**Historia de JMeter**

**Versión oficial 2001**

**Inició su construcción 1998**

Es un software de código abierto que fue desarrollado por Stefano Mazzochi de la fundación de software Apache, diseñado para pruebas de carga funcionales y medidas de performance.

Puede usar JMeter para analizar y medir el rendimiento de la aplicación web o una variedad de servicios. La prueba de rendimiento significa probar una aplicación web contra una carga pesada, tráfico de usuarios múltiples y concurrentes. JMeter originalmente se usa para probar aplicaciones web o aplicaciones FTP. Hoy en día, se utiliza para una prueba funcional, una prueba de servidor de base de datos, etc.

* **jmeter.bat**

run JMeter (in GUI mode by default)

* **jmeterw.cmd**

run JMeter without the windows shell console (in GUI mode by default)

* **jmeter-n.cmd**

drop a JMX file on this to run a CLI mode test

* **jmeter-n-r.cmd**

drop a JMX file on this to run a CLI mode test remotely

* **jmeter-t.cmd**

drop a JMX file on this to load it in GUI mode

* **jmeter-server.bat**

start JMeter in server mode

* **mirror-server.cmd**

runs the JMeter Mirror Server in CLI mode

* **shutdown.cmd**

Run the Shutdown client to stop a CLI mode instance gracefully

* **stoptest.cmd**

Run the Shutdown client to stop a CLI mode instance abruptly

## **Solicitud FTP**

Este controlador le permite enviar una solicitud de "recuperar archivo" FTP o "cargar archivo" a un servidor FTP. Si va a enviar múltiples solicitudes al mismo servidor FTP, considere usar un Elemento de configuración de valores predeterminados de solicitud FTP para que no tenga que ingresar la misma información para cada Controlador generativo de solicitud FTP. Al descargar un archivo, puede almacenarse en el disco (Archivo local) o en los Datos de respuesta, o en ambos.

## **Solicitud HTTP**

Este muestreador le permite enviar una solicitud HTTP / HTTPS a un servidor web. También le permite controlar si JMeter analiza o no los archivos HTML para imágenes y otros recursos integrados y envía solicitudes HTTP para recuperarlos. Se recuperan los siguientes tipos de recursos incrustados:

* imágenes
* applets
* hojas de estilo (CSS) y recursos a los que se hace referencia desde esos archivos
* guiones externos
* marcos, iframes
* imágenes de fondo (cuerpo, tabla, TD, TR)
* sonido de fondo

El analizador predeterminado es **org.apache.jmeter.protocol.http.parser.LagartoBasedHtmlParser** . Esto se puede cambiar utilizando la propiedad " **htmlparser.className** "; consulte **jmeter.properties** para obtener más detalles.

Non GUI Mode

Para realizar pruebas de carga, debe ejecutar JMeter en este modo (sin la GUI) para obtener los mejores resultados. Para hacerlo, use las siguientes opciones de comando:

**-n** This specifies JMeter is to run in cli mode

**-t** [name of JMX file that contains the Test Plan].

**-l** [name of JTL file to log sample results to].

**-j** [name of JMeter run log file].

**-r** Run the test in the servers specified by the JMeter property "remote\_hosts"

**-R** [list of remote servers] Run the test in the specified remote servers

**-g** [path to CSV file] generate report dashboard only

**-e** generate report dashboard after load test

**-o** output folder where to generate the report dashboard after load test. Folder must not exist or be empty

The script also lets you specify the optional firewall/proxy server information:

**-H** [proxy server hostname or ip address]

**-P** [proxy server port]

jmeter -n -t my\_test.jmx -l log.jtl -H my.proxy.server -P 8000

Los http request son síncronos, se mantienen esperando en hasta que llegue el response durante un cierto tiempo.

**Servidor Web**

En cuanto a software, un servidor web tiene muchas partes encargadas del control sobre como tienen acceso los usuarios a los archivos, por lo menos un servidor *HTTP. UN* servidor HTTP es una pieza de software que comprende [URLs](https://developer.mozilla.org/es/docs/Glossary/URL) (direcciones web) y [HTTP](https://developer.mozilla.org/es/docs/Glossary/HTTP) (el protocolo que tu navegador usa para ver las paginas web).

Al nivel más básico, siempre que un navegador necesite un archivo almacenado en un servidor web, el navegador lo solicita via HTTP. Cuando la petición llega al servidor web correcto (hardware), el servidor *HTTP*  (software) envía el archivo antes solicitado, tambien a través de HTTP.

Para publicar un sitio web, necesitarás un servidor web dinámico o estático.

Un **servidor web estático**, o pila, consiste en una computadora (hardware) con un servidor HTTP (software). Llamamos a este "estático" debido a que el servidor envía los archivos almacenados "tal cual" a tu navegador.

Un **servidor web dinámico** consiste en un servidor web estático con un software extra , lo mas común es que sea una *aplicación* servidor *y una*  base de datos*. Llamamos a esto* "dinámico" por que la aplicación servidor actualiza los archivos almacenados antes de enviarlos mediante el servidor HTTP.

Por ejemplo, para ver la página que ves en tu navegador finalmente, el servidor aplicacion puede mostrar el diseño HTML con contenido desde una base de datos. sitios como MDN or Wikipedia tienen cientos de paginas web, pero estas no son realmente archivos HTML, se trata de una estrucura HTML y una gigantesca base de datos. Esto hace mas fácil y rapido el mantenimiento y entrega del contenido.

Applicantion server

**Application Servers:**

To understand this better, we’ll have to know what a Web Server is.

A Web Server will send the HTML as output for all requests sent to the server. It does not have any backend logic.

But, Application server is where complex and large applications reside. It can have backend logic and makes the application dynamic by interacting with the Database servers. Application deployment happens on the application server. An application server can also handle the services and requests which a web server can handle. This means that there is an in-built web server in application servers.

A web server, even though may to be necessary, people still to prefer to use it for numerous reasons including the language and the technology supported by the Application servers. Application servers also perform other services apart from web page generation.

Some of the services offered by application servers include clustering, fail-over, and load-balancing, so developers can focus on implementing the business logic.

**Example of Web servers:**

1. PHP – Apache

2. Java – Tomcat

3. Windows – IIS

**Example of Application Servers**

1. PHP – Zend Server

2. Java – Jb, Geronimo

3. Windows – Windows Server

Backend

El backend es la parte del desarrollo web que se encarga de que toda la lógica de una página web funcione. Se trata del conjunto de acciones que pasan en una web pero que no vemos como, por ejemplo, la comunicación con el servidor.

**El encargado de desarrollar el back-end se pasa el día entero toqueteando diferentes y variados códigos de programación en lenguajes como PHP, Java o .Net.** El desarrollador debe ser meticuloso y cuidadoso en su trabajo ya que un pequeño despiste y ¡bum!… Página web caída.

Antes de seguir con este artículo, quería comentarte que quizá hayas oído llamar backend a la parte administrativa de una página web. Donde se hacen los ajustes, como por ejemplo pasa en WordPress. Bueno, pues también se llama Backend a esto. Pero realmente Backend es lo que están explicando los chicos de Ensalaza. :)

Algunas de las funciones que se gestionan en la parte del back-end son:

* El Desarrollo de funciones que simplifiquen el proceso de desarrollo.
* Acciones de lógica.
* Conexión con bases de datos.
* Uso de librerías del servidor web (por ejemplo para implementar temas de caché o para comprimir las imágenes de la web).

Además, tiene que velar por la seguridad de los sitios web que gestiona y optimizar al máximo los recursos para que las páginas sean ligeras.

## **¿Qué debe caracterizar al desarrollador de back-end?**

Sentimos decirte que no todo el mundo puede ser programador back-end. Bueno, mentimos, estudiando puedes serlo, pero tiene que gustarte mucho para no cansarte ni volverte loco. Ser programador es sinónimo de ser constante, serio con tu trabajo y con alta capacidad de atención y concentración.

Además, hay una serie de ramas que debes controlar para saber programar back:

* Un buen desarrollador de backend debe saber configurar los servidores. O al menos, **saber lo suficiente para hablar con el responsable de sistemas** y explicarle sus necesidades.
* Debería manejar al menos un CMS, por ejemplo WordPress.
* Saber de Metodologías de programación.
* Conocer al dedillo los fundamentos de la programación y manejarse con soltura en al menos uno de ellos (en Ensalza, por ejemplo, nos especializamos en el desarrollo con PHP)
* Ser MUY lógico (pero mucho, mucho, mucho…) y ordenado (pero mucho, mucho también)

## **¿Qué diferencia al backend del frontend?**

Sin necesidad de ser bilingües ni ser un gran lumbreras, entendemos que si el back-end es el que desarrolla la parte de atrás de una web, lo que no se ve, el front-end… ¿será el que diseña lo de delante / lo que se ve?

Pues sí.

El desarrollador de front-end se encarga de implementar todo lo relacionado con la parte visible, lo que «toca» el usuario cuando navega por la web. Resumiendo al máximo el front-end trabaja con *HTML*, *CSS* y *JavaScript*.

Su trabajo es mucho más visual que el del back-end, pero ambos trabajan mano a mano en cada proyecto porque no se entienden uno sin el otro.

## **¿Por qué es tan importante el BackEnd?**

Es cierto que no pueden vivir el backend sin el front y viceversa. Sin embargo, y especialmente para Ensalza, la parte del back se hace especialmente necesaria e imprescindible para desarrollar proyectos a medida, que es en lo que somos expertos nosotros.

Para una página web corporativa siempre puede haber un diseñador gráfico que haga un diseño nuevo y que un experto en front-end se encargue de maquetarla en WordPress. Con este formato, a priori, no hace falta ningún desarrollador de back-end y sin embargo se puede crear una web nueva. En este caso, **el desarrollo de back-end se lo dejamos a WordPress y el front-end se «aprovechará» de esas funcionalidades para maquetar la web**.

Pero, ¿qué ocurre en [desarrollos a medida](https://blog.ensalza.com/desarrollo-web-a-medida/?utm_source=referral&utm_medium=link&utm_campaign=rafa-arjonilla&utm_content=post) o en esos proyectos que requieren «un puntito más» de desarrollo? Pues que la **figura de los desarrolladores back-end resulta ESENCIAL para agencias de desarrollo web**. ¿Qué seríamos nosotros sin nuestros freaks del teclado?

**Marcas de Bases de Datos**

Oracle

MySQL

SQLite

MongoDB

Cassandra

MariaDB

SQL Server

Postgre SQL

**Servidor proxy**

Un servidor proxy es un equipo dedicado o un sistema de software que se ejecuta en un equipo de cómputo que actúa como intermediario entre un dispositivo de punto final, como una computadora, y otro servidor del cual un usuario o cliente solicita un servicio. El servidor proxy puede existir en la misma máquina que un servidor de firewall o puede estar en un servidor independiente, que reenvía las solicitudes a través del firewall.

Una ventaja de un servidor proxy es que su caché puede servir a todos los usuarios. Si uno o más sitios de internet se solicitan con frecuencia, es probable que estos estén en el caché del proxy, lo que mejorará el tiempo de respuesta del usuario. Un proxy también puede registrar sus interacciones, lo que puede ser útil para la solución de problemas.

A continuación se muestra un ejemplo sencillo de cómo funcionan los servidores proxy:

Cuando un servidor proxy recibe una solicitud de un recurso de internet (como una página Web), busca en su caché local de páginas anteriores. Si encuentra la página, la devuelve al usuario sin necesidad de reenviar la solicitud a internet. Si la página no está en la memoria caché, el servidor proxy, que actúa como un cliente en nombre del usuario, utiliza una de sus propias direcciones IP para solicitar la página desde el servidor en internet. Cuando se devuelve la página, el servidor proxy la relaciona con la solicitud original y la envía al usuario.

Los servidores proxy se utilizan para fines legales e ilegales. En la empresa, un servidor proxy se utiliza para facilitar la seguridad, control administrativo o servicios de caché, entre otros propósitos. En un contexto de computación personal, los servidores proxy se utilizan para permitir la privacidad del usuario y el surf anónimo. Los servidores proxy también se pueden utilizar para el propósito opuesto: supervisar el tráfico y minar la privacidad del usuario.

Para el usuario, el servidor proxy es invisible; todas las solicitudes de internet y las respuestas devueltas parecen estar directamente con el servidor de internet dirigido. (El proxy no es realmente invisible, su dirección IP tiene que ser especificada como una opción de configuración para el navegador u otro programa de protocolo.)

Los usuarios pueden acceder a proxies web en línea o configurar navegadores web para usar constantemente un servidor proxy. Los parámetros del navegador incluyen opciones manuales y detectadas automáticamente para proxies HTTP, [SSL](https://searchdatacenter.techtarget.com/es/respuesta/Proceso-de-reconocimiento-de-SSL-Claves-publicas-y-privadas-explicadas), FTP y SOCKS. Los servidores proxy pueden servir a muchos usuarios o sólo uno por servidor. Estas opciones se llaman proxies compartidos y dedicados, respectivamente. Hay una serie de razones para los proxies y por lo tanto una serie de tipos de servidores proxy, a menudo en categorías superpuestas.

Usuarios > Virtual Users } Threads

**Mecanismos de JMeter**

Samplers

Controllers

Listeners

Configuration Elements

Assertions

Timers

Pre-Processors

Post-Processors

Miscellaneous Features

**Más conceptos**

Performance testing es una prueba no funcional

Se utilizan para validar si una aplicación cumple o no con los requisitos no funcionales.

Se usarán herramientas de pruebas de performance para enviar requests utilizando users en un modelo de negocios para verificar si cumple o no con los requisitos no funcionales.

**Un sampler** es un mecanismo que permite enviar requests a un web server.

Se mide la sintonía entre la aplicación y el ambiente

Son imprescindibles en el ciclo de vida de SW

**Cookies**

La aplicación le instruye al browser a guardar información para que la aplicación pueda ser mas rapida, mas sencilla.

Las cookies son archivos que crean los sitios web que visitas y guardan datos de navegación para que disfrutes de una experiencia online más sencilla. Gracias a ellas, los sitios web no cierran tu sesión, recuerdan tus preferencias y te proporcionan contenido relevante según tu ubicación.

**Proceso de Negocio Obtener curp**

1. Navegar a la pagina utilizando la URL <https://consultas.curp.gob.mx/CurpSP/gobmx/inicio.jsp>
2. Dar click en la pestaña Ingresa los datos para buscar la CURP utilizando el icono “+”.
3. Ingresar tu Nombre(s) en el campo nombre.
4. Ingresar tu primer apellido en el campo primer apellido.
5. Ingresar tu segundo apellido en el campo segundo apellido.
6. Marcar tu sexo en el apartado sexo.
7. Ingresar tu fecha de nacimiento en el formato correcto.
8. Seleccionar tu entidad en el campo Entidad Federativa de nacimiento.
9. Ingresar el codigo de verificacion en el campo codigo de verificacion. (El código se muestra en la parte superior derecha de la página).
10. Dar click en el botón Buscar .

En performance testing el proceso de negocio conocido como business process o Workflow no se buscan todas las variantes como si fuera una prueba funcional en un test case.

Se realiza una sola combinación para lograr un objetivo. HAPPY PATH

**Metodología de Performance Testing**

**Fases Estratégicas**

1: Discover (Descubrir el objetivo de la aplicación, que componentes contiene, qué herramientas utiliza, en qué consiste).

2: Planning( Establecer fechas y orden de trabajo dependiendo el trabajo a hacerse la herramienta a utilizar).

3: Design(Cual workflow se debe de documentar, se identifican datos de prueba y se documentan, los detailed steps(engloban user action que resulta en un request), los procesos de negocio).

Se generan el Performances test plan y el Business Processes Documented .

4: Construction(Fase en la cual ocurre toda la construcción de test scripts, scenarios, validar el entorno, obtener la información de prueba y que los técnicos del equipo estén disponibles para monitorear la aplicación).

**Fases Iterativas**

5: Execution(Se corren ciertos tipos de pruebas, identificar, rastrear y resuelve problemas de ambiente, monitorea las aplicación y el servidor, ejecuta pruebas de performance.

A los scripts se les debe dar mantenimiento en cada ciclo de prueba.

6: Analysis & Optimization(

Todo se documenta bajo un template recibido.

**HTTP Authorization Manager**

El Administrador de autorización le permite especificar uno o más inicios de sesión de usuario para páginas web que están restringidas mediante la autenticación del servidor. Usted ve este tipo de autenticación cuando usa su navegador para acceder a una página restringida, y su navegador muestra un cuadro de diálogo de inicio de sesión. JMeter transmite la información de inicio de sesión cuando encuentra este tipo de página.

**SLA Service Level Agreement**

Un acuerdo de nivel de servicio (*service level agreement*, SLA) es un contrato entre un proveedor de servicios y sus clientes internos o externos que documenta qué servicios proporcionará el proveedor y define los estándares de servicio que el proveedor está obligado a cumplir.

En caso de la curp es tiempo de respuesta esperado para la transacción.

**Key Business Transactions / Critical Business Transactions**

Las transacciones más importantes o críticas en una aplicación.

**Workarounds**

Pasos a seguir cuando una prueba no funciona correctamente.

**SME Subject Matter Expert**

La persona experta en el tema.

**Naming conventions de Hexaware**

BP<#>\_NameofBP

**TLA:** Three Letter Acronym

### **Correlación**

Empecemos por la definición de correlación: La correlación es la relación que existe entre una variable que está siendo utilizada en la sesión del Usuario que navega la aplicación y los valores de las mismas en el servidor.

Esto quiere decir que cuando grabemos el script del escenario que queremos testear algunas variables que son dinámicas van a tomar valores fijos (Hardcodeados) por lo que al querer correr el escenario con JMeter (o la herramienta que sea) nos vamos a encontrar con errores del servidor ya que se le está mandando valores fijos cuando los valores de esas variables deben ser dinámicos.

**Expresiones regulares y su uso en correlaciones para guardar datos de aplicación o de communication/Diálogo HTTP:**

**Expresiones regulares en Unix y javascript.**

Las expresiones regulares son una serie de caracteres que forman un patrón, normalmente representativo de otro grupo de caracteres mayor, de tal forma que podemos comparar el patrón con otro conjunto de caracteres para ver las coincidencias.

**(^)** La expresión encaja o coincide al principio de la línea, por ejemplo ^A

**(?)** La expresión encaja o coincide al final de la línea, por ejemplo A?

**(\)** Escapa el carácter siguiente a la contrabarra, haciendo que deje de ser un carácter especial, ejemplo in \\*

**([ ])** La expresión encaja o coincide con alguno de los caracteres incluidos entre los corchetes, también se pueden especificar rangos, por ejemplo [aeiou] o [0-9].

**[^ ]** La expresión encaja o coincide con cualquier carácter excepto los que se encuentran entre corchetes, ejemplo, todos los caracteres excepto del 0 al 9: [^0-9]

**(.)** La expresión encaja respecto a un único carácter, de cualquier valor, excepto al final de la línea.

**(\*)** La expresión coincide con 0 o más caracteres de la expresión que le precede.

**\{x,y\}** La expresión encaja o coincide desde X hasta Y respecto a lo que la ocurrencia que le precede.

**\{x\}** La expresión encaja exactamente X ocurrencias de lo que le precede.

**\{x,\}** La expresión encaja exactamente X o más ocurrencias de lo que le precede.

Se permite colocar espacios

**$ # Listar todas las líneas que comienzan por "From:" de nuestros emails:**

$ grep '^From: ' /usr/mail/$USER

**$ # Buscar líneas con al menos una letra de la a a la z (incluidas mayúsculas) de un fichero concreto.**

$ grep '[a-zA-Z]' search-file.txt

**$ # Lista cualquier carácter que no sea ni letra ni número de un fichero concreto.**

$ grep '[^a-zA-Z0-9] search-file.txt

**$ # Buscar teléfonos del tipo 999-9999 de un fichero concreto.**

$ grep '[0-9]\{3\}-[0-9]\{4\}' search-file.txt

**$ # Buscar líneas de un fichero con solamente 1 carácter:**

$ grep '^.$' search-file.txt

**$ # Buscar líneas que comienzan con el carácter "."**

$ grep '^\.' search-file.txt

**$ # Buscar líneas que comienzan por un "." y dos letras en minúscula:**

$ grep '^\.[a-z][a-z]' search-file.txt

Una expresión regular es un patrón que define un conjunto de cadenas sin enumerar todos sus elementos.

Una expresión regular está formada de caracteres y metacaracteres que tienen una función definida.

La principal utilidad de las expresiones regulares es la de describir un conjunto de cadenas, lo que resulta de utilidad en editores de texto y aplicaciones para buscar y manipular texto. Además, otro uso es la validación de un formato específico en una cadena de caracteres dada, como por ejemplo fechas, correos electrónicos o identificadores.

* **^**: el emparejamiento se debe realizar desde el principio de la cadena.
* **[A-Z]**: cualquier carácter entre la A mayúscula y la Z mayúscula.
* **{1,2}**: uno o dos caracteres.
* **\s**: un espacio en blanco.
* **\d**: un dígito.
* **{4}**: cuatro dígitos.
* **\s**: un espacio en blanco.
* **([B-D]|[F-H]|[J-N]|[P-T]|[V-Z])**: cualquier carácter entre la B mayúscula y la Z mayúscula, excepto las vocales.
* **{3}**: tres caracteres.
* **$**: el emparejamiento se debe realizar hasta el final de la cadena.

**Defina una expresión regular para encontrar (guardar) el valor de jsessionid obtenido de este fragmento de respuesta HTTP.**

<form action="/CurpSP/gobmx/consultaXDatos.do;jsessionid=4RZszr3-ir6BZJrYarbLP3rf\_DHpHAylrOmAXZTyAVMr6IsGC2tZ!-1644272920" method="post" accept-charset="UTF-8" role="form" id="consultaXDatos" name="consultaXdatos" autocomplete="off" novalidate="novalidate" class="bv-form"><button type="submit" class="bv-hidden-submit" style="display: none; width: 0px; height: 0px;" disabled="disabled"></button>

<input type="hidden" name="strtipo" value="A">

<div class="row">

<div class="col-sm-4">

<div class="form-group has-feedback">

<label class="control-label" for="nombre">Nombre(s)<span class="form-text">\*</span>:</label>

<input

**const regex = /jsessionid=[^=]+/gm;**

**()**

De forma similar que los corchetes, los paréntesis sirven para agrupar caracteres, sin embargo existen varias diferencias fundamentales entre los grupos establecidos por medio de corchetes y los grupos establecidos por paréntesis:

* Los caracteres especiales conservan su significado dentro de los paréntesis.
* Los grupos establecidos con paréntesis establecen una "etiqueta" o "punto de referencia" para el motor de búsqueda que puede ser utilizada posteriormente como se denota más adelante.
* Utilizados en conjunto con la barra "|" permite hacer búsquedas opcionales. Por ejemplo la expresión regular "al (este|oeste|norte|sur) de" permite buscar textos que den indicaciones por medio de puntos cardinales, mientras que la expresión regular "este|oeste|norte|sur" encontraría "este" en la palabra "esteban", no pudiendo cumplir con este propósito.
* Utilizados en conjunto con otros caracteres especiales que se detallan posteriormente, ofrece funcionalidad adicional.

**[ ]**

La función de los corchetes en el lenguaje de las expresiones regulares es representar "clases de caracteres", o sea, agrupar caracteres en grupos o clases. Son útiles cuando es necesario buscar uno de un grupo de caracteres. Dentro de los corchetes es posible utilizar el guión "-" para especificar rangos de caracteres. Adicionalmente, losmetacaracteres pierden su significado y se convierten en literales cuando se encuentran dentro de los corchetes. Por ejemplo, como vimos en la entrega anterior "\d" nos es útil para buscar cualquier carácter que representa un dígito. Sin embargo esta denominación no incluye el punto "." que divide la parte decimal de un número. Para buscar cualquier carácter que representa un dígito o un punto podemos utilizar la expresión regular "[\d.]".

**Defina una expresión regular para encontrar (guardar) el valor de rpid obtenido de este fragmento de respuesta HTTP.**

<!DOCTYPE html>

<html lang=en><head><meta charset=utf-8 /><meta name=viewport content="width=device-width,initial-scale=1,maximum-scale=1,user-scalable=0,shrink-to-fit=no"/><meta name=theme-color content="#000000"/><script type="text/javascript" src="/ruxitagentjs\_ICA27SVfjqrux\_10169190624124732.js" data-dtconfig="rid=RID\_551357822|rpid=853029560|domain=walmart.com.mx|reportUrl=/rb\_3770e195-dd69-4828-8cae-1067062e79aa|app=5fc495de4df505d6|featureHash=ICA27SVfjqrux|srsr=25000|rdnt=1|uxrgce=1|bp=2|srms=1,1,,,|uxrgcm=100,25,300,3;100,25,300,3|dpvc=1|lastModification=1565204434506|dtVersion=10169190624124732|tp=500,50,0,1|uxdcw=1500|agentUri=/ruxitagentjs\_ICA27SVfjqrux\_10169190624124732.js"></script>

**$re = '/rpid=[0-9]+/m';**

**Determine si el valor para el campo ‘featureHash’ deberia guardarse (correlacionarse),Si acaso si, explique el por que, si acaso no, explique el por que.**

<!DOCTYPE html>

<html lang=en><head><meta charset=utf-8 /><meta name=viewport content="width=device-width,initial-scale=1,maximum-scale=1,user-scalable=0,shrink-to-fit=no"/><meta name=theme-color content="#000000"/><script type="text/javascript" src="/ruxitagentjs\_ICA27SVfjqrux\_10169190624124732.js" data-dtconfig="rid=RID\_551357822|rpid=1651945784|domain=walmart.com.mx|reportUrl=/rb\_3770e195-dd69-4828-8cae-1067062e79aa|app=5fc495de4df505d6|featureHash=ICA27SVfjqrux|srsr=25000|rdnt=1|uxrgce=1|bp=2|srms=1,1,,,|uxrgcm=100,25,300,3;100,25,300,3|dpvc=1|lastModification=1565204434506|dtVersion=10169190624124732|tp=500,50,0,1|uxdcw=1500|agentUri=/ruxitagentjs\_ICA27SVfjqrux\_10169190624124732.js"></script>

**Token de función hash**

Un hash tiene la misión de ofrecer un resumen de ‘x’ bits a la salida que hace la función de huella digital en base a un documento de entrada de cualquier tipo de longitud.

Los hashses tienen una particularidad muy especial: es un proceso de un solo sentido. Si el algoritmo de este tipo está correctamente desarrollado debe ser casi imposible (muy difícil matemáticamente hablando) obtener el texto de entrada a partir de la salida hash.

Es decir, sacar el hash de algo es sencillo, pero hacer el camino inverso es prácticamente imposible en un buen algoritmo.

Un mismo texto siempre da el mismo mismo resultado. Si cambias cualquier cosa en el texto dará un hash totalmente diferente, por lo que puede, por ejemplo comprobarse si un fichero ha sido modificado sin necesidad de revisar letra por letra.

De ese modo, un hash tiene la capacidad de demostrar si una cantidad de datos se ha visto alterada, sin importar el tamaño de esa cantidad de datos.

Dentro de la seguridad informática y la criptografía el hash se usa sobre todo para garantizar la seguridad e integridad en diferentes campos y para generar identificadores únicos a su correspondiente información.

Los casos de uso más comunes para este algoritmo están en la comprobación de contraseñas en servicios web de todo tipo, la detección de archivos corruptos o manipulados, el firmado digital de los documentos, el proceso de prueba de trabajo del proceso de minar bitcoins e incluso en la creación de direcciones Bitcoin.

En todos los casos expuestos en el párrafo anterior se utiliza porque cuando un atacante malicioso quiere alterar un parámetro cualquiera de una contraseña, un archivo, un documento o una transacción de Bitcoin, el hash resultante varía y se puede detectar fácilmente la manipulación.

Si debe guardarse debido a que una feature hash funciona para la seguridad, sirve para poder hacer identificadores únicos a partir de un archivo o password.

Si yo utilizo un texto único como password siempre se generará el mismo hash pero si yo altero algo de ese texto, el hash será totalmente diferente y por eso nos podría servir para la correlación si ocupamos saber que x número de bits ocupa nuestro texto.

**Bit =** dígito binario unidad de medida de información que equivale a la selección entre dos alternativas que tienen el mismo grado de probabilidad.

A diferencia del sistema decimal, que utiliza diez dígitos (0, 1, 2, 3, 4, 5, 6, 7, 8 y 9), el sistema binario apela a sólo dos (0 y 1). Un bit, por lo tanto, puede representar a uno de estos dos valores (0 ó 1).

* **0 0** - Los dos están "apagados"
* **0 1** - El primero está "apagado" y el segundo "encendido"
* **1 0** - El primero está "encendido" y el segundo "apagado"
* **1 1** - Los dos están "encendidos"

**name=\”([a-zA-Z]+)\”**

**title=\”(.t)\”**

**([a-z])[A-Z]+.? (.)t?**