Sizing Up PQ-Signatures, Summary of Results

Full blog post (to appear) at blog.cloudflare.com/sizing-up-post-quantum-signatures

The goal is to measure the impact of larger TLS signatures for the Web: how fast they are and whether they work at all.

Setup

To emulate the impact of larger signatures in TLS, we add extra dummy certificates to the certificate chain. These are 1kB self-signed invalid certificates. As adding superfluous irrelevant certificates is a common misconfiguration, most clients have learned to ignore them. TLS 1.3 actually stipulates these should be ignored. We tested hundreds of browsers and found that none of them rejected a TLS handshake with such an additional dummy certificate. However, in preliminary testing we found that a small, but significant amount of clients had issues with them. We don't want to ruin anyone's connection, so we decided to use new background connections for this purpose instead.

Word-wide on a small percentage of our challenge pages (those with the CAPTCHA), we pick a number 1≤*n*≤59, a random key and send this key in two separate requests to:

- `0.tls-size-experiment-c.cloudflareresearch.com`
- *[n]*`.tls-size-experiment-1.cloudflareresearch.com`

The first, **the control**, is a normal webpage that stores the TLS handshake time under the key that's been sent. The real action happens at the second, **the live**, which adds the *n* dummy certificates to its chain. The live also stores handshake time under the given key. We could call it "experimental" instead of "live", but the benign control connection is also an important part of the experiment. Indeed, it allows us to see if live connections are missing.

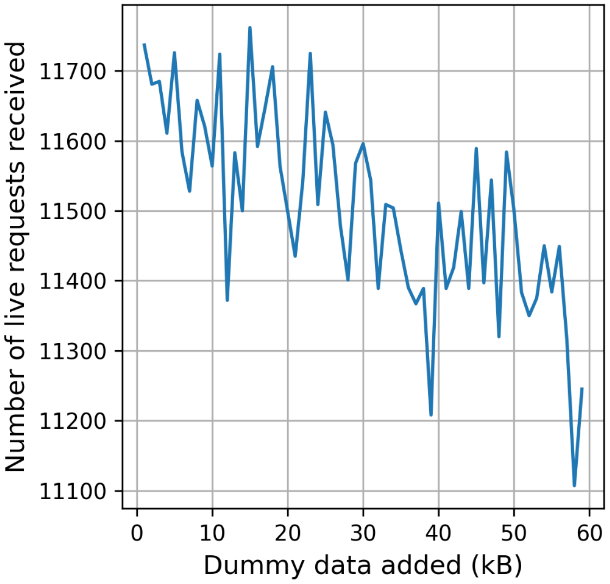
Results

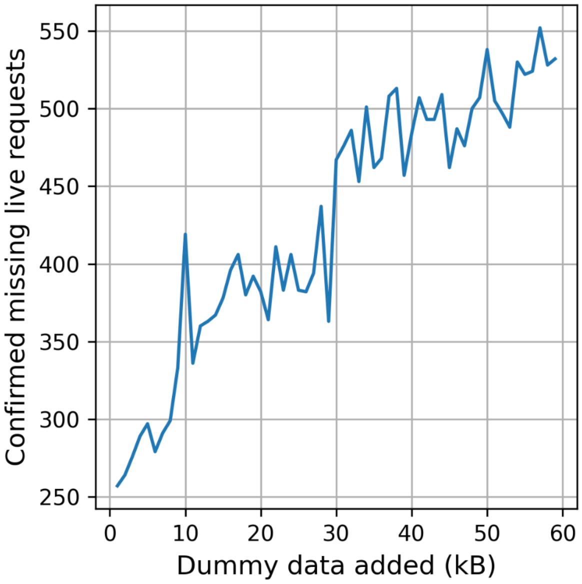
Over 16 days we've received 653,491 live connections from 324,789 different truncated IPs (to 24 bits, "/24", for IPv4 and 48 bits for IPv6) and 9,732 different ASNs.

Can clients handle the larger handshakes?

The control connection was missing for 2.6% of the live connections. This is not alarming: we expect some connections to be missing for harmless reasons, such as the user browsing away from the challenge page. There are, however, significantly more live connections without control connection at 3.6%.

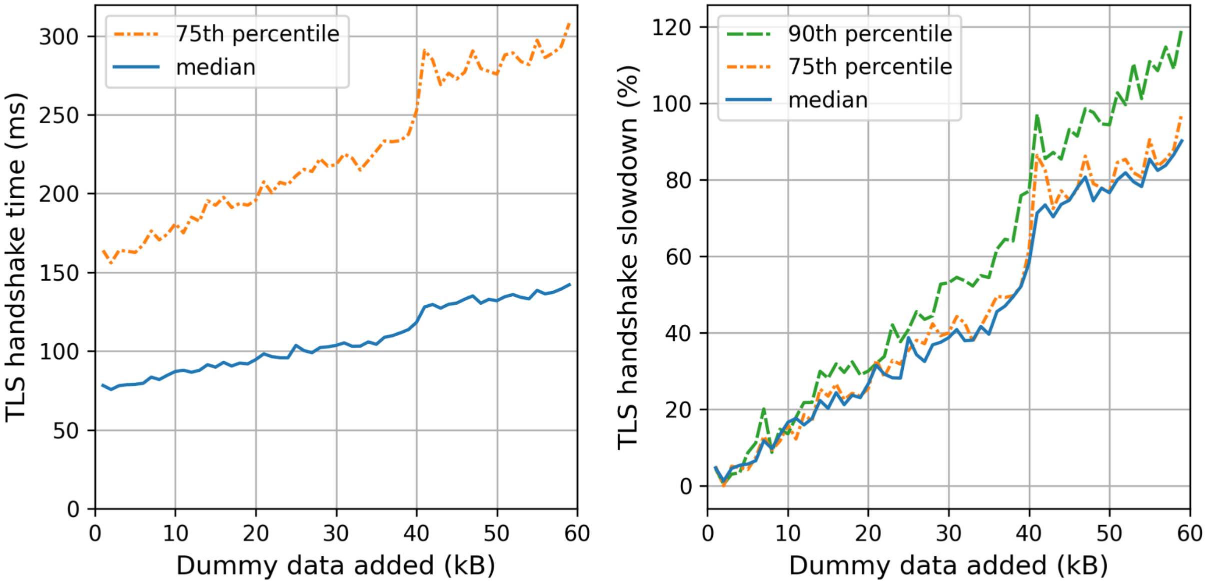
In the graph below on the left we break the number of received live connections down by the number of dummy certificates added. Because we pick the number of certificates randomly, the graph is noisy. To get a clearer picture, we started storing the number of certificates added in the corresponding control request, which gives us the graph on the right. The bumps at 10kB and 30kB suggest that there are clients or middleboxes that cannot handle these handshake sizes.





Handshake times with larger signatures

What is the effect on the handshake time? The graph on the left shows the weighted median and 75th percentile TLS handshake times for different amounts of dummy data added. We use the weight so that every truncated IP contributes equally. On the right we show the slowdowns for each size, relative to the handshake time of the control connection.



Looking at the figures, there is a slope until 40kB, where a wall appears. The marked increase corresponds to the initial congestion window of 30 packets. (Note that the default initial congestion window is 10.)

Adding 35kB fits within *our* initial congestion window. Nonetheless, the median handshake with 35kB extra is 40% slower. The slowest 10% are even worse off, taking 60% as much time. Thus even though we stay within the congestion window, the added data is not for free at all.

For Dilithium2 as a drop-in replacement we need around 17kB extra. That also fits within *our* initial congestion window with a median slowdown of 20%, which gets worse for the tail-end of users. For the normal initial congestion window of ten, we expect the slowdown to be much worse — around 60–80%.

There are several caveats to point out:

- These experiments used an initial congestion window of 30 packets instead of ten. With a smaller initial congestion window of ten, which is the default for most systems, we would expect the wall to move from 40kB to around 10kB.
- Because of our presence all across the world, our RTTs are fairly low. Thus the *congestion window wall* is smaller for us.
- Challenge pages are served, by design, to those clients that we expect to be bots. This adds a significant bias because bots are generally hosted at well-connected providers, and so are closer than users.
- HTTP/3 was not supported by the server we used for the endpoint. Support for IPv6 was only added ten days into the experiment and accounts for 5.1% of the measurements.
- Actual TLS handshakes differ in size much more than tested in this setup due to differences in certificate sizes and extensions and other factors.