Adapting Server Frameworks to Support HTTP/2 in Proxy Settings

Master Thesis Defense

Junyu Pu

Outline

- Introduction
- HTTP/2 New Features
- Proxy with HTTP/2
- HTTP/2 Support in SANE
- Evaluation
- Conclusion

Introduction

HTTP/2 is the new protocol for the internet for next decade. Research and adapt current technologies to embrace this new protocol become necessary.

- What is the main purposes of this thesis?
- What are the benefits we can gain from upgrading to HTTP/2?
- Why we want to introduce HTTP/2 into SANE?

Introduction

What is the main purpose of this thesis?

- A **guide** to embrace the new HTTP
- **Research** the possibility to apply new technologies in current projects
- Give **example** to adapt current server frameworks to support HTTP/2
- **Estimate** the performance by enabling HTTP/2

Introduction

What are the benefits we can gain from upgrading to HTTP/2?

- We expect a better performance by such as:
  - Decrease the traffic during transactions
  - Increase system robustness and tolerance
- More secure
- Compatibility

Introduction

Why introduce HTTP/2 into SANE?

- SANE is specified memory modify proxy to achieve certain functions.
- An application level proxy can help us to build a general method to upgrade system frameworks to support HTTP/2 in future.
- SANE only uses POST and GET request, which is clear other factors to analyze the performance

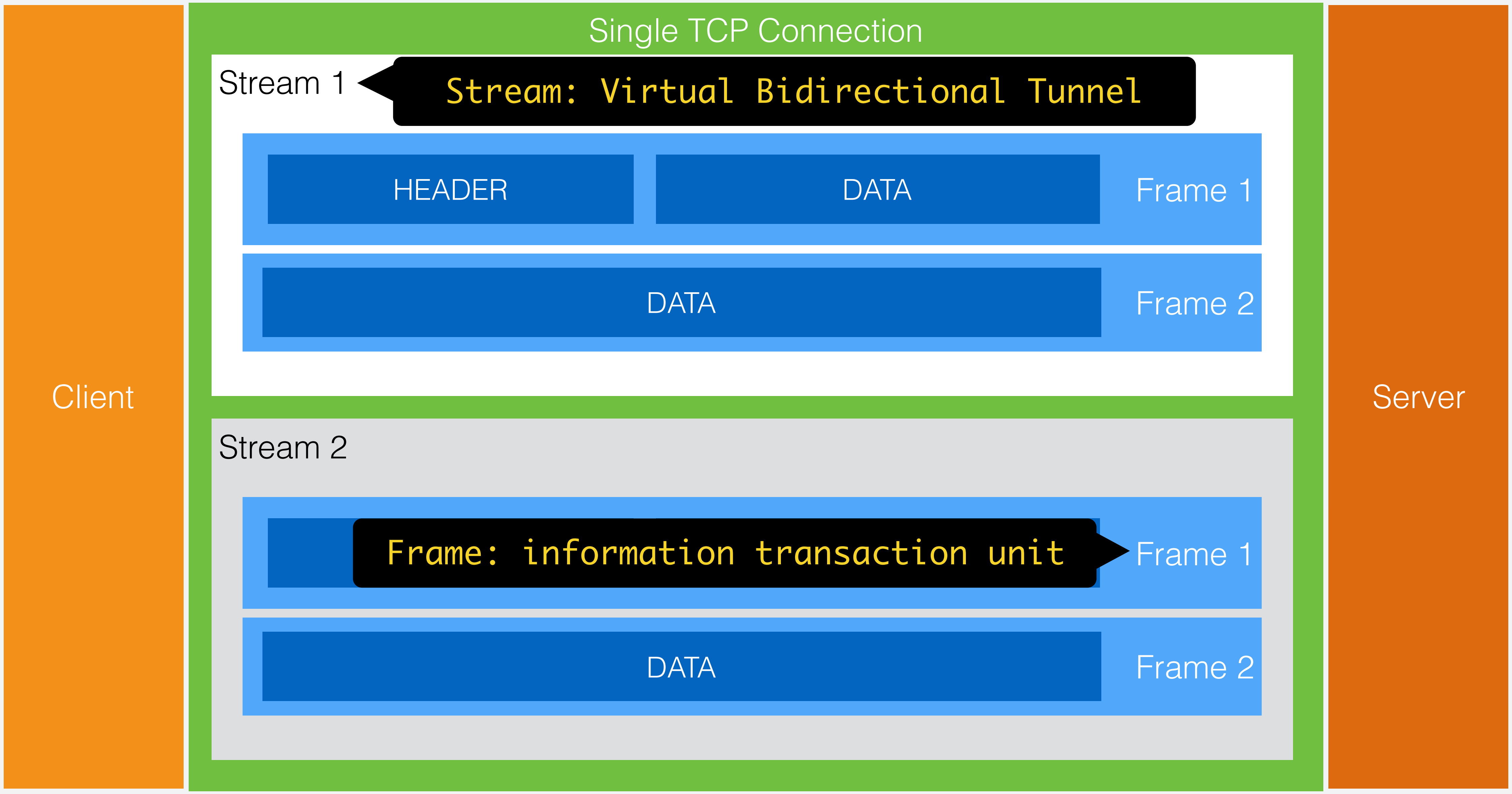
HTTP/2 New Features

Before focusing on the proxy with HTTP/2 setting, there are few new features in HTTP/2 need to be discussed first.

- Binary Frame Layer
- Multiplexing
- Header Compression
- Server Push
- Priority

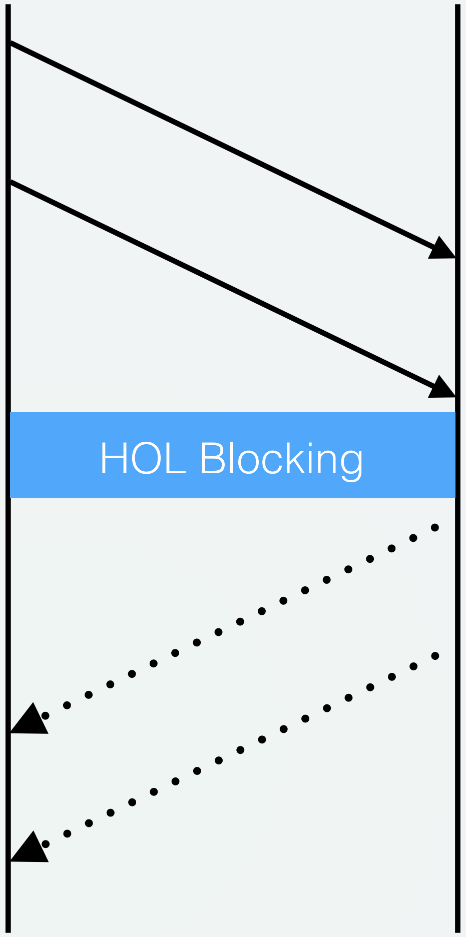
HTTP/2 New Features

Binary Frame Layer

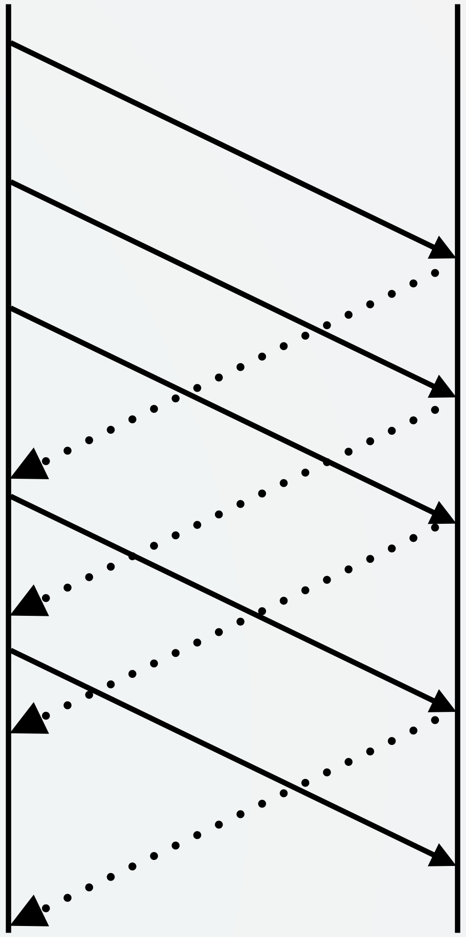


HTTP/2 New Features

Multiplex



HTTP/1.1 with pipelining



HTTP/2 Multiplex

- All communications are performed within a single TCP connection.
- HTTP/2 can contain multiple streams and exchanging messages in parallel within a single TCP connection
- Solves Head-of-line blocking problem

HTTP/2 New Features

Header Compression

- Header compressed based on HPACK (RFC7541)
- Binary Coded based on Huffman code
- Consist of Static Table and Dynamic Table
  - Static Table: contain predefined and unchangeable list of header fields
  - Dynamic Table: contain writeable and updated header fields
- Purpose: reduce header size

HTTP/2 New Features

Server Push

- Server can send multiple responses to a single client request
- Client decides accept or reject server push
- Server push reduces requests number and improve communication efficiency also provide better web browsing experience for users
- *Not suitable for one-response-one-request type application*

HTTP/2 New Features

Priority

- HTTP message can be split into many individual frames in HTTP/2
- Allow frames from multiple streams to be multiplexed and reordered
- HTTP/2 standard allows each stream to have an associated weight and dependency
- Server can decide the priority of each frames

Proxy with HTTP/2

Proxy is a middleware between server and client. It is a frequently used component for web applications. How can we upgrade to HTTP/2?

- Before Upgrade to HTTP/2
- Server/Client Frameworks Upgrading
- Proxy Support
- Three Architecture Proxies
- Proxy Concept with HTTP/2 Setting
- Build Proxy with HTTP/2 Setting

Proxy with HTTP/2

Before Upgrade to HTTP/2

Application Layer Protocol Negotiation

- It is a TLS extension for secure connection
- It helps server and client negotiation which protocol to use
- If server does not support HTTP/2, the connection will be fall back to HTTP/1.1
- HTTPS becomes nearly defector for HTTP/2

Proxy with HTTP/2

Before Upgrade to HTTP/2

HTTP Request/Response

- Request and response are binary frames
- The frames can be header or data
- This allows multiplexing by associate with the same stream ID
- Hard to generate a HTTP/2 request/response with other underneath libraries help

Proxy with HTTP/2

Before Upgrade to HTTP/2

Header Compression

- HTTP/2 enables header compression by default
- Based on HPACK algorithm

Server Push

- Need server supports and configuration
- Not efficient for one request-response application

Proxy with HTTP/2

Before Upgrade to HTTP/2

Summary

- HTTP/2 is complicated and much more harder than HTTP/1.1 to manually coding
- However, application above HTTP/2 does not need to worry about to much since API and method did not change a lot
- **Use third-party library to support HTTP/2 become necessary**

Proxy with HTTP/2

Server/Client Frameworks Upgrading

Client

- On shipped modern browser already support HTTP/2
- Third-party library add HTTP/2 support on Android
- iOS supports HTTP/2 with system API: NSURLSession
- libcurl supports HTTP/2 with last few updates

Server/Client Frameworks Upgrading

Server

- Apache supports HTTP/2 since version 2.3.14 by implemented mod_http2 module
- nginx supports HTTP/2 with previous support for SPDY
- New server frameworks such as H2O supports HTTP/2 natively since it is developed for HTTP/2

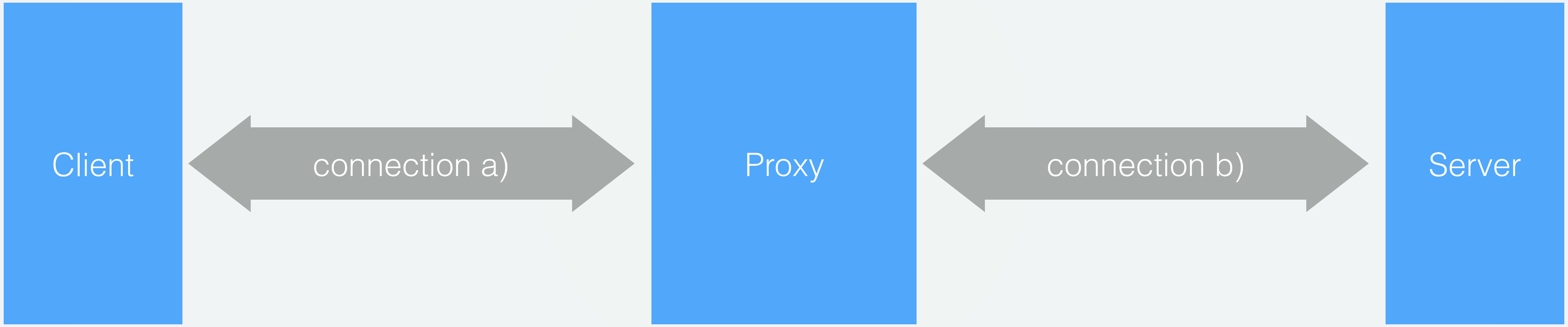
Proxy with HTTP/2

Proxy Support

- Proxy is a middleware between client and server, the connection should be upgrade to support HTTP/2
- Proxy behalf as client and server at the same time
- Fully HTTP/2 support products are not on the market right now
- ***Solution: adapt current frameworks to support proxy with HTTP/2 partially.***

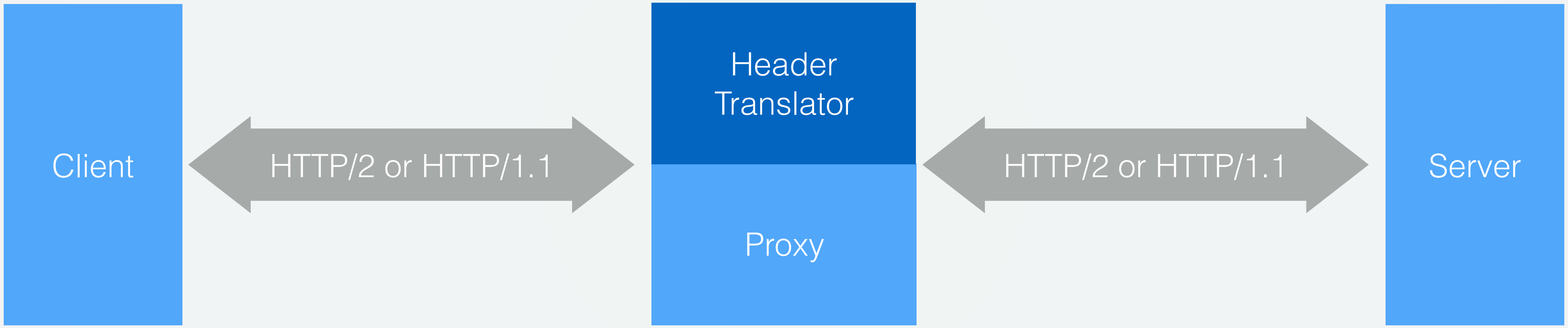
Proxy with HTTP/2

Three Architecture Proxies



- Upgrade Proxy: only connection b) supports HTTP/2
- Straightforward Proxy: both connections support HTTP/2
- Downgrade Proxy: only connection a) support HTTP/2

Proxy Concept with HTTP/2 Setting



- Header Translator Module to change the header from HTTP/1.1 to HTTP/2 if needed
- Using web proxy to get resource from destination server

Proxy with HTTP/2

Build Proxy with HTTP/2 Setting

- Using nghttp2 can easily build a proxy to adapt HTTP into current system frameworks
- It is easy to configure and supports all the three different architectures of HTTP/2 involved proxies
- Corporate with traditional web proxy can build SPDY alike proxy

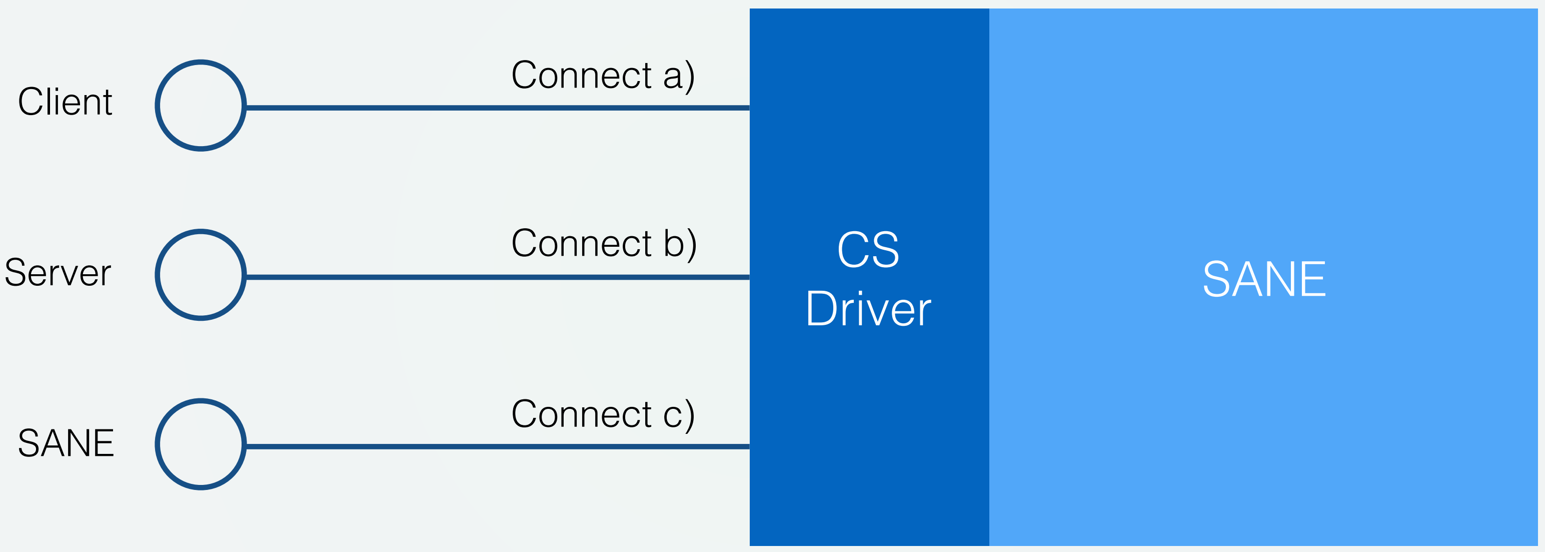
HTTP/2 Support in SANE

Proxy is a middleware between server and client. It is a frequently used component for web applications. How can we upgrade to HTTP/2?

- Deconstruct SANE
- Adapt SANE to Support HTTP/2
    - With Translator Module
    - With System Library

HTTP/2 Support in SANE

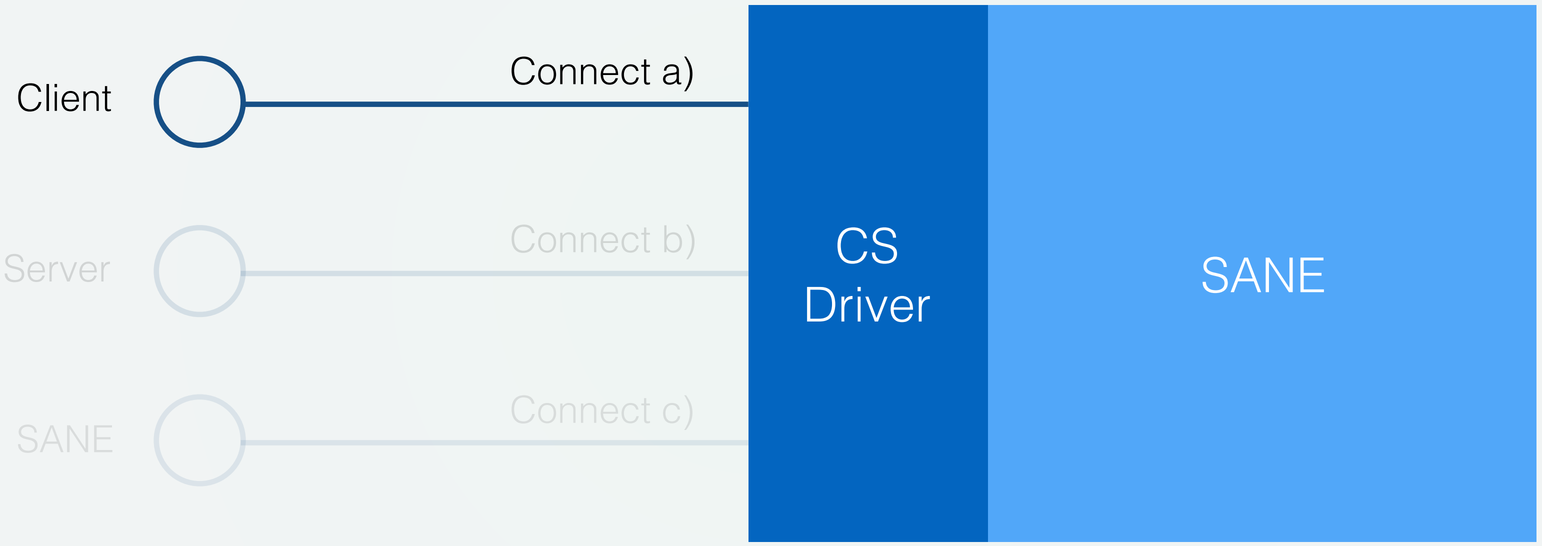
Deconstruct SANE



- Request and response from outside of SANE are handled by Crowdsourcing Driver.
- There are 3 connection outside SANE
- ***HTTP/2 support in SANE need upgrade these 3 connections***

HTTP/2 Support in SANE

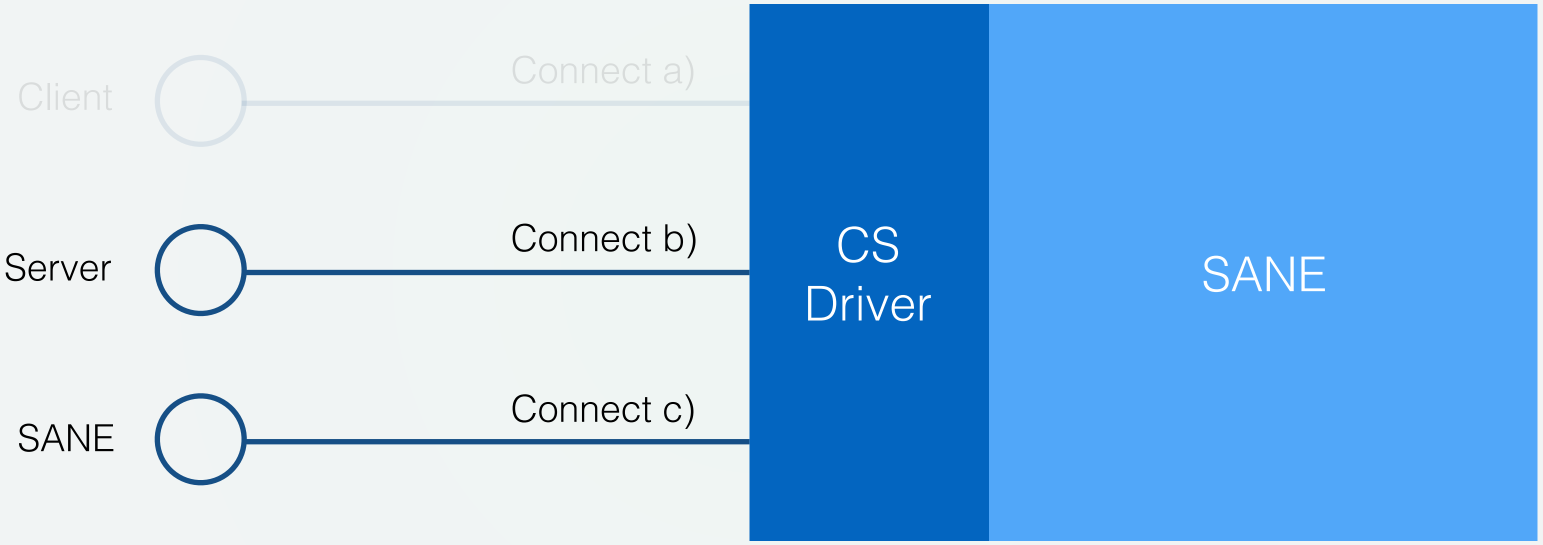
Deconstruct SANE



- Requirement to support HTTP/2 in connect a)
  - Client should support HTTP/2
  - SANE server framework should upgrade to support HTTP/2

HTTP/2 Support in SANE

Deconstruct SANE



- Requirement to support HTTP/2 in connect b) & c)
  - SANE acts as client, HTTP Library need upgrade
  - Server and another SANE also need upgrade to support HTTP/2

HTTP/2 Support in SANE

Adapt SANE to Support HTTP/2 with Translator Module

- Translator module responsible for HTTP/2 and HTTP/1.1 request
    - Pro: Easy to implement, and without changing the current architecture of SANE
    - Con: Performance drawback and limit

HTTP/2 Support in SANE

Adapt SANE to Support HTTP/2 with System Library

- Adapt SANE support HTTP/2 natively with curl_lib for HTTP requests
  - Pro: more efficient, less layer for system
  - Con: need change the code and rewrite the HTTP Library

HTTP/2 Support in SANE

Adapt SANE to Support HTTP/2 with lib_curl

```
…
    $ch = curl_init();
    curl_setopt($ch, CURLOPT_HTTP_VERSION,
     CURL_HTTP_VERSION_2_0);
    curl_setopt($ch, CURLOPT_URL,$connect);
    curl_setopt($ch, CURLOPT_POST, 1);
    curl_setopt($ch, CURLOPT_RETURNTRANSFER, 1 );
    curl_setopt($ch, CURLOPT_POSTFIELDS, $params);
    curl_setopt($ch, CURLOPT_HEADER, 1);
    curl_setopt($ch, CURLINFO_HEADER_OUT, true);
    $result = curl_exec($ch);
    curl_close($ch);
 …
```

CURL_HTTP_VERSION_2_0

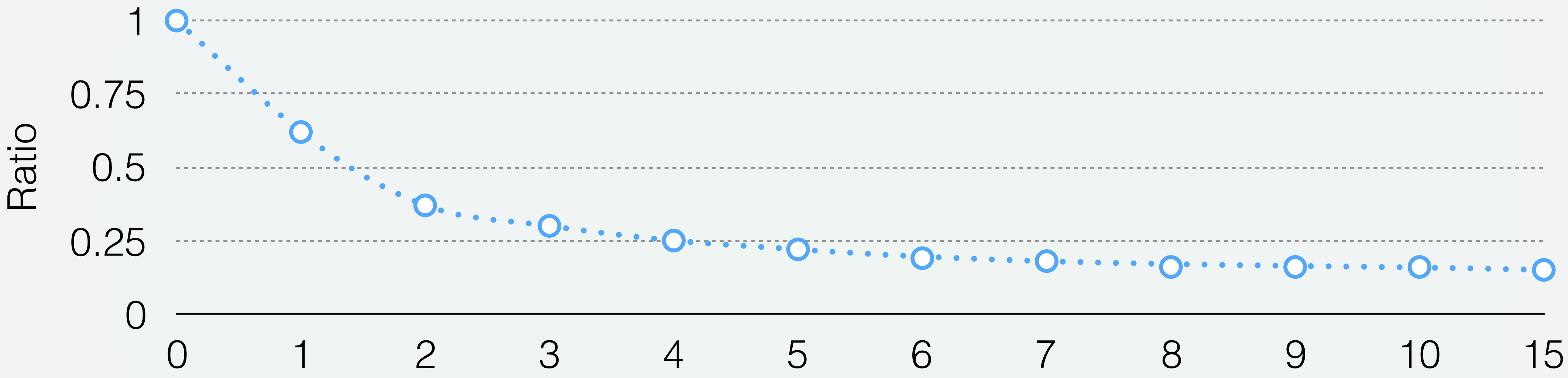
Evaluation

The performance improvement because of the new feature of HTTP/2

- Header Compression Ratio
- Proxy with HTTP/2 Setting
- SANE with HTTP/2 setting

Evaluation

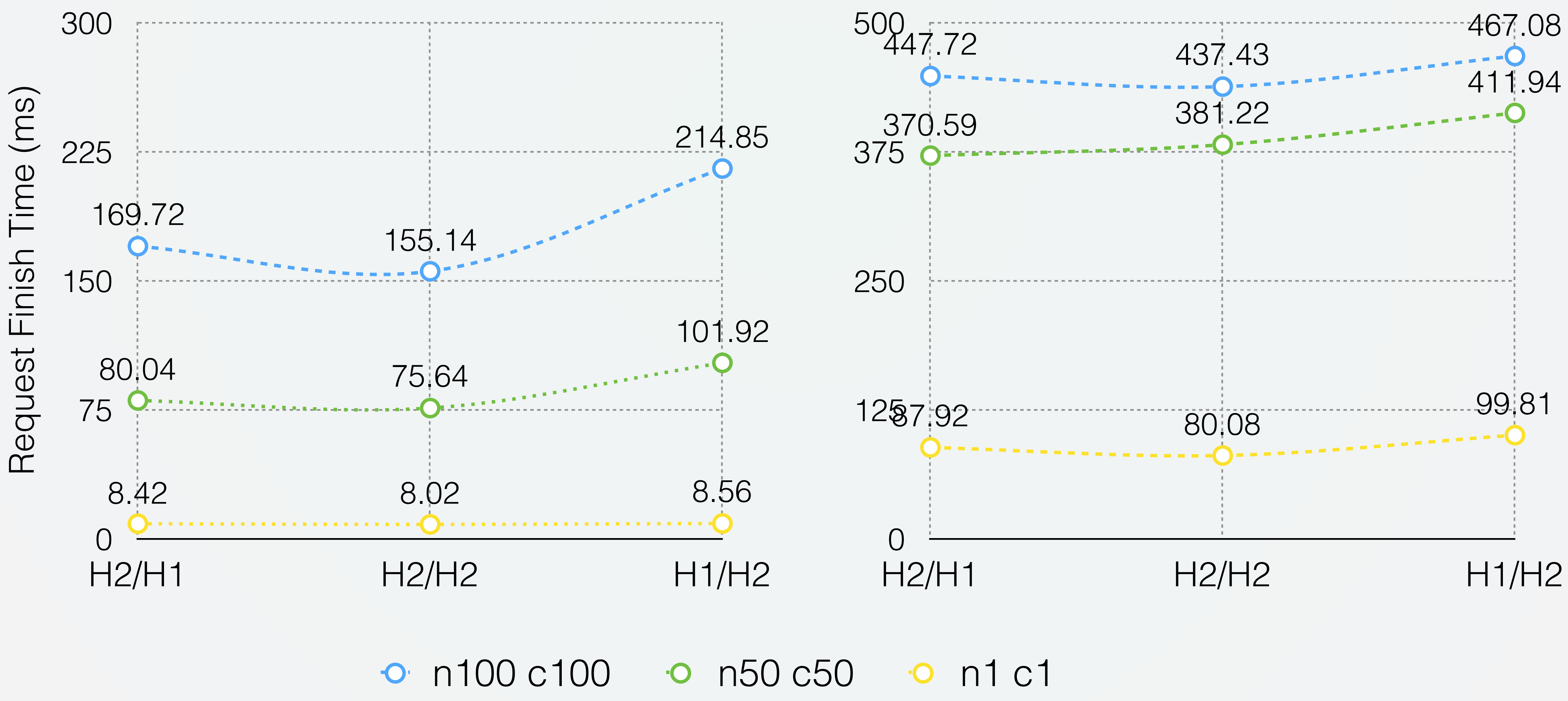
Header Compression Ratio



- HPACK can dramatically reduce the header size and decrease the traffic from header transaction.
- Dynamic Table contributes most in header compression
- Header Compression is related to header complexity and redundancy.

Evaluation

Proxy with HTTP/2 Setting



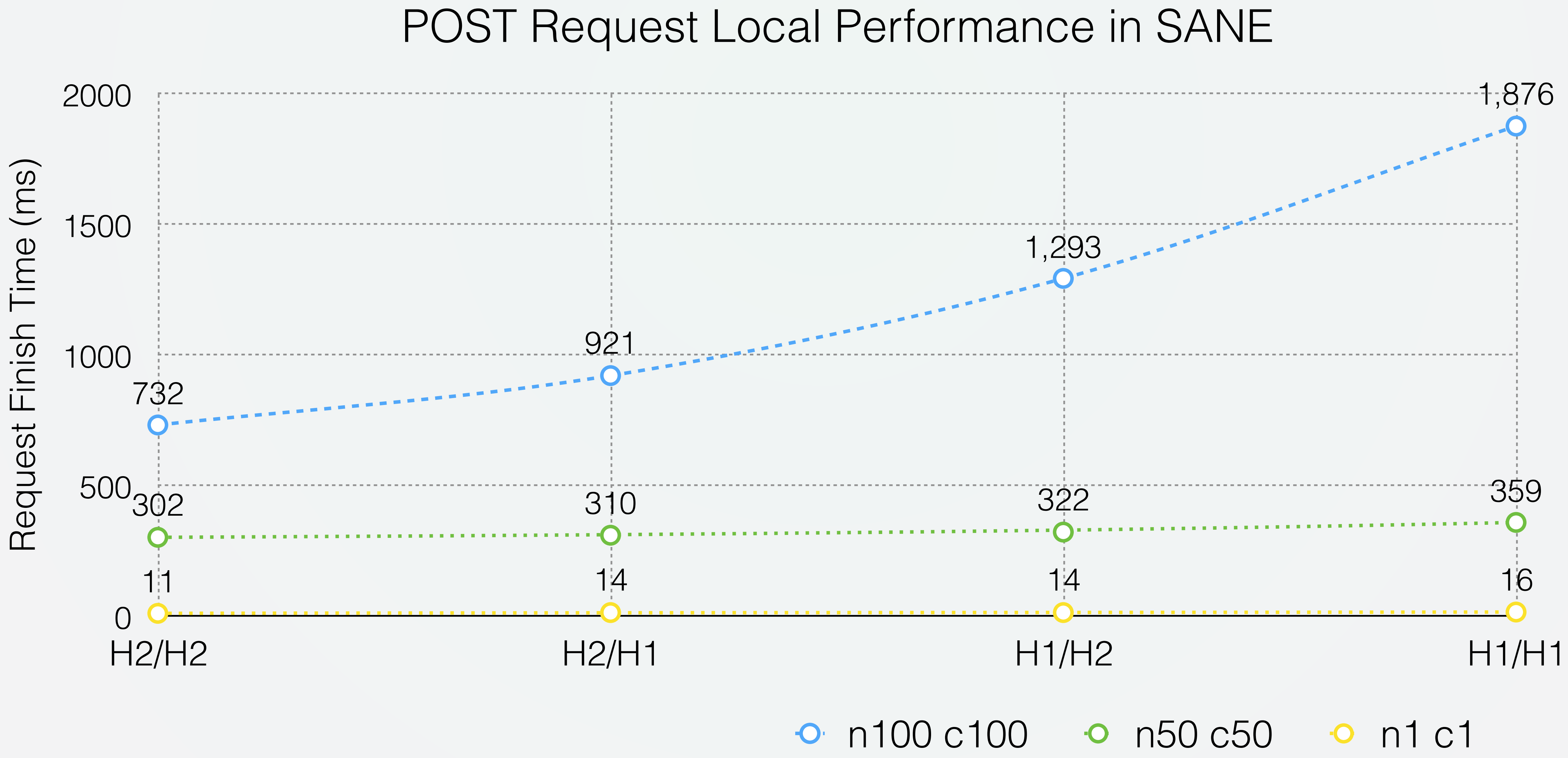
Evaluation

Proxy with HTTP/2 Setting

- Different architecture of proxies will affect the performance.
- HTTP/2 enabled on the client side is preferable than server side.
- Straightforward Proxy has most performance improvement.
- Other factors affect the performance.

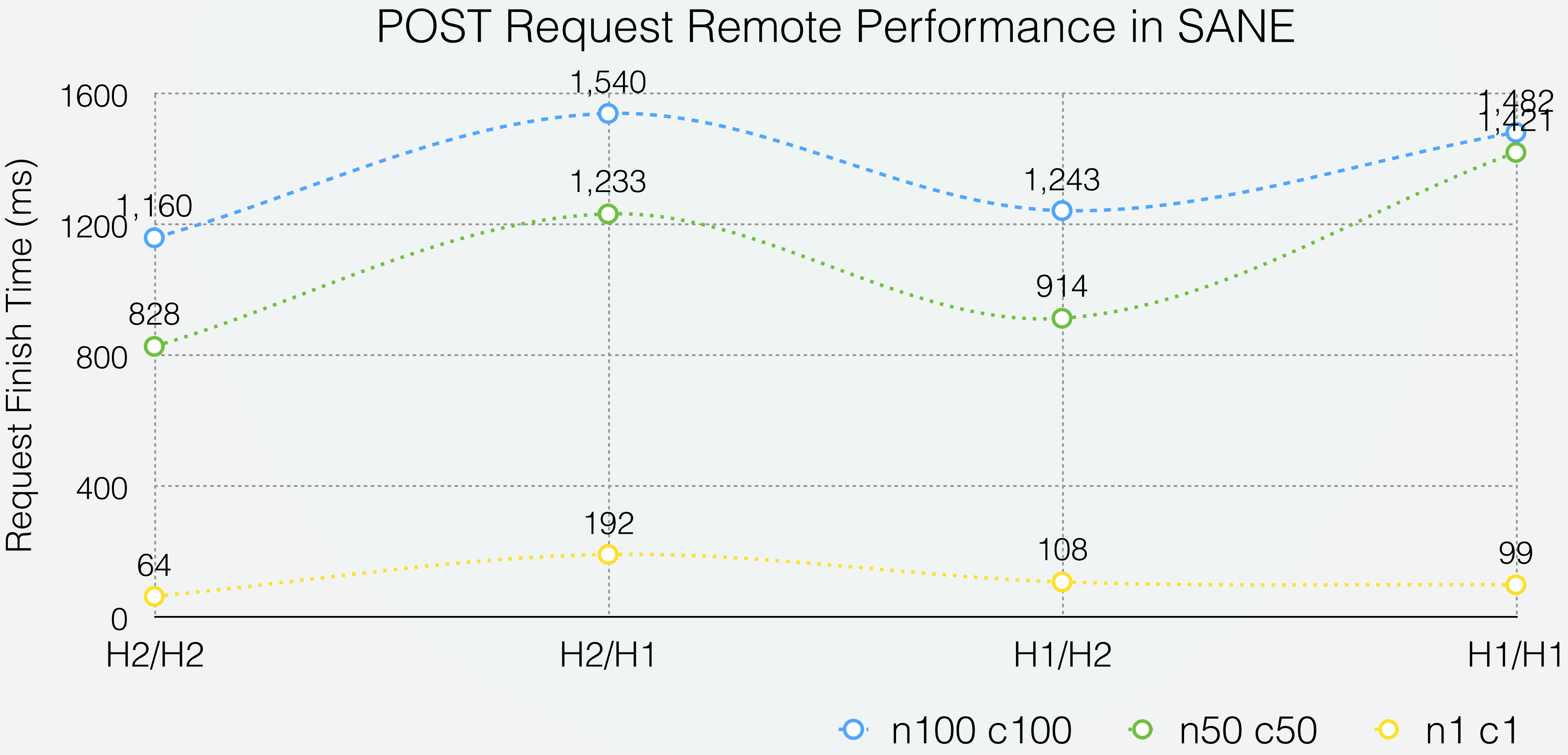
Evaluation

SANE with HTTP/2 Setting



Evaluation

SANE with HTTP/2 Setting



Evaluation

SANE with HTTP/2 Setting

- Enable HTTP/2 can improve the performance in SANE
- Enable HTTP/2 on slower connection side can bring more benefit
- HTTP/2 increases the concurrence for SANE
- Straightforward proxy architecture has more performance improvement than others

Conclusion

- HTTP/2 and Proxy
- SANE with HTTP/2 Support
- Recommendations and Future Work

Conclusion

HTTP/2 and Proxy

- Header compression contributes most in performance improvement.
- HTTP/2 support is easy to adapt and integrate.
- Application supports HTTP/2 relies on underneath infrastructure.
- Cross-protocol proxy is feasible and performance influenced by different architectures.
- Straightforward Proxy performance surpasses the cross-protocol proxy.

Conclusion

SANE with HTTP/2 Support

- Adapting SANE to support HTTP/2 is feasible.
- SANE with HTTP/2 support can reduce traffic bandwidth and improve throughput.
- The security and anonymity of SANE do not jeopardize with HTTP/2 enabled.
- Caveat, for latency sensitive request should be careful about enabling HTTP/2 in the system.

Conclusion

Recommendations and Future Work

Recommendations

- All the tools used in the thesis is under fast developing, bugs and unknown issues may affects the results
- Common softwares does not fully support HTTP/2, be ware of the tool you choose when upgrading to HTTP/2
- Not wise to code for HTTP/2 support from scratch, due to the complicity and binary nature

Recommendations and Future Work

Future Work

- Adapting Cache Proxy support HTTP/2 on back-end. For example, HTTP/2 supports in Squid is under developing
- Adapting MapBiquios with HTTP/2 support. SANE is only one part of MapBiquious, using the methods presented in this thesis can adapt whole MapBiquious system to support HTTP/2.

Questions

Master Thesis Defense

Junyu Pu

Thanks

Master Thesis Defense

Junyu Pu