v. 3, n. 31, 2023

Journal of Engineering Research

VULNERABILIDADES DE SEGURIDAD Y DEFENSAS EN APLICACIONES AJAX (ING. EN TIC'S EN EL TECNM CAMPUS APIZACO)

José Antonio Cruz Zamora
Profesor del Departamento de Sistemas y Computación en la carrera de Ingeniería en Tecnologías de la Información y Comunicaciones en el Instituto Tecnológico de Apizaco del Tecnológico Nacional de México, Apizaco Tlaxcala

Elizabeth Cuatecontzi Cuahutle
Profesora del Departamento de Sistemas y Computación en la carrera de Ingeniería en Tecnologías de la Información y Comunicaciones, y de la Maestría en Sistemas Computacionales en el Instituto Tecnológico deApizaco del Tecnológico Nacional de México, Apizaco Tlaxcala

Miquelina Sánchez Pulido
Profesora del Departamento de Sistemas y Computación en la carrera de Ingeniería enTecnologías de la Información en el Instituto Tecnológico de Apizaco del Tecnológico Nacional de México, Apizaco, Tlaxcala

Haydee Patricia Martínez Hernández
Profesora del Departamento de Ingeniería Eléctrica y Electrónica e imparte clases en la Ingeniería Electrónica e Ingeniería mecatrónica, en la Maestría de Ingeniería Mecatrónica y es Coordinadora del Doctorado en Ciencias de la Ingeniería

Marcedeli Romero Brav0
estudiante de la carrera de Ingeniería en Tecnologías de la Información y Comunicaciones en el Instituto Tecnológico de Apizaco del Tecnológico Nacional de México, Apizaco Tlaxcal

Resumen: En este artículo se realiza un análisis de las vulnerabilidades que se tienen en aplicaciones desarrolladas con arquitectura y herramientas AJAX, ya que al tener una mayor área de exposición del lado de cliente se corre un mayor riesgo porque existe una mayor área de ataque. Se realiza una revisión del desarrollo web y las aplicaciones de internet enriquecidas, así como la arquitectura y tecnologías de AJAX para pasar a una revisión de las principales amenazas que marca el top ten de OWASP y sus sugerencias de solución para crear mejores prácticas en el desarrollo de software seguro.
Palabras clave: Aplicaciones AJAX, Vulnerabilidades OWASP, Aplicaciones RIA, Tecnologías AJAX.

INTRODUCCIÓN

La Web 1.0 se inicia con el desarrolló del HTML por Teen Berners Lee, estando en el CERN (Luján, 2015), en el cual se puede navegar en una forma amigable, ya que al desarrollar una serie de marcas que permiten organizar el despliegue de materiales como texto e imágenes, generaliza las posibilidades de acceso por personas que no necesitan ser estudiosos expertos en el área de cómputo ni que dominen el entorno de una consola, además Teen Berners Lee desarrolló la World Wild Web como una herramienta de navegación en un equipo NEC que permitió una calidad gráfica única. En ese tiempo se detona el desarrollo de sitios que solo comparten contenido y en el que los usuarios pueden acceder a ellos, el que crea el sitio pone contenido para que sea leído por los que lo acceden, es decir comunicación en un solo sentido.

La Web 2.0 se le conoce a la aparición de desarrollos colaborativos, el crecimiento de sitios en donde uno no sólo accede a información, sino que puede participar en la construcción de este, puede modificar o

aportar el propio contenido, se caracteriza por la proliferación de redes sociales y sitios de chat, donde se participa activamente y se genera comunicación en grupos que uno puede restringir para que sólo participen un grupo específico de amigos.

LiveScript aparece por pimera vez en la versión beta de Netscape Navigator 2.0 en septiembre de 1995, y el 4 de diciembre, en un comunicado conjunto con Sun, es renombrado como JavaScipt. Su núcleo (Kernel) contiene objetos como Array, Date, Math, Number, String y operadores, estructuras de control y sentencias. Amplía el lenguaje en el cliente, añadiendo objetos que permiten controlar el navegador y su DOM. En el servidor añade objetos que son útiles cuando JavaScript se ejecuta en este (Luján, 2015).

En el año 2002, en un documento publicado por Macromedia (ahora Adobe) aparece publicado el término RIA acrónimo de Rich Internet Application (Aplicaciones Ricas en Internet) y busca tener ventajas sobre los desarrollos tradicionales. Se busca realizar aplicaciones WEB con las funcionalidades de las aplicaciones de escritorio, con la ventaja de que no es necesario que el usuario tenga instalada la aplicación en su equipo ya que es accesible desde cualquier navegador, por ende, tendremos aplicaciones multiplataforma que funcionarán con un navegador y acceso a internet (Pérez Mata, 2008).

AJAX se presenta por primera vez en el artículo "Ajax: A New Approach to Web Aplicaction" publicado por Jesse James Garret el 18 de febrero de 2005. El término es un acrónimo de Asinchronous JavaScript + XML. Se refiere a un Técnica basada en los Navegadores en donde los elementos de una aplicación web son recuperados de una manera asíncrona y en segundo plano, lo cual permite un despliegue de elementos de la página sin recargar la página completa. Es una técnica RIA que permite el desarrollo de aplicaciones interactivas en el navegador y mantiene una comunicación asíncrona con el servidor.

Al incrementar el código del lado del cliente, se genera una mayor área de ataque, por lo cual las aplicaciones se encuentran más expuestas si no se realiza una serie de prácticas adecuadas para el desarrollo de software seguro. En el presente documento se hará una revisión a las aplicaciones AJAX, a la arquitectura y tecnologías que utiliza, las vulnerabilidades y defensas que se pueden (realizar) poner en práctica para desarrollar aplicaciones AJAX seguras y finalmente las conclusiones.

DESARROLLO APLICACIONES AJAX

AJAX permite implementar con gran cantidad de código JavaScript, que se ejecute con robustez y rendimiento aceptable. En AJAX existen bloques constructores que aceleran el proceso de desarrollo de mediana a gran escala. Estos bloques se clasifican en cuatro categorías:

Snippets: Fragmentos de código se agregan sin grandes cambios.

Widgets: elementos pequeños de la Interfaz de usuario que son autocontenidos: Calendarios, menú jerárquico, menú de acordeón.

Frameworks: entorno de ejecución del lado del cliente con utilidad de funciones y widgets.

Frameworks avanzados: Un entorno con herramientas de desarrollo y componentes del lado del servidor que también incluye elementos del lado del cliente (Bazán, 2008).

Una aplicación WEB tradicional, requiere recargar la página cada vez que se realiza una petición o se Accesa un link, mientras que, en AJAX, el cambio se realiza a nivel de segmento, con lo cual se tiene los beneficios de una respuesta rápida, menor tráfico de datos, mayor interactividad y una mejor experiencia

de usuario.

Trabajar con AJAX puede tener las ventajas de poder comenzar a desarrollar fácilmente con snippets y widgets, adoptar algún framework robusto de desarrollo, incorporar el diseño centrado en el usuario buscando mejorar la usabilidad; sin embargo, se puede caer en el engaño de que sólo se centre en el procesamiento del lado del cliente y olvidarse de la seguridad que implica la interacción del cliente con el servidor y la gran cantidad de código que se encuentra del lado del cliente y que pudiera ser aprovechado por un hacker o usuario con malas intenciones.

ARQUITECTURA Y TECNOLOGÍAS

La arquitectura de una aplicación WEB tradicional es como se muestra en la figura l . El cliente realiza una petición, el navegador la solicita al servidor y existe un tiempo de espera en el momento que se están transfiriendo los datos, en el de la petición y en el de la carga del contenido. Y cada vez que el cliente solicite se generará tiempo de espera en lo que el servidor la recibe, la procesa, la construye en HTML y la regresa.

Figura 1. Modelo tradicional de una aplicación WEB

En la figura 2. Se muestra la arquitectura en AJAX. Una vez cargada la aplicación AJAX, el motor AJAX se encarga de realizar las peticiones en forma asíncrona, con la diferencia de que el usuario puede seguir trabajando y al realizarse una petición, una vez que recibe los cambios, el manejador de eventos se encarga de hacer las modificaciones de la página por medio de DOM. Es decir, no sólo se carga la página, sino el código del motor AJAX que realizará la comunicación con el servidor, de manera que el usuario estará interactuando con el motor y cuando ocurra un evento que lo requiera, el motor AJAX realizará la petición al servidor y actualizará la página para el usuario (Oxlaj Mangandi, 2008).

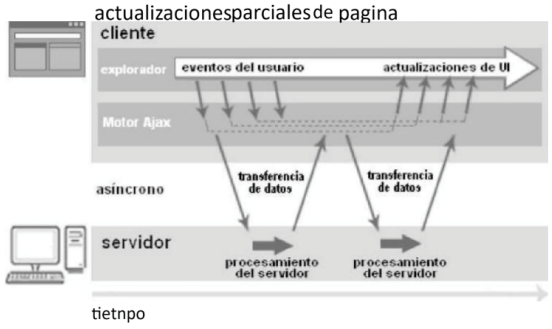

Figura 2. Modelo AJAX de una aplicación WEB

Las tecnologías que forman AJAX son: XHTML y CSS para crear una presentación basada en estándares; DOM, para la Interacción y manipulación dinámica del contenido; XML, XSLT y JSON, para intercambiar y manipular datos; XMLHttpRequest, para el intercambio asíncrono de información. JavaScript que permita aglutinar todas las tecnologías como se muestra en la figura 3 (Eguíluz Pérez, 2012).

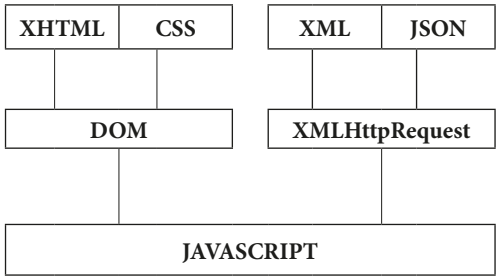

Figura 3. Tecnologías agrupadas bajo el concepto AJAX

VULNERABILIDADES DE SEGURIDAD Y DEFENSAS:

Las IO principales vulnerabilidades de las aplicaciones WEB de OWASP en 2020 son:

- Inyección
- Autenticación rota
- Exposición de datos sensibles
- Entidades externas XML (XXE)
- Control de acceso roto
- Errores de configuración de seguridad
- Secuencias de comandos entre sitios (XSS)
- Deserialización insegura
- Uso de componentes con vulnerabilidades conocidas
- Registro y seguimiento insuficientes

INYECCIÓN

Ocurre una inyección de código cuando un usuario malintencionado, envía datos incorrectos con la intención de que la aplicación realice algo diferente de la función para la que fue programada.

En JavaScript se puede crear una función que elimine las etiquetas HTML y caracteres definidos que indiquen que se puede estar tecleando alguna instrucción como entrada de datos; también se puede limitar la longitud de la cadena de entrada en los formularios, evitando la inyección de código mal intencionado.

Además, la construcción del desarrollo debiera incluir un ORM que filtre las consultas SQL en vez de realizarlas directamente, trasladando a un modelo de objetos donde se recuperen los datos por persistencia.

Autenticación Rota.

El proceso de Autenticación se realiza del lado del Servidor, por lo que se puede utilizar un patrón de diseño modelo -vista — controlador, que permita realizar la autenticación de los datos enviados en la petición y un control de sesiones.

Por el lado del cliente, se puede bloquear cuando se realice más de cierto número de intentos o verificar si el que intenta ingresar es una persona con chequeo de imágenes o caracteres ofuzcados dentro de una imagen, para evitar ataques automáticos con lista de nombres.

Cerrar la sesión después de un cierto tiempo de inactividad e invalidar cuando se establece el cierre de sesión.

Se debe establecer normas para la construcción de contraseñas robustas, que incluyan caracteres especiales, números y letras.

Exposición de datos sensibles.

El código de las aplicaciones JavaScript está disponible para que pueda ser visualizado del lado del cliente, por lo que se han generado ofuscadores de código que utilizan diversos mecanismos para hacer casí imposible de entender el código fuente, manteniendo el funcionamiento de la aplicación. Un ejemplo lo podemos ver con el siguiente código fuente:

```
// Calculate salary for each employee in
“aEmployees”. // “aEmployees” is array of
“Employee” objects.


function CalculateSalary(aEmployees)

    var nEmpIndex = 0; while (nEmpIndex < aEmployees.
    length)
        var oEmployee = aEmployees[nEmpIndex];
        oEmployee.fSalary =
        CalculateBaseSalary(oEmployee.nType,
        oEmployee.nWorkingHours); if (oEmployee.
        bBonusAllowed == true)


        oEmployee.fBonus =
        CalculateBonusSalary(oEmployee.nType,
        oEmployee.nWorkingHours, oEmployee.fSalary);
        else oEmployee.Bonus = 0;


        oEmployee.sSalaryColor =
        GetSalaryColor(oEmployee.fSalary + oEmployee.
        fBonus); nEmpIndex++;
```

Una vez que se pasa por el ofuscador el código resultante es:

```
function c(g) {var m=0;while(m<g.length) {var r=g[m];r.l=d(r.n,r.o);if(r.j==true) {
r.k=e(r.n,r.o,r.l);} else r.k=0;}r.t=f(r.l+r.k);m++;}}
```

Es importante, además, cifrar los datos en tránsito y almacenar las contraseñas, utilizando sólidas funciones hash adaptativas.

ENTIDADES EXTERNAS XML (XXE)

Este tipo de ataque ocurre cuando la entrada XML tiene una referencia externa y ésta es procesada por el analizador XML que se encuentra configurado débilmente.

En JavaScript se puede trabajar con JSON, evitando entradas XML.

CONTROL DE ACCESO REMOTO

JavaScript no puede realizar conexiones con otros dominios externos, por lo que si un atacante obliga al navegador a dirigirse a otra URL se genera el problema de dominios diferentes.

Se debe aplicar la autenticación multifactorial para todos los puntos de acceso. Se deben restringir los privilegios.

SECUENCIAS DE COMANDOS ENTRE SITIOS (XSS)

Consiste en inyectar scripts maliciosos del lado del cliente en un sitio web y utilizar el sitio web como método de propagación.

Con JavaScript se puede realizar la validación de cadenas para todo tipo de entradas, eliminando la posibilidad de inyección de scripts y se puede limitar el tamaño sólo a la longitud del dato de entrada, así como el tipo de carácter que se esté tecleando para agregarlos a la cadena.

USO DE COMPONENTES CON VULNERABILIDADES CONOCIDAS

En el proceso de desarrollo se debe establecer qué tipo de componentes validados pueden ser utilizados por los desarrolladores y evitar componentes que tengan vulnerabilidades en los repositorios de código.

CONCLUSIONES

Desde la aparición de las Aplicación de WEB con características de Internet enriquecido se trabaja con mayor interactividad, por lo cual los desarrollos que aplican el grupo de tecnologías para tener aplicaciones AJAX se ha incrementado; sin embargo, al incrementar el código del lado del cliente se genera una mayor área para los usuarios malintencionados y se incrementa la posibilidad de vulnerabilidades en el software desarrollado. Se deben integrar mayores prácticas en el proceso de desarrollo cuando se realizan aplicaciones AJAX, tener repositorios con componentes probados y sin vulnerabilidades y utilizar solo los framewroks robustos que ya mostraron su confiabilidad para el desarrollo de software seguro.

REFERENCIAS

Luján, M. S. (2015). Programación de Aplicaciones WEB. Alicante: Club Universitario.

Bazán, P. (2008). AJAX: un análisis tecnológico y posibilidades metodológicas. Buenos Aires, Argentina: Laboratorio de Investigación en Nuevas Tecnologías Informáticas .

Eguíluz Pérez, J. (2012). Introducción a AJAX. Madrid, España: www.librosweb.es.

0x1aj Mangandi, L. A. (2008). Fuerzas y debilidades de AJAXcomo nuevo enfoque para el desarrollo de aplicaciones WEB. Guatemala : Universidad de San Carlos de Guatemala.

Pérez Mata, M. (2008). Evaluación y pruebas de aplicaciones RIA con AJAX. Cataluña, España: Universidad Politécnica de Cataluña.

NOTAS BIOGRÁFICAS

El MISSI José Antonio Cruz Zamora es profesor de la carrera de Ingeniería en Tecnologías de la Información y Comunicaciones en el Instituto Tecnológico de Apizaco del Tecnológico Nacional de México, coautor del modelo de Proyectos Integradores para el desarrollo de competencias profesionales del SNIT, colaborador en 20 reuniones Nacionales de Innovación y diseño Curricular Basada en competencias, ha publicado más de 20 artículos en congresos y revistas de otros índices.

La MDIS Elizabeth Cuatecontzi Cuahutle en el Instituto Tecnológico de Apizaco es profesora de tiempo completo del departamento de Sistemas y Computación, en la carrera de Ing. en Tecnologías de la información y Comunicaciones y colaboradora en la Maestría en Sistemas Computacionales en la línea de Investigación de Ing. de Software, ha sido Jefa del Departamento de Gestión Tecnológica y Vinculación de Enero 2007 a febrero de 2010, Jefa del Departamento de Sistemas y Computación de Marzo de 2010 a noviembre de 2013. Actualmente es jefa de proyectos de Docencia, coordinando las actividades para la reacreditación de la carrera de Ingeniería en Tecnologías de la información y Comunicaciones por el CONAIC A. y, coordinadora del módulo de especialidad de Ing. de Software de la carrera de Ing. en Tecnologías de la Información.

La Lic. Miquelina Sánchez Pulido es Profesora del Departamento de Sistemas y Computación en la Carrera de Ingeniería en Tecnologías de la Información y Comunicaciones en el Instituto Tecnológico de Apizaco del Tecnológico Nacional de México, actualmente es Jefa de Proyectos de Vinculación del Departamento de Sistemas y Computación y ha publicado diversos artículos en congresos y revistas.

La Dra. Haydee Patricia Martínez Hernández es Profesora Investigadora en el Tecnológico Nacional de México /Tecnológico de

Apizaco, Docente de la Academia de del Depto. De Ing. Eléctrica y Electrónica. Miembro del Consejo de la Maestría en Ingeniería Mecatrónica.

Miembro del Claustro Doctoral en el Doctorado en Ciencias de la Ingeniería. Enseñanza de Electrónica Digital, Analógica, Control y

Automatización, Programación de Microcontroladores y Planes de negocios a estudiantes de Licenciatura y Maestría. Coordinadora del Doctorado en Ciencias de la Ingeniería. Manejo de software de Ensamblador, CCS, Proteus, LabView, Origi y simuladores para microcontroladores, realizó depósito de materiales semiconductores para la realización de dispositivos foto y electroluminiscentes

La Alumna Marcedeli Romero Bravo es alumna del octavo semestre de la carrera de Ingeniería en Tecnologías de la Información y Comunicaciones en el Tecnológico Nacional de México/ Instituto Tecnológico de Apizaco.