

ESTA SEMANA

 12

SOURCE CODE VPN login portal of the Argentina National

mrdump



3
REP

0
LIKES

DarkForums Members

 Member

 Joined:

Jun 2025

 Threads:

3

 Posts:

11

 Credits:

3.00 [+]

1 DAY

36 minutes ago



CONSEJO DE LA MAGISTRATURA
PODER JUDICIAL DE LA NACIÓN

Successfully breached the VPN login portal of the Argentina National Judiciary.Full access to the internal network has been obtained.

Check out the full details on our official channel:
 <https://t.me/+VPtUaXgFzmY5ZGFk>

PM

★ RATE

ADD FEEDBACK

 LIKE

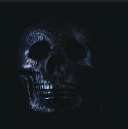
 REPORT

1

OBRAS SOCIALES 2023

Pages (2): 1 2 Next »

ARGENTINE NATIONAL SOCIAL WORKS DATABASE LEAK
by blackmind05 - Monday June 19, 2023 at 06:11 PM



blackmind05

MVP User

Posts: 9

Threads: 9

Joined: Jun 2023

06-19-2023, 06:11 PM

ARGENTINE NATIONAL SOCIAL WORKS DATABASE LEAK 20.471.935 LINES

SAMPLE:

TIPODOC,NUMDOC,NOMBRE,SEXO,NACDIA,NACMES,NACANO,CP,PROVINCIA,CUIL,OBRASOCDU,23813155,RAMOS ISAAC RAFAEL,M,20,05,1974,4612,Jujuy,20238131554,000109 DU,26675186,MAMANI NATALIA MA
BEL ,F,13,06,1978,4612,julio,20238131554,000109

Hidden Content

You must register or login to view this content.

TELEGRAMA: @DATALEAKSALE05
CANAL: @ blackmind05

2

OBRAS SOCIALES 2025

42

[DUMP] UPCNBA - Argentina - 31,000+ People

aero



52 minutes ago

#1



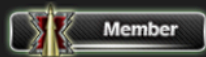
7

REP

0

LIKES

DarkForums Members



Joined: May 2025

Threads: 9

Posts: 10

Credits: 19.00 [+]

3 WEEKS

These files contain full names, emails, phone numbers, addresses, banking information, birthdates, ID documents, and family relationships per person.

The dump was made using access to tu.upcnba.org, with data from over 31,000 individuals, not including the relatives listed within each .json. All the data is included in a .tar file, along with a Flask-based API in Python to run locally.

Read the README.txt inside the uploaded .rar to better understand how it works.

Preview Image - <https://ibb.co/PZN25Nwh>

Preview .Tar Image - <https://ibb.co/nqhkww73v>

Full Download Link - <https://gofile.io/d/Ys9o9g>

Full Download Link (Mirror) - <https://gofile.io/d/eBkTxq>

Contact - t.me/fuerzafederal or t.me/nawzyn

More content on t.me/aeropuertualog

PM RATE ADD FEEDBACK

LIKE

REPORT

EN PARAGUAY

ámbito • Mundo • Bitcoin

9 de junio 2025 - 15:04

"Paraguay legaliza el Bitcoin": hackearon al presidente Santiago Peña y difundieron un falso comunicado sobre la criptomoneda



La cuenta de X del presidente de Paraguay, Santiago Peña, fue hackeada y utilizada para difundir una falsa adopción de Bitcoin como moneda oficial. El gobierno desmintió el mensaje y anunció una investigación.



ESCUCHAR EL RESUMEN DE LA NOTA

00:00

powerbeans

00:39

nota

A07:2021: RESPUESTAS Y ERRORES

- Un ataque siempre comienza con una fase de reconocimiento en la que el atacante intentará recopilar la mayor cantidad de información técnica (a menudo, propiedades de nombre y versión) sobre el objetivo, como el servidor de aplicaciones, los marcos, las bibliotecas, etc.
- Los mensajes de error implementados incorrectamente en el caso de la funcionalidad de autenticación se pueden utilizar para fines de enumeración de ID de usuario y contraseña.
- Una aplicación debe responder (tanto HTTP como HTML) de manera genérica.

A07:2021: RESPUESTAS Y ERRORES

- Mediante cualquiera de los mecanismos de autenticación (inicio de sesión, restablecimiento de contraseña o recuperación de contraseña), una aplicación debe responder con un mensaje de error genérico independientemente de si:
 - El ID de usuario o la contraseña eran incorrectos.
 - La cuenta no existe.
 - La cuenta está bloqueada o deshabilitada.
 - También se debe tener en cuenta la función de registro de cuenta, y se puede aplicar el mismo enfoque de mensaje de error genérico con respecto al caso en el que existe el usuario.

A07:2021: RESPUESTAS Y ERRORES

- El objetivo es evitar la creación de un factor de discrepancia, lo que permite a un atacante montar una acción de enumeración de usuarios contra la aplicación.
- Es interesante notar que la propia lógica empresarial puede traer un factor de discrepancia relacionado con el tiempo de procesamiento necesario.
- De hecho, dependiendo de la implementación, el tiempo de procesamiento puede ser significativamente diferente según el caso (éxito o fracaso), lo que permite a un atacante montar un ataque Time-based (delta de algunos segundos, por ejemplo).

A07:2021: RESPUESTAS Y ERRORES

- Mensajes login incorrectos:
 - "Login for User foo: invalid password."
 - "Login failed, invalid user ID."
 - "Login failed; account disabled."
 - "Login failed; this user is not active."
- Mensajes login correctos:
 - "Login failed; Invalid user ID or password."

A07:2021: RESPUESTAS Y ERRORES

- Mensajes password recovery incorrectos:
 - "We just sent you a password reset link."
 - "This email address doesn't exist in our database."
- Mensajes password recovery correctos:
 - "If that email address is in our database, we will send you an email to reset your password."

A07:2021: RESPUESTAS Y ERRORES

- Mensajes Account creation incorrectos:
 - "This user ID is already in use."
 - "Welcome! You have signed up successfully."
- Mensajes Account creation correctos:
 - "A link to activate your account has been emailed to the address provided."

A07:2021: RESPUESTAS Y ERRORES

- La aplicación puede devolver un código de error HTTP diferente según la respuesta al intento de autenticación.
- Puede responder con un 200 para un resultado positivo y un 403 para un resultado negativo.
- Aunque se muestra una página de error genérica a un usuario, el código de respuesta HTTP puede diferir, lo que puede filtrar información sobre si la cuenta es válida o no.
- La divulgación de errores también se puede utilizar como factor de discrepancia.
- [Error Handling Cheat Sheet](#)

A07:2021: ATAQUES AUTOMÁTICOS

- Hay varios tipos diferentes de ataques automatizados que los atacantes pueden utilizar para intentar comprometer las cuentas de los usuarios.
- Los tipos más comunes se enumeran a continuación:
 - Brute Force : varias contraseñas de un diccionario u otra fuente en una sola cuenta.
 - Credential Stuffing: Prueba de pares de nombre de usuario / contraseña obtenidos de la infracción de otro sitio.
 - Password Spraying: Prueba de una única contraseña débil contra un gran número de cuentas diferentes.

A07:2021 - BRUTE FORCE

- Un ataque de fuerza bruta puede manifestarse de muchas formas diferentes, pero principalmente consiste en que un atacante configure valores predeterminados, realice solicitudes a un servidor utilizando esos valores y luego analice la respuesta.
- Un atacante puede usar un ataque de diccionario o un ataque de fuerza bruta tradicional (con determinadas clases de caracteres, por ejemplo: alfanumérico, especial, sensible a mayúsculas y minúsculas).
- Considerando el método dado, el número de intentos, la eficiencia del sistema que realiza el ataque y la eficiencia estimada del sistema que es atacado, el atacante puede calcular aproximadamente cuánto tiempo tomará enviar todos los valores predeterminados elegidos.

A07:2021 - BRUTE FORCE: WORDLIST

- El bruteforce clásico de basa en la combinación de caracteres hasta llegar al correcto.
- Se debe limitar o contextualizar este tipo de ataque para que el conjunto de valores no sea muy grande
- Ej:
 - DNI: 8 dígitos [0-9]{8}
 - fibertel: (014|004)[0-9]{7}

A07:2021 - BRUTE FORCE: CRUNCH

- Crunch es un generador de wordlist, que permite generar combinaciones y permutaciones de caracteres.
- Ejemplo para generar combinaciones de 4 números y 4 letras minúsculas.

```
crunch 8 8 -t %%%%@@@@@
```

Crunch will now generate the following amount of data:

41127840000 bytes 39222 MB 38 GB 0 TB 0 PB Crunch will now generate the following number of lines: 4.569.760.000

A07:2021 - BRUTE FORCE: DICCIONARIOS

- Los ataques de fuerza bruta por diccionario se realizan atacando con una lista predefinida de credenciales.
- Por ejemplo:
 - Passwords
 - Usernames
- Combinando estos usuarios y contraseñas podemos generar un ataque de diccionario de millones de combinaciones.

A07:2021 - BRUTE FORCE: HYDRA

- **THC-Hydra** es una herramienta para realizar bruteforcing
- cuenta de base con más de 30 protocolos compatibles (sistemas operativos, webs, bases de datos, etc)
- viene por defecto en Kali linux

A07:2021 - BRUTE FORCE: HYDRA

- Demo DVWA sobre docker hydra instalado local:

```
docker run --rm --name dvwa -d -p 80:80 vulnerables/web-dvwa

hydra -L top-username-shortlist.txt \
-P darkweb2017-top100.txt \
-f -V -s \
80 localhost http-post-form \
"/login.php:username=^USER^&password=^PASS^&Login=Login:Login f
```

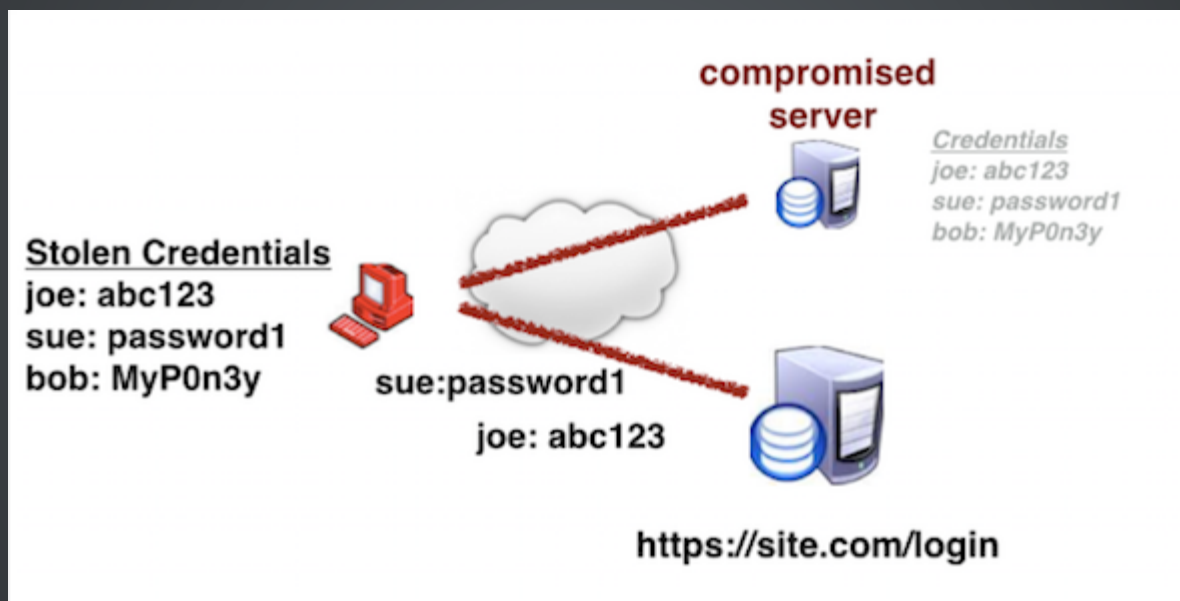
A07:2021 - CREDENTIAL STUFFING

- **Credential stuffing** es la inyección automática de pares de credenciales robadas de usuario / contraseña para obtener acceso fraudulento a las cuentas de los usuarios.
- Este es un subconjunto de la categoría de ataque de fuerza bruta:
 - una gran cantidad de credenciales se ingresan automáticamente en los sitios web hasta que potencialmente coinciden con una cuenta existente, que el atacante puede luego secuestrar para sus propios fines.

CREDENTIAL STUFFING: ANATOMÍA

- El atacante adquiere nombres de usuario y contraseñas desde algun leak o un sitio vulnerable.
- El atacante verifica las cuentas para probar las credenciales robadas en muchos sitios web (por ejemplo, sitios de redes sociales o mercados en línea).
- Los inicios de sesión exitosos (generalmente 0,1-0,2% del total de intentos de inicio de sesión) permiten al atacante hacerse cargo de la cuenta que coincide con las credenciales robadas.
- El atacante drena las cuentas robadas de valor, números de tarjetas de crédito y otra información de identificación personal.
- El atacante también puede usar la información de la cuenta en el futuro para otros propósitos nefastos (por ejemplo, para enviar spam o crear más transacciones).

CREDENTIAL STUFFING: DIAGRAMA



CREDENTIAL STUFFING: EJEMPLO

- A continuación se muestran extractos tomados de publicaciones que analizan infracciones a gran escala.
- La evidencia respalda que estas infracciones fueron el resultado del uso de Credential stuffing.

CREDENTIAL STUFFING: EJEMPLO

- Sony, Breach de 2011: "Deseo destacar que dos tercios de los usuarios cuyos datos estaban tanto en el conjunto de datos de Sony como en el breach de Gawker a principios de este año utilizaron la misma contraseña para cada sistema". Fuente: [Agile Bits](#) - Fuente: [Wired](#)
- Yahoo, 2012 Breach: "¿Qué tienen en común Sony y Yahoo!? ¡Contraseñas! ". Fuente: [Troy Hunt](#).
- Dropbox, Breach de 2012: "Los nombres de usuario y las contraseñas a los que se hace referencia en estos artículos fueron robados de servicios no relacionados, no de Dropbox. Luego, los atacantes utilizaron estas credenciales robadas para intentar iniciar sesión en sitios de Internet, incluido Dropbox ". Fuente: [Dropbox](#)
- Esta cadena de eventos conecta a Sony, Yahoo y Dropbox.

CREDENTIAL STUFFING: REFERENCIAS.

- Credential Stuffing Prevention Cheat Sheet
- Ramification of Credential Stuffing
- What Happens to Stolen Data After a Breach?
- How Third Party Password Breaches Put Your Website at Risk

ÚLTIMO PWNED WEBSITES IN HAVEIBEENPWNED

<https://haveibeenpwned.com/PwnedWebsites>

¿QUÉ SON LOS STEALER LOGS?

- Son archivos generados por un tipo específico de malware conocido como "Stealer", el "ladrón de información". Este malware está diseñado para infiltrarse en sistemas con un objetivo claro: obtener información sensible del usuario.
- Una vez que un Stealer se instala en un dispositivo, comienza a recopilar una amplia gama de datos (inicio de sesión, detalles de tarjetas de crédito, historiales de navegación, cookies, y cualquier otra información).
- Todos estos datos recopilados se almacenan en los Stealer logs.

DEMO

A07:2021: PROTECCIÓN ATAQUES AUTOMÁTICOS

- Se pueden implementar diferentes mecanismos de protección para protegerse contra estos ataques.
- En muchos casos, estas defensas no brindan una protección completa, pero cuando varias de ellas se implementan con un enfoque de defensa en profundidad, se puede lograr un nivel razonable de protección.

A07:2021: MFA

- La autenticación multifactor (MFA) es, con mucho, la mejor defensa contra la mayoría de los ataques relacionados con contraseñas, incluidos los ataques de fuerza bruta, y el análisis sugiere que habría detenido el 99,9% de las amenazas a las cuentas.
- Como tal, debe implementarse siempre que sea posible; sin embargo, dependiendo de la audiencia de la aplicación, puede que no sea práctico o factible hacer cumplir el uso de MFA.
- [Multifactor Authentication Cheat Sheet](#)

A07:2021: BLOQUEO DE CUENTA

- La protección más común contra estos ataques es implementar el bloqueo de la cuenta, que evita más intentos de inicio de sesión durante un período después de una cierta cantidad de inicios de sesión fallidos.
- El contador de inicios de sesión fallidos debe asociarse con la cuenta en sí, en lugar de la dirección IP de origen, para evitar que un atacante realice intentos de inicio de sesión desde una gran cantidad de direcciones IP diferentes.

A07:2021: BLOQUEO DE CUENTA

- Hay una serie de factores diferentes que deben tenerse en cuenta al implementar una política de bloqueo de cuenta para encontrar un equilibrio entre seguridad y usabilidad:
 - El número de intentos fallidos antes de que se bloquee la cuenta (umbral de bloqueo).
 - El período de tiempo en el que deben producirse estos intentos (ventana de observación).
 - Cuánto tiempo está bloqueada la cuenta (duración del bloqueo).

A07:2021: BLOQUEO DE CUENTA

- En lugar de implementar una duración de bloqueo fija (por ejemplo, diez minutos), algunas aplicaciones utilizan un bloqueo exponencial, donde la duración del bloqueo comienza como un período muy corto (por ejemplo, un segundo), pero se duplica después de cada intento fallido de inicio de sesión.

A07:2021: BLOQUEO DE CUENTA

- Al diseñar un sistema de bloqueo de cuentas, se debe tener cuidado para evitar que se use para causar una denegación de servicio al bloquear las cuentas de otros usuarios.
- Una forma de hacerlo es permitir que el usuario de la función de contraseña olvidada inicie sesión, incluso si la cuenta está bloqueada.

A07:2021: CAPTCHA

- El uso de un CAPTCHA eficaz puede ayudar a prevenir los intentos de inicio de sesión automatizados en las cuentas.
- Sin embargo, muchas implementaciones de CAPTCHA tienen debilidades que permiten resolverlas mediante técnicas automatizadas o pueden ser subcontratadas a servicios que pueden resolverlas.
- Como tal, el uso de CAPTCHA debe verse como un control de defensa en profundidad para hacer que los ataques de fuerza bruta consuman más tiempo y sean más costosos, en lugar de una medida preventiva.
- Puede ser más fácil requerir que se resuelva un CAPTCHA solo después de una pequeña cantidad de intentos fallidos de inicio de sesión, en lugar de requerirlo desde el primer inicio de sesión.

A07:2021: USO DE PROTOCOLOS DE AUTENTICACIÓN QUE NO REQUIEREN CONTRASEÑA

A07:2021: OAUTH

- Open Authorization (OAuth) es un protocolo que permite a una aplicación autenticarse frente a un servidor como usuario, sin requerir contraseñas ni ningún servidor de terceros que actúe como proveedor de identidad.
- Utiliza un token generado por el servidor y proporciona cómo deben ocurrir los más los flujos de autorización, de modo que un cliente, como una aplicación móvil, pueda decirle al servidor qué usuario está utilizando el servicio.
- La recomendación es utilizar e implementar OAuth 1.0a o OAuth 2.0, ya que se descubrió que la primera versión (OAuth1.0) es vulnerable a la fijación de sesiones.

A07:2021: OPENID

- OpenId es un protocolo basado en HTTP que utiliza proveedores de identidad para validar que un usuario es quien dice ser.
- Es un protocolo muy simple que permite que un proveedor de servicios crear un single sign-on(SSO) inicial.
- Esto permite al usuario reutilizar una única identidad dada a un proveedor de identidad OpenId confiable y ser el mismo usuario en varios sitios web, sin la necesidad de proporcionar la contraseña a ningún sitio web, excepto el proveedor de identidad OpenId.
- Debido a su simplicidad y que proporciona protección de contraseñas, OpenId ha sido bien adoptado.
- Algunos de los proveedores de identidad más conocidos para OpenId son Stack Exchange, Google, Facebook y Yahoo!
- Para entornos no empresariales, OpenId se considera una opción segura y, a menudo, mejor, siempre que el proveedor de identidad sea de confianza.

A07:2021: SAML

- Security Assertion Markup Language (SAML) compete con OpenId.
- La versión más recomendada es la 2.0, ya que tiene características muy completas y proporciona una gran seguridad.
- A diferencia de OpenId, está basado en XML
- Si bien OpenId se ha apoderado de la mayor parte del mercado SAML suele ser la opción para aplicaciones empresariales.
- La razón de esto es a menudo que hay pocos proveedores de identidad OpenId que se consideren de clase empresarial.
- Lo que significa que la forma en que validan la identidad del usuario no tiene altos estándares requeridos para la identidad empresarial
- [SAML Security Cheat Sheet](#)

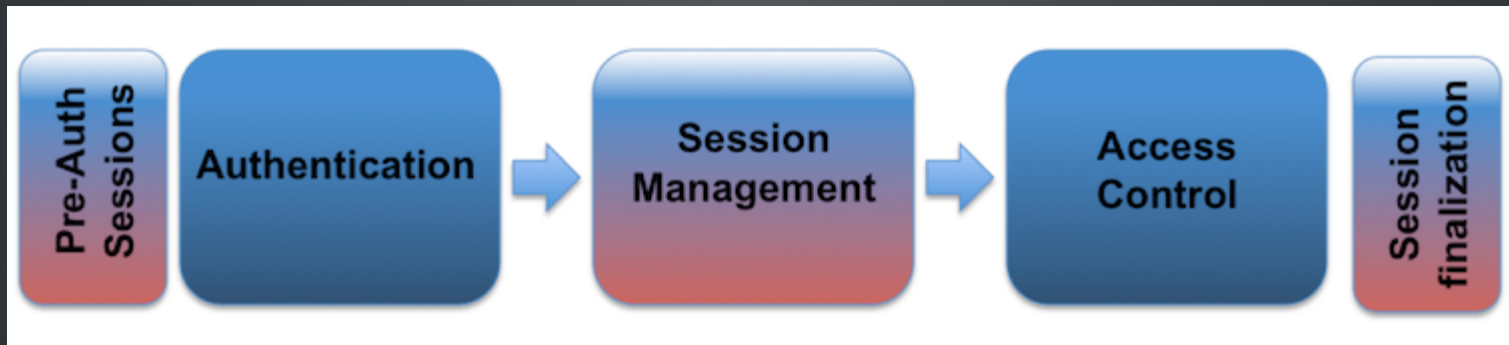
A07:2021: GESTIÓN DE SESIONES

- Una sesión web es una secuencia de transacciones de solicitud y respuesta HTTP de red asociadas con el mismo usuario.
- Las aplicaciones web modernas y complejas requieren la retención de información o estado sobre cada usuario durante la duración de múltiples solicitudes.
- Por lo tanto, las sesiones brindan la capacidad de establecer variables, como derechos de acceso y configuraciones de localización, que se aplicarán a todas y cada una de las interacciones que un usuario tenga con la aplicación web durante la duración de la sesión.

A07:2021: GESTIÓN DE SESIONES

- HTTP es un protocolo sin estado, donde cada par de solicitud y respuesta es independiente de otras interacciones web.
- Por lo tanto, para introducir el concepto de sesión, es necesario implementar capacidades de gestión de sesiones que vinculen los módulos de autenticación y control de acceso (o autorización) comúnmente disponibles en aplicaciones web.
- Una vez que se ha establecido una sesión autenticada, la ID de sesión (o token) es temporalmente equivalente al método de autenticación más fuerte utilizado por la aplicación.

A07:2021: GESTIÓN DE SESIONES



A07:2021: GESTIÓN DE SESIONES

- El ID de sesión o el token vincula las credenciales de autenticación del usuario (en forma de sesión de usuario) al tráfico HTTP del usuario y los controles de acceso adecuados impuestos por la aplicación web.
- La divulgación, captura, predicción, fuerza bruta o fijación de la ID de sesión dará lugar a ataques de secuestro de sesión (o secuestro lateral), en los que un atacante puede suplantar completamente a un usuario víctima en la aplicación web.

A07:2021: SESSION ID

- Para mantener el estado autenticado y realizar un seguimiento del progreso de los usuarios dentro de la aplicación web, las aplicaciones proporcionan a los usuarios un identificador de sesión (ID de sesión o token) que se asigna en el momento de la creación de la sesión, y que el usuario y la aplicación web comparten e intercambian durante la duración de la sesión (se envía en cada solicitud HTTP).
- El ID de sesión es un par nombre = valor.

A07:2021: SESSION ID FINGERPRINTING

- El nombre utilizado por el ID de sesión no debe ser extremadamente descriptivo ni ofrecer detalles innecesarios sobre el propósito y el significado del ID.
- Los nombres de ID de sesión utilizados en aplicaciones web más comunes se pueden descubrir fácilmente con fingerprinting:
 - PHPSESSID (PHP)
 - JSESSIONID (J2EE)
 - ASP.NET_SessionId (ASP.NET)
- Por lo tanto, el nombre de ID de sesión puede revelar las tecnologías y lenguajes de programación utilizados por la aplicación web.
- Se recomienda cambiar el nombre de ID de sesión predeterminado del marco de desarrollo web a un nombre genérico, como id.

A07:2021: SESSION ID LENGTH

- El ID de sesión debe ser lo suficientemente largo para evitar ataques de fuerza bruta, donde un atacante puede revisar todo el rango de valores de ID y verificar la existencia de sesiones válidas.
- La longitud del ID de sesión debe ser de al menos 128 bits (16 bytes).

A07:2021: SESSION ID ENTROPY

- El ID de sesión debe ser impredecible (lo suficientemente aleatorio) para evitar ataques de adivinación, donde un atacante puede adivinar o predecir el ID de una sesión válida mediante técnicas de análisis estadístico.

A07:2021: SESSION ID VALUE

- El contenido (o valor) de la ID de sesión debe carecer de significado para evitar ataques de divulgación de información, donde un atacante puede decodificar el contenido de la ID y extraer detalles del usuario, la sesión o el funcionamiento interno de la aplicación web.
- El ID de sesión debe ser simplemente un identificador del lado del cliente y su valor nunca debe incluir información confidencial (o PII).
- El significado y la lógica comercial o de aplicación asociados con la ID de sesión deben almacenarse en el lado del servidor, y específicamente, en objetos de sesión o en una base de datos o repositorio de administración de sesiones.

A07:2021: SESSION ID

- Hay varios mecanismos disponibles en HTTP para mantener el estado de la sesión dentro de las aplicaciones web, como:
 - cookies (encabezado HTTP estándar)
 - parámetros de URL (reescritura de URL - RFC2396)
 - argumentos de URL en solicitudes GET
 - argumentos de cuerpo en solicitudes POST como campos de formulario ocultos (Formularios HTML).

A07:2021: SESSION ID - VÍA GET

```
http://www.ejemplo.com/index.php?ID=883002889306
```

- Ventajas: (Mayormente de usabilidad)
 - Siempre funciona, no importa que el browser tenga cookies off
 - Facilita el acceso para el usuario si es un sitio persistente en el tiempo y se guarda en favoritos
- Desventajas: (Mayormente de Seguridad)
 - Cualquier persona en el mismo equipo podría revisar el historial o los favoritos y seguir el mismo URL
 - La información del URL puede ser almacenada en un Log por intermediarios (Firewalls o servidores proxy)
 - Es bastante fácil cambiar el URL y el ID de sesión asociado usando el navegador, facilita un ataque
 - Cuando el usuario navega a otro sitio web, el ID de la sesión puede ser enviada a ese sitio web a través del campo HTTP_REFERER

A07:2021: SESSION ID - VIA POST

- Uso de campos ocultos en un formulario, por POST:

```
<FORM METHOD=POST ACTION="http://www.ejemplo.com">  
<INPUT TYPE="hidden" NAME="sessionid" VALUE="883002889306">
```

- Ventajas:
 - No es tan obvio como enviar el ID en el URL, un poco menos atacable.
 - Permite a los usuarios almacenar el URL de la página sin info de la sesión.
 - Siempre funciona; no importa que el browser tenga cookies off
- Desventajas:
 - A pesar de que requiere un mayor nivel de conocimiento, se puede atacar
 - El contenido de la página tiende a ser más compleja

A07:2021: COOKIES

- El mecanismo de intercambio de ID de sesión basado en cookies proporciona múltiples características de seguridad en forma de atributos de cookies que se pueden utilizar para proteger el intercambio de ID de sesión:

(Parte de la respuesta enviada por el servidor al navegador cliente)

```
Set-Cookie: sessionId="883002889306"; path="/";  
domain="www.ejemplo.com"; expires="2008-09-01 00:00:00GMT";  
version=0
```

A07:2021: COOKIES

- Ventajas:
 - El uso cuidadoso de cookies persistentes puede ser usado para regular el acceso de los usuarios a través del tiempo.
 - Hay más opciones disponibles para manejar los timeouts de la sesión.
 - La funcionalidad de las cookies está disponible en todos los navegadores.
- Desventajas:
 - Las cookies persistentes son almacenadas como simples archivos de texto en el equipo del usuario, así que resulta relativamente fácil copiarla a otros dispositivos.
 - Las Cookies tienen limitación de tamaño, así que no pueden ser usadas para almacenar grandes cantidades de información del estado de la sesión.
 - Las Cookies son enviadas con cada página y-o archivo solicitado por el browser usando la directiva SET-COOKIE

A07:2021: COOKIES ATTRIBUTES

- **Secure:** indica a los navegadores web que solo envíen la cookie a través de una conexión HTTPS (SSL / TLS) cifrada
 - Garantiza que un atacante no pueda simplemente capturar el ID de sesión del tráfico del navegador web.
- **HttpOnly:** El atributo cookie indica a los navegadores web que no permitan que los scripts (por ejemplo, JavaScript o VBscript) tengan la capacidad de acceder a las cookies a través del objeto DOM `document.cookie`.
 - Esta protección de ID de sesión es obligatoria para evitar el robo de ID de sesión a través de ataques XSS.
- **SameSite:** permite que un servidor defina un atributo de cookie haciendo imposible que el navegador envíe esta cookie junto con solicitudes entre sitios.
 - El objetivo principal es mitigar el riesgo de fuga de información de origen cruzado y proporciona cierta protección contra ataques de falsificación de solicitudes entre sitios.

A07:2021: COOKIES ATTRIBUTES

- **Domain:** El atributo de cookie de dominio indica a los navegadores web que solo envíen la cookie al dominio especificado y a todos los subdominios.
 - Si el atributo no está configurado, de forma predeterminada, la cookie solo se enviará al servidor de origen.
- **Path:** El atributo de cookie de ruta indica a los navegadores web que solo envíen la cookie al directorio o subdirectorios especificados (o rutas o recursos) dentro de la aplicación web.
 - Si el atributo no está configurado, de forma predeterminada, la cookie solo se enviará para el directorio (o ruta) del recurso solicitado y la configuración de la cookie.
- **Max-Age o Expires:** Los mecanismos de gestión de sesiones basados en cookies pueden hacer uso de dos tipos de cookies, cookies no persistentes (o de sesión) y cookies persistentes.
 - Si una cookie presenta los atributos Max-Age (que tiene preferencia sobre Expires) o Expires, se considerará una cookie persistente y el navegador web la almacenará en el disco hasta la fecha de vencimiento.

A07:2021: HTML5 WEB STORAGE

- localStorage y sessionStorage, son mecanismos para almacenar pares nombre-valor del lado del cliente.
- A diferencia de las cookies HTTP, los contenidos de localStorage y sessionStorage no se comparten automáticamente dentro de las solicitudes o respuestas del navegador y se utilizan para almacenar datos del lado del cliente.
- Refs:
 - [Web Storage APIs](#)
 - [LocalStorage API](#)
 - [SessionStorage API](#)

A07:2021: HTML5 LOCALSTORAGE

- Los datos almacenados usando la API localStorage son accesibles por páginas que se cargan desde el mismo origen.
- Debido al potencial acceso concurrente desde ventanas / subprocesos separados, los datos almacenados usando localStorage pueden ser susceptibles a problemas de acceso compartido

A07:2021: HTML5 SESSIONSTORAGE

- La API sessionStorage almacena datos dentro del contexto de la ventana desde la que se llamó, lo que significa que la pestaña 1 no puede acceder a los datos que se almacenaron desde la pestaña 2.

A07:2021:SESSION ID LIFE CYCLE

- Hay dos tipos de mecanismos de gestión de sesiones para aplicaciones web,
 - permisivos: El mecanismo permisivo permite que la aplicación web acepte inicialmente cualquier valor de ID de sesión establecido por el usuario como válido, creando una nueva sesión para él
 - estrictos: el mecanismo estricto obliga a que la aplicación web solo acepte valores de ID de sesión que hayan sido generados previamente por el Aplicación web.
- Los tokens de sesión deben ser manejados por el servidor web si es posible o generados mediante un generador de números aleatorios criptográficamente seguro.

A07:2021:SESSION ID LIFE CYCLE

- Los ID de sesión deben considerarse no confiables, como cualquier otra entrada de usuario procesada por la aplicación web, y deben validarse y verificarse minuciosamente.
- Dependiendo del mecanismo de gestión de sesión utilizado, el ID de sesión se recibirá en un parámetro GET o POST, en la URL o en un encabezado HTTP (por ejemplo, cookies).
- Si las aplicaciones web no validan y filtran los valores de ID de sesión no válidos antes de procesarlos, se pueden usar potencialmente para explotar otras vulnerabilidades web, como la inyección de SQL si los ID de sesión se almacenan en una base de datos relacional, o XSS persistente si los ID de sesión son almacenados y reflejados posteriormente por la aplicación web.

A07:2021:SESSION ID LIFE CYCLE

- La aplicación web debe renovar o regenerar el ID de sesión después de cualquier cambio de nivel de privilegio dentro de la sesión de usuario asociada.
- El escenario más común en el que la regeneración del ID de sesión es obligatoria es durante el proceso de autenticación, ya que el nivel de privilegio del usuario cambia del estado no autenticado (o anónimo) al estado autenticado.
- Se deben considerar otros escenarios comunes, como cambios de contraseña, cambios de permisos o cambio de un rol de usuario regular a un rol de administrador dentro de la aplicación web.

A07:2021:SESSION ID LIFE CYCLE

Los Frameworks de desarrollo web más comunes proporcionan funciones y métodos de sesión para renovar el ID de sesión, como:

- `request.getSession(true)` & `HttpSession.invalidate()` (J2EE)
- `Session.Abandon()` & `Response.Cookies.Add()` (ASP .NET),
- `session_start()` & `session_regenerate_id()` (PHP)

A07:2021:SESSION EXPIRATION

- Para minimizar el período de tiempo que un atacante puede lanzar ataques sobre sesiones activas y secuestrarlas, es obligatorio establecer tiempos de expiración para cada sesión, estableciendo la cantidad de tiempo que una sesión permanecerá activa.
- La expiración insuficiente de la sesión por parte de la aplicación web aumenta la exposición de otros ataques basados en sesiones, ya que para que el atacante pueda reutilizar una ID de sesión válida y secuestrar la sesión asociada, debe seguir activa.
- Cuanto más corto sea el intervalo de sesión, menor será el tiempo que un atacante tiene para usar la ID de sesión válida.

A07:2021: IDLE TIMEOUT

- Todas las sesiones deben implementar un tiempo de espera inactivo o inactivo.
- Este tiempo de espera define la cantidad de tiempo que una sesión permanecerá activa en caso de que no haya actividad en la sesión, cerrando e invalidando la sesión en el período de inactividad definido desde la última solicitud HTTP recibida por la aplicación web para una ID de sesión determinada.
- Ej:
 - Netflix: "continuar viendo"
 - Homebankings: "sigues ahí?"

A07:2021: MANUAL SESSION EXPIRATION

- Las aplicaciones web deben proporcionar mecanismos que permitan a los usuarios conscientes de la seguridad cerrar activamente su sesión una vez que hayan terminado de usar la aplicación web.
- Logout:
 - Las aplicaciones web deben proporcionar un botón de cierre de sesión (cerrar sesión, salir o cerrar sesión) visible y fácilmente accesible que esté disponible en el encabezado o menú de la aplicación web y accesible desde cada recurso y página de la aplicación web, de modo que el usuario pueda cerrar manualmente la sesión en en cualquier momento.
 - Cerrar sesión en instagram u otras apps no cumplen con esto.

A07:2021: ATTACKS

- Por como funciona una sesión típicamente independientemente del método que se utilice → el problema es el ID de sesión que identifica al usuario.
- Problemas comunes:
 - **Session Prediction** → Que se pueda predecir el id de sesión
 - **Session hijacking** → Que se pueda capturar el id de sesión
 - **Session fixation** → Que se pueda fijar el id de sesión desde el cliente

A07:2021: SESSION PREDICTION

- El ataque de predicción de sesión se centra en predecir los valores de ID de sesión que permiten a un atacante eludir el esquema de autenticación de una aplicación.
- Al analizar y comprender el proceso de generación de ID de sesión, un atacante puede predecir un valor de ID de sesión válido y obtener acceso a la aplicación.

A07:2021: SESSION PREDICTION

- En el primer paso, el atacante debe recopilar algunos valores de ID de sesión válidos.
- Luego, deben comprender la estructura del ID de sesión, la información que se usa para crearlo y el algoritmo de cifrado o hash que usa la aplicación para protegerlo.
- Algunas implementaciones incorrectas utilizan ID de sesiones compuestas por nombre de usuario u otra información predecible, como la marca de tiempo o la dirección IP del cliente.
- En el peor de los casos, esta información se utiliza en texto sin cifrar o se codifica utilizando algún algoritmo débil como la codificación base64.

Ejemplo: DVWA Vulnerability: Weak Session IDs

A07:2021: SESSION HIJACKING

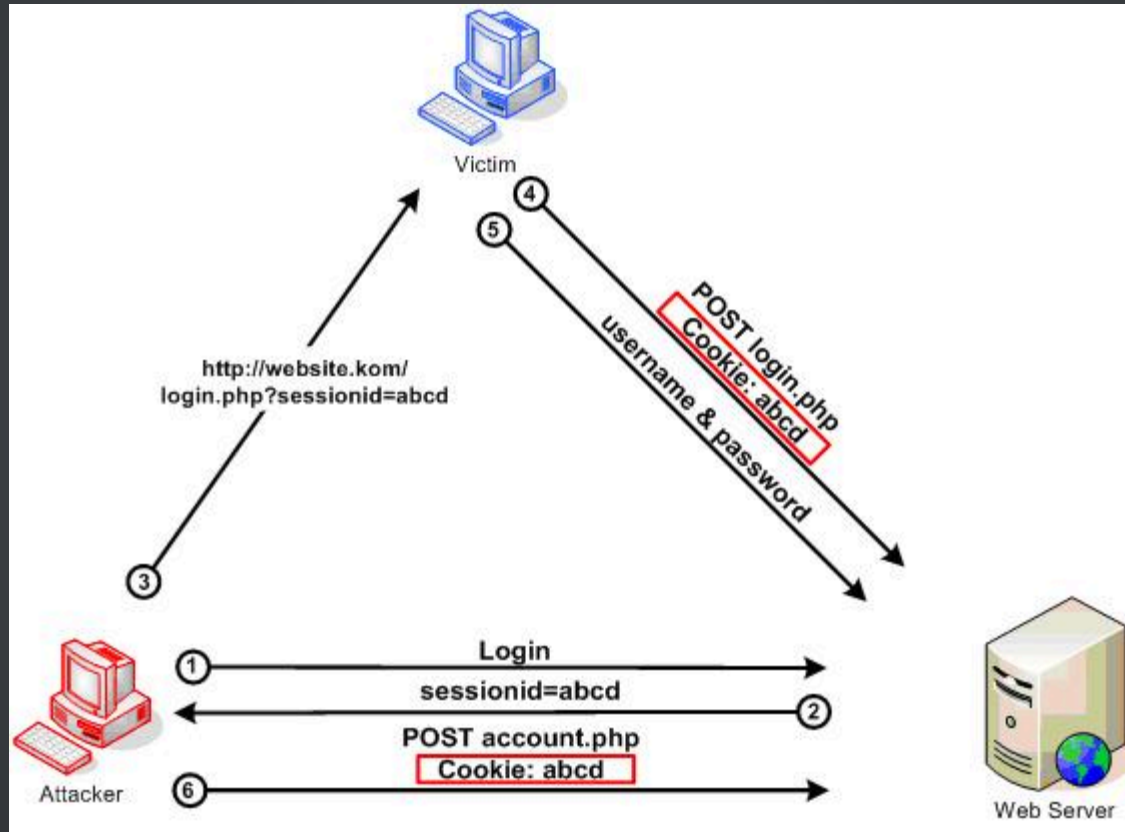
- El ataque de secuestro de sesión compromete el token de sesión al robar o predecir un token de sesión válido para obtener acceso no autorizado al servidor web.
- El token de sesión podría verse comprometido de diferentes formas; los más comunes son:
 - Predictable session token;
 - Session Sniffing;
 - Client-side attacks (XSS, malicious JavaScript Codes, Trojans, etc);
 - Man-in-the-middle attack
 - Man-in-the-browser attack

A07:2021: SESSION FIXATION

- Session Fixation es un ataque que permite a un atacante secuestrar una sesión de usuario válida.
- Al autenticar a un usuario, no asigna una nueva ID de sesión, por lo que es posible utilizar una ID de sesión existente.
- El ataque consiste en obtener una ID de sesión válida (por ejemplo, conectándose a la aplicación), inducir a un usuario a autenticarse con esa ID de sesión y luego secuestrar la sesión validada por el usuario mediante el conocimiento de la ID de sesión utilizada.
- El ataque de fijación de sesión es una clase de Session Hijacking

Ejemplo: DVWA Cookie `fix` value

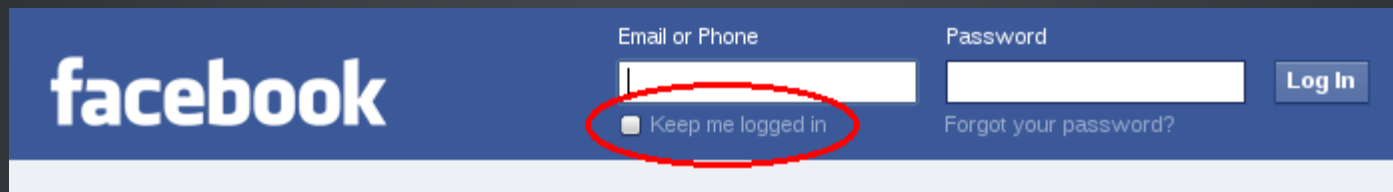
A07:2021: SESSION FIXATION



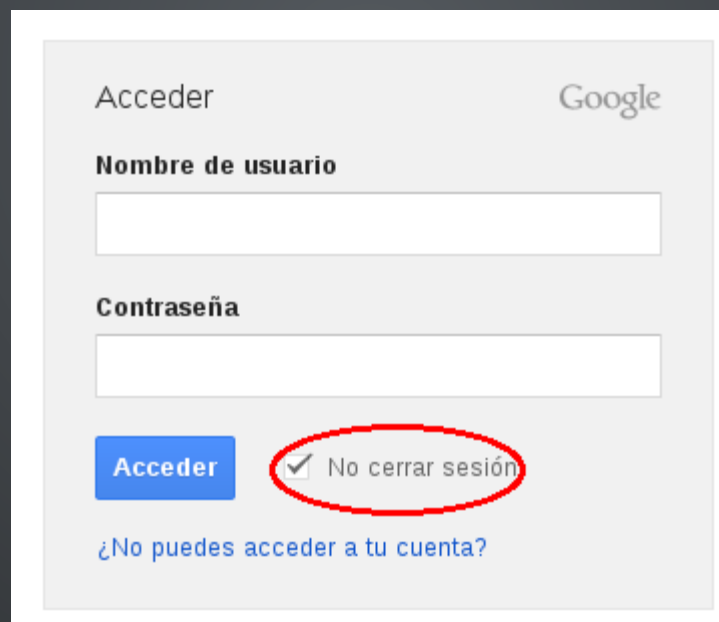
A07:2021: SESSION ATTACKS REMEMBER ME

- La Cookie de "Remember me" es equivalente al sello que te estampa en la mano el portero de un bar.
- Cuando volvés y mostrás el sello, él simplemente va a dejar que entres directamente al bar.
- Seguramente, sin pedir documento, entrada, etc.
- En el caso de las cookies, se generará una nueva SESSID Cookie pero mantendrá la REMEMBERME
- El resultado son dos SESSID validos para el mismo usuario.

A07:2021: SESSION ATTACKS

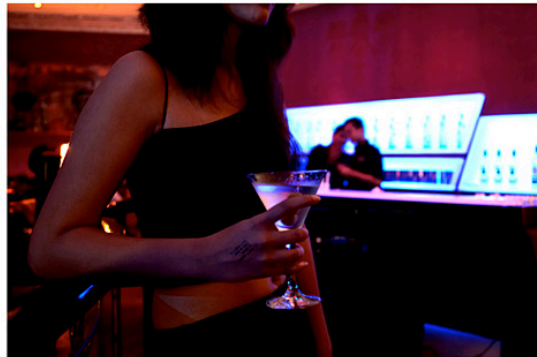


The image shows the Facebook login interface. The Facebook logo is on the left. To its right are two input fields: 'Email or Phone' and 'Password'. Below the 'Email or Phone' field is a checkbox labeled 'Keep me logged in', which is circled in red. To the right of the 'Password' field is a 'Log In' button and a link that says 'Forgot your password?'.



The image shows the Google login interface. At the top left is the word 'Acceder' and at the top right is the Google logo. Below these are two input fields: 'Nombre de usuario' and 'Contraseña'. Below the 'Contraseña' field is a blue 'Acceder' button and a checkbox labeled 'No cerrar sesión', which is circled in red. At the bottom is a link that says '¿No puedes acceder a tu cuenta?'.

A07:2021: SESSION ATTACKS REMEBER ME



Description: Clubs in India use an ink stamp to identify their patrons. For Enigma, we replaced the simple logo stamp with a call-a-cab number and a reminder against drinking and driving.

Brief: Marriott's nightclub Enigma wanted a low-budget campaign that cautioned their patrons against drinking and driving.

Results: Not only did the cab company receive approximately 30 calls a night from Enigma, the exercise generated such strong word-of-mouth that it was taken up by the Mumbai Police and has now been adopted by several other clubs in the city.



A07:2021: SESSION ATTACKS

- Chrome : se inicia sesion en el browser
- Smartphones..always loggedin

DEMO MOODLE

JSON WEB TOKEN

- **JSON Web Token** (JWT) es un estándar abierto (**RFC-7519**) basado en JSON para crear un token que sirva para enviar datos entre aplicaciones o servicios y garantizar que sean válidos y seguros.

JSON WEB TOKEN

- Debido a que los JWT son solo cadenas seguras para URL, son fáciles de transmitir a través de parámetros de URL, etc.
- JWT puede almacenar tantos datos JSON como se desee y pueden decodificarse en un objeto JSON.

JSON WEB TOKEN

- Están firmados criptográficamente lo que significa que cualquier parte de confianza que tenga un JWT puede saber si el token se ha modificado o cambiado.
- Esto significa que si su aplicación o servicio de API genera un token que dice que alguien es un usuario "gratis" y luego alguien modifica el token para decir que es un usuario "administrador", podrá detectar esto y actuar en consecuencia.
- Esta propiedad hace que los JWT sean útiles para compartir información entre partes a través de la web donde la confianza es difícil de conseguir.

JSON WEB TOKEN

- Los JWT se utilizan normalmente como identificadores de sesión para aplicaciones web, aplicaciones móviles y servicios API.
- Pero, a diferencia de los identificadores de sesión tradicionales, como las cookies, que actúan como nada más que un puntero a los datos reales del usuario en el lado del servidor, los JWT normalmente contienen datos del usuario directamente.
- La razón principal por la que los JWT se han vuelto populares en los últimos años (que solo existen desde 2014) es que pueden contener datos JSON arbitrarios.

JSON WEB TOKEN

- El beneficio de un JWT sobre un ID de sesión tradicional es que:
 - Los JWT no tienen estado y pueden contener datos de usuario directamente
 - Debido a que los JWT no tienen estado, no es necesario implementar ninguna sesión del lado del servidor (sin base de datos de sesión, caché de sesión, etc.)
- Debido a que los JWT no tienen estado, cuando una aplicación del lado del servidor recibe un JWT, puede validarlo usando solo la "clave secreta" que se usó para crearlo, evitando así la penalización de rendimiento de hablar con una base de datos o caché en el backend, que agrega latencia a cada solicitud.

JWT: ESTRUCTURA

- Hay implementación en multitud de lenguajes. Es una cadena formada por 3 partes:

```
[Base64(HEADER)] . [Base64(PAYLOAD)] . [Base64(SIGNATURE)]
```

JWT: ESTRUCTURA

- **Header:** indica el tipo de token y el algoritmo de firma.
 - Se codifica en Base64.

```
{  
  "alg": "HS256",  
  "typ": "JWT"  
}
```

(indica que esto es un "JWT" y se firmará con HMAC SHA-256)

JWT: ESTRUCTURA

- **Payload:** la información que queremos almacenar en el token en formato JSON y codificado en Base64URL
- El payload solo es codificado en base64 no es seguro guardar información sensible.

```
{  
  "sub": "1234567890",  
  "name": "John Doe",  
  "iat": 1516239022  
}
```

JWT: ESTRUCTURA

- **Signature:** se aplica un algoritmo de hash sobre la cabecera, el payload y una clave secreta y se pasa a Base64URL

```
HMACSHA256(  
  base64UrlEncode(header) + "." +  
  base64UrlEncode(payload),  
  
  your-256-bit-secret  
)
```


JWT: EJEMPLO

```
header = '{"alg":"HS256","typ":"JWT"}'
payload = '{"loggedInAs":"admin","iat":1422779638}'
key = 'secretkey'
unsignedToken = encodeBase64(header) + '.' + encodeBase64(payload)
signature = HMAC-SHA256(key, unsignedToken)
token = encodeBase64(header) + '.' + encodeBase64(payload)
+ '.' + encodeBase64(signature)
```

```
#El token es:
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.
eyJsb2dnZWZjbGkiOiJWRtaW4iLCJpYXQiOiJlMjI3Nzk2Mzh9.
gZSraSYS8EXBxLN_oWnFSRgCzcmJmMjLiuyu5CSpyHI
```

JWT: SEGURIDAD

- Debido a que los JWT se utilizan para identificar al cliente, si uno es robado o comprometido, un atacante tiene acceso completo a la cuenta del usuario de la misma manera que lo haría si el atacante hubiera comprometido el nombre de usuario y la contraseña del usuario.
- Los JWT se pueden configurar para que caduquen automáticamente después de un período de tiempo establecido (un minuto, una hora, un día, lo que sea), los atacantes solo pueden usar su JWT para acceder al servicio hasta que caduque.

JWT: SEGURIDAD

- En general, los tokens deben tratarse como contraseñas y protegerse como tales.
- Nunca deben compartirse públicamente y deben guardarse en almacenes de datos seguros.
- Para las aplicaciones basadas en navegador, esto significa nunca almacenar sus tokens en HTML5 Local Storage
- [JSON Web Token Cheat Sheet](#)