Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: December 18, 2012

M. McCall
Akamai

June 16, 2012

HTTP Extension to provide timing data for performance measurement
draft-mccall-httpbis-timing-measurements-00

Abstract

This memo presents a method for allowing an HTTP server to gather relevant performace data from compliant user-agents. Though the HTML 5 standard supplies Nav-Timings there is no standard way for that data to be communicated back to an origin. The intention of this document is to describe a method in which that can be done cleanly and scalably.

Status of this Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at http://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on December 18, 2012.

Table of Contents

1. Overview

One of the primary goals of the HTTP 2.0 is to improve performance of web pages. While there have been many performance improvements in the entire web stack since the first iteration of HTTP - from server, to network, to browser - there are now important efforts being made to improve the protocol which delivers much of the world's content every day. Indeed, it's no surprise that having a faster website has shown to improve customer conversion rates and user engagement, and with richer websites becoming more prevalent, improving performance is a natural next step in the evolution of the protocol. While there are many different proposals on how best to speed up HTTP, this document focuses on quantifying the performance of an HTTP session.

In its current state, HTTP has no means to expose performance metrics for an HTTP session. Because of this, today, most web page performance measurement is done one of two ways:

Synthetic transactions

Real-user Measurement

Synthetic transactions tend to rely on a instrumented browser that has the ability measure DNS resolution time, first byte time, object download time, DOM processing time, and other browser events. There are many synthetic transaction options currently available, ranging from 3rd-party solutions provided by companies like Keynote [Keynote] and Gomez [Gomez] , to open-source projects like WebPageTest [WPT] . While these solutions are valuable, they do have limitations. In particular, most these measurements are performed in controlled, "clean room" environment. These "clean rooms" often have better-performing networks, browsers, and computers than a real user might have, and thus, tend to provide a best-case measurement of a site's performance.

Real-user measurement is normally performed by instrumenting the target webpage with JavaScript to capture measurements similar to those available in synthetic transactions. Recently, the W3C has submitted a few specification drafts [Nav-Timing] which provide an interface for browsers to expose performance timing data, and most modern browsers support the Navigation Timing interface. While the measurements available using browser-side RUM are extremely valuable in quantifying an end-user experience, this solution also comes with limitations. In particular, running JavaScript on the client does not come without some performance impact. In addition, instrumenting the webpage with timing measurements can potentially be non-trivial and add to development time for web page designers.

This document outlines an alternative to both synthetic measurements and real-user measurements in their current forms by adding new request and response headers to the HTTP protocol. By coupling client-side metrics into the HTTP session, end-user web page performance can be easily quantified without the need and overhead of JavaScript running in the client's browser, and without having to rely on third parties to provide performance data. Additionally, since the server is privy to the performance of the session, it is possible for it to modify the content delivered accordingly.

At a high level, the HTTP Timing Measures Extension would work in a manner similar to the following:

A new client request header, X-Accept-Measurement, is sent at the initiation of an HTTP request.

A negotiation takes place between the server and the client to determine which measurements will be sent by the client and a timing interval for the measurements. The server requests measurements by sending an X-Send-Measurement response header.

Assuming the negotiation is successful, the client sends measurements after the timing interval has expired to the server in an HTTP POST request with an X-Timing-Measures header.

We believe that adding this functionality to HTTP is important for a variety of reasons. CDNs, in particular, compete through performance attributes. However, there are a number of issues that make showing value through performance data difficult: Web site owners tend to be hesitant to add new JavaScript to their sites out of fear of slowing the site down, and synthetic measurements don't truly show what real users are experiencing. Even if a CDN were to transparently insert measurement code into a web page, there are risks involved. By adding measurements directly into HTTP, the playing field is leveled by having the protocol itself provide the de-facto method of measuring web site performance.

2. Supported Client Measurements

The W3C spec for Navigation Timing provides an excellent interface for collecting base page metrics. However, it is limited to network and browser events, and in those cases only for the base page as it is served. AJAX-initiated events and object timings are not exposed. The draft W3C spec for Resource Timing goes further, but to date has no browser support.

In addition to the base measurements outlined in Navigation Timing, the client should optionally support providing measurements for all resources required to render the page within the timing interval specified by the server. Having this level of measurement granularity would allow one to collect measurements for objects that are delivered outside of the base page measurement, and be useful in illustrating end user experience.

At a minimum, clients should support the measurements outlined in the Navigation Timing spec. Supporting other resource timings should be optional, since it could conceivably generate a large number of performance metrics to be sent back to the server.

3. Delivering Measurement Data

Once the client and server have negotiated the collection of the measurements, data should be sent back to the collecting server via an empty HTTP POST request, and encapsulated in an X-Timing-Measures header.

However, there may be a case in which the amount of data to be sent back to the server is too much to encapsulate in an HTTP header. An HTTP POST also may not be desirable in cases when POSTs interfere with application logic. In this case, an optional HTTP header, X-Timing-Measures-Collector can be specified in the initial response to indicate to the client where to send the data. If the X-Timing-Measures-Collector header exists, data will be sent in the body of the POST request, in the format previously described.

In some cases, it may be necessary to specify a Timing Measures Collector on a per-page basis, in which case an optional meta element can be added to an HTML page. Such a meta element might look like:

```
<meta http-equiv="Timing-Measures-Collector"
content="http://www.example.com/beacon.htm">
```

By default, the X-Timing-Measures-Collector attribute is null, and data is POSTed back to the page as described earlier.

4. Illustrative Use Case

This use case demonstrates a successful performance measurement negotiation and data posting. In it, we show how a user agent and site (www.example.com) would negotiate and transfer measurement data.

```
Request1:
UA --> Server (www.example.com)

GET /index.html HTTP/2.0
Host: www.example.com
X-Accept-Measurement: allowed
Response1:
Server (www.example.com) --> UA

HTTP/2.0 200 OK
X-Send-Measurement:
    dns;firstByte;domReady;domLoaded;onLoad;resources:TI:300000
```

We see that the first UA request has indicated that it is willing to provide HTTP performance measurements (X-Accept-Measurement: allowed), and that the server has replied with the measurements it is requesting (X-Send-Measurement: dns;firstByte;domReady;domLoaded;onLoad;resources:TI:300000). The X-Send-Measurement response header sent by the server is followed with a semicolon-delimited list of desired measurements, followed by 'TI', which is the timing interval of the desired measurement data expressed as milliseconds.

```
Request2:
UA --> Server (www.example.com)

POST /index.html HTTP/2.0
Host: www.example.com
X-Timing-Measures:
    dns:23;firstByte:120;domReady:1200;
    domLoaded:1300;onLoad:1331;resources:1400

Response2:
Server (www.example.com) --> UA

HTTP/2.0 200 OK
```

We now see the UA sending back data to the initial server via an empty HTTP POST request, including the X-Timing-Measures header containing measurements that happened in the timing interval requested by the server in the initial handshake. The header contains key-value pairs of the requested measurements, where the

values are represented as milliseconds of the measurement's duration. Depending on which measurements are supported by the browser, all or some of the measurements requested by the server are returned.

5. Proposed HTTP protocol extension

To implement Timing Measures, additions of new HTTP headers are required. The details of the request and response headers are included as an initial starting point for discussions, rather than as a final proposal.

Pseudo-BNF grammar for these headers might look like:

```
X-Accept-Measurement        = "Accept-Measurement:" measure-resp
measure-resp                = "accept|not-accepted"

X-Send-Measurement          = "Send-Measurement:" measure-type:TI
measure-type                = measurement
TI                          = timing-interval

X-Timing-Measures           = "Timing-Measures:" measure-type:value
measure-type                = measurement
value                       = value

X-Timing-Measures-Collector = "Timing-Measures-Collector:" collector-url
collector-url               = collector
```

The Accept-Measurement, Send-Measurement headers are optional. If a negotiation is successful, the Timing-Measures header is required. The Timing-Measures-Collector header is optional. All measurement values, as well as TI, are 32-bit integers expressed as milliseconds. The collector-url is a string representing the URI data should be sent back to.

Measurements are submitted via an HTTP POST request and must contain the Timing-Measures header, and should not contain any data in the body of the request. Measurements are sent as key-value pairs in the form of measurement:value delimited by semicolons. If the Timing-Measures-Collector header is sent or the relevant HTML meta tag exists, the client must send its data to the specified collector-url in the body of a POST request, using the data format specified above.

The user agent must discard the measurements after the Timing Interval for the requested metrics expires.

6. Security Considerations

User-agent MUST follow 'same origin' considerations when sending timing data

7. IANA Considerations

The new header fields

Accept-Measurement

Send-Measurement

Timing-Measures

Timing-Measures-Collector

should require IANA assignment

8. Closing Thoughts

Outside of adoption, the goal of this document is to get the community thinking about measuring HTTP performance. While many of the current HTTP 2.0 proposals have been designed to improve performance, to date, none discuss quantifying the intended improvements.

It is well known by the author that there are shortcomings and unanswered questions in this document. In particular, there is a lingering question about whether or not the timing measurements should exist within the context of a server connection, or whether they are best viewed within the scope of a user session. Another question arises as to what should happen if the user agent closes the connection before the timing interval has expired. With the optional X-Timing-Measures-Collector header that can be used to send the measurement results to another host, it is the belief of the author that measurements should exist as part of a user session, and not necessarily tied to a particular connection. Community consensus on this matter is encouraged.

Another concern is that some of the measurements outlined in the "Supported Client Measurements" section are more browser-specific than some may be comfortable with. However, while network timing is most obviously relevant for HTTP, it is also possible to infer browser and hardware performance by looking at the time elapsed processing and rendering the document. If a server had a full performance profile of the user agent it was communicating with, it could potentially serve different content or make other decisions based on the UA's performance characteristics.

One glaring omission is the security of the proposed extension. While performance data would seem to be of low value to malicious users, at the very least the security of the extension should be considered.

Finally, the grammar for the new headers needs to be properly defined. The grammar outlined in the "Proposed HTTP Protocol Extension" section is woefully incomplete and syntactically incorrect, and is included only for illustrative purposes.

In closing, the intention is that these, as well as other shortcomings in this document, are fleshed out by the community. The more people thinking about these issues, the faster and better the web will be.

9. References

9.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

9.2. Informative References

[Keynote] "Keynote Systems", <http://www.keynote.com>.

[Gomez] "Gomez Networks", <http://www.gomeznetworks.com>.

[WPT] "WebPageTest", <http://www.webpagetest.org>.

[Nav-Timing] "Nav-Timing Spec", <http://www.w3.org/TR/navigation-timing/>.

Author's Address

Mike McCall
Akamai Technologies