Speeding up Tor with SPDY

Master's Thesis

Andrey Uzunov

Supervisor: Christian Grothoff
Chair for Network Architectures and Services
Department of Informatics
Technische Universität München

July 25, 2013

Agenda

- SPDY
- SPDY and Tor
- HTTP over Tor vs SPDY over Tor

SPDY

What is SPDY?

- ”An experimental protocol for a faster web” – alternative to HTTP
  - uses one persistent TCP connection per host: only one TCP handshake and slow start
  - traffic reduction by compressing headers
  - pushes data to the client even before being asked for it
  - requests have priorities, so they may be answered in different order

SPDY Support

- Client support:
  - Firefox
  - Chrome
  - Opera
  - Internet Explorer 11 (currently as preview release)
  - Multiple mobile browsers
- Server support:
  - Google services
  - Facebook
  - Twitter
  - Wordpress
  - Apache
  - nginx

SPDY Versions

- Versions
  - SPDY/2 – widely supported
  - SPDY/3 – widely supported
  - SPDY/3.1
  - SPDY/4 alpha versions – Google continues working on it
  - SPDY/3 is the base for HTTP/2.0 – IETF works on it
- SPDY version is negotiated in TLS handshake via extension: Next Protocol Negotiation
- HTTP/2.0 will be used with and without TLS

C Libraries

- *spdylay*
  - http://spdylay.sourceforge.net/
  - supports most(all?) of the features of SPDY/2 and SPDY/3
  - low level API, e.g. user calls all TLS connect/read/write functions
  - *shrpx* SPDY-to-HTTP and HTTP-to-SPDY proxy
- *libmicrospdy*
  - together with *libmicrohttpd* http://www.gnu.org/software/libmicrohttpd/
  - supports SPDY/3, only GET so far
  - similar API to *libmicrohttpd*
  - *microspdy2http* proxy

libmicrospdy vs Apache(HTTP and HTTPS)

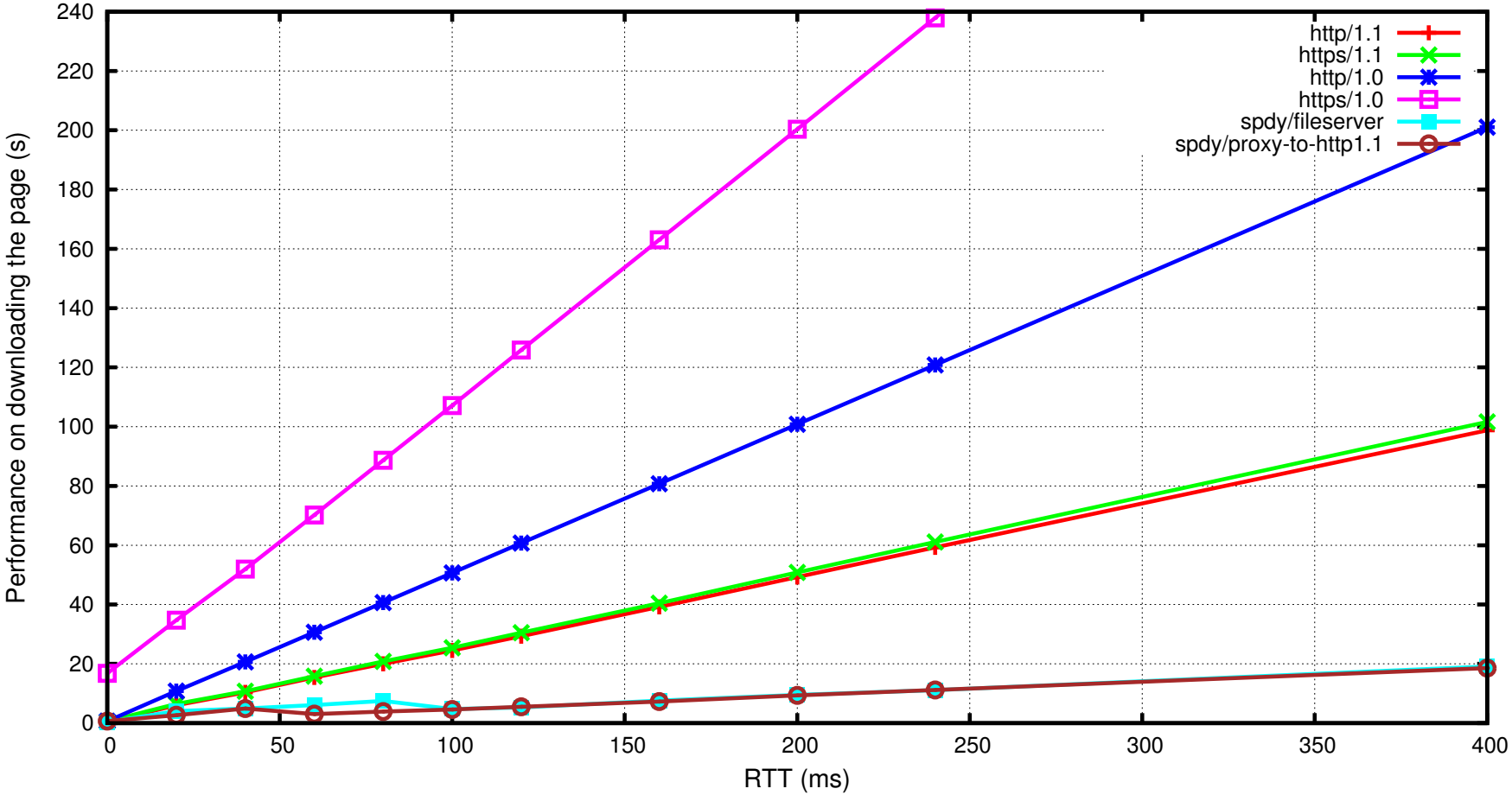

Figure: Download time for the entire webpage (1,315 resources; 3,560,921 B) in relation to network latency, allowing 6 parallel TCP connections for all HTTP setups.

Gaining more from SPDY 1/2

SPDY PUSH:

- sending multiple responses to just a single request
- not specified when and what should be pushed

Idea:

- learn the order in which the client requests resources
- predict resource(s) to push with certain probability
- learn from own mistakes

Gaining more from SPDY 2/2

Problems to address:

- pushed resource may be cached by client
- different clients – different request sequences
- mistakes will increase the traffic

SPDY PUSH for proxies:

- before pushing the proxy has to request the resource from the target server

SPDY and Tor

Speeding up Tor with SPDY – Idea

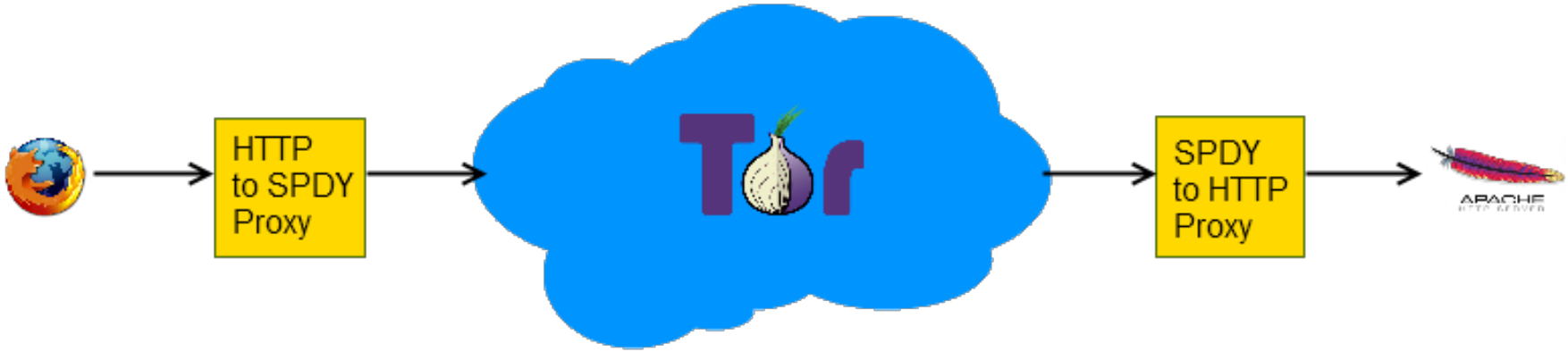

Figure: Using SPDY over Tor.

Goals:

- faster loading times at the client
- less traffic within Tor network

Implementation

- HTTP-to-SPDY proxy on *localhost*
  - *libmicrohttpd* and *spdylay*
  - receives only HTTP requests from Tor Browser (HTTPS goes directly to the SOCKS proxy)
  - uses *torsocks* to communicate with the SOCKS proxy
- SPDY/3 without TLS is used
- SPDY-to-HTTP proxy at exit nodes
  - *libmicrospdy* and *curl*
  - listens on 192.0.2.80:9980
- Open issue: how to find such an exit node?
  - Idea: Tor client chooses exit node with explicitly mentioned ”192.0.2.80:9980” in the exit policy

HTTP over Tor
vs
SPDY over Tor

Test Setup

- Load page in frame and use onload event to measure time
- Using always the same circuit:
  - entry: Unnamed (89.16.176.158) / United Kingdom
  - middle: Ramsgate (38.229.70.61) / USA
  - exit with proxy: spdytor1 (131.159.15.90) / Garching, Germany
- HTTPS Everywhere is disabled
- Browser cache is disabled
- HTTP pipelining is disabled when using the HTTP-to-SPDY proxy
- SPDY headers compression in proxies is enabled in both requests and responses just for the tests
- Traffic between the middle node and the exit node is measured

Test Single Large Page

- Large page on the same physical machine as the exit node:
  - 1315 resources
  - 3,560,921 B total
- Time per page load:

Setup	Load Time (ms)	Relative Time
HTTP	267,827	100.0 %
HTTPS	117,558	43.9 %
Firefox SPDY	64,047	23.9 %
Proxy SPDY	49,165	18.4 %

Test Single Large Page – Traffic per Page Load

Setup	Requests Traffic (B)	Rel	Resp. Traffic (B)	Rel	Total Traffic (B)	Rel
HTTP	1,122,415	100.0%	4,785,214	100.0%	5,907,629	100.0%
HTTPS	1,028,972	91.7%	4,818,302	100.7%	5,847,274	99.0%
Firefox SPDY	975,959	87.0%	4,305,313	90.0%	5,281,272	89.4%
Proxy SPDY	646,982	57.6%	4,141,389	86.5%	4,788,371	81.1%

Test Alexa Top 100 Pages

- Alexa Top 100 as of 6th of July, 2013
- 57 sites are frame-friendly and have HTTP version
- 60 seconds timeout per page load
- Numerous timeouts, especially on AJAX requests
- First set of tests
  - standard settings
  - big variations for same pages on different page loads
  - 5 sites were removed – not enough data for them
- Second set of tests
  - JavaScript is disabled
  - TLS requests are disabled
  - pages are much faster

Test Alexa Top 100 Pages – Page Load Time

	With JS	No JS
Average Page Load Time – HTTP (ms)	18,708	9,536
Average Page Load Time – Proxy SPDY (ms)	17,278	7,673
Difference (ms)	1,430	1,863
SPDY to HTTP Relation	92.4 %	80.5 %

Test Alexa Top 100 Pages – Traffic

	With JS	No JS
Requests Traffic – HTTP (B)	156,221	60,128
Requests Traffic – Proxy SPDY (B)	126,804	46,638
Difference (B)	29,416	13,489
SPDY to HTTP Relation	81.2 %	77.6 %
Responses Traffic – HTTP (B)	1,101,606	538,842
Responses Traffic – Proxy SPDY (B)	1,389,776	516,072
Difference (B)	-279,170	22,770
SPDY to HTTP Relation	125.1 %	95.8 %
Total Traffic – HTTP (B)	1,266,827	598,969
Total Traffic – Proxy SPDY (B)	1,516,581	562,710
Difference (B)	-249,754	36,259
SPDY to HTTP Relation	119.7 %	93.9 %

Issues

- Privacy issues
  - Cross-origin identifier unlinkability
    - Original SPDY: a connection per host
    - Best approach: a connection per tab
  - Shorter session timeout should be used – less statistical data for SPDY PUSH
  - SPDY PUSH: client should not cancel pushed resources – mistakes will be costly

Check http://dev.online6.eu/spdytor/ for information how to run the proxies

Thank you