

Performance Ajax tools

Skill Level: Intermediate

Judith M. Myerson (jmyerson@bellatlantic.net)
Systems engineer and architect

13 May 2008

Wasting server resources can impact the performance of Ajax applications, resulting in excessive HTTP requests, high memory consumption, and the need for an unusual amount of polling to make applications work. Regular developerWorks author Judith Myerson suggests some open source tools and Firefox add-ons you can use to improve or solve problems with your Ajax applications.

Introduction

In my developerWorks article "[Speed up your Ajax applications while dodging Web services vulnerabilities](#)," I discuss excessive bandwidth, frequent HTTP requests, and memory leaks as being contributors to network bottlenecks. The article shows how to eliminate or reduce bottlenecks to improve the performance of your Ajax applications.

In this article, you'll learn of some useful performance tools that can help you achieve the following:

- Reduce the number of HTTP requests your application makes
- Track down I/O disk issues
- Analyze network traffic
- Discover excessive calls
- Reduce memory consumption
- Solve other performance problems

The tools I describe can be divided into two categories: open source tools and Firefox add-ons.

Open source tools

This section covers the following tools:

- **Apache Bench:** Simulates a load on a server
- **Tsung:** Tests multi-protocol loads
- **Bonnie++:** Tracks down I/O disk issues
- **Wireshark:** Analyzes network traffic
- **Comet server application tools:** Used for longer-lived connections, higher volumes of concurrency, lower latency, and lower server load

This article assumes that you have already optimized JavaScript and HTML codes to remove excessive calls and unnecessary tags and redundant lines. If you have not done so, I suggest using the Dojo Toolkit (an Ajax toolkit) to eliminate excessive calls before you attempt to test server performance with Firefox (see [Resources](#) for a link to Dojo as well as links to the other tools introduced here).

Apache Bench

If you need to simulate a load on an Apache server (or any Web server), you can use Apache Bench, which is included in the standard Apache HTTPd distribution. This benchmarking tool will launch connections to your Web server to simulate multiple users. Use the following three options to get the information you want; otherwise, the default will not give representative benchmarking results.

Options	Description	Default
-n requests	Get the number of requests the server can process per second	Perform just one single request
-c concurrency	Get the number of simultaneous requests to perform per server before performance suffers	Perform one HTTP request at a time; that is, no concurrency
-t timelimit	Set the number of seconds to spend for the benchmarking session up to 50000	There is no time limit

Other options allow you to control precisely the request to send, the proxy settings, user authentication, and cookies.

While benchmarking with Apache Bench, you may want to log client requests for later analysis. You can do this with Apache Module `mod_log_config`, located in `mod_log_config.c`

Logs are written in a customizable format and may be written directly to a file or to an external program. Conditional logging is provided so that individual requests may be included or excluded from the logs based on characteristics of the request.

This module provides the following:

Directives	Description
------------	-------------

TransferLog	Create a log file
LogFormat	Set a custom format
CustomLog	Define a log file

The TransferLog and CustomLog directives can be used multiple times on each server to cause each request to be logged to multiple files.

Tsung

Tsung is an open source, multi-protocol, distributed load testing tool. Its purpose is to simulate users in order for developers to test the scalability and performances of IP-based client/server applications. You can use it to do load and stress testing of your HTTP servers running Ajax applications. It can be distributed on several client machines and is able to simulate thousands of virtual users concurrently in various scenarios.

Bonnie++

Bonnie++ lets you track down disk I/O issues that could result, for example, in slowing down SQL queries in any Ajax application and affecting the performance of your database as well. For example, simple `EXPLAIN` statements in the queries and a large amount of query log analysis will slow down the queries.

Bonnie++, written in C++, can be compiled under different UNIX flavors and run on Windows®-based machines. Here is a sample listing of Bonnie++ commands that you can use on your UNIX® machine:

Listing 1. Bonnie++ commands

```
-u root \
-d /mnt/vol-001/test-vol001/testfiles \
-s 2016M \
-n 10:102400:1024:1024 \
-m bonnie \
-q > /var/tmp/bonnie_test.csv 2> /var/tmp/bonnie_test.out
```

The following table explains what each command does:

Command	Description
-u root	Run as root.
-s 2016M	Specify the file size created for the sequential tests. The option specifies twice the system memory size (1GB) to avoid testing the cache.
-d /mnt/vol-001/test-vol001/testfiles	Specify where test files are to be created.
-n 10:102400:1024:1024	Specify what the file size is, what the largest number of files is, what the smallest number of files is, and how many directories are to be created. The option specifies file size is 10MB,

	maximum number of files is 102400, minimum number files is 1024, and the number of directories is 1024.
-m bonnie	Give a Machine Name for the output.
-q	Quiet mode so that we can redirect the output to files -- <code>stdout</code> is the csv format output and <code>stderr</code> is the human-readable format output.

You can change the specifications. You can run each Bonnie++ test as many times as you want (5 or 10 times), but be sure to allow a 60-second sleep between each run to let everything settle, so you can average the results.

Wireshark

Wireshark, formerly known as Ethereal, is a multiplatform network protocol analyzer. It can read live data from Ethernet, Token-Ring, FDDI, serial (PPP and SLIP), 802.11 wireless LAN, ATM connections, and any device supported on Linux® by the libpcap interface.

You can inspect hundreds of protocols, analyze live and offline capture, and work with a three-pane packet browser. You can choose to dump network traffic, dump and analyze traffic, or interactively dump and analyze network traffic. You can generate a capture file from an ASCII hexdump of packets and then edit and merge capture files.

Included in the Wireshark distribution are the following `man` pages. Find out what each does by typing the `man` command at the prompt on UNIX / POSIX systems, or view the HTML files from the Start menu on Windows systems.

Command	Description
capinfos	Print information about capture files
dumpcap	Dump network traffic
editcap	Edit and/or translate the format of capture files
mergcap	Merge two or more capture files into one
text2pcap	Generate a capture file from an ASCII hexdump of packets
tshark	Dump and analyze network traffic
wireshark-filter	Wireshark filter syntax and reference
wireshark	Interactively dump and analyze network traffic

Comet Server implementation tools

Real-time or highly collaborative applications require a significant amount of Ajax polling to make them work. The browser explicitly requests chunks of data used to

update the current page. It regularly polls the server, asking it if a new event has occurred (for example, every five seconds, asking the server: "Any new events?")

The problem is that when polling is repeated for new events and gets no responses from the server (for example, "No new events, try again later"), high polling frequencies may waste server resources and bandwidth by processing negative responses. Applications with predictable information updates or those that tolerate relatively infrequent updates can use polling without excessive waste.

You may want to consider switching to a Comet server implementation such as Cometd, Lightstreamer, KnowNow, or lighttpd. Comet applications use long-lived HTTP connections between the client and server to allow the server to respond lazily, *pushing* new data to the client as it becomes available. They use one of two strategies: *streaming* and *long polling*.

In an application using streaming Comet, the browser opens a single persistent connection to the server for all Comet events. Each time the server sends a new event, the browser interprets it. Neither side closes the connection. With long polling, the browser makes an Ajax-style request to the server. The server is kept open until it has new data to send to the browser. After sending such an event, the server closes the connection, and the browser immediately opens a new one.

In addition to longer-lived connections, Comet servers are optimized for higher volumes of concurrency, lower latency, and lower server load than typical Web servers.

A closer look at Javacode:Dojotool kit

Before you test Ajax applications for heavy loads, excessive polling, and/or network traffic bottlenecks, take a closer look at your JavaScript code. Moving a line of your code to another part of the code could make a difference in performance. If you're using an Ajax toolkit such as Dojo, you may have configured the Web server to cache JavaScript files aggressively and send status information quickly. You also may have reduced the number of tags on a page by leveraging CSS. For example, instead of doing four lines of tags (see Listing 2):

Listing 2. Four lines of tags

```
<table><tr>
<td>Hello Dojo</td>
</tr>
</table>
```

Just do:

```
<div class="script2">Hello Dojo</div>
```

Be sure that you do not have redundant CSS scripts.

After you have done a few other things within the application, you may have profiling code to supplement Firebug (a Firefox extension). You might discover that the code was making excessive calls to a component because it was inside a loop. Moving this code outside the loop could fix the problem.

Firefox add-ons

Two useful Firefox add-ons are LiveHTTPHeaders (used to view normal HTTP traffic), and Firebug (used to measure load times of resources). Other extensions of interest include RAMBack (used to issue a notification on freeing up memory), Load Time Analyzer (used to measure load times of Web pages), YSlow (used to analyze Web pages), iMacors for Firefox (used to test Web applications), and ColorBlind Ext (used to aid the color blind).

Except for LiveHTTPHeaders, which you can directly download from a Web site, follow these steps to get other add-ons:

1. Open Firefox.
2. Click **Tools** and then **Add-ons**.
3. On the bottom right corner of the dialog box, click **Get Extensions**.
4. If you see a Security Warning, click **OK**. You can choose to select or unselect the box to alert you when you are about to view an encrypted page.
5. Select the add-ons you want to download and install.
6. When you click **Install**, Firefox will close.
7. Reopen Firefox.

LiveHTTPHeaders

LiveHTTPHeaders allows easy viewing of all normal HTTP traffic, XMLHttpRequests, and Comet traffic, including the viewing of full headers in the Firefox browser. You can see what kind of Web server the remote site is using to communicate with your Ajax application and even see the cookies sent by a remote site. Header Monitor and Header Spy extensions obtain headers from LiveHTTPHeaders for display on the statusbar panel. Header Monitor displays any HTTP response header of a top-level document returned by a Web server. Header Spy lets you view both response and request headers on up to five statusbar panels

Load analysis tools

You can use Firebug to view not only script errors and XMLHttpRequest information, but also to measure the load times of various resources such as script and image files. Google's Load Time Analyzer tool lets you measure and graph how long Web

pages take to load in Firefox. YSlow analyzes Web pages and tells you why they're slow, using criteria based on Yahoo's rules for high-performance Web sites. YSlow is licensed under the Mozilla Public License, with portions licensed by third parties under other license terms.

The developerWorks Ajax resource center

Check out the [Ajax resource center](#), your one-stop shop for free tools, code, and information on developing Ajax applications. The [active Ajax community forum](#), hosted by Ajax expert Jack Herrington, will connect you with peers who might just have the answers you're looking for right now.

Response performance tools

iMacros for Firefox records and replays all Web activities and includes support for many Ajax elements. You can use it for functional, performance, and regression testing of Web applications. The built-in STOPWATCH command captures precise Web page response times.

With Fasterfox, you can tweak many network and rendering settings such as simultaneous connections, pipelining, cache, DNS cache, and initial paint delay. You can increase speed dynamically with the unique pre-fetching mechanism, which recycles idle bandwidth by silently loading and caching all of the links on the page you are browsing.

RAMBack causes Firefox to issue an internal notification to free up memory that is otherwise held for performance purposes. It lets you clear internal Firefox caches. Unlike other extensions that work with Firefox versions up to 2.0.11, this extension requires Firefox 3.0a8pre-3.0b2.

User performance tools

ColorBlindExt helps the color blind with Web browsing by processing images and text on the page according to the type of user's color blindness. A color blindness detection test is included for raising awareness of the condition.

Conclusion

Development teams can take advantage of a wide range of performance-enhancing tools available for Ajax applications. To improve the performance of servers, client-side tasks, and networks, you must plan ahead on creating, testing, and deploying any Ajax performance improvement projects. Resolving any performance issues as soon as they are discovered can make a significant impact on the effective development, as well as end use, of Ajax applications.

Resources

Learn

- The [entire series](#) by Judith M. Myerson offers information on how to work with Web services in enterprise-wide SOAs.
- Judith M. Myerson's series, "[Use SLAs in a Web services context](#)" has details on service-level agreements.
- Learn more about [TsunG](#), [Bonnie++](#), [Wireshark](#), and the [Dojo ToolKit](#).
- Find out about Comet server implementation tools: [Cometd](#), [Lightstreamer](#), [KnowNow](#), or [lighttpd](#).
- Get details on [Firefox](#) and [LiveHTTPHeaders](#).
- Want more information on Ajax tools? Read about them in "[Survey of Ajax tools and techniques](#)" (Gal Shachor, Yoav Rubin, Shmulik London, Shmuel Kallner, developerWorks, July 2007).
- Read Judith M. Myerson's [The Complete Book of Middleware](#), which focuses on the essential principles and priorities of system design and emphasizes the new requirements brought forward by the rise of e-commerce and distributed integrated systems.
- Get the business insight and the technical know-how to ensure successful systems integration by reading [Enterprise Systems Integration, Second Edition](#).
- Bring your organization into the future with [RFID in the Supply Chain](#), which explains business processes, operational and implementation problems, risks, vulnerabilities, and security and privacy.
- Go into the nuts and bolts of developing a service-level agreement in this [IBM® Redbook for Domino administrators](#).
- Visit the [technology bookstore](#) for books on these and other technical topics.
- Check out the [Ajax Resource Center](#), your one-stop shop for information on the Ajax programming model, including articles and tutorials, discussion forums, blogs, wikis, events, and news. If it's happening, it's covered here.

Get products and technologies

- [IBM trial products for download](#): Build your next development project with IBM trial software, available for download directly from developerWorks.

Discuss

- [Participate in the discussion forum for this content](#).
- [developerWorks blogs](#): Get involved in the developerWorks community.

About the author

Judith M. Myerson

Judith M. Myerson is a systems architect and engineer. Her areas of interest include open source tools, middleware technologies, enterprise-wide systems, database technologies, application development, network management, security, performance management, RFID technologies, and project management.

Trademarks

UNIX is a registered trademark of The Open Group in the United States and other countries.

Microsoft, Windows, Windows NT, and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both.

Linux is a registered trademark of Linus Torvalds in the United States, other countries, or both.