Ajax fingerprinting for Web 2.0 Applications

Shreeraj Shah
Founder, Net Square
shreeraj@net-square.com

Introduction

Fingerprinting is an age old concept and one that adds great value to assessment methodologies. There are several tools available for fingerprinting operating systems (*nmap*), Web servers (*httprint*), devices, etc. Each one of these tools uses a different method – inspecting the TCP stack, ICMP responses, HTTP responses. With this evolution of Web 2.0 applications that use Ajax extensively, it is important to fingerprint Ajax tools, framework or library used by a particular web site or a page. This paper describes the method of doing Ajax fingerprinting with a simple prototype serving as an example.

Ajax fingerprinting can help in deriving the following benefits:

- ***Vulnerability detection*** – Knowledge of the framework on which a web application is running, allows the mapping of publicly known vulnerabilities found for that particular framework. Example – DWR client side vulnerability
- ***Architecture enumeration*** – On the basis of derived information from fingerprinting it is possible to guess application architecture and inner working of a system. Example – Atlas (.NET application framework), DWR (Servelet/JavaScript combo)
- ***Assessment methodology*** – Derived information from the fingerprinting phase can help in defining future assessment path and vulnerability detection methods. Example – Deciding on JavaScript-scanning

Methodology

Ajax fingerprinting is a very simple process. It can be done with one request to the target page. It is possible to advance this technique with multiple requests by adding some complexity to help in detecting obfuscation mechanisms. Let's focus on the simple technique first. Fetch an HTML page from the target and look for JavaScript dependencies residing in the "script" tag as shown below:

```
<script src="./src/prototype.js"></script>
```

It is easy to grab these lines with regular expressions. This snippet to extract the "src" attribute is written using simple ruby code. The entire code of this script is listed in Exhibit 1.

```
a_script=page.scan(/<script.*?>/)
a_script.each do |temp|
  if(temp.scan("src").length > 0)
    temp += "</script>"
    doc=Document.new temp
    root = doc.root
    all_path += root.attributes["src"]+"|"
    puts root.attributes["src"]
  end
end
```

Grab all script dependencies and compare them with entries in the local database. We are trying to fingerprint on the basis of "filename". Here is a simple signature for, say, a DWR framework.

```
# DWR fingerprinting ...
auth.js
engine.js
util.js
```

If JavaScript source dependencies are found with the list of filenames, then probable candidates can be derived using the following simple routine. In this routine, each file name residing in the database is compared against the set of "src"s found in the HTML page.

```
File.open("ajaxfinger-db",'r') do |temp|
  while line = temp.gets
    if not (/^#/.match(line))
      if(all_path.scan(line.chomp).length>0)
        puts "[+]  "+line.chomp + " [..Found..]"
      end
    else
      puts line
    end
  end
end
```

This is a very simple way of fingerprinting the target application. For example, if you run it against, say, *digg.com*.

```
D:\ajaxfinger>ajaxfinger.rb http://digg.com

---Scanning for scripts---
/js/4/utils.js
/js/4/xmlhttp.js
/js/4/wz_dragdrop.js
/js/4/hover.js
/js/4/label.js
/js/4/dom-drag.js
/js/4/prototype.js
/js/4/scriptaculous.js
/js/4/lightbox.js
/js/4/swfobject.js
/js/4/hbxdigg.js
/js/4/digg_hbx_migration.js
/js/tooltip.js


...Continued on the next page
```

```
---Fingerprinting Ajax frameworks---
# Prototype fingerprinting ...
[+]  prototype.js [..Found..]
# script.aculous fingerprinting
[+]  dragdrop.js [..Found..]
[+]  scriptaculous.js [..Found..]
# Dojo toolkit ...
# DWR fingerprinting ...
# Moo.fx fingerprinting ...
# Rico fingerprinting ...
# Mochikit fingerprinting ...
# Yahoo UI! fingerprinting ...
# xjax fingerprinting ...
# GWT fingerprinting ...
# Atlas fingerprinting ...
# jQuery fingerprinting ...
```

As you can see in above result, *digg.com*'s index page extracts files for *prototype* and *script.aculous* framework. Ajax technology fingerprinting for this web site/page can be done on the basis of this information.

Accuracy and unveiling obfuscation

It is possible that the name of the file may change or that the file name is "*prototype.js*". It is customized and has nothing to do with the "prototype" framework. To deal with *false positives* or *false negatives,* function-level signatures may be required to be captured in terms of function name for each one of these frameworks and compared with all target JavaScript files. More downloads from the target server, in addition to JavaScript parsing functionality, is needed. This can be done with a JavaScript interpreter such as *rbNarcissus*. This approach is not implemented in this version of the script.

Conclusion

Ajax fingerprinting can play a key role when doing black box assessment with Web 2.0 applications. This method can help security professionals in enumeration of Ajax frameworks. Consequently, the information obtained can lead to architecture exposure and known vulnerabilities.

Exhibit 1 – ajaxfinger.rb *[Ruby code for simple implementation]*

Note: I will upload latest script (This may get changed from time-to-time) on my site http://www.net-square.com/ns_freetools.shtml

```
# Ajax fingerprinting script using local database (ajaxfinger-db)
# Author: Shreeraj Shah (shreeraj@net-square.com)

require 'open-uri'
require 'rexml/document'
include REXML

if (ARGV.length != 1)
  puts "\nUsage:\n"
  puts "ajaxfinger.rb <Target URL>\n"
  puts "Example:\n"
  puts "ajaxfinger.rb http://digg.com\n"
  Kernel.exit(1)
end

url = ARGV[0]
html = open(url)
page = html.read
all_path = ""

puts "\n---Scanning for scripts---"
a_script=page.scan(/<script.*?>/)
a_script.each do |temp|
  if(temp.scan("src").length > 0)
    temp += "</script>"
    doc=Document.new temp
    root = doc.root
    all_path += root.attributes["src"]+"|"
    puts root.attributes["src"]
  end
end

puts "\n---Fingerprinting Ajax frameworks---"
File.open("ajaxfinger-db",'r') do |temp|
  while line = temp.gets
    if not (/^#/.match(line))
      if(all_path.scan(line.chomp).length>0)
        puts "[+]  "+line.chomp + " [..Found..]"
      end
    else
      puts line
    end
  end
end
```

Exhibit 2 – ajaxfinger-db *[Database in cleartext for name comparison]*

```
# Prototype fingerprinting ...
prototype.js
# script.aculous fingerprinting
builder.js
controls.js
dragdrop.js
effects.js
scriptaculous.js
slider.js
unittest.js
# Dojo toolkit ...
dojo.js.uncompressed.js
dojo.js
# DWR fingerprinting ...
auth.js
engine.js
util.js
DWRActionUtil.js
# Moo.fx fingerprinting ...
Moo.js
Function.js
Array.js
String.js
Element.js
Fx.js
Dom.js
Ajax.js
Drag.js
Windows.js
Cookie.js
Json.js
Sortable.js
Fxpack.js
Fxutils.js
Fxtransition.js
Tips.js
Accordion.js
# Rico fingerprinting ...
rico.js
# Mochikit fingerprinting ...
MochiKit.js
```

Exhibit 2 – ajaxfinger-db *…Contd.*

```
# Yahoo UI! fingerprinting ...
animation.js
autocomplete.js
calendar.js
connection.js
container.js
dom.js
event.js
logger.js
menu.js
slider.js
tabview.js
treeview.js
utilities.js
yahoo.js
yahoo-dom-event.js
# xjax fingerprinting ...
xajax.js
xajax_uncompressed.js
# GWT fingerprinting ...
gwt.js
search-results.js
# Atlas fingerprinting ...
AtlasRuntime.js
AtlasBindings.js
AtlasCompat.js
AtlasCompat2.js
# jQuery fingerprinting ...
jquery.js
jquery-latest.pack.js
jquery-latest.js
```