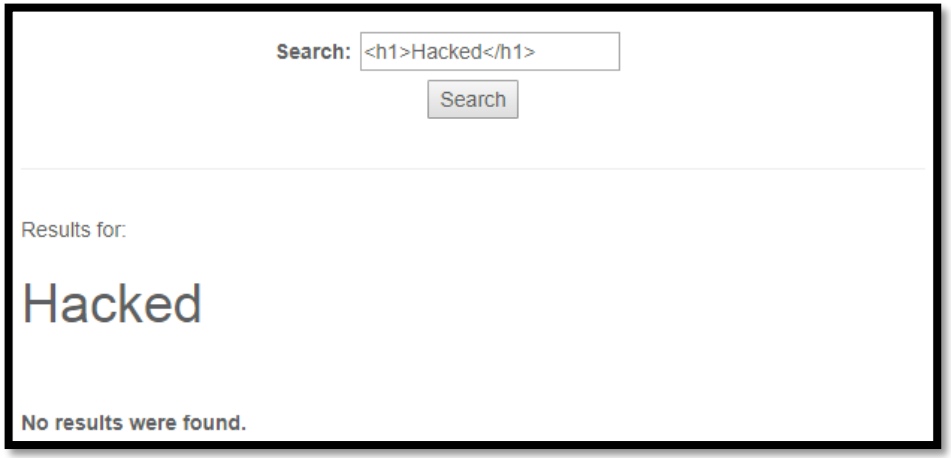
Cross Site Scripting:

Es un vector de ataque que intenta inyectar código malicioso en nuestros sitios web.

Normalmente se produce por malas prácticas de programación o desconocimiento del vector de ataque.

Vulnerabilidades Comunes

XSS



Vulnerabilidades Comunes

XSS

- Nunca confiemos en la Data que escribe el usuario.
- Escapemos los caracteres, ejemplo “<“ → “<”
- Nunca renderizemos la data del usuario directamente

```
echo “Results for" . $_GET[‘search'];

echo “Results for" . Htmlspecialchars($_GET[‘search']);
```

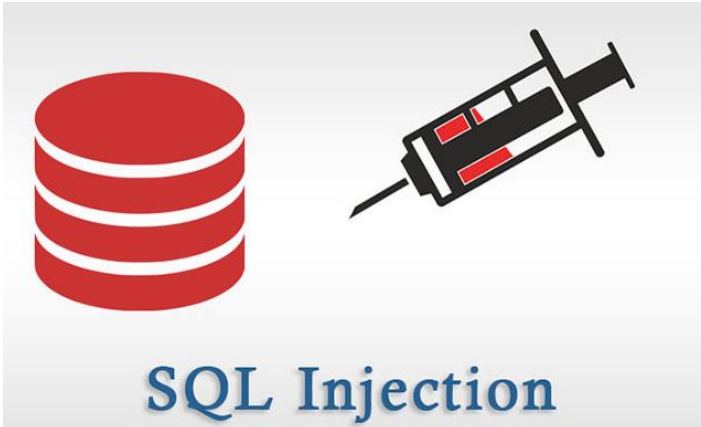
Vulnerabilidades Comunes

SQL Injection

SQL Injection:

Es un vector de ataque que intenta inyectar código SQL en nuestras consultas del servidor hacia la Base de Datos para obtener más información de la que se le permite o destruir nuestros datos

Normalmente se produce por malas prácticas de programación o desconocimiento del vector de ataque.



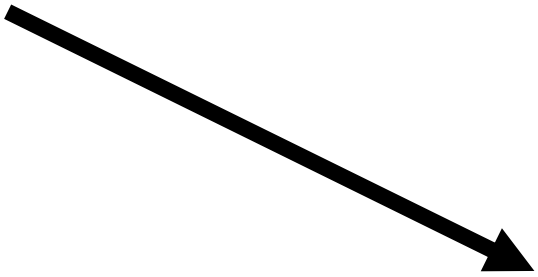
Vulnerabilidades Comunes

SQL Injection

Enter your last name: Smith Go!

```
SELECT * FROM user_data WHERE last_name = ?
```

USERID	FIRST_NAME	LAST_NAME	CC_NUMBER	CC_TYPE	COOKIE	LOGIN_COUNT
102	John	Smith	2435600002222	MC		0
102	John	Smith	4352209902222	AMEX		0



Enter your last name: Smith' OR 1=1 -- Go!

```
SELECT * FROM user_data WHERE last_name = 'Smith' OR 1=1 --'
```

USERID	FIRST_NAME	LAST_NAME	CC_NUMBER	CC_TYPE	COOKIE	LOGIN_COU
101	Joe	Snow	987654321	VISA		0
101	Joe	Snow	2234200065411	MC		0
102	John	Smith	2435600002222	MC		0
102	John	Smith	4352209902222	AMEX		0

Vulnerabilidades Comunes

SQL Injection

- Nunca confiemos en la Data que escribe el usuario.
- Sanitizemos nuestros campos
- Usemos parámetros de SQL en lugar de concatenar la data en un solo String.

```
// prepare and bind
$stmt = $conn->prepare("INSERT INTO MyGuests (firstname, lastname, email) VALUES (?, ?, ?)");
$stmt->bind_param("sss", $firstname, $lastname, $email);
```

Vulnerabilidades Comunes

Web Services

Autenticación

Estamos en la era de los Servicios WEB, ya sean SOAP o REST

Si no somos cuidadosos con nuestros servicios podemos exponer más datos de los deseados, por lo que debemos de manejar protocolos de autenticación y autorización.

No debemos publicar o exponer a internet más de la cuenta

Vulnerabilidades Comunes

Web Services

Enumeración

Los ataques de enumeración se pueden dar de distintas formas y con distinto objetivo.

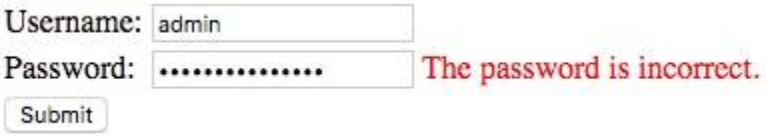
En cualquiera de los casos, el atacante desea obtener un listado de recursos, usuarios o datos en general apoyándose ya sea en mensajes de error muy descriptivos o Web Services sin la verificación adecuada.

Vulnerabilidades Comunes

Web Services

Enumeración de Usuarios

Uno de los errores más comunes es dar más información de la cuenta en los mensajes de error, por ejemplo en login o registro, que el atacante sepa cuales cuentas ya existen, o son invalidas, etc.



Vulnerabilidades Comunes

Web Services

Enumeración de Usuarios

Vulnerabilidades Comunes

Web Services

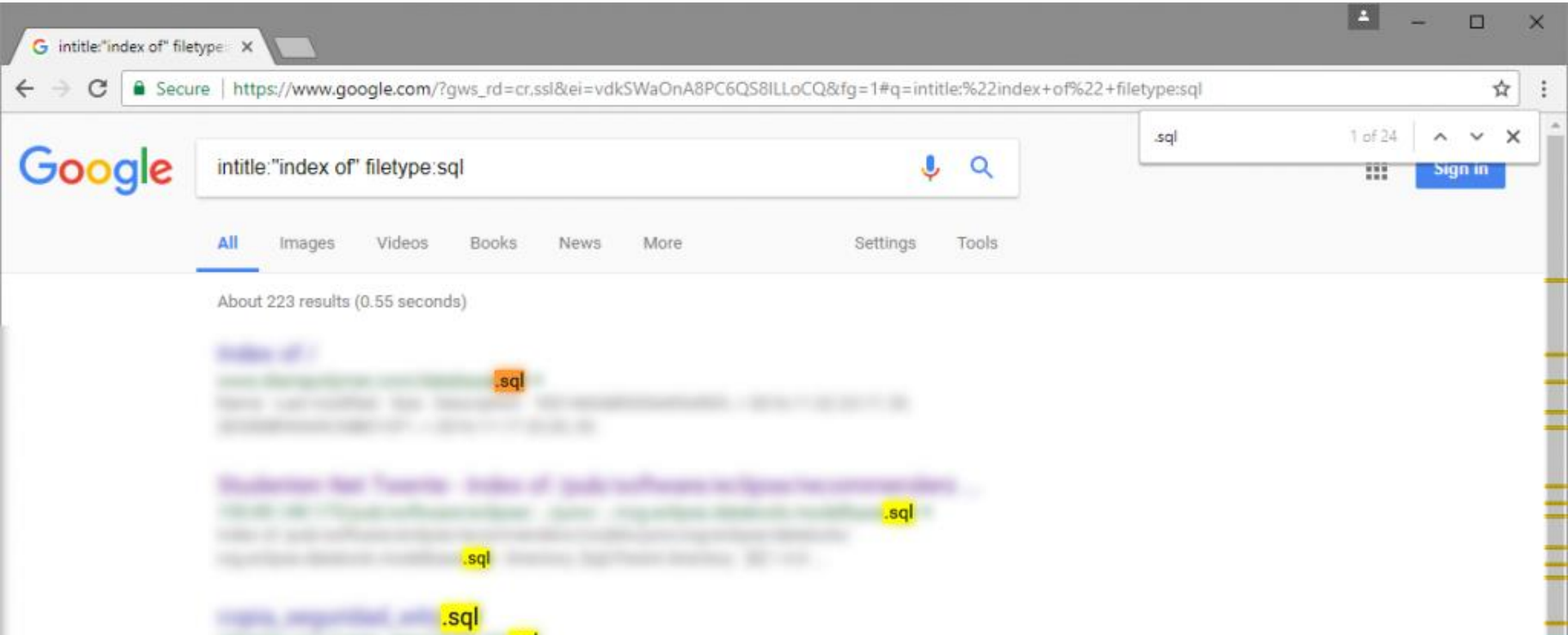
Enumeración de Recursos y directorios

Esto se da por malas configuraciones en nuestros servidores, que permiten al atacante hacer un escaneo de nuestro servidor encontrando directorios y recursos (posiblemente vulnerables), y es uno de los ataques más exitosos utilizando Google Hacking

Vulnerabilidades Comunes

Web Services

Enumeración de Recursos y directorios



Vulnerabilidades Comunes

Web Services

Enumeración de datos:

Esta se da cuando nuestros servicios no ejecutan las validaciones adecuadas en las consultas o modificaciones a nuestra base de datos.

La forma común de los Web services con esta vulnerabilidad es:

```
http://server.com/user/{id}
```

```
http://server.com/documents/{id}
```

Vulnerabilidades Comunes

Frontend Validations

- Recordemos que nuestro código del Frontend está en manos del cliente… y un posible atacante.
- Validaciones de autenticación y autorización siempre deben de complementarse del lado del servidor.

Vulnerabilidades Comunes

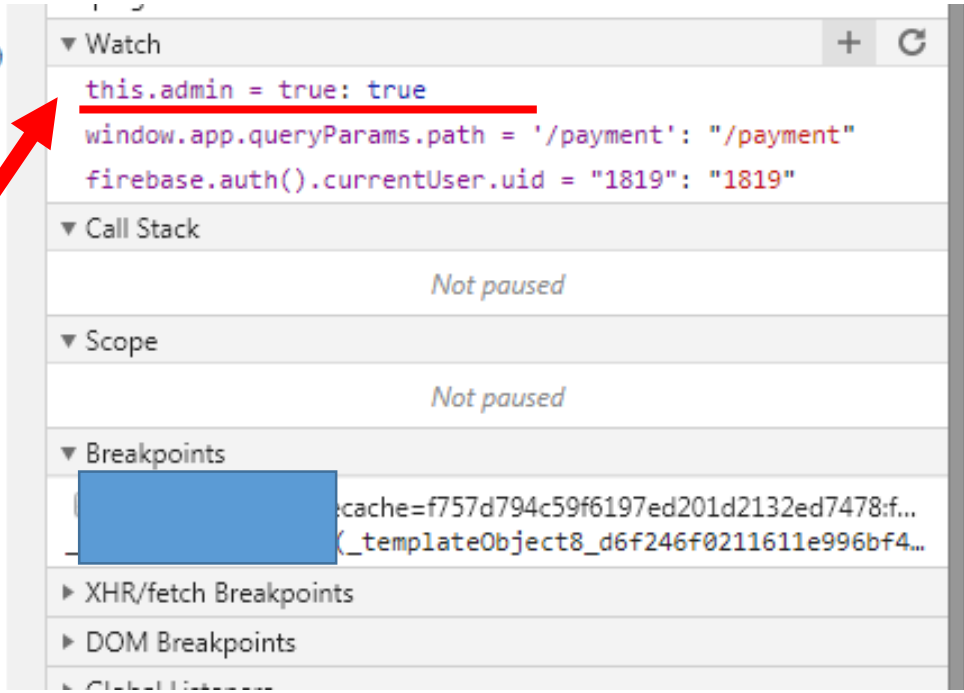
Frontend Validations

```
301         babelHelpers.classCallCheck(this, [redacted]
302         _this11 = babelHelpers.possibleConstructorReturn(this, babelHelpers.getPrototypeOf([redacted]).call(this))
303         _this11.admin = !1;
304         return _this11
305     }
306     babelHelpers.createClass[redacted] [{
307         key: "render",
308         value: function render() {
309             return (0,
310                 [redacted]tml$1)(_templateObject8_d6f246f0211611e996bf4db44374c7b8([redacted], this.admin ? (0,
311                 [redacted]tml$1)(_templateObject9_d6f246f0211611e996bf4db44374c7b8()) [redacted]
312         }
313     }, {
314         key: "firstUpdated",
315         va[redacted]
316             if ("/payment" == window.app.queryParams.path)
317                 this.admin = "1819" == firebase.auth().currentUser.uid
318         }
319     }], [{
320         key: "properties",
321         get: function get() {
322             return {
```

Vulnerabilidades Comunes

Frontend Validations

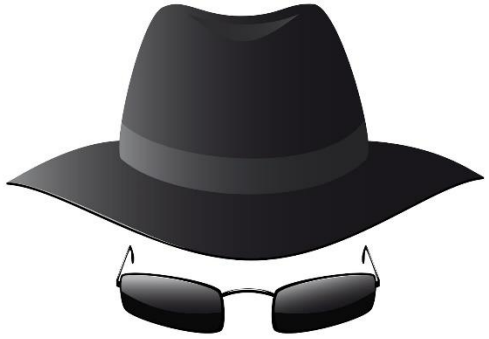
```
iew);
ctorReturn(this, babelHelpers.getPrototypeOf [redacted] call(this))

ect8_d6f246f0211611e996bf4db44374c7b8( , this.admin ? (0,
9_d6f246f0211611e996bf4db44374c7b8()) : "")

roperties) {
Params.path)
e.auth().currentUser.uid
```



Bonus

WebGoat

https://github.com/WebGoat



HACKING 101

¿Preguntas?

INVERSIONES INTELECTUALES

¡ Muchas Gracias !

Jorge Alejandro Cajas Mota

jac.mota@gmail.com
@cajasmota
https://github.com/Motojo/Slides