Mobile Access to Web Sites

Users are increasingly using a variety of mobile devices to access web sites including mobile phones and tablets. These devices and the networks they use to connect to the Internet have many different properties, all of which can impact how the web pages load.

These features include screen dimensions, connection handling, browser identification (user agent) as well as network speeds and, more importantly, latency. These properties can have a large impact on how a web page is loaded on a mobile device, examples of which are discussed further in this document.

Our agent technology can simulate these conditions, causing the web site to respond according to the type of device and connection being emulated (for example, the request can be configured to appear to be made from an iPhone on a 3G connection).

As site owners continue to update their existing web sites, as well as offering dedicated mobile web sites , it becomes increasingly important to test such sites using monitors configured as mobile devices, as site changes can have a significant impact to performance for mobile customers, which they may not be aware of.

This is even more important where a mobile site is delivered from a different platform than the traditional site. **It is not a case of monitoring either your desktop or mobile site, but monitoring BOTH.**

Likewise, load testing your mobile site using mobile connection characteristics and download speeds, is important to understand the performance of your mobile site and how variations in performance under load affect the experience of end users on mobile devices and connections.

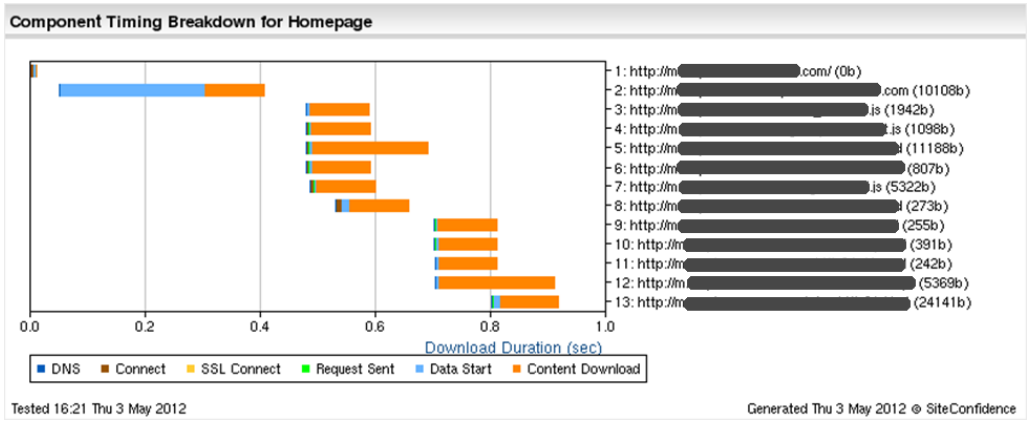
Impact of Latency

All networks exhibit latency, which is the amount of time it takes a packet in a network to complete a 'round trip' between the requesting device and the destination server. Mobile networks such as 3G have higher latency than traditional networks and, whilst a 200ms round trip for an example 3G connection may not sound like much, it can have a large impact.

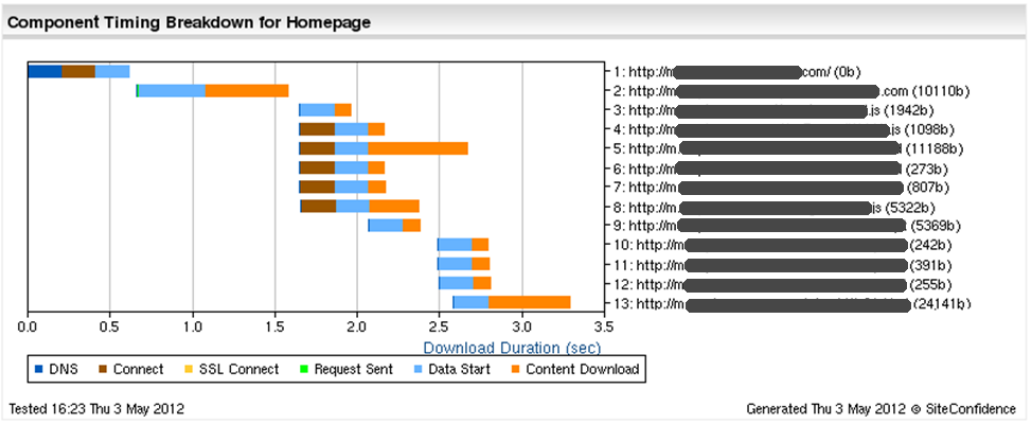
For instance, looking at an example 8 step transaction through a well-known eCommerce web site where we browse for a product and then make a purchase, adding just 200ms of additional latency (without any other differences) increases the overall transaction from ~28 seconds to ~73 seconds:

Time / Date	Test Type	Test Server	Speed (sec)	Status	Result Code	Step (if failed)	
11:49:11 Wed 9 May 2012	Manual	10003	28.64	✓	1: Site OK	All OK	zero latency
11:54:10 Wed 9 May 2012	Manual	10003	27.66	✓	1: Site OK	All OK	
12:03:35 Wed 9 May 2012	Manual	10003	72.50	✓	1: Site OK	All OK	200ms latency
12:08:08 Wed 9 May 2012	Manual	10003	73.28	✓	1: Site OK	All OK	

So, how does just 200ms of additional latency make such a difference? Let's take a look at the home page of their mobile site:



Zero additional latency



200ms additional latency

The home page now takes ~3.3 seconds to download with 200ms additional latency as opposed to ~0.9 seconds. As expected, the page make up doesn't really change, but the timings do. This is particularly noticeable when you look at the scale of the x-axis.

The first noticeable impact of the additional latency is on the initial request (object 1). This is not even a page, but a 302 redirect to the 'real' page. This additional 200ms of latency has meant the redirect has not been delivered for ~610ms, compared with ~10ms. This is because of the additional time taken to conduct the DNS lookup (1 round trip), the connection to the host (1 round trip) as well as receiving the 302 response header (1 round trip).

The main page (object 2) is then requested. As this is on the same host, there is no additional latency cost for a DNS lookup or connection, but there is an additional 1 round trip cost seen in the data start (i.e. for the response to start being received by the client agent).

This main page is compressed, so only 10KB of data needs to be transferred. However, because of TCP slow start this is still impacted by the additional latency so now takes ~0.5 seconds to receive as opposed to ~0.1 seconds.

At this point the main page has now been downloaded and can be parsed to allow the rest of the page to be requested. With no additional latency, the page is ready to be parsed at ~0.4 seconds. With 200ms additional latency, this increases to just over 1.5 seconds.

All of the resources for this mobile site come from a single `m.[domain].com` host. The iPhone 4 supports 6 concurrent connections per host, so objects 3 to 8 begin to download, with objects 4 to 8 also incurring the additional cost of the latency for setting up extra connections (object 3 downloads on the connection that is now available from object 2). The data start is also impacted as the responses for those requests are delayed by the increased latency.

We can see that the request for object 9 does not need to set up a new connection (i.e. no brown bar on the waterfall) because object 3 has finished downloading so there is now a connection available.

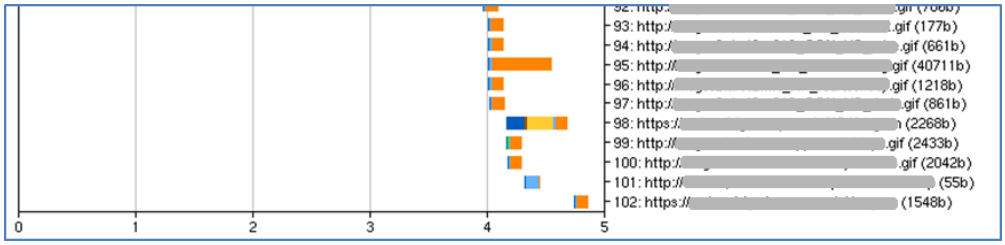
The content has also taken longer to download. For example, the `logo.png` file (object 5) has taken ~0.6 seconds just for the content to be downloaded compared with ~0.2 seconds with no additional latency. This is largely because the connection is still subject to TCP slow start as the earlier requests on these connections only downloaded small amounts of data.

As we can see, even with a small (61KB compressed) home page, the additional latency seen with mobile networks can have a significant impact on the overall time taken to download a page.

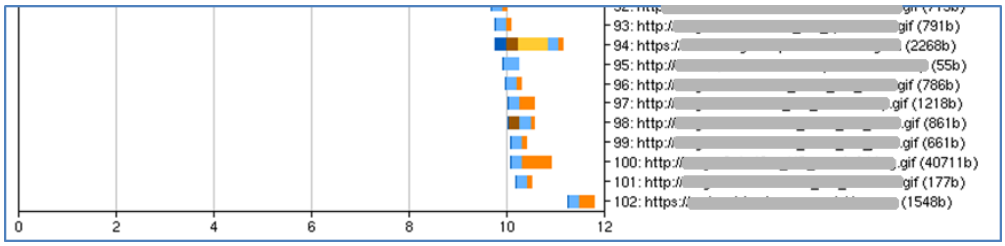
Non Mobile Site?

Although the increase seen above is significant, at least this is a site specifically tailored for mobile devices, so it is smaller than the traditional web site.

For those serving the standard web site to mobile users, things can be far worse. Here's another high street brand whose web site does not have a mobile version. Again, just comparing the difference with an additional 200ms of latency, the time taken to download their main product landing page increases from ~4.8 to ~11.7 seconds:



Zero additional latency

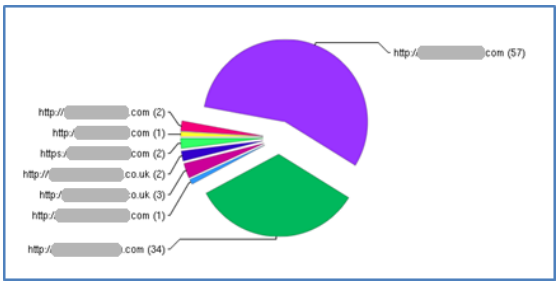


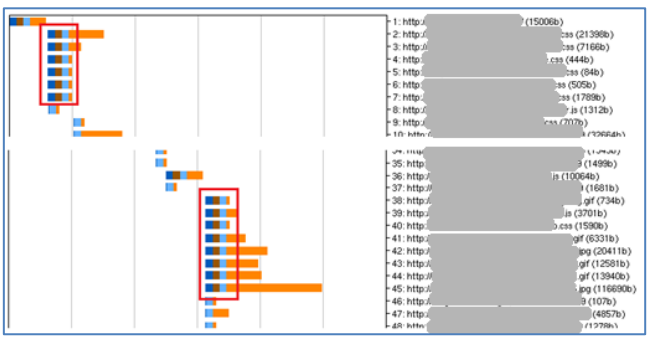
200ms additional latency

The biggest cause of the increase on the whole page however is the additional data start time across all of the objects on the page. Even though the iPhone 4 supports up to 6 concurrent connections per host, this particular page still needs to make 102 HTTP requests.

As each of these are based on a request => response, the additional latency causes each to have a longer data start time (the delay between the request being sent, and the response coming back to the client).

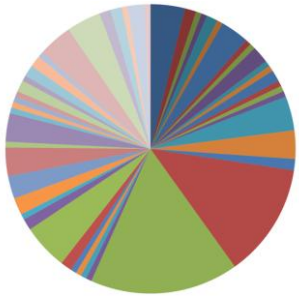
Although a number of these requests occur in parallel, there are numerous connections required to download all of the objects from the 8 different hosts that make up the page. Each of these connections has been impacted by the additional latency on the connect times.





These have been highlighted here in red.

Things are even worse for the product page, which has over 150 requests being made across 50 different hosts (each host shown in the pie chart opposite).



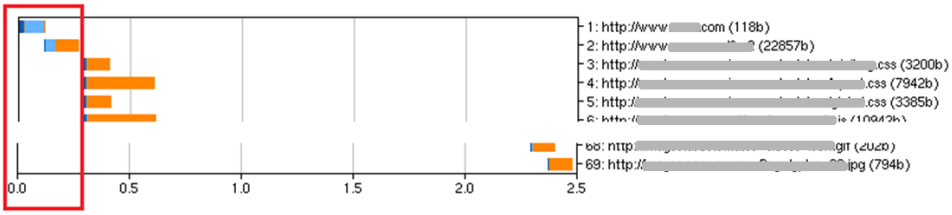
As such, almost all of the requests require a new connection to be established, many of which also need a DNS lookup, each requiring 1 round trip, effectively adding latency for every request.

This has the effect of increasing the page load from 5.9 seconds to 18.1 seconds, just because of the additional 200ms of latency on each round trip.

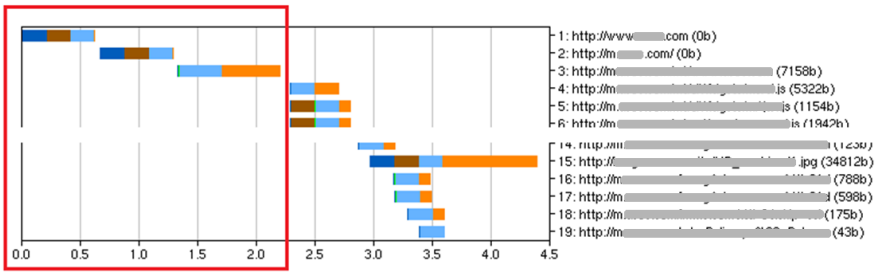
Redirects to Mobile Site

One of the most common features implemented on web sites which have a mobile specific site is to redirect requests for the traditional web site to the mobile version, when they come from a mobile device.

For instance, if we look at the home page of a well-known online clothing retailer, we can see that even the traditional home page request gets a 302 redirect to the 'proper' page. This results in the main page not being loaded for ~0.25 seconds.



The benefit of presenting the request as coming from a mobile device (for example an iPhone 4) is that you can see how the web site loads for users of these devices. For example, hitting the same home page, we now get 2 redirects before we can request the 'proper' page:



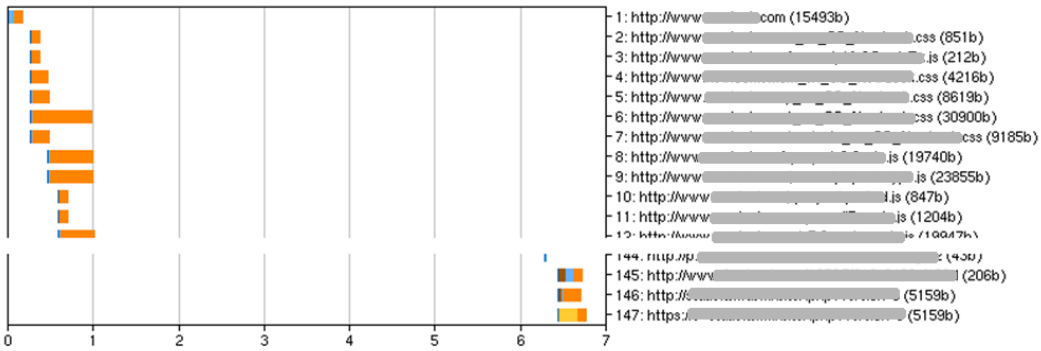
This immediately shows that customers accessing the retailer's home page on an iPhone 4 over 3G will have to wait ~2¼ seconds before the main page has even loaded. As this goes across two different hosts (`www.` and `m.`), there is the expense of an additional DNS lookup and connection delay.

As well as the main page, the rest of the objects (images, style sheets etc.) still have to load. Although we can see from the two waterfall charts above, there are far less objects for the mobile site (19 requests in total compared with 69 on the traditional site) – the mobile site still loads in ~4.3 seconds compared to ~2.5 seconds for the traditional site.

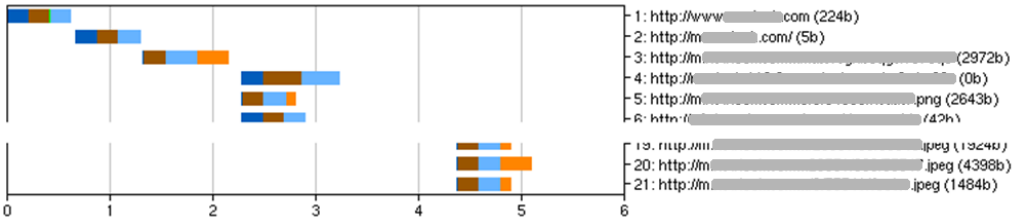
A better solution would be for the mobile site to be delivered directly from the initial request. The site must be using server side browser detection anyway (to deliver the initial redirect to the `m.[domain].com` host) and this could mean the main page object being delivered to the mobile device in just over a second instead of 2¼ seconds.

Closing Connections?

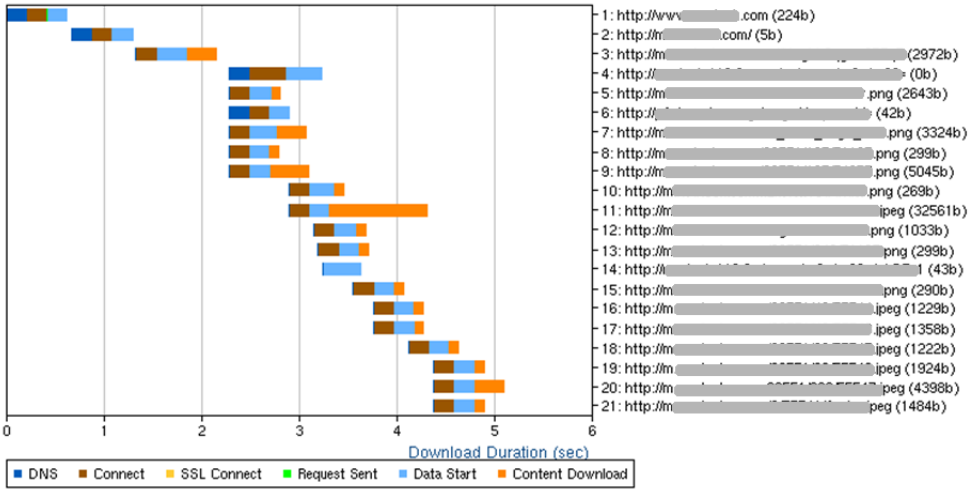
When comparing another high street retailer's home page, as delivered to a traditional desktop browser and a mobile device, at a first glance we can see the site is optimised for mobile delivery. For instance, there are 147 requests for the main site:



This compares with 21 requests for a monitor configured as a Samsung Galaxy S2 on 3G:



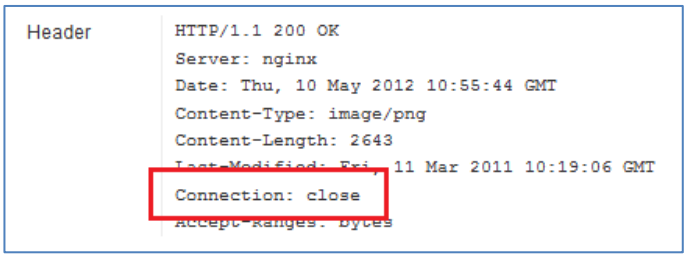
These are delivered in ~6.8 and ~5.0 seconds respectively, so not a big difference. However, focusing on the mobile delivered site, there is still room for improvement. If we look at the waterfall chart for the mobile site, we see a lot of 'brown bars':



This 'brown bar' timing relates to the connect time. We know this is impacted by the additional latency seen on typical mobile networks; however it seems odd in this example because the majority of these additional connect times are to the same `m.[domain].com` host.

One of the benefits of using the same hosts for delivering content (as opposed to domain sharding) is that you will typically benefit from saving on a DNS lookup (1 RTT) and a connection request (1 RTT). However, this site does not appear to benefit from the latter here.

If we look closer at the HTTP response headers, we can see why this is happening:



When loading this high street retailer's web site as a mobile device type (including the iPhone 3, iPhone 4, Samsung Galaxy S2 and Blackberry Bold 9900), the `m.[domain].com` host is returning the directive to close the connection.

This is why additional connections need to be set up, introducing additional delay. Although some of these occur in parallel, there is an aggregated 2.2 seconds of 'wasted' time on these additional connects. This delays the page by ~1 second of elapsed time.

This however does not happen on their traditional site, so the recommendation would to look at the infrastructure serving the mobile content to understand why this is happening, with a view to removing it so connections can be reused (which is the typical behaviour with HTTP 1.1).

Conclusion

The most significant improvement recommended for mobile sites is to reduce the number of requests as these are so badly impacted by latency. For instance, the upfront redirect on the home page means that at least one extra round trip delay is incurred.

Combining JavaScript files and also using CSS sprites or data URIs for the images would also reduce the number of requests being made, all helping limit the impact of the additional latency seen on mobile networks.

Setting up monitoring configured as a mobile device is as important as traditional monitoring, regardless of whether you use the same infrastructure to deliver the content. As we have seen, sites can load significantly differently for mobile users.