“Universidad Internacional de La Rioja en México”

Seguridad en Sistemas, Aplicaciones y Datos Masivos

Proyecto:

Actividad 2: Seguridad en Aplicaciones Ajax

Profesor:

OMAR URIEL DOMINGUEZ MENDOZA

Autor:

Juan Luis Cruz Aristeo.

Fecha de entrega:

22/07/2024

Índice

Introducción Ajax.

Las tecnologías de la información han avanzado y evolucionado a gran velocidad y uno de los objetivos que desean cumplir y por los cuales las tecnologías se han desarrollado tanto, es que los procesos y servicios que se realizan por medio de la web, sean más eficientes, rápidos y confiables, para que los usuarios tengan una experiencia agradable en el uso de las páginas web, esto se refiere que sean rápidas, que los datos e información carguen completos, que se actualicen de forma automática sin interrumpir la experiencia del usuario.

Para entender por qué se han enfocado en esto, es importante destacar las diferencias entre la web actual y la web de hace muchos años. Entre las diferencias más resaltadas encontramos que muchos servicios se han migrado a internet para ser realizados por este medio. En el caso de México, existen procesos como trámites de RFC, trámites de cédula profesional y trámites de predio, entre muchos otros. De igual forma, otra diferencia es la creciente área del comercio electrónico, donde hoy en día podemos encontrar una gran variedad de páginas de ventas o ecommerce que ofrecen una amplia gama de productos, desde ropa hasta consumibles, entre otras cosas.

Entonces, si suponemos la creciente migración de servicios en la actualidad hacia el internet, podemos comprender por qué fue necesario el rápido desarrollo de tecnologías hacia las páginas web para que sean de respuesta rápida, eficientes y confiables. Entre estas tecnologías encontramos AJAX, mejor conocido como (Asynchronous JavaScript And XML), que es una combinación de tecnologías usadas para el desarrollo de páginas web, que hace posible que los usuarios disfruten de estar en una página web, mientras en segundo plano la página va actualizando cierta información, pero esto sin interrumpir o volviéndola a cargar, hoy en día esta tecnología es usada en redes sociales, como twitter, Facebook, Instagram, etc. Es así que el objetivo de este entregable es explicar cómo funciona AJAX, cuáles son sus vulnerabilidades y cómo se podría defender alguien ante esas vulnerabilidades.

AJAX

Para poder empezar por entender AJAX, existe un término que debemos definir antes:

La arquitectura de las aplicaciones web ricas de Internet (RIA)

¿Qué son las RIAs?

Ría se refiere a un enfoque de desarrollo web que busca ofrecer una experiencia de usuario similar a la de las aplicaciones de escritorio, utiliza tecnologías web avanzadas. Las RIAs combinan la interactividad y capacidad de respuestas que tienen las aplicaciones de escritorio, con la fácil accesibilidad de las aplicaciones web.

Entre las tecnologías que componen las RIAs, encontramos que AJAX forma parte importante de estas realizando los siguientes trabajos.

1. **Renderización Dinámica**: Utiliza tecnologías como AJAX para actualizar partes de la página sin necesidad de recargarla completamente.
2. **Asincronía:** Utiliza AJAX para enviar y recibir datos del servidor sin interferir con la experiencia del usuario.

Entendiendo así que son las RIAs y la importancia de AJAX en ellas, podemos adentrarnos a la tecnología AJAX.

¿Qué es Ajax?

Para poder entender que es AJAX, hay que empezar por entender el desarrollo de una página web, ya que de esta forma será más sencillo entender cómo funciona AJAX para esto se describirá de forma breve que lenguajes componen una página web tradicional, y como se desarrollan mediante estos lenguajes resaltando que es solo una de las tantas formas en las que se puede desarrollar una página web y solo se ejemplifica para poder entender cómo es que trabaja la tecnología AJAX.

En primer lugar, HTML (HyperText Markup Language): Este lenguaje es el que se encarga de definir la estructura y el contenido de la página, es decir la información, esto mediante elementos y etiquetas, estas se encargan de crear encabezados, párrafos, listas, enlaces, imágenes, etc.

En segundo lugar, tenemos CSS (Cascading Style Sheets): Este lenguaje se encarga de diseñar la presentación y el diseño, pues su trabajo es controlar el aspecto visual, como

colores, fuentes, márgenes y disposición, generalmente es donde cargamos también la parte responsiva de una página.

En tercer lugar, encontramos a JavaScript: Este lenguaje permite agregar interactividad y comportamiento dinámico a las paginas, pues con él nos encargamos de manipular contenido HTML y CSS, responder a eventos del usuario y comunicarse con servidores.

En resumen, si ejemplificáramos de forma general como es que estos lenguajes se relacionan en el desarrollo de una página, sería como un cuerpo humano, HTML cumpliría la función del esqueleto de un cuerpo, CSS sería como la forma del cuerpo el color de piel, cabello, ojos, altura, etc. Finalmente, JavaScript, serían las funciones de ese cuerpo, como saltar, correr, hablar. De esta forma se puede apreciar cómo se relacionan los lenguajes entre sí.

Ahora podemos entender mejor Ajax, esta tecnología se encarga de que las páginas web, sean más rápidas y dinámicas, esto lo logra mediante el siguiente proceso. Anteriormente una página para poder actualizar información tenía que volver a cargar la página para actualizarla, pero AJAX se encarga de actualizar solo partes de la página sin tener que recargarla.

¿Cómo logra esto?

Un ejemplo breve que describiría como logra esto AJAX, es de la siguiente forma:

1.- El usuario entra a una página y realiza una acción (por ejemplo, hace clic en un botón de la página).

2.- Aquí entra la tecnología AJAX, JavaScript crea una solicitud HTTP y la envía al servidor en segundo plano.

3.- Una vez que el servidor lo procesa devuelve los datos.

4.- Finalmente JavaScript procesa la respuesta y actualiza solo la parte relevante sin recargarla completamente.

En resumen, AJAX permite que las aplicaciones web puedan enviar y recibir datos de un servidor de forma asíncrona, esto mejora en gran medida la experiencia de los usuarios, ya que se actualizan sin interrumpir la navegación y recargar toda la página. Pues JavaScript se encarga de realizar las solicitudes HTTP en segundo plano y manejar la respuesta de forma más dinámica, al final tenemos una página fluida y rápida.

Ahora que entendemos cómo es que trabaja AJAX de forma breve, procedemos a describir lo siguiente.

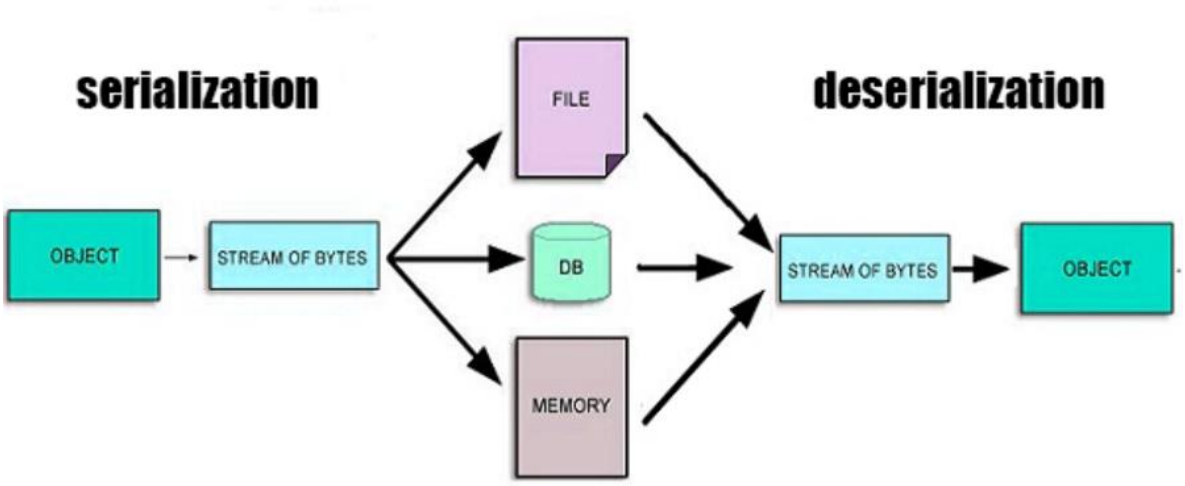
vulnerabilidades en la seguridad de AJAX.

A pesar de que AJAX introduce grandes beneficios, también introduce varias vulnerabilidades las cuales pueden ser explotadas si no se manejan correctamente, a continuación, se describen algunas vulnerabilidades a detalle:

- **Serialización de objetos JS con formato incorrecto**

La serialización es el proceso de convertir objetos en un formato que se pueda almacenar o transmitir. JavaScript permite la programación orientada a objeto (POO). Si la serialización no se hace correctamente se pueden introducir vulnerabilidades que un atacante puede explotar para inyectar datos maliciosos, esto sucede cuando los datos serializados no se validan adecuadamente antes de ser utilizados por el servidor.

Ejemplo: si alguien pudiera manipular un objeto JavaScript serializado para incluir código malicioso. Cuando este objeto manipulado es deserializado y ejecutado por el servidor, podría causar un comportamiento inesperado, comprometiendo la seguridad del sistema.



- **Cross-site scripting**

Como sabemos uno de los principales problemas o vulnerabilidades que se tiene con la tecnología AJAX es que no se validen los datos ingresados usuarios por los usuarios. Entonces tenemos el Cross-Site Scripting (XSS) el cual es una vulnerabilidad que permite a un atacante inyectar scripts maliciosos en las páginas. Esto ocurre debido a la falta de validación y filtrado de los datos ingresados por un usuario. Entonces puede ser utilizado para robar información sensible, como cookies.

Esto funciona de la siguiente forma:

1. Un usuario identifica un punto de entrada en una aplicación web e inyecta un script malicioso.
2. Cuando un usuario accede a la página que contiene el script malicioso, el navegador lo ejecuta como parte de la página. Lo que permite al atacante robar información, realizar acciones en nombre del usuario o manipular la interfaz.
3. Otro dato importante es que existen tipos de ataques XSS (persistentes o no persistentes) estos se diferencian en el tiempo en que se ejecutan, ya sea instantáneamente o se almacena para posteriormente afectar múltiples usuarios.

- **Validación solo de cliente en rutinas AJAX**

Las aplicaciones AJAX a menudo realizan validaciones en el lado del cliente (navegador del usuario) antes de enviarlos al servidor. El problema está en que confiar únicamente en esta validación es peligroso por alguien puede eludir estas validaciones y enviar datos maliciosos. Esto puede permitir la inyección de datos no deseados y comprometer la integridad del sistema.

Por ejemplo, un formulario web, este solo valida datos en el navegador puede ser manipulado por un atacante para enviar datos incorrectos al servidor, causando vulnerabilidades como inyecciones SQL o XSS.

- **Acceso entre dominios y política del mismo origen**

AJAX normalmente no puede acceder a datos de otros dominios debido a la política de seguridad del mismo origen, que es una medida de seguridad que se implementó por los navegadores. Pero los desarrolladores pueden utilizar técnicas como JSONP para quitar esta restricción lo que puede introducir riesgos si no se valida correctamente el flujo de datos entre dominios. Estas técnicas pueden permitir inyección de contenido malicioso mediante condigo Java Script.

Las vulnerabilidades mencionadas son una de tantas ya descubiertas, es importante entender que cada día se encuentran nuevas vulnerabilidades y siempre hay que estar atento a nuevas actualizaciones por parte de grupos donde se comparte información actualizada de nuevos tipos de vulnerabilidades descubiertas sobre la tecnología AJAX.

Principales mecanismos de defensa por implementar en toda la arquitectura de una aplicación web AJAX.

Si bien conocer la vulnerabilidad de esta tecnología es de suma importancia, de igual forma es de vital importancia conocer mecanismos de defensa para proteger una página web, que haga uso de tecnologías como AJAX. Para poder defender y a su vez poder ser más proactivos, por lo tanto, se describirán algunos mecanismos de defensa a continuación:

- **Validación de Entrada**

Como se ha resaltado con anterioridad las validaciones de entrada son una de las vulnerabilidades de las páginas web, que hacen uso de la tecnología AJAX y otras más, entonces debido a que es una vulnerabilidad muy conocida, se debe tener en las páginas web, validaciones de entrada que garanticen que los datos que ingresen los usuarios solo pasen aquellos bien formados. Esto ayuda a evitar el ingreso de datos malintencionados. Esta validación debe de hacerse desde el principio de preferencia, es decir desde que se reciban.

- **Codificación de salida**

Este mecanismo de defensa, va orientado a los ataques XXSS, donde un atacante puede insertar y ejecutar contenido malicioso dentro de una página web. De esta forma

entendemos que todas las variables de una página web. Deben de estar protegidas. Esto se logra mediante el proceso de pasar todas las variables por una validación y posteriormente escaparlas, así logramos la resistencia a la inyección.

Recordando que el término “escapado”, se refiere al proceso de transformar ciertos caracteres en cadena de texto. Con el objetivo de evitar que estos sean interpretados como código en contextos específicos, como HTML, JavaScript, CSS o URL.

Ejemplo:
Original: <script>alert('Hola');</script>

Escapado: <script>alert('Hola');</script>

- ## Política de seguridad de contenido (CSP)

La política de seguridad de contenido, se define como una capa adicional a la seguridad de las aplicaciones web. Esta medida de seguridad implementada por los navegadores web, permite que los administradores de los sitios web puedan controlar los recursos que pueden cargarse y ejecutarse en la página. Esto se logra mediante directivas o encabezados.

Funciona de la siguiente forma, cuando el navegador carga una página, verifica las directivas de CSP, las cuales indicar que tipos de contenido y de que fuentes están permitidos. Si alguno no cumple con las respectivas directivas simplemente se bloquea.

Es importante tomar en cuenta que estas políticas ayudan a que, en una página web, pueda defenderse ante ataque de inyección de scripts, puesto que con CSP se puede prevenir al restringir las fuentes de los scripts permitidos. En conclusión, los desarrolladores pueden especificar políticas detalladas que permiten solo los recursos necesarios para la funcionalidad de la página minimizando la superficie del ataque.

- ## Validación de Datos JSON y XML

La validación de datos JSON y XML, es un proceso de seguridad de aplicaciones web, la cual asegura que los datos recibidos en estos formatos sean correctos y seguros antes de ser

procesado. Lo que ayuda a evitar vulnerabilidades como el envenenamiento de XML y otras formas de inyección de código.

Recordando que JSON, es un formato de intercambio de datos ligero y fácil de leer para los humanos y de escribir para las maquinas. Es usado comúnmente para enviar datos entre un servidor y el cliente.

La importancia de validación de datos JSON y XML es vital pues se asegura de que los datos recibidos no contengan errores ni código malicioso que pueda comprometer la seguridad de la aplicación. Sin una validación adecuada, las aplicaciones son susceptibles a ataques que puedan resultar en la divulgación de información o corrupción de datos.

- ## Hoja de trucos de seguridad AJAX

A continuación, se harán descripciones de algunas prácticas recomendadas por el OWASP (Open Web Application Security Project) que es un organismo que se dedica a la seguridad de las aplicaciones web, sus materiales están disponibles de forma gratuita y son fácilmente accesibles.

- ## Referencia en texto (‘.innerText’) :

En el contexto de desarrollo web seguro, una de las practicas recomendadas por OWASP es utilizar **‘.innerText’** en lugar de **‘.innerHTML’** cuando se manipulen elementos DOM en JavaScript. Esta recomendación se debe, a que **‘.innerText’** codifica automáticamente el texto, Evitando así la mayoría de los problemas de Cross-Site Scripting (XSS).

- ## No usar (‘eval( )’) en JavaScript

La función **‘eval( )’** en JavaScript toma una cadena de texto como argumento y lo ejecuta como si fuera código JavaScript. Esto hace que se extremadamente peligroso y desaconsejado por varias razones. Pero la principal es la seguridad ya que, si una página permite que los usuarios ingresen texto que se procesa con eval, alguien puede hacer mal uso de esto ingresando código que se ejecutara y lo que puede llevar a una ejecución de código malicioso, como consecuencia puede resultar en Cross-Site Scripting (XSS).

Y como todo se hace extrema concienciación en las siguientes recomendaciones:

- **Nunca transmita secretos al cliente.**
- **No realice el cifrado en el código de lado del cliente.**
- **No ejecute lógica que afecta la seguridad**
- **Utilice la protección CSRF**
- **Evite crear XML o JSON a mano, utilice el marco.**

Para obtener más información sobre esta y otras prácticas recomendadas para evitar XSS en aplicaciones web, puedes consultar la hoja de referencia de OWASP en el siguiente enlace:

OWASP Cross-Site Scripting (XSS) Prevention Cheat Sheet: https://cheatsheetseries.owasp.org/cheatsheets/AJAX_Security_Cheat_Sheet.html

Conclusión

En la actualidad es un hecho que todas las empresas u organizaciones que se dedican al desarrollo web, mantenimiento y soporte tienen claro que tan necesario es seguir de forma rigurosa aquellas recomendaciones y marcos que se brindan con la intención de que las páginas web sean cada vez más seguras, aunque en realidad es imposible que sean perfectamente defendidas, si pueden tener un alto nivel en la fortaleza de sus páginas para no ser afectadas.

AJAX es una tecnología conformada por tecnologías, por lo tanto hay que señalar que la protección de la tecnología depende de la protección de cada una de las tecnologías que lo conforman, hoy en día existen un gran número de vulnerabilidades documentadas por organizaciones y profesionales para poder hacer frente a ellas mediante mecanismos de defensa que ellos mismos han documentado y publicado, por lo tanto para que una organización pueda llevar acabo los mecanismos de defensa podría tener equipos completamente dedicados la investigación e implementación de los mismos que deben llevarse a cabo para poder evitar intrusiones y problemas con las páginas que hagan uso de la tecnología AJAX.

Por otro lado, durante la investigación de este entregable, nos encontramos con la importancia de conocer cómo se desarrollan las páginas web, para poder entender el funcionamiento de AJAX, es por eso que se tuvo la necesidad de dar una breve explicación sobre cómo se desarrolla una página web. AJAX es sin duda una herramienta que hace que las páginas web de hoy en día sean rápidas y eficientes, el simple hecho de actualizar partes

de forma asíncrona es el gran beneficio de esta tecnología, nos dimos cuenta que redes sociales como Facebook, Instagram y YouTube son ejemplos claros de cómo funciona la tecnología, otro ejemplo seria el Gmail, que mientras se deje abierta la página se puede actualizar tu bandeja de correos sin necesidad de volver a cargar la página.

En lo personal, tenía conocimiento sobre cómo se desarrolla una página web tradicional, pero no tenía conocimiento de la tecnología AJAX, encontré que es de suma importancia el hecho de agregar esta tecnología, pero también durante la investigación encontré la importancia de estar informado sobre las vulnerabilidades de las páginas web, al final de cuentas termina sumando conocimiento para poder evitar problemas futuros en el desarrollo de páginas web, lo mejor de todo es que existen muchas páginas que brindan información con todas las vulnerabilidades que existen y como podrías implementar mecanismos de defensa.

BIBLIOGRAFIA

- Cloudflare. (n.d.). *OWASP Top 10*. Retrieved July 16, 2024, from https://www.cloudflare.com/es-es/learning/security/threats/owasp-top-10/

- Mozilla Developer Network (MDN). (n.d.). *Fetching data*. Retrieved July 16, 2024, from https://developer.mozilla.org/es/docs/Learn/JavaScript/Client-side_web_APIs/Fetching_data

- OWASP. (n.d.). *XML External Entity (XXE) Prevention Cheat Sheet*. Retrieved July 16, 2024, from https://cheatsheetseries.owasp.org/cheatsheets/XML_External_Entity_Prevention_Cheat_Sheet.html

- OWASP. (n.d.). *Cross-Site Scripting (XSS) Prevention Cheat Sheet*. Retrieved July 16, 2024, from https://cheatsheetseries.owasp.org/cheatsheets/Cross_Site_Scripting_Prevention_Cheat_Sheet.html

- Mozilla Developer Network (MDN). (n.d.). *Content Security Policy (CSP)*. Retrieved July 16, 2024, from https://developer.mozilla.org/en-US/docs/Web/HTTP/CSP

- Vargas, B. (n.d.). *¿Qué es la arquitectura AJAX?*. Retrieved July 16, 2024, from https://www.byronvargas.com/web/que-es-la-arquitectura-ajax/#google_vignette

- Google. (n.d.). *Serialización de objetos en JavaScript*. Retrieved July 16, 2024, from https://www.google.com/search?q=serializacion+de+objetos+enjavascript&oq=serializacio

n+de+objetos+enjavascript&gs_lcrp=EgZjaHJvbWUyBggAEEUYOTIJCAEQIRgKGKAB
MgkIAhAhGAoYoAHSAQg4NzI2ajBqN6gCALACAA&sourceid=chrome&ie=UTF-8

- IBM. (n.d.). *Asynchronous JavaScript and XML (AJAX) overview*. Retrieved July 16, 2024, from https://www.ibm.com/docs/es/rational-soft-arch/9.6.1?topic=page-asynchronous-javascript-xml-ajax-overview

- Amazon Web Services (AWS). (n.d.). *What is AJAX?*. Retrieved July 16, 2024, from https://aws.amazon.com/es/what-is/ajax/