SMig: A Stream Migration Extension For HTTP/2

Xianghang Mi    Feng Qian    Xiaofeng Wang

Department of Computer Science
Indiana University Bloomington

ACM CoNEXT 2016
HTTP is Evolving

2015
HTTP/2.0

1999
HTTP/1.1

1996
HTTP/1.0

1991
HTTP/0.9
HTTP/2 Features

- Stream Multiplexing
- Server Push
- Header Compression
- Binary Format
- Stream Prioritization
- Flow Control
HTTP/2 Features

- Stream Multiplexing
- Server Push
- Header Compression
- Binary Format
- Stream Prioritization
- Flow Control
HTTP/2 Features

Server Push

Header Compression

Stream Multiplexing

Binary Format

HTTP/2 Features

Stream Prioritization

Flow Control
HTTP/2 Features

- Server Push
- Stream Multiplexing
- Header Compression
- Binary Format
- Stream Prioritization
- Flow Control
According to W3Techs, by Dec 2016, 10.8% of the top 10 million websites are using HTTP/2.
Motivations of SMig: Sender-side HoLB Problem

Request For A Small File
Request For A Large File

Client

HTTP/2 Conn

Server

TCP Send Buff
4MB

4MB
Motivations of SMig: Sender-side HoLB Problem

- Request For A Small File
- Request For A Large File

Client

HTTP/2 Conn

Server

TCP Send Buff

4MB
Motivations of SMig: Sender-side HoLB Problem

Client

Request For A Small File
Request For A Large File

HTTP/2 Conn

TCP Send Buff
4MB

Server
Motivations of SMig: Sender-side HoLB Problem

- Request For A Small File
- Request For A Large File

Client

HTTP/2 Conn

Server

TCP Send Buff

4MB
Motivations of SMig: Sender-side HoLB Problem

Client

Request For A Small File

Request For A Large File

HTTP/2 Conn

Server

TCP Send Buff

4MB
Motivations of SMig: Sender-side HoLB Problem

- Request For A Small File
- Request For A Large File

Client -> HTTP/2 Conn -> Server

TCP Send Buff: 4MB
Motivations of SMig: Sender-side HoLB Problem
Motivations of SMig: Sender-side HoLB Problem

Table: Download Time for 10KB file (10Mbps BW, 50ms RTT)

<table>
<thead>
<tr>
<th>Concurrent Download</th>
<th>HTTPS(HTTP/1.1)</th>
<th>HTTP/2</th>
</tr>
</thead>
<tbody>
<tr>
<td>No</td>
<td>0.05</td>
<td>0.05</td>
</tr>
<tr>
<td>Yes</td>
<td>0.14</td>
<td>8.40</td>
</tr>
</tbody>
</table>

HoLB increases the small file download time by up to 70x, compared to HTTP/1.1!
Motivations of SMig: Sender-side HoLB Problem

**Table:** Download Time for 10KB file (10Mbps BW, 50ms RTT)

<table>
<thead>
<tr>
<th>Concurrent Download</th>
<th>HTTPS(HTTP/1.1)</th>
<th>HTTP/2</th>
</tr>
</thead>
<tbody>
<tr>
<td>No</td>
<td>0.05</td>
<td>0.05</td>
</tr>
<tr>
<td>Yes</td>
<td>0.14</td>
<td>8.40</td>
</tr>
</tbody>
</table>

HoLB increases the small file download time by up to 70x, compared to HTTP/1.1!

HoLB frequently happens in the real world (see paper for measurements)
Motivations of SMig: How to Handle Sender-side HoLB?

Start With Separate Conns?  
Response size is **Unknown** before request

Stream Prioritization?  
No help for HoLB in **TCP** layer

**Our Solution:** migrate an on-going stream of large file transfer to an idle connection.
Motivations of SMig: How to Handle Sender-side HoLB?

Start With Separate Conns?

Response size is Unknown before request

Stream Prioritization?

No help for HoLB in TCP layer

Our Solution: migrate an on-going stream of large file transfer to an idle connection.
Motivations of SMig: How to Handle Sender-side HoLB?

**Start With Separate Conns?**
- Response size is **Unknown** before request

**Stream Prioritization?**
- No help for HoLB in **TCP** layer

**Our Solution:** migrate an on-going stream of large file transfer to an idle connection.
SMig: Stream Migration Extension for HTTP/2

Stream Migration: migrate an on-going stream from one HTTP/2 connection to another.

Request For a Small File of 10KB

Request For a Large File of 50MB

Size of TCP Send Buff: 4MB
SMig: Stream Migration Extension for HTTP/2

Stream Migration: migrate an on-going stream from one HTTP/2 connection to another.
SMig: Stream Migration Extension for HTTP/2

Stream Migration: migrate an on-going stream from one HTTP/2 connection to another.

- Request For a Small File of 10KB
- Request For a Large File of 50MB
- Size of TCP Send Buff: 4MB
Stream Migration: migrate an on-going stream from one HTTP/2 connection to another.

HTTP/2 Conn 1

HTTP/2 Conn 2

TCP Send Buff 1

TCP Send Buff 2
SMig: Stream Migration Extension for HTTP/2

Stream Migration: migrate an on-going stream from one HTTP/2 connection to another.
Stream Migration: migrate an on-going stream from one HTTP/2 connection to another.

Request For a Small File of 10KB

Request For a Large File of 50MB

Size of TCP Send Buff: 4MB
SMig: Stream Migration Extension for HTTP/2

Stream Migration: migrate an on-going stream from one HTTP/2 connection to another.
Design of Smig: Migration Frame

Migration Frame expresses the intent of initiating a stream migration.

The flags ensure correct cross-connection ordering of frames (details in the paper).

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Length (24)</td>
<td></td>
</tr>
<tr>
<td>Type = OxA</td>
<td></td>
</tr>
<tr>
<td>Flags</td>
<td>ACK or END_STREAM</td>
</tr>
<tr>
<td>R</td>
<td></td>
</tr>
<tr>
<td>Stream Identifier (31)</td>
<td></td>
</tr>
<tr>
<td>dstCID: Destination Connection Identifier (96)</td>
<td></td>
</tr>
<tr>
<td>dstSID: Destination Stream Identifier (31)</td>
<td></td>
</tr>
</tbody>
</table>
Design of Smig: Migration Scenarios

A migration can be initiated by either a client or server. If no idle connection exists, SMig will create a new one.
A migration can be initiated by either a client or server. If no idle connection exists, SMig will create a new one.
A migration can be initiated by either a client or server. If no idle connection exists, SMig will create a new one.

- **(a)** Initiated by server w/ idle conn.
- **(b)** Initiated by client w/ idle conn.
- **(c)** Initiated by server w/o idle conn.
- **(d)** Initiated by client w/o idle conn.
A migration can be initiated by either a client or server. If no idle connection exists, SMig will create a new one.
SMig incurs low overhead for migration in common usage scenarios.

SMig strategically manages idle connections to strike a balance between resource usage and performance.

Various migration policies can be applied (examples shown soon).

SMig can work with HTTP/2 server push.

No new security vulnerability is introduced by SMig.
Other Design Considerations (Details in the Paper)

SMig incurs low overhead for migration in common usage scenarios.

SMig strategically manages idle connections to strike a balance between resource usage and performance.

Various migration policies can be applied (examples shown soon).

SMig can work with HTTP/2 server push.

No new security vulnerability is introduced by SMig.
SMig incurs low overhead for migration in common usage scenarios.

SMig strategically manages idle connections to strike a balance between resource usage and performance.

Various migration policies can be applied (examples shown soon).

SMig can work with HTTP/2 server push.

No new security vulnerability is introduced by SMig.
SMig incurs low overhead for migration in common usage scenarios.

SMig strategically manages idle connections to strike a balance between resource usage and performance.

Various migration policies can be applied (examples shown soon).

SMig can work with HTTP/2 server push.

No new security vulnerability is introduced by SMig.
Other Design Considerations (Details in the Paper)

SMig incurs low overhead for migration in common usage scenarios.

SMig strategically manages idle connections to strike a balance between resource usage and performance.

Various migration policies can be applied (examples shown soon).

SMig can work with HTTP/2 server push.

No new security vulnerability is introduced by SMig.
## Implementation of SMig

<table>
<thead>
<tr>
<th>Component</th>
<th>PL</th>
<th>LOC</th>
<th>OS Platforms</th>
</tr>
</thead>
<tbody>
<tr>
<td>HTTP/2 Client and Server</td>
<td>C++</td>
<td>7.5K</td>
<td>Linux/OS X</td>
</tr>
<tr>
<td>SMig extension</td>
<td>C++</td>
<td>1K</td>
<td>Linux/OS X</td>
</tr>
</tbody>
</table>
### Evaluation of SMig: Experimental Setup

#### Client & Server Setting

<table>
<thead>
<tr>
<th>Node</th>
<th>OS</th>
<th>CPU</th>
<th>Memory</th>
</tr>
</thead>
<tbody>
<tr>
<td>Client</td>
<td>OS X 10.10</td>
<td>2.7GHz Intel Core i5 CPU</td>
<td>8GB</td>
</tr>
<tr>
<td>Server</td>
<td>Ubuntu 14.04</td>
<td>3GHz Intel Core2 Duo E8400 CPU</td>
<td>4GB</td>
</tr>
</tbody>
</table>

#### Network Setting

<table>
<thead>
<tr>
<th>Type</th>
<th>Network Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>Wired</td>
<td>An emulated 10Mbps link with 50ms RTT</td>
</tr>
<tr>
<td>Cellular</td>
<td>A commercial LTE network</td>
</tr>
</tbody>
</table>
## Evaluation of SMig: Experimental Setup

### Client & Server Setting

<table>
<thead>
<tr>
<th>Node</th>
<th>OS</th>
<th>CPU</th>
<th>Memory</th>
</tr>
</thead>
<tbody>
<tr>
<td>Client</td>
<td>OS X 10.10</td>
<td>2.7GHz Intel Core i5 CPU</td>
<td>8GB</td>
</tr>
<tr>
<td>Server</td>
<td>Ubuntu 14.04</td>
<td>3GHz Intel Core2 Duo E8400 CPU</td>
<td>4GB</td>
</tr>
</tbody>
</table>

### Network Setting

<table>
<thead>
<tr>
<th>Type</th>
<th>Network Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>Wired</td>
<td>An emulated 10Mbps link with 50ms RTT</td>
</tr>
<tr>
<td>Cellular</td>
<td>A commercial LTE network</td>
</tr>
</tbody>
</table>
Evaluation Methodology

**Workload:** concurrent small & large file downloads (10 KB vs. 50 MB) in four scenarios. SMig migrates the large file.

- **NoMig:** SMig is disabled
- **MigSW:** server initiates the migration for the large file once it receives its request.
- **MigSP:** server initiates the migration after sending 100KB response data (for chunked mode encoding).
- **MigCP:** client initiates the migration once it receives the response header.
Evaluation: Small File Download Time over Wired Network

Figure: SMig’s Impact on Small File Download (Wired)
Evaluation: Small File Download Time over LTE

**Figure:** SMig’s Impact on Small File Download (Cellular)
Evaluation: Impact of Migration on Large File Download Time

**Figure**: SMig’s Impact on Large File Download (Wired)
SMig: an HTTP/2 extension allowing a client or server to migrate an on-going HTTP/2 stream from one connection to another.

SMig eliminates sender-side HoLB. It reduces the delay-sensitive file download time by up to 99% when concurrent transfers occur.

SMig brings other benefits and usage scenarios (see the paper for details).
SMig: an HTTP/2 extension allowing a client or server to migrate an on-going HTTP/2 stream from one connection to another.

SMig eliminates sender-side HoLB. It reduces the delay-sensitive file download time by up to 99% when concurrent transfers occur.

SMig brings other benefits and usage scenarios (see the paper for details).
SMig: an HTTP/2 extension allowing a client or server to migrate an on-going HTTP/2 stream from one connection to another.

SMig eliminates sender-side HoLB. It reduces the delay-sensitive file download time by up to 99% when concurrent transfers occur.

SMig brings other benefits and usage scenarios (see the paper for details).
Thanks!