# Proposed Necko Web Transport Priorities

Mar 06 2013 - :mcmanus

**Problem Statement**

The top level necko priorities are performance and support of Firefox OS. This document was undertaken to focus attention on the important but tractable problems related to those goals within web transport. It isn't meant to be a comprehensive survey but hopefully it is a useful short summary of our current status. It is acknowledged that there are projects not considered here outside the scope of web transport that also need to be considered as part of necko performance and Firefox OS work.

Traditional web transport mechanisms are slower and less secure than they could be. HTTP succeeds at providing ubiquitous low overhead interconnection but underperforms on networks with high latency or variable rates of contention. Furthermore, in practice the decision of whether to utilize SSL and protect a user's privacy from being invaded through simple passive observation is generally based on the financial liability of the service operator rather than the interests of the user which results predictably in perceived low rates of SSL deployment.

**Big Picture Focus Areas**

- Identify work areas that will drive the adoption of secure transports (presumably SSL/TLS/DTLS) through performance.
- Reduce latency and improve bandwidth utilization with a focus on non-traditional network profiles with higher latencies, variable rates of contention, and more mobility.

**Known Barriers**

- SSL Cost. Certificates cost money, CDNs charge higher per-byte rates for SSL, and operational costs of SSL associated with obtaining certificates, certificate expiration, and accidental key disclosures are additional costs.
- SSL performance is slower than plaintext. On average, It takes firefox an additional 325ms to make a SSL connection ready to use vs its plaintext equivalent. The overhead for chrome is around 175ms.
- Mixed content rules make SSL deployment require inter-organization coordination that goes against the distributed web service model of the Internet.
- Intermediaries defeat (often unwittingly) many traditional enhancements to HTTP/1. Anti-Virus software creates a ubiquitous, but often overlooked, intermediary. SSL provides a path to successful deployment of some of these ideas, but embedded intermediaries can have problems with advances in SSL as well. (e.g. tls intolerance)
- Dependence on legacy operating system TCP stacks prevent deploying fixes to the most critical problems of non-SSL specific performance: congestion control implementations, connection management, head of line blocking, etc..

**General Focus Areas**

- SPDY and HTTP2 development, standardization, and evangelization. These are low risk steps that complement our big picture focus areas. These protocols improve performance over high latency networks, safely enable greater parallelism for a variety of levels of contention, and encourage improved security. However, they reach their incremental benefits fairly quickly. We should actively seek to use the evolving standard as a lever to increase SSL utilization rates on the web.
- SSL performance improvements related to the latency of SSL transactions. Reducing the overhead of SSL removes a barrier to its use and is necessary to make us competitive with Chrome.
- Experiments in TWOC (Transports Without OS Constraints). Build experimental UDP based transports that deploy congestion control, retransmission policies, connection management, and security better suited for the web of tomorrow. This is a high risk item but has the potential to make a much larger step forward than other directions. Google's effort in this space is called QUIC.

**Specific Work Items**

All of these items are meant to support the broader focus areas. The list is not exhaustive and can be expected to evolve more rapidly than the more general lists. There are of course other work items compatible with our focus areas that are best pursued by a non platform development team - they are not listed here and of course to the extent that another development team (e.g. security) could also prioritize them that work should be gleefully shared. The items listed are not yet in any order.

- SPDY server push
- SPDY/4 updated compression
- TLS False Start (bugs 658222 and 713933)
- Integrate OCSP caching with HTTP disk cache
- Support OCSP stapling
    - integerate cache anticipation with stapling to minimize the server-hello overflow problem
- Track SSL utilization rates (bug 848360),
- HTTP2 draft implementations to track standardization
- Track TLS server hello size overflow problem (bug 848139)
    - If data indicates it and when standard has progressed integrate TLS-Cached-Info extension to help. Potentially contribute to mod_ssl on server side as well.
- Brian's proposal to make EV revocation checking of intermediate certificates match DV
- Work with bsmith to deploy largely ocsp-less runtimes via a combination of stapling, policy, and crlsets.
- Experiment with different pconn expiration policies and their tradeoffs with getting extra transactions per connection vs the error rate.
- TWOC
    - Define performance success criteria
    - Evaluate QUIC wrt ideal TWOC. Socialize to privacy teams, security teams, etc..
    - Prototype or do data gathering on any QUIC/TWOC gap.
    - Make decision on our own initiative vs trying to partner-influence quic vs standing back and waiting..