

Performance Evaluation of WebSocket Protocol for Implementation of Full-Duplex Web Streams

Oleg Bilovus

Università degli Studi di Salerno

1st Scalability Research Forum

WebSocket

Oleg Bilovus

Background

- HTTP polling
- HTTP long polling
- Streaming

WebSocket
protocol

- Definition
- Handshake
- Upgrade Request
- Upgrade Response
- Frame
- API

Performance vs
TCP Socket

- Performance Evaluation
- WebSocket sequence diagram
- Network traffic
- Handshake overhead
- Frame overhead
- Results
- Data Transfer Time
- Connection
- Data

Conclusion

WebSocket

2024-06-03

We will talk about WebSockets and compare its performance with TCP Socket. But, before diving into analyzing the performance we need to understand why we needed WebSockets and what they are.

Outline

Background

HTTP polling

HTTP long polling

Streaming

WebSocket protocol

Definition

Handshake

Frame

API

Performance vs TCP Socket

Performance Evaluation

WebSocket sequence diagram

Network traffic

Data Transfer Time

Conclusion

WebSocket
Oleg Bilovus

Background

- HTTP polling
- HTTP long polling
- Streaming

WebSocket protocol

- Definition
- Handshake
- Upgrade Request
- Upgrade Response
- Frame
- API

Performance vs TCP Socket

- Performance Evaluation
- WebSocket sequence diagram
- Network traffic
- Handshake overhead
- Frame overhead
- Results
- Data Transfer Time
- Connection
- Data

Conclusion

2024-06-03

WebSocket

- Background

Background

- Outline

Outline

Background

- HTTP polling
- HTTP long polling
- Streaming

WebSocket protocol

- Definition
- Handshake
- Frame
- API

Performance vs TCP Socket

- Performance Evaluation
- WebSocket sequence diagram
- Network traffic
- Data Transfer Time

Conclusion

Background

- ▶ Historically, creating web applications that need bidirectional communication between a client and a server has required an abuse of HTTP to poll the server for updates while sending upstream notifications as distinct HTTP calls.

WebSocket
Oleg Bilovus

Background

- HTTP polling
- HTTP long polling
- Streaming

WebSocket protocol

- Definition
- Handshake
- Upgrade Request
- Upgrade Response
- Frame
- API

Performance vs TCP Socket

- Performance Evaluation
- WebSocket sequence diagram
- Network traffic
- Handshake overhead
- Frame overhead
- Results
- Data Transfer Time
- Connection
- Data

Conclusion

Background

2024-06-03

WebSocket

- Background
- Background

Background

Historically, creating web applications that need bidirectional communication between a client and a server has required an abuse of HTTP to poll the server for updates while sending upstream notifications as distinct HTTP calls.

Background

- ▶ Historically, creating web applications that need bidirectional communication between a client and a server has required an abuse of HTTP to poll the server for updates while sending upstream notifications as distinct HTTP calls.

WebSocket
Oleg Bilovus

Background

- HTTP polling
- HTTP long polling
- Streaming

WebSocket protocol

- Definition
- Handshake
- Upgrade Request
- Upgrade Response
- Frame
- API

Performance vs TCP Socket

- Performance Evaluation
- WebSocket sequence diagram
- Network traffic
- Handshake overhead
- Frame overhead
- Results
- Data Transfer Time
- Connection
- Data

Conclusion

Background

2024-06-03

WebSocket

- Background
- Background

Background

Historically, creating web applications that need bidirectional communication between a client and a server has required an abuse of HTTP to poll the server for updates while sending upstream notifications as distinct HTTP calls.

Background

- ▶ Historically, creating web applications that need bidirectional communication between a client and a server has required an abuse of HTTP to poll the server for updates while sending upstream notifications as distinct HTTP calls.

WebSocket
Oleg Bilovus

Background

- HTTP polling
- HTTP long polling
- Streaming

WebSocket protocol

- Definition
- Handshake
- Upgrade Request
- Upgrade Response
- Frame
- API

Performance vs TCP Socket

- Performance Evaluation
- WebSocket sequence diagram
- Network traffic
- Handshake overhead
- Frame overhead
- Results
- Data Transfer Time
- Connection
- Data

Conclusion

Background

2024-06-03

WebSocket

- Background
- Background

Background

Historically, creating web applications that need bidirectional communication between a client and a server has required an abuse of HTTP to poll the server for updates while sending upstream notifications as distinct HTTP calls.

Background

- ▶ Historically, creating web applications that need bidirectional communication between a client and a server has required an abuse of HTTP to poll the server for updates while sending upstream notifications as distinct HTTP calls.

WebSocket
Oleg Bilovus

Background

- HTTP polling
- HTTP long polling
- Streaming

WebSocket protocol

- Definition
- Handshake
- Upgrade Request
- Upgrade Response
- Frame
- API

Performance vs TCP Socket

- Performance Evaluation
- WebSocket sequence diagram
- Network traffic
- Handshake overhead
- Frame overhead
- Results
- Data Transfer Time
- Connection
- Data

Conclusion

Background

2024-06-03

WebSocket

- Background
- Background

Background

Historically, creating web applications that need bidirectional communication between a client and a server has required an abuse of HTTP to poll the server for updates while sending upstream notifications as distinct HTTP calls.

Background

- ▶ Historically, creating web applications that need bidirectional communication between a client and a server has required an abuse of HTTP to poll the server for updates while sending upstream notifications as distinct HTTP calls.

WebSocket
Oleg Bilovus

Background

- HTTP polling
- HTTP long polling
- Streaming

WebSocket protocol

- Definition
- Handshake
- Upgrade Request
- Upgrade Response
- Frame
- API

Performance vs TCP Socket

- Performance Evaluation
- WebSocket sequence diagram
- Network traffic
- Handshake overhead
- Frame overhead
- Results
- Data Transfer Time
- Connection
- Data

Conclusion

Background

2024-06-03

WebSocket

- Background
- Background

Background

Historically, creating web applications that need bidirectional communication between a client and a server has required an abuse of HTTP to poll the server for updates while sending upstream notifications as distinct HTTP calls.

Background

- ▶ Historically, creating web applications that need bidirectional communication between a client and a server has required an abuse of HTTP to poll the server for updates while sending upstream notifications as distinct HTTP calls.

WebSocket
Oleg Bilovus

Background

- HTTP polling
- HTTP long polling
- Streaming

WebSocket protocol

- Definition
- Handshake
- Upgrade Request
- Upgrade Response
- Frame
- API

Performance vs TCP Socket

- Performance Evaluation
- WebSocket sequence diagram
- Network traffic
- Handshake overhead
- Frame overhead
- Results
- Data Transfer Time
- Connection
- Data

Conclusion

Background

2024-06-03

WebSocket

- Background
- Background

Background

Historically, creating web applications that need bidirectional communication between a client and a server has required an abuse of HTTP to poll the server for updates while sending upstream notifications as distinct HTTP calls.

Background

- ▶ Historically, creating web applications that need bidirectional communication between a client and a server has required an abuse of HTTP to poll the server for updates while sending upstream notifications as distinct HTTP calls.

WebSocket
Oleg Bilovus

Background

- HTTP polling
- HTTP long polling
- Streaming

WebSocket protocol

- Definition
- Handshake
- Upgrade Request
- Upgrade Response
- Frame
- API

Performance vs TCP Socket

- Performance Evaluation
- WebSocket sequence diagram
- Network traffic
- Handshake overhead
- Frame overhead
- Results
- Data Transfer Time
- Connection
- Data

Conclusion

Background

2024-06-03

WebSocket

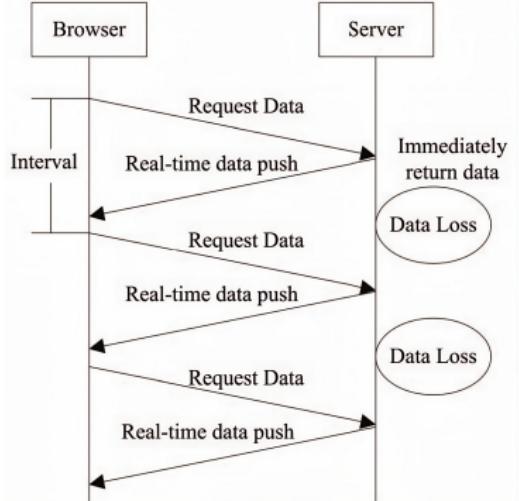
- Background
- Background

Background

Historically, creating web applications that need bidirectional communication between a client and a server has required an abuse of HTTP to poll the server for updates while sending upstream notifications as distinct HTTP calls.

HTTP polling

Check whether the server is changed in a while, thereby performing incremental updates.



WebSocket
Oleg Bilovus

Background
HTTP polling
HTTP long polling
Streaming

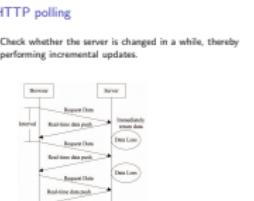
WebSocket protocol
Definition
Handshake
Upgrade Request
Upgrade Response
Frame
API

Performance vs TCP Socket
Performance Evaluation
WebSocket sequence diagram
Network traffic
Handshake overhead
Frame overhead
Results
Data Transfer Time
Connection
Data

Conclusion

WebSocket
Background
HTTP polling
HTTP polling

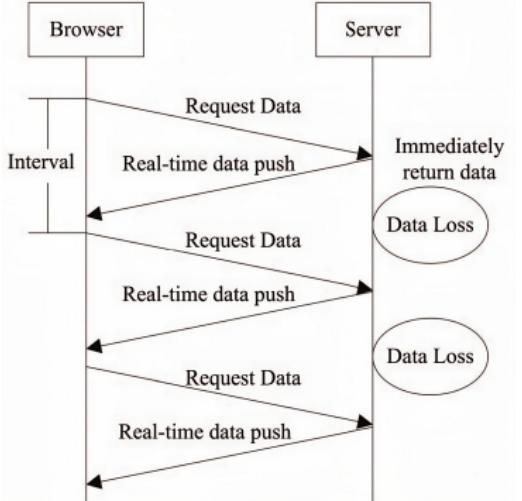
2024-06-03



A client can send data and ask for data at the same time. But, if client has no data and server has no data, a request and response will still be generated with all the HTTP headers and thus wasting resources. No real-time data because while the client waits, an event could occur and the client will know about it only when the timeout expires.

HTTP polling

Check whether the server is changed in a while, thereby performing incremental updates.



► How often to query?

WebSocket
Oleg Bilovus

Background
HTTP polling
HTTP long polling
Streaming

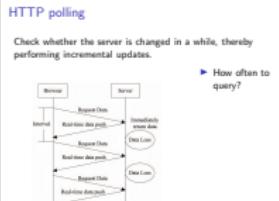
WebSocket protocol
Definition
Handshake
Upgrade Request
Upgrade Response
Frame
API

Performance vs TCP Socket
Performance Evaluation
WebSocket sequence diagram
Network traffic
Handshake overhead
Frame overhead
Results
Data Transfer Time
Connection
Data

Conclusion

WebSocket
Background
HTTP polling
HTTP polling

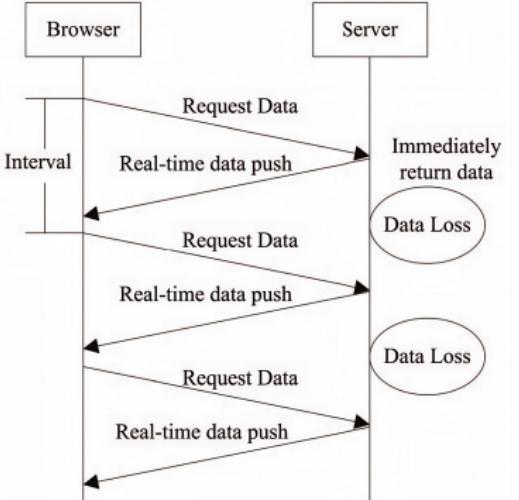
2024-06-03



A client can send data and ask for data at the same time. But, if client has no data and server has no data, a request and response will still be generated with all the HTTP headers and thus wasting resources. No real-time data because while the client waits, an event could occur and the client will know about it only when the timeout expires.

HTTP polling

Check whether the server is changed in a while, thereby performing incremental updates.



- ▶ How often to query?
- ▶ Continuously short interval requests will be washed away the server.

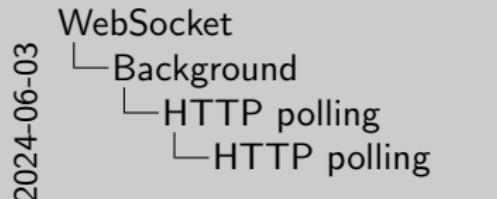
WebSocket
Oleg Bilovus

Background
HTTP polling
HTTP long polling
Streaming

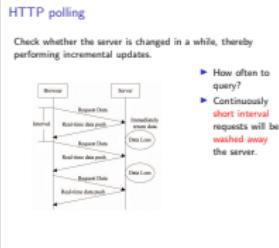
WebSocket protocol
Definition
Handshake
Upgrade Request
Upgrade Response
Frame
API

Performance vs TCP Socket
Performance Evaluation
WebSocket sequence diagram
Network traffic
Handshake overhead
Frame overhead
Results
Data Transfer Time
Connection
Data

Conclusion

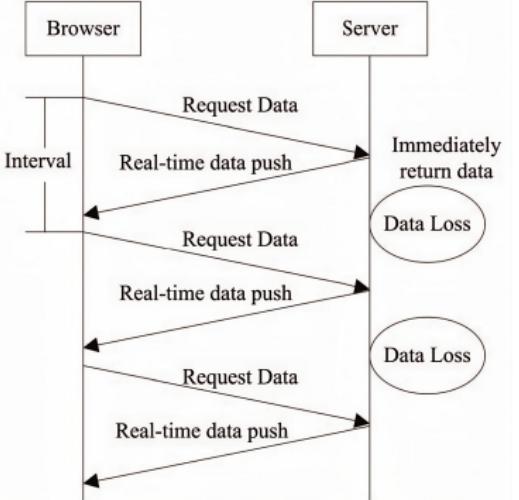


4 / 35



HTTP polling

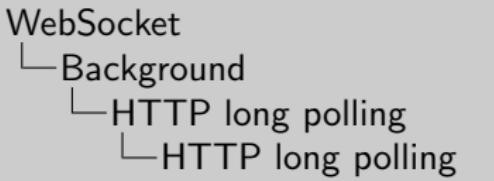
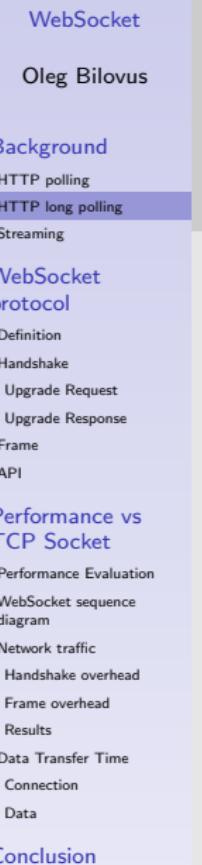
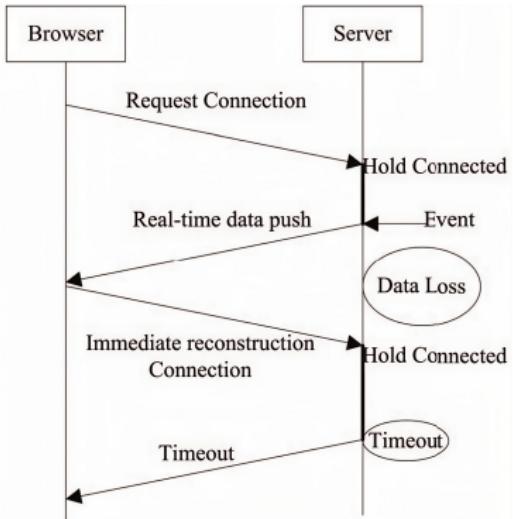
Check whether the server is changed in a while, thereby performing incremental updates.



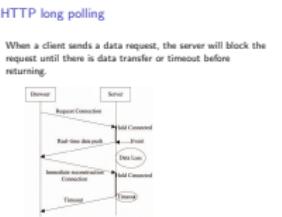
- ▶ How often to query?
- ▶ Continuously short interval requests will be washed away the server.
- ▶ Long interval will require more time to reach the client, no real-time data.

HTTP long polling

When a client sends a data request, the server will block the request until there is data transfer or timeout before returning.



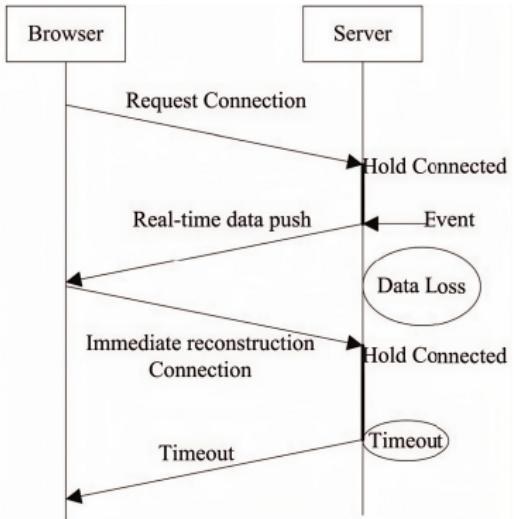
2024-06-03



Can hold the connection up to a certain time, after that a timeout is exceeded and need a new connection. No bidirectional because the client may only send data the first time, but then it will only receive until a timeout and another request is made. In the normal polling we could have bidirectional because the interval was shorter.

HTTP long polling

When a client sends a data request, the server will block the request until there is data transfer or timeout before returning.



► Solve the short polling frequency to access the server.

WebSocket
Oleg Bilovus
Background
HTTP polling
HTTP long polling
Streaming
WebSocket protocol
Definition
Handshake
Upgrade Request
Upgrade Response
Frame
API
Performance vs TCP Socket
Performance Evaluation
WebSocket sequence diagram
Network traffic
Handshake overhead
Frame overhead
Results
Data Transfer Time
Connection
Data
Conclusion

2024-06-03
WebSocket
Background
HTTP long polling
HTTP long polling

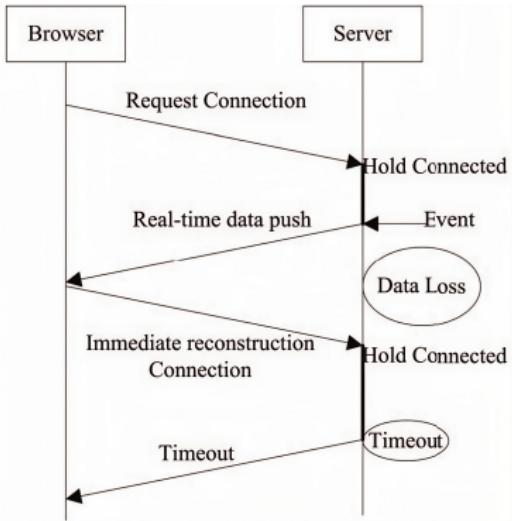
HTTP long polling
When a client sends a data request, the server will block the request until there is data transfer or timeout before returning.

Solve the short polling frequency to access the server.

Can hold the connection up to a certain time, after that a timeout is exceeded and need a new connection. No bidirectional because the client may only send data the first time, but then it will only receive until a timeout and another request is made. In the normal polling we could have bidirectional because the interval was shorter.

HTTP long polling

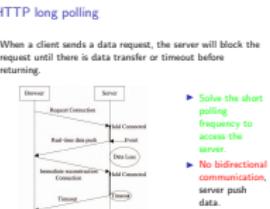
When a client sends a data request, the server will block the request until there is data transfer or timeout before returning.



- ▶ **Solve the short polling frequency to access the server.**
- ▶ **No bidirectional communication, server push data.**

WebSocket
Oleg Bilovus
Background
HTTP polling
HTTP long polling
Streaming
WebSocket protocol
Definition
Handshake
Upgrade Request
Upgrade Response
Frame
API
Performance vs TCP Socket
Performance Evaluation
WebSocket sequence diagram
Network traffic
Handshake overhead
Frame overhead
Results
Data Transfer Time
Connection
Data
Conclusion

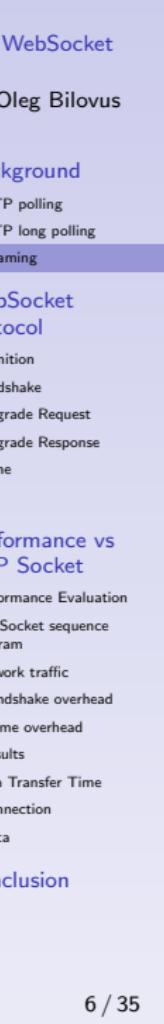
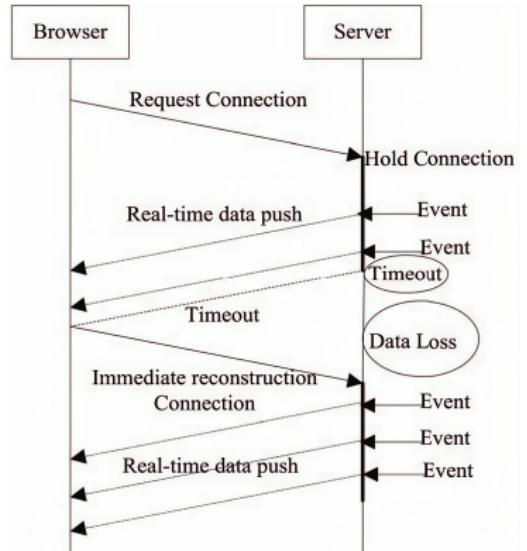
2024-06-03
WebSocket
Background
HTTP long polling
HTTP long polling



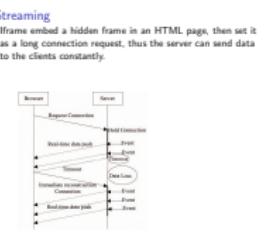
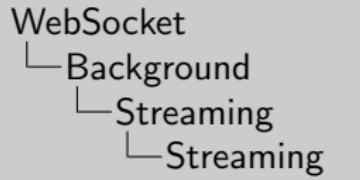
Can hold the connection up to a certain time, after that a timeout is exceeded and need a new connection. No bidirectional because the client may only send data the first time, but then it will only receive until a timeout and another request is made. In the normal polling we could have bidirectional because the interval was shorter.

Streaming

Iframe embed a hidden frame in an HTML page, then set it as a long connection request, thus the server can send data to the clients constantly.



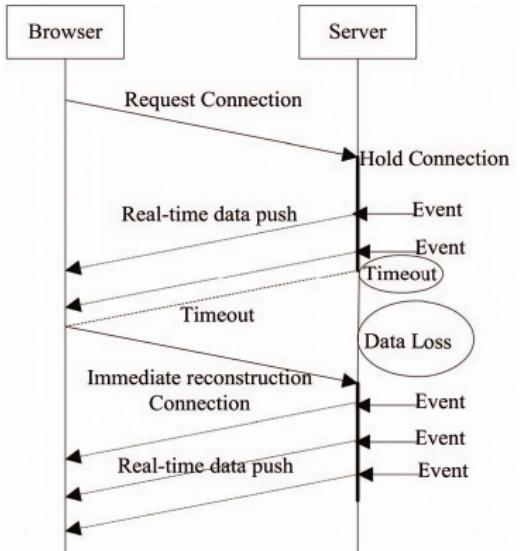
2024-06-03



iframe is a html page inside another. Because the server need to keep the connections alive.

Streaming

Iframe embed a hidden frame in an HTML page, then set it as a long connection request, thus the server can send data to the clients constantly.



- ▶ It can send multiple events from a single request.

WebSocket
Oleg Bilovus

Background
HTTP polling
HTTP long polling
Streaming

Websocket protocol
Definition
Handshake
Upgrade Request
Upgrade Response
Frame
API

Performance vs TCP Socket
Performance Evaluation
WebSocket sequence diagram
Network traffic
Handshake overhead
Frame overhead
Results
Data Transfer Time
Connection
Data

Conclusion

2024-06-03

WebSocket
Background
Streaming
Streaming

iframe is a html page inside another. Because the server need to keep the connections alive.

Streaming
Iframes embed a hidden frame in an HTML page, then set it as a long connection request, thus the server can send data to the clients constantly.

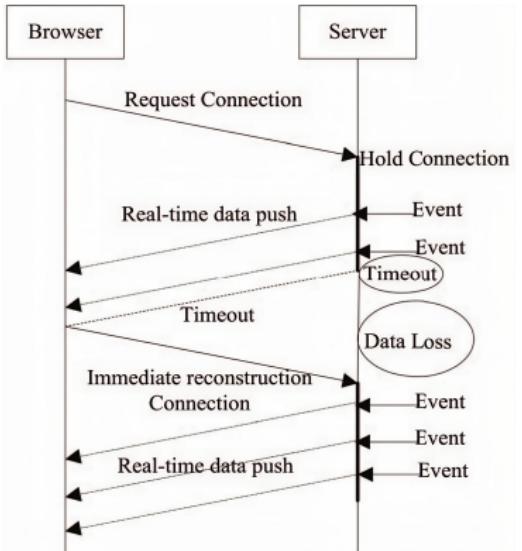
▶ It can send multiple events from a single request.

The diagram illustrates the communication flow between a Browser and a Server via an iframe:

- The Browser initiates a "Request Connection" to the Server.
- The Server responds with a "Hold Connection".
- The Server performs a "Real-time data push" to the Browser.
- The Browser sends an "Event" to the Server.
- The Server responds with another "Event".
- A "Timeout" occurs on the Server side.
- The Server handles "Data Loss".
- The Server performs an "Immediate reconstruction" to establish a new connection.
- The Server sends a "Connection" event to the Browser.
- The Browser sends an "Event" to the Server.
- The Server responds with another "Event".
- The Server performs a "Real-time data push" to the Browser.

Streaming

Iframe embed a hidden frame in an HTML page, then set it as a long connection request, thus the server can send data to the clients constantly.



- ▶ It can send multiple events from a single request.
- ▶ But, it increases the burden on the server, causing the server performance degradation, or even collapse.

WebSocket
Oleg Bilovus

Background
HTTP polling
HTTP long polling
Streaming

Websocket protocol
Definition
Handshake
Upgrade Request
Upgrade Response
Frame
API

Performance vs TCP Socket
Performance Evaluation
WebSocket sequence diagram
Network traffic
Handshake overhead
Frame overhead
Results
Data Transfer Time
Connection
Data

Conclusion

2024-06-03

WebSocket
Background
Streaming
Streaming

iframe is a html page inside another. Because the server need to keep the connections alive.

Streaming
Iframes embed a hidden frame in an HTML page, then set it as a long connection request, thus the server can send data to the clients constantly.

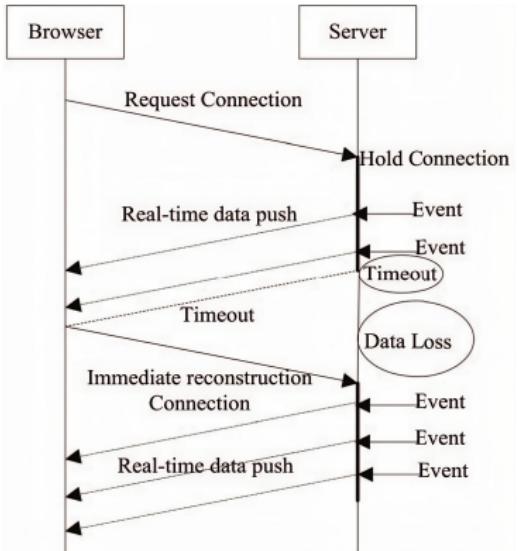
- ▶ It can send multiple events from a single request.
- ▶ But, it increases the burden on the server, causing the server performance degradation, or even collapse.

The diagram illustrates the structure of an iframe and its interaction with a server:

- A **Browser** contains an **Iframe**, which in turn contains a **Server**.
- The **Browser** initiates a **Request Connection** to the **Server** within the Iframe.
- The **Server** responds with a **Hold Connection**.
- The **Server** then performs a **Real-time data push** to the **Browser**.
- The **Browser** sends an **Event** back to the **Server**.
- This cycle repeats with another **Real-time data push** and an **Event** from the **Browser**.
- After the second event, a **Timeout** occurs at the **Server** side.
- The **Server** handles **Data Loss** during this timeout period.
- Upon timeout, the **Server** initiates **Immediate reconstruction** and sends a **Connection** back to the **Browser**.
- The **Browser** responds with another **Event**.
- The final step shows a **Real-time data push** from the **Server** to the **Browser**.

Streaming

Iframe embed a hidden frame in an HTML page, then set it as a long connection request, thus the server can send data to the clients constantly.



- ▶ It can send multiple events from a single request.
- ▶ But, it increases the burden on the server, causing the server performance degradation, or even collapse.
- ▶ No bidirectional communication.

WebSocket
Oleg Bilovus

Background
HTTP polling
HTTP long polling
Streaming

WebSocket protocol
Definition
Handshake
Upgrade Request
Upgrade Response
Frame
API

Performance vs TCP Socket
Performance Evaluation
WebSocket sequence diagram
Network traffic
Handshake overhead
Frame overhead
Results
Data Transfer Time
Connection
Data

Conclusion

2024-06-03

WebSocket
Background
Streaming
Streaming

Streaming
Iframes embed a hidden frame in an HTML page, then set it as a long connection request, thus the server can send data to the clients constantly.

- ▶ It can send multiple events from a single request.
- ▶ But, it increases the burden on the server, causing the server performance degradation, or even collapse.
- ▶ No bidirectional communication.

The diagram illustrates the communication flow between a Browser and a Server using an iframe:

- The process starts with a **Request Connection** from the Browser to the Server.
- The Server responds with a **Hold Connection**.
- The Server then performs a **Real-time data push** to the Browser.
- The Browser sends an **Event** back to the Server.
- This cycle repeats with another **Real-time data push** and an **Event** from the Browser.
- After two such cycles, a **Timeout** occurs on the Server side.
- The Server handles **Data Loss**, which leads to **Immediate reconstruction**.
- A new **Connection** is established, and the process resumes with another **Real-time data push** and an **Event** from the Browser.

Outline

Background

HTTP polling

HTTP long polling

Streaming

WebSocket protocol

Definition

Handshake

Frame

API

Performance vs TCP Socket

Performance Evaluation

WebSocket sequence diagram

Network traffic

Data Transfer Time

Conclusion

WebSocket	
Oleg Bilovus	
Background	
HTTP polling	
HTTP long polling	
Streaming	
WebSocket protocol	
Definition	
Handshake	
Upgrade Request	
Upgrade Response	
Frame	
API	
Performance vs TCP Socket	
Performance Evaluation	
WebSocket sequence diagram	
Network traffic	
Handshake overhead	
Frame overhead	
Results	
Data Transfer Time	
Connection	
Data	
Conclusion	

RFC 6455

Keywords

- ▶ The WebSocket Protocol enables two-way communication between a client running untrusted code in a controlled environment to a remote host that has opted-in to communications from that code.

WebSocket
Oleg Bilovus

Background
HTTP polling
HTTP long polling
Streaming

WebSocket protocol
Definition

Handshake
Upgrade Request
Upgrade Response
Frame
API

Performance vs TCP Socket
Performance Evaluation
WebSocket sequence diagram
Network traffic
Handshake overhead
Frame overhead
Results
Data Transfer Time
Connection
Data

Conclusion

2024-06-03

WebSocket

- WebSocket protocol
 - Definition
 - RFC 6455

RFC 6455
Keywords

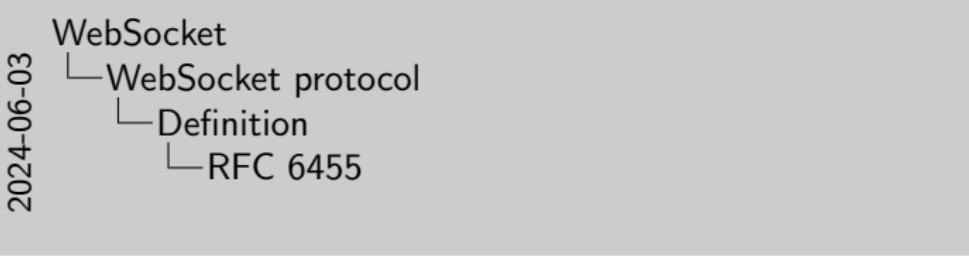
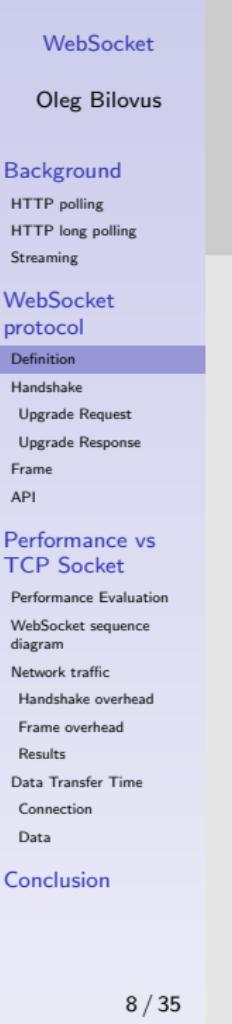
▶ The WebSocket Protocol enables two-way communication between a client running untrusted code in a controlled environment to a remote host that has opted-in to communications from that code.

opted-in is important because with polling any HTTP server would accept it, but here additional steps are needed. Handshake means client and server have to agree that they can both use the protocol and the server has to prove it. Message framing because we do not want to send every time the headers. TCP means it is reliable, no messages will be lost.

RFC 6455

Keywords

- ▶ The WebSocket Protocol enables **two-way communication** between a client running untrusted code in a controlled environment to a remote host that has opted-in to communications from that code.



RFC 6455
Keywords

▶ The WebSocket Protocol enables **two-way communication** between a client running untrusted code in a controlled environment to a remote host that has opted-in to communications from that code.

opted-in is important because with polling any HTTP server would accept it, but here additional steps are needed. Handshake means client and server have to agree that they can both use the protocol and the server has to prove it. Message framing because we do not want to send every time the headers. TCP means it is reliable, no messages will be lost.

RFC 6455

Keywords

- ▶ The WebSocket Protocol enables **two-way communication** between a **client** running untrusted code in a controlled environment to a remote host that has opted-in to communications from that code.

WebSocket
Oleg Bilovus

Background
HTTP polling
HTTP long polling
Streaming

WebSocket protocol
Definition

Handshake
Upgrade Request
Upgrade Response
Frame
API

Performance vs TCP Socket
Performance Evaluation
WebSocket sequence diagram
Network traffic
Handshake overhead
Frame overhead
Results
Data Transfer Time
Connection
Data

Conclusion

2024-06-03

WebSocket

- WebSocket protocol
 - Definition
 - RFC 6455

RFC 6455
Keywords

▶ The WebSocket Protocol enables **two-way communication** between a **client** running untrusted code in a controlled environment to a remote host that has opted-in to communications from that code.

opted-in is important because with polling any HTTP server would accept it, but here additional steps are needed. Handshake means client and server have to agree that they can both use the protocol and the server has to prove it. Message framing because we do not want to send every time the headers. TCP means it is reliable, no messages will be lost.

RFC 6455

Keywords

- ▶ The WebSocket Protocol enables **two-way communication** between a **client** running untrusted code in a controlled environment to a **remote host** that has opted-in to communications from that code.

WebSocket
Oleg Bilovus

Background
HTTP polling
HTTP long polling
Streaming

WebSocket protocol
Definition

Handshake
Upgrade Request
Upgrade Response
Frame
API

Performance vs TCP Socket
Performance Evaluation
WebSocket sequence diagram
Network traffic
Handshake overhead
Frame overhead
Results
Data Transfer Time
Connection
Data

Conclusion

2024-06-03

WebSocket

- WebSocket protocol
 - Definition
 - RFC 6455

RFC 6455
Keywords

▶ The WebSocket Protocol enables **two-way communication** between a **client** running untrusted code in a controlled environment to a **remote host** that has opted-in to communications from that code.

opted-in is important because with polling any HTTP server would accept it, but here additional steps are needed. Handshake means client and server have to agree that they can both use the protocol and the server has to prove it. Message framing because we do not want to send every time the headers. TCP means it is reliable, no messages will be lost.

RFC 6455

Keywords

- ▶ The WebSocket Protocol enables **two-way communication** between a **client** running untrusted code in a controlled environment to a **remote host** that has **opted-in** to communications from that code.

WebSocket
Oleg Bilovus

Background
HTTP polling
HTTP long polling
Streaming

WebSocket protocol
Definition

Handshake
Upgrade Request
Upgrade Response
Frame
API

Performance vs TCP Socket
Performance Evaluation
WebSocket sequence diagram
Network traffic
Handshake overhead
Frame overhead
Results
Data Transfer Time
Connection
Data

Conclusion

WebSocket
2024-06-03

- WebSocket
- └ WebSocket protocol
- └ Definition
- └ RFC 6455

RFC 6455
Keywords

▶ The WebSocket Protocol enables **two-way communication** between a **client** running untrusted code in a controlled environment to a **remote host** that has **opted-in** to communications from that code.

opted-in is important because with polling any HTTP server would accept it, but here additional steps are needed. Handshake means client and server have to agree that they can both use the protocol and the server has to prove it. Message framing because we do not want to send every time the headers. TCP means it is reliable, no messages will be lost.

RFC 6455

Keywords

- ▶ The WebSocket Protocol enables **two-way communication** between a **client** running untrusted code in a controlled environment to a **remote host** that has **opted-in** to communications from that code.
- ▶ The protocol consists of an opening handshake followed by basic message framing, layered over TCP.

WebSocket
Oleg Bilovus

Background
HTTP polling
HTTP long polling
Streaming

WebSocket protocol
Definition

Handshake
Upgrade Request
Upgrade Response
Frame
API

Performance vs TCP Socket
Performance Evaluation
WebSocket sequence diagram
Network traffic
Handshake overhead
Frame overhead
Results
Data Transfer Time
Connection
Data

Conclusion

2024-06-03

WebSocket

- WebSocket protocol
 - Definition
 - RFC 6455

RFC 6455
Keywords

▶ The WebSocket Protocol enables **two-way communication** between a **client** running untrusted code in a controlled environment to a **remote host** that has **opted-in** to communications from that code.

▶ The protocol consists of an opening handshake followed by basic message framing, layered over TCP.

opted-in is important because with polling any HTTP server would accept it, but here additional steps are needed. Handshake means client and server have to agree that they can both use the protocol and the server has to prove it. Message framing because we do not want to send every time the headers. TCP means it is reliable, no messages will be lost.

RFC 6455

Keywords

- ▶ The WebSocket Protocol enables **two-way communication** between a **client** running untrusted code in a controlled environment to a **remote host** that has **opted-in** to communications from that code.
- ▶ The protocol consists of an opening **handshake** followed by basic message framing, layered over TCP.

WebSocket
Oleg Bilovus

Background
HTTP polling
HTTP long polling
Streaming

WebSocket protocol
Definition

Handshake
Upgrade Request
Upgrade Response
Frame
API

Performance vs TCP Socket
Performance Evaluation
WebSocket sequence diagram
Network traffic
Handshake overhead
Frame overhead
Results
Data Transfer Time
Connection
Data

Conclusion

2024-06-03

WebSocket

- WebSocket protocol
 - Definition
 - RFC 6455

RFC 6455
Keywords

▶ The WebSocket Protocol enables **two-way communication** between a **client** running untrusted code in a controlled environment to a **remote host** that has **opted-in** to communications from that code.

▶ The protocol consists of an opening **handshake** followed by basic message framing, layered over TCP.

opted-in is important because with polling any HTTP server would accept it, but here additional steps are needed. Handshake means client and server have to agree that they can both use the protocol and the server has to prove it. Message framing because we do not want to send every time the headers. TCP means it is reliable, no messages will be lost.

RFC 6455

Keywords

- ▶ The WebSocket Protocol enables **two-way communication** between a **client** running untrusted code in a controlled environment to a **remote host** that has **opted-in** to communications from that code.
- ▶ The protocol consists of an opening **handshake** followed by basic **message framing**, layered over TCP.

WebSocket
Oleg Bilovus

Background
HTTP polling
HTTP long polling
Streaming

WebSocket protocol
Definition

Handshake
Upgrade Request
Upgrade Response
Frame
API

Performance vs TCP Socket
Performance Evaluation
WebSocket sequence diagram
Network traffic
Handshake overhead
Frame overhead
Results
Data Transfer Time
Connection
Data

Conclusion

2024-06-03

WebSocket

- WebSocket protocol
 - Definition
 - RFC 6455

RFC 6455
Keywords

► The WebSocket Protocol enables **two-way communication** between a **client** running untrusted code in a controlled environment to a **remote host** that has **opted-in** to communications from that code.

► The protocol consists of an opening **handshake** followed by basic **message framing**, layered over TCP.

opted-in is important because with polling any HTTP server would accept it, but here additional steps are needed. Handshake means client and server have to agree that they can both use the protocol and the server has to prove it. Message framing because we do not want to send every time the headers. TCP means it is reliable, no messages will be lost.

RFC 6455

Keywords

- ▶ The WebSocket Protocol enables **two-way communication** between a **client** running untrusted code in a controlled environment to a **remote host** that has **opted-in** to communications from that code.
- ▶ The protocol consists of an opening **handshake** followed by basic **message framing**, layered over **TCP**.

WebSocket
Oleg Bilovus

Background
HTTP polling
HTTP long polling
Streaming

WebSocket protocol
Definition

Handshake
Upgrade Request
Upgrade Response
Frame
API

Performance vs TCP Socket
Performance Evaluation
WebSocket sequence diagram
Network traffic
Handshake overhead
Frame overhead
Results
Data Transfer Time
Connection
Data

Conclusion

WebSocket
2024-06-03

- WebSocket protocol
 - Definition
 - RFC 6455

RFC 6455
Keywords

▶ The WebSocket Protocol enables **two-way communication** between a **client** running untrusted code in a controlled environment to a **remote host** that has **opted-in** to communications from that code.

▶ The protocol consists of an opening **handshake** followed by basic **message framing**, layered over **TCP**.

opted-in is important because with polling any HTTP server would accept it, but here additional steps are needed. Handshake means client and server have to agree that they can both use the protocol and the server has to prove it. Message framing because we do not want to send every time the headers. TCP means it is reliable, no messages will be lost.

RFC 6455

Keywords

- ▶ The WebSocket Protocol enables **two-way communication** between a **client** running untrusted code in a controlled environment to a **remote host** that has **opted-in** to communications from that code.
- ▶ The protocol consists of an opening **handshake** followed by basic **message framing**, layered over **TCP**.
- ▶ The goal of this technology is to provide a mechanism for browser-based applications that need two-way communication with servers.

WebSocket
Oleg Bilovus

Background
HTTP polling
HTTP long polling
Streaming

WebSocket protocol
Definition

Definition
Handshake
Upgrade Request
Upgrade Response
Frame
API

Performance vs TCP Socket
Performance Evaluation
WebSocket sequence diagram
Network traffic
Handshake overhead
Frame overhead
Results
Data Transfer Time
Connection
Data

Conclusion

2024-06-03

WebSocket
└ WebSocket protocol
 └ Definition
 └ RFC 6455

RFC 6455
Keywords

▶ The WebSocket Protocol enables **two-way communication** between a **client** running untrusted code in a controlled environment to a **remote host** that has **opted-in** to communications from that code.
▶ The protocol consists of an opening **handshake** followed by basic **message framing**, layered over **TCP**.
▶ The goal of this technology is to provide a mechanism for browser-based applications that need two-way communication with servers.

opted-in is important because with polling any HTTP server would accept it, but here additional steps are needed. Handshake means client and server have to agree that they can both use the protocol and the server has to prove it. Message framing because we do not want to send every time the headers. TCP means it is reliable, no messages will be lost.

RFC 6455

Keywords

- ▶ The WebSocket Protocol enables **two-way communication** between a **client** running untrusted code in a controlled environment to a **remote host** that has **opted-in** to communications from that code.
- ▶ The protocol consists of an opening **handshake** followed by basic **message framing**, layered over **TCP**.
- ▶ The goal of this technology is to provide a mechanism for **browser-based** applications that need two-way communication with servers.

WebSocket
Oleg Bilovus

Background
HTTP polling
HTTP long polling
Streaming

WebSocket protocol
Definition

Definition
Handshake
Upgrade Request
Upgrade Response
Frame
API

Performance vs TCP Socket
Performance Evaluation
WebSocket sequence diagram
Network traffic
Handshake overhead
Frame overhead
Results
Data Transfer Time
Connection
Data

Conclusion

2024-06-03

WebSocket

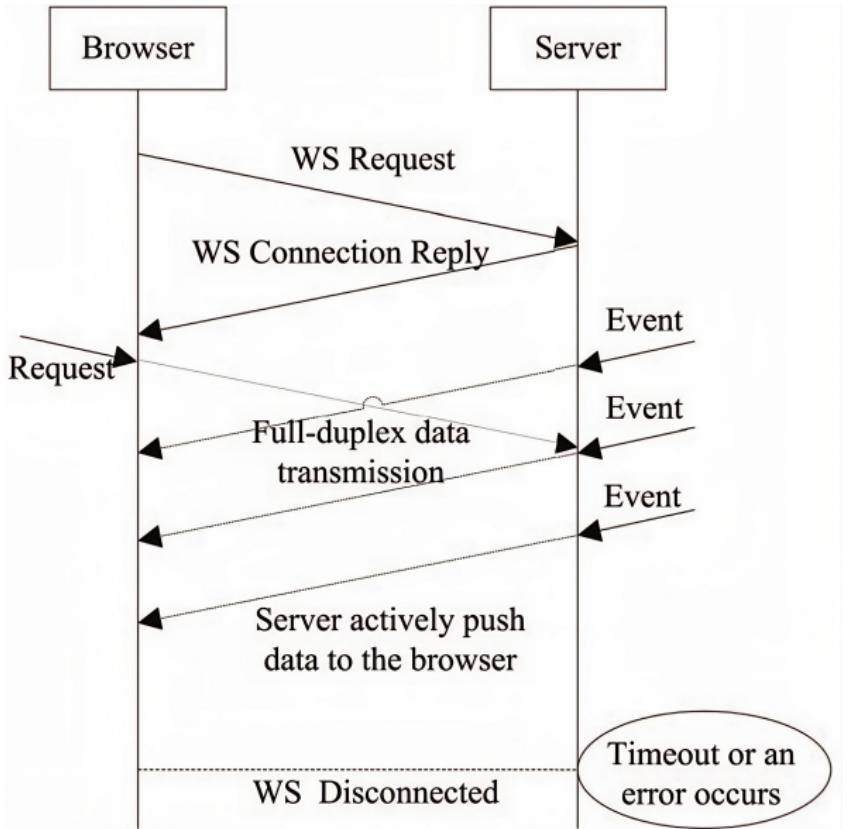
- WebSocket protocol
 - Definition
 - RFC 6455

RFC 6455
Keywords

- ▶ The WebSocket Protocol enables **two-way communication** between a **client** running untrusted code in a controlled environment to a **remote host** that has **opted-in** to communications from that code.
- ▶ The protocol consists of an opening **handshake** followed by basic **message framing**, layered over **TCP**.
- ▶ The goal of this technology is to provide a mechanism for **browser-based** applications that need two-way communication with servers.

opted-in is important because with polling any HTTP server would accept it, but here additional steps are needed. Handshake means client and server have to agree that they can both use the protocol and the server has to prove it. Message framing because we do not want to send every time the headers. TCP means it is reliable, no messages will be lost.

WebSocket



WebSocket
Oleg Bilovus

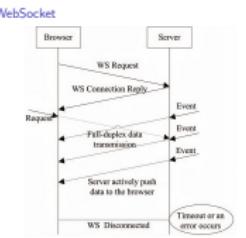
Background
HTTP polling
HTTP long polling
Streaming

WebSocket protocol
Definition
Handshake
Upgrade Request
Upgrade Response
Frame
API

Performance vs TCP Socket
Performance Evaluation
WebSocket sequence diagram
Network traffic
Handshake overhead
Frame overhead
Results
Data Transfer Time
Connection
Data

Conclusion

WebSocket
2024-06-03
└ WebSocket protocol
 └ Definition
 └ WebSocket



There is the initial handshake, after that, client and server can send and receive data at any moment without further interaction. There is no timeout. If it disconnects, it is because of an error and to establish the connection, the handshake has to be done again.

Handshake

- ▶ For WebSocket-based communication, a **WebSocket session** should be established first.

WebSocket
Oleg Bilovus

Background
HTTP polling
HTTP long polling
Streaming

WebSocket protocol
Definition
Handshake

- Upgrade Request
- Upgrade Response
- Frame
- API

Performance vs TCP Socket
Performance Evaluation
WebSocket sequence diagram
Network traffic
Handshake overhead
Frame overhead
Results
Data Transfer Time
Connection
Data

Conclusion

2024-06-03

WebSocket

- WebSocket protocol
 - Handshake
 - Handshake

▶ For WebSocket-based communication, a **WebSocket session** should be established first.

Handshake

- ▶ For WebSocket-based communication, a **WebSocket session** should be established first.
- ▶ To establish a session, client sends a WebSocket **Upgrade Request** to the server, upon which server responds with a WebSocket **Upgrade Response**.

WebSocket
Oleg Bilovus

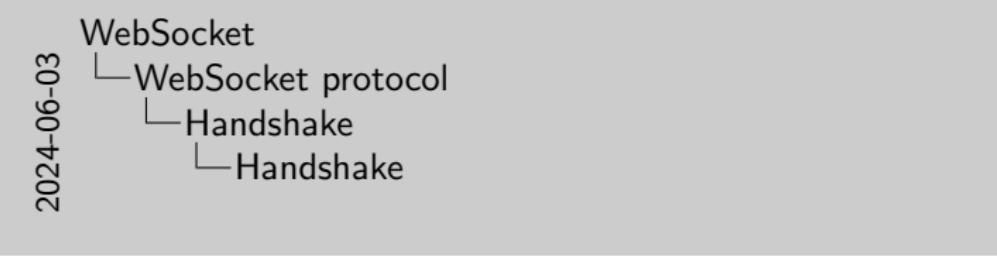
Background
HTTP polling
HTTP long polling
Streaming

WebSocket protocol
Definition
Handshake

Upgrade Request
Upgrade Response
Frame
API

Performance vs TCP Socket
Performance Evaluation
WebSocket sequence diagram
Network traffic
Handshake overhead
Frame overhead
Results
Data Transfer Time
Connection
Data

Conclusion



With the Upgrade Response, the server proves that it can communicate with WebSockets.

Handshake

- ▶ For WebSocket-based communication, a **WebSocket session** should be established first.
- ▶ To establish a session, client sends a WebSocket **Upgrade Request** to the server, upon which server responds with a WebSocket **Upgrade Response**.

Handshake

- ▶ For WebSocket-based communication, a **WebSocket session** should be established first.
- ▶ To establish a session, client sends a WebSocket **Upgrade Request** to the server, upon which server responds with a WebSocket **Upgrade Response**.
- ▶ From this point forward, the client and server can **send data back and forth in asynchronous full-duplex mode**.

WebSocket
Oleg Bilovus

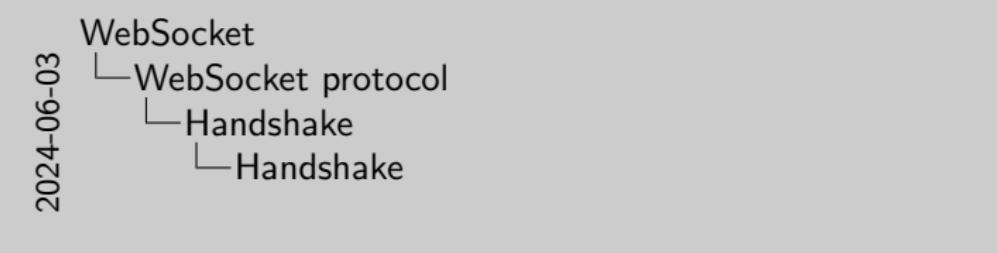
Background
HTTP polling
HTTP long polling
Streaming

WebSocket protocol
Definition
Handshake

- Upgrade Request
- Upgrade Response
- Frame
- API

Performance vs TCP Socket
Performance Evaluation
WebSocket sequence diagram
Network traffic
Handshake overhead
Frame overhead
Results
Data Transfer Time
Connection
Data

Conclusion



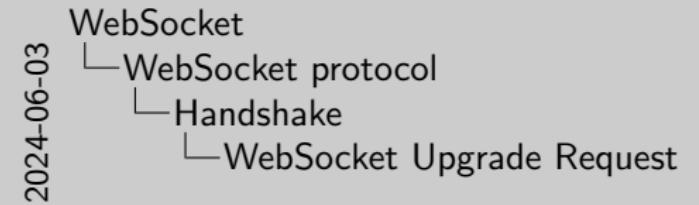
With the Upgrade Response, the server proves that it can communicate with WebSockets.

- Handshake
- ▶ For WebSocket-based communication, a **WebSocket session** should be established first.
 - ▶ To establish a session, client sends a WebSocket **Upgrade Request** to the server, upon which server responds with a WebSocket **Upgrade Response**.
 - ▶ From this point forward, the client and server can **send data back and forth in asynchronous full-duplex mode**.

WebSocket Upgrade Request

```
GET /chat HTTP/1.1
Host: server.example.com
Upgrade: WebSocket
Connection: Upgrade
Sec-WebSocket-Key:
dGh1IHNhbXBsZSBub25jZQ==
Origin: http://example.com
Sec-WebSocket-Protocol:
chat, superchat
Sec-WebSocket-Version: 13
```

WebSocket
Oleg Bilovus
Background
HTTP polling
HTTP long polling
Streaming
WebSocket protocol
Definition
Handshake
Upgrade Request
Upgrade Response
Frame
API
Performance vs TCP Socket
Performance Evaluation
WebSocket sequence diagram
Network traffic
Handshake overhead
Frame overhead
Results
Data Transfer Time
Connection
Data
Conclusion



Different URI can be used to identify different endpoints. A URI can be regular HTTP, another can be WebSocket.

```
GET /chat HTTP/1.1
Host: server.example.com
Upgrade: WebSocket
Connection: Upgrade
Sec-WebSocket-Key:
dGh1IHNhbXBsZSBub25jZQ==
Origin: http://example.com
Sec-WebSocket-Protocol:
chat, superchat
Sec-WebSocket-Version: 13
```

WebSocket Upgrade Request

► HTTP GET request.

GET /chat HTTP/1.1

Host: server.example.com

Upgrade: WebSocket

Connection: Upgrade

Sec-WebSocket-Key:

dGh1IHNhbXBsZSBub25jZQ==

Origin: http://example.com

Sec-WebSocket-Protocol:

chat, superchat

Sec-WebSocket-Version: 13

WebSocket	Oleg Bilovus
Background	
HTTP polling	
HTTP long polling	
Streaming	
WebSocket protocol	
Definition	
Handshake	
Upgrade Request	
Upgrade Response	
Frame	
API	
Performance vs TCP Socket	
Performance Evaluation	
WebSocket sequence diagram	
Network traffic	
Handshake overhead	
Frame overhead	
Results	
Data Transfer Time	
Connection	
Data	
Conclusion	

WebSocket
└ WebSocket protocol
 └ Handshake
 └ WebSocket Upgrade Request

2024-06-03

WebSocket Upgrade Request
► HTTP GET request.
GET /chat HTTP/1.1
Host: server.example.com
Upgrade: WebSocket
Connection: Upgrade
Sec-WebSocket-Key: dGh1IHNhbXBsZSBub25jZQ==
Origin: http://example.com
Sec-WebSocket-Protocol: chat, superchat
Sec-WebSocket-Version: 13

WebSocket Upgrade Request

```
GET /chat HTTP/1.1
Host: server.example.com
Upgrade: WebSocket
Connection: Upgrade
Sec-WebSocket-Key:
dGh1IHNhbXBsZSBub25jZQ==
Origin: http://example.com
Sec-WebSocket-Protocol:
chat, superchat
Sec-WebSocket-Version: 13
```

- ▶ HTTP GET request.
- ▶ URI to identify endpoint.

WebSocket

Oleg Bilovus

Background

- HTTP polling
- HTTP long polling
- Streaming

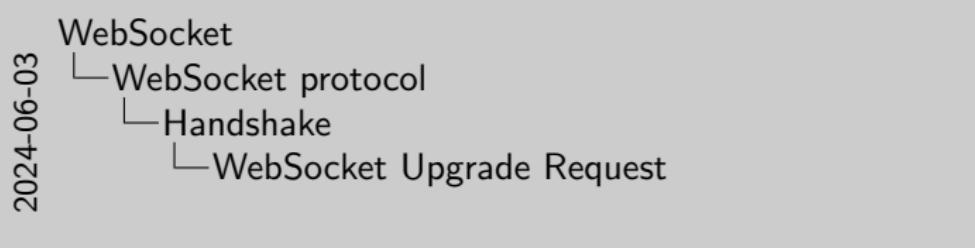
WebSocket protocol

- Definition
- Handshake
- Upgrade Request
- Upgrade Response
- Frame
- API

Performance vs TCP Socket

- Performance Evaluation
- WebSocket sequence diagram
- Network traffic
- Handshake overhead
- Frame overhead
- Results
- Data Transfer Time
- Connection
- Data

Conclusion



Different URI can be used to identify different endpoints. A URI can be regular HTTP, another can be WebSocket.

WebSocket Upgrade Request

▶ HTTP GET request.

▶ URI to identify endpoint.

```
GET /chat HTTP/1.1
Host: server.example.com
Upgrade: WebSocket
Connection: Upgrade
Sec-WebSocket-Key:
dGh1IHNhbXBsZSBub25jZQ==
Origin: http://example.com
Sec-WebSocket-Protocol:
chat, superchat
Sec-WebSocket-Version: 13
```

WebSocket Upgrade Request

```
GET /chat HTTP/1.1
Host: server.example.com
Upgrade: WebSocket
Connection: Upgrade
Sec-WebSocket-Key:
dGh1IHNhbXBsZSBub25jZQ==
Origin: http://example.com
Sec-WebSocket-Protocol:
chat, superchat
Sec-WebSocket-Version: 13
```

- ▶ HTTP GET request.
- ▶ URI to identify endpoint.
- ▶ Headers indicating the will to switch from regular HTTP to WebSocket.

WebSocket
Oleg Bilovus

Background
HTTP polling
HTTP long polling
Streaming

WebSocket protocol
Definition
Handshake
Upgrade Request
Upgrade Response
Frame
API

Performance vs TCP Socket
Performance Evaluation
WebSocket sequence diagram
Network traffic
Handshake overhead
Frame overhead
Results
Data Transfer Time
Connection
Data

Conclusion

WebSocket
2024-06-03
WebSocket protocol
Handshake
WebSocket Upgrade Request

Different URI can be used to identify different endpoints. A URI can be regular HTTP, another can be WebSocket.

WebSocket Upgrade Request

▶ HTTP GET request.
▶ URI to identify endpoint.
▶ Headers indicating the will to switch from regular HTTP to WebSocket.

GET /chat HTTP/1.1
Host: server.example.com
Upgrade: WebSocket
Connection: Upgrade
Sec-WebSocket-Key:
dGh1IHNhbXBsZSBub25jZQ==
Origin: http://example.com
Sec-WebSocket-Protocol:
chat, superchat
Sec-WebSocket-Version: 13

WebSocket Upgrade Request

```
GET /chat HTTP/1.1
Host: server.example.com
Upgrade: WebSocket
Connection: Upgrade
Sec-WebSocket-Key:
dGh1IHNhbXBsZSBub25jZQ==
Origin: http://example.com
Sec-WebSocket-Protocol:
chat, superchat
Sec-WebSocket-Version: 13
```

- ▶ HTTP GET request.
- ▶ URI to identify endpoint.
- ▶ Headers indicating the will to switch from regular HTTP to WebSocket.
- ▶ A key the server has to use to prove that it can use WebSockets.

WebSocket
Oleg Bilovus

Background
HTTP polling
HTTP long polling
Streaming

WebSocket protocol
Definition
Handshake
Upgrade Request
Upgrade Response
Frame
API

Performance vs TCP Socket
Performance Evaluation
WebSocket sequence diagram
Network traffic
Handshake overhead
Frame overhead
Results
Data Transfer Time
Connection
Data

Conclusion

2024-06-03
WebSocket
└ WebSocket protocol
 └ Handshake
 └ WebSocket Upgrade Request

Different URI can be used to identify different endpoints. A URI can be regular HTTP, another can be WebSocket.

WebSocket Upgrade Request

▶ HTTP GET request.
▶ URI to identify endpoint.
▶ Headers indicating the will to switch from regular HTTP to WebSocket.
▶ A key the server has to use to prove that it can use WebSockets.

```
GET /chat HTTP/1.1
Host: server.example.com
Upgrade: WebSocket
Connection: Upgrade
Sec-WebSocket-Key:
dGh1IHNhbXBsZSBub25jZQ==
Origin: http://example.com
Sec-WebSocket-Protocol:
chat, superchat
Sec-WebSocket-Version: 13
```

WebSocket Upgrade Request

```
GET /chat HTTP/1.1
Host: server.example.com
Upgrade: WebSocket
Connection: Upgrade
Sec-WebSocket-Key:
dGh1IHNhbXBsZSBub25jZQ==
Origin: http://example.com
Sec-WebSocket-Protocol:
chat, superchat
Sec-WebSocket-Version: 13
```

- ▶ HTTP GET request.
- ▶ URI to identify endpoint.
- ▶ Headers indicating the will to switch from regular HTTP to WebSocket.
- ▶ A key the server has to use to prove that it can use WebSockets.
- ▶ **WebSocket protocols.**

WebSocket
Oleg Bilovus

Background
HTTP polling
HTTP long polling
Streaming

WebSocket protocol
Definition
Handshake
Upgrade Request
Upgrade Response
Frame
API

Performance vs TCP Socket
Performance Evaluation
WebSocket sequence diagram
Network traffic
Handshake overhead
Frame overhead
Results
Data Transfer Time
Connection
Data

Conclusion

WebSocket
2024-06-03
└WebSocket protocol
 └Handshake
 └WebSocket Upgrade Request

WebSocket Upgrade Request

```
GET /chat HTTP/1.1
Host: server.example.com
Upgrade: WebSocket
Connection: Upgrade
Sec-WebSocket-Key:
dGh1IHNhbXBsZSBub25jZQ==
Origin: http://example.com
Sec-WebSocket-Protocol:
chat, superchat
Sec-WebSocket-Version: 13
```

- ▶ HTTP GET request.
- ▶ URI to identify endpoint.
- ▶ Headers indicating the will to switch from regular HTTP to WebSocket.
- ▶ A key the server has to use to prove that it can use WebSockets.
- ▶ **WebSocket protocols.**

WebSocket Upgrade Request

```
GET /chat HTTP/1.1
Host: server.example.com
Upgrade: WebSocket
Connection: Upgrade
Sec-WebSocket-Key:
dGh1IHNhbXBsZSBub25jZQ==
Origin: http://example.com
Sec-WebSocket-Protocol:
chat, superchat
Sec-WebSocket-Version: 13
```

- ▶ HTTP GET request.
- ▶ URI to identify endpoint.
- ▶ Headers indicating the will to switch from regular HTTP to WebSocket.
- ▶ A key the server has to use to prove that it can use WebSockets.
- ▶ WebSocket protocols.
- ▶ WebSocket version.

WebSocket
Oleg Bilovus
[Background](#)
HTTP polling
HTTP long polling
Streaming
[WebSocket protocol](#)
Definition
Handshake
Upgrade Request
Upgrade Response
Frame
API
[Performance vs TCP Socket](#)
Performance Evaluation
WebSocket sequence diagram
Network traffic
Handshake overhead
Frame overhead
Results
Data Transfer Time
Connection
Data
[Conclusion](#)

2024-06-03
WebSocket
└ WebSocket protocol
 └ Handshake
 └ WebSocket Upgrade Request

Different URI can be used to identify different endpoints. A URI can be regular HTTP, another can be WebSocket.

WebSocket Upgrade Request

```
GET /chat HTTP/1.1
Host: server.example.com
Upgrade: WebSocket
Connection: Upgrade
Sec-WebSocket-Key:
dGh1IHNhbXBsZSBub25jZQ==
Origin: http://example.com
Sec-WebSocket-Protocol:
chat, superchat
Sec-WebSocket-Version: 13
```

- ▶ HTTP GET request.
- ▶ URI to identify endpoint.
- ▶ Headers indicating the will to switch from regular HTTP to WebSocket.
- ▶ A key the server has to use to prove that it can use WebSockets.
- ▶ WebSocket protocols.
- ▶ WebSocket version.

WebSocket Upgrade Response

HTTP/1.1 101 Switching
protocols

Upgrade: WebSocket

Connection: Upgrade

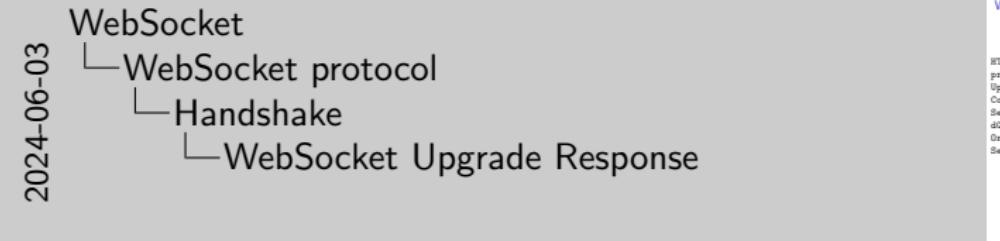
Sec-WebSocket-Accept:

dGh1IHNhbXBsZSBub25jZQ==

Origin: http://example.com

Sec-WebSocket-Protocol: chat

WebSocket	Oleg Bilovus
Background	
HTTP polling	
HTTP long polling	
Streaming	
WebSocket protocol	
Definition	
Handshake	
Upgrade Request	
Upgrade Response	
Frame	
API	
Performance vs TCP Socket	
Performance Evaluation	
WebSocket sequence diagram	
Network traffic	
Handshake overhead	
Frame overhead	
Results	
Data Transfer Time	
Connection	
Data	
Conclusion	



There is a specific algorithm to generate this Header from a key.

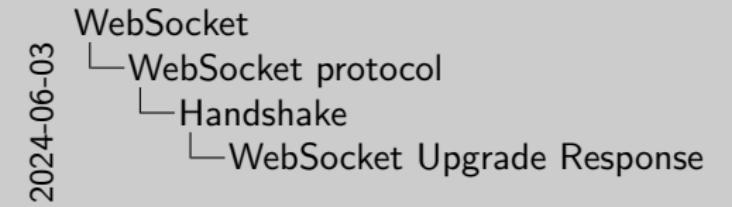
HTTP/1.1 101 Switching
protocols
Upgrade: WebSocket
Connection: Upgrade
Sec-WebSocket-Accept:
dGh1IHNhbXBsZSBub25jZQ==
Origin: http://example.com
Sec-WebSocket-Protocol: chat

WebSocket Upgrade Response

HTTP/1.1 101 Switching
protocols
Upgrade: WebSocket
Connection: Upgrade
Sec-WebSocket-Accept:
dGh1IHNhbXBsZSBub25jZQ==
Origin: http://example.com
Sec-WebSocket-Protocol: chat

► Server confirms it
supports WebSocket.

WebSocket
Oleg Bilovus
Background
HTTP polling
HTTP long polling
Streaming
WebSocket protocol
Definition
Handshake
Upgrade Request
Upgrade Response
Frame
API
Performance vs TCP Socket
Performance Evaluation
WebSocket sequence diagram
Network traffic
Handshake overhead
Frame overhead
Results
Data Transfer Time
Connection
Data
Conclusion



There is a specific algorithm to generate this Header from a key.

HTTP/1.1 101 Switching
protocols
Upgrade: WebSocket
Connection: Upgrade
Sec-WebSocket-Accept:
dGh1IHNhbXBsZSBub25jZQ==
Origin: http://example.com
Sec-WebSocket-Protocol: chat

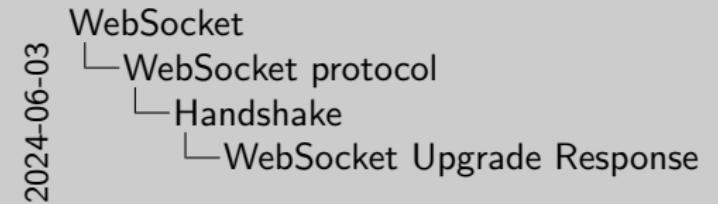
► Server confirms it
supports WebSocket.

WebSocket Upgrade Response

HTTP/1.1 101 Switching
protocols
Upgrade: WebSocket
Connection: Upgrade
Sec-WebSocket-Accept:
dGh1IHNhbXBsZSBub25jZQ==
Origin: http://example.com
Sec-WebSocket-Protocol: chat

- ▶ Server confirms it supports WebSocket.
- ▶ Server proves that it can use WebSocket.
Client checks it.

WebSocket
Oleg Bilovus
Background
HTTP polling
HTTP long polling
Streaming
WebSocket protocol
Definition
Handshake
Upgrade Request
Upgrade Response
Frame
API
Performance vs TCP Socket
Performance Evaluation
WebSocket sequence diagram
Network traffic
Handshake overhead
Frame overhead
Results
Data Transfer Time
Connection
Data
Conclusion



There is a specific algorithm to generate this Header from a key.

WebSocket Upgrade Response

HTTP/1.1 101 Switching
protocols
Upgrade: WebSocket
Connection: Upgrade
Sec-WebSocket-Accept:
dGh1IHNhbXBsZSBub25jZQ==
Origin: http://example.com
Sec-WebSocket-Protocol: chat

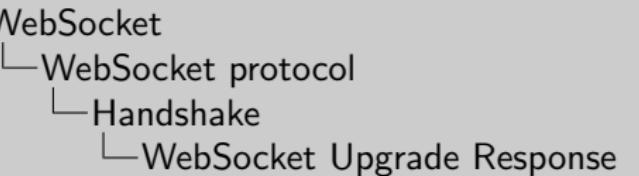
▶ Server confirms it supports WebSocket.
▶ Server proves that it can use WebSocket.
Client checks it.

WebSocket Upgrade Response

HTTP/1.1 101 Switching
protocols
Upgrade: WebSocket
Connection: Upgrade
Sec-WebSocket-Accept:
dGh1IHNhbXBsZSBub25jZQ==
Origin: http://example.com
Sec-WebSocket-Protocol: chat

- ▶ Server confirms it supports WebSocket.
- ▶ Server proves that it can use WebSocket. Client checks it.
- ▶ **Server tells which protocol it supports.**

WebSocket	2024-06-03
Oleg Bilovus	
Background	
HTTP polling	
HTTP long polling	
Streaming	
WebSocket protocol	
Definition	
Handshake	
Upgrade Request	
Upgrade Response	
Frame	
API	
Performance vs TCP Socket	
Performance Evaluation	
WebSocket sequence diagram	
Network traffic	
Handshake overhead	
Frame overhead	
Results	
Data Transfer Time	
Connection	
Data	
Conclusion	



WebSocket Upgrade Response

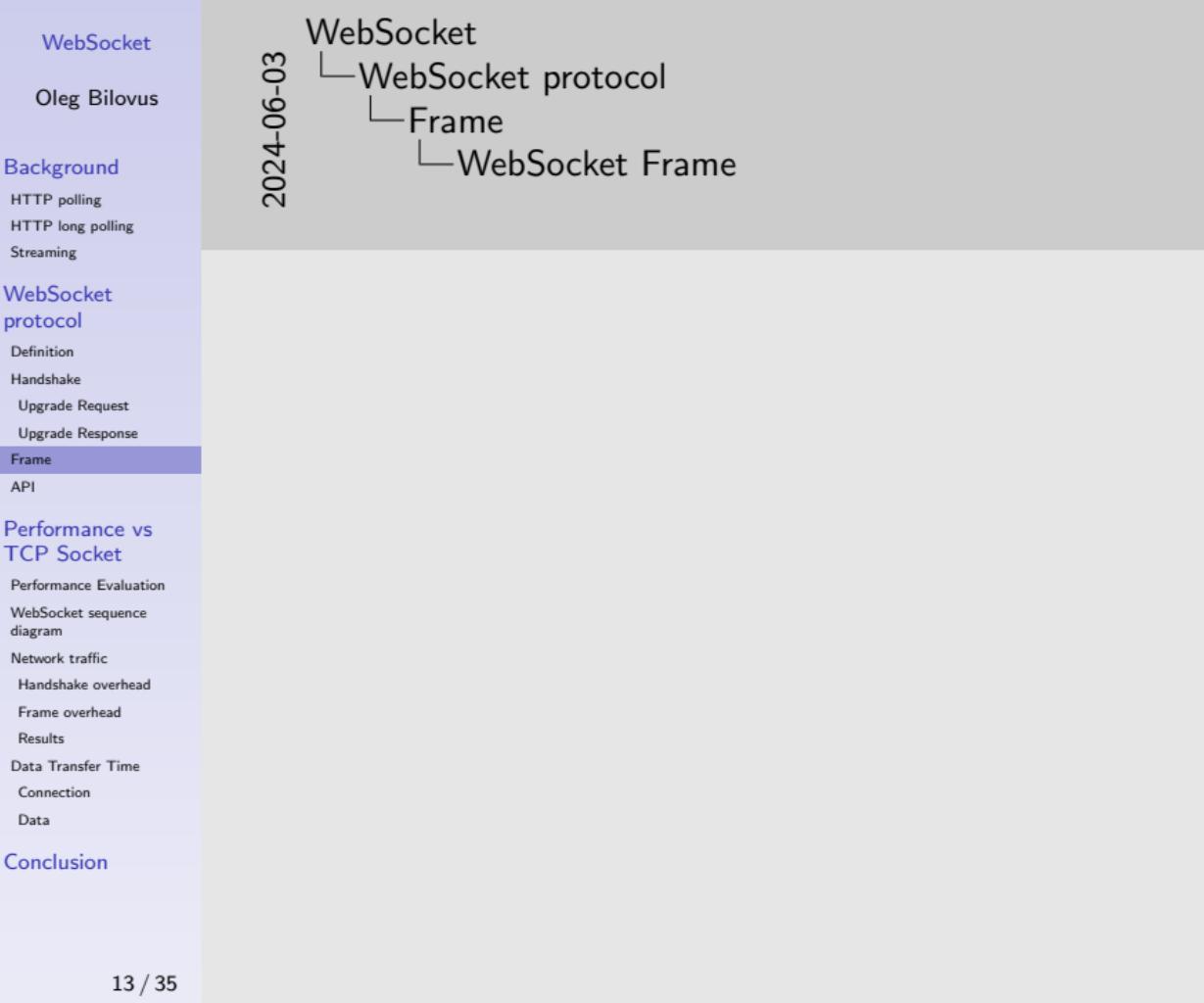
HTTP/1.1 101 Switching protocols
Upgrade: WebSocket
Connection: Upgrade
Sec-WebSocket-Accept: dGh1IHNhbXBsZSBub25jZQ==
Origin: http://example.com
Sec-WebSocket-Protocol: chat

- ▶ Server confirms it supports WebSocket.
- ▶ Server proves that it can use WebSocket. Client checks it.
- ▶ **Server tells which protocol it supports.**

There is a specific algorithm to generate this Header from a key.

WebSocket Frame

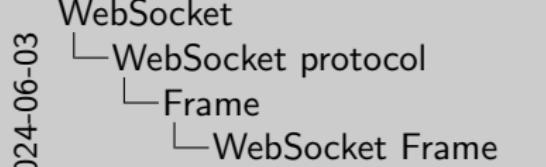
- ▶ After the handshake is successful, client and server can communicate in full-duplex by using frames.



WebSocket Frame

- ▶ After the handshake is successful, client and server can **communicate in full-duplex** by using frames.
- ▶ The added **overhead** to the payload data is **minimal** because it does not send all the HTTP headers for each frame.

WebSocket	Background
Oleg Bilovus	HTTP polling HTTP long polling Streaming
	WebSocket protocol
	Definition Handshake Upgrade Request Upgrade Response
	Frame
	API
	Performance vs TCP Socket
	Performance Evaluation WebSocket sequence diagram Network traffic Handshake overhead Frame overhead Results Data Transfer Time Connection Data
	Conclusion



- ▶ After the handshake is successful, client and server can **communicate in full-duplex** by using frames.
- ▶ The added **overhead** to the payload data is **minimal** because it does not send all the HTTP headers for each frame.

WebSocket Frame

- ▶ After the handshake is successful, client and server can **communicate in full-duplex** by using frames.
- ▶ The added **overhead** to the payload data is **minimal** because it does not send all the HTTP headers for each frame.
- ▶ Each frame adds **at least 2 bytes of overhead** to the payload data. Depending on the length of the payload data and the direction of the communication, the length of the overhead **may increase up to 14 bytes**.

WebSocket
Oleg Bilovus

Background
HTTP polling
HTTP long polling
Streaming

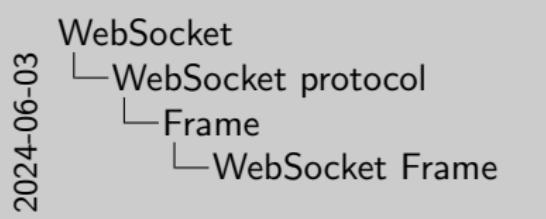
WebSocket protocol
Definition
Handshake
Upgrade Request
Upgrade Response

Frame

API

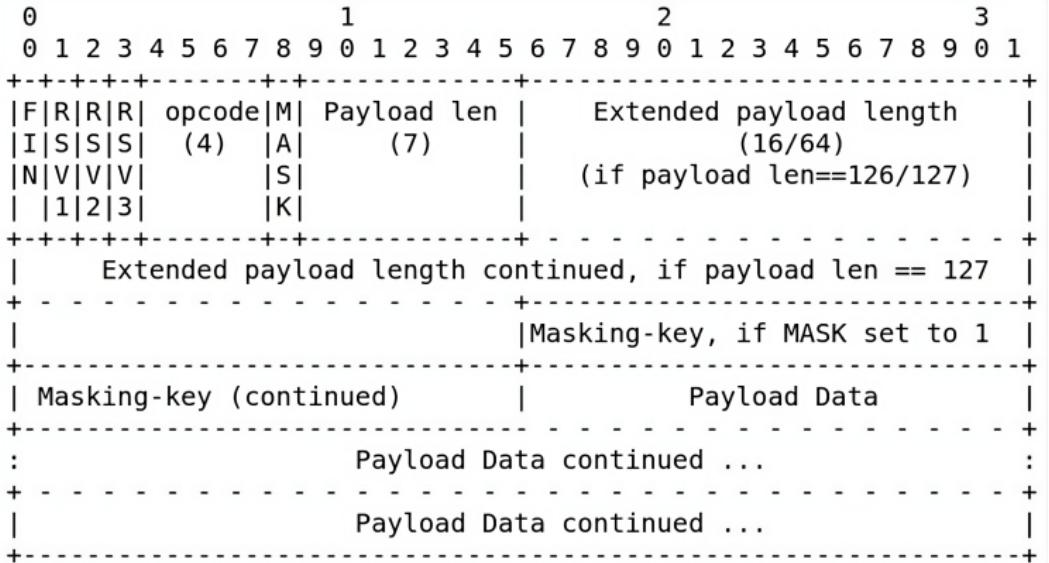
Performance vs TCP Socket
Performance Evaluation
WebSocket sequence diagram
Network traffic
Handshake overhead
Frame overhead
Results
Data Transfer Time
Connection
Data

Conclusion



- WebSocket Frame
- ▶ After the handshake is successful, client and server can **communicate in full-duplex** by using frames.
 - ▶ The added **overhead** to the payload data is **minimal** because it does not send all the HTTP headers for each frame.
 - ▶ Each frame adds **at least 2 bytes of overhead** to the payload data. Depending on the length of the payload data and the direction of the communication, the length of the overhead **may increase up to 14 bytes**.

WebSocket Frame Structure



WebSocket
Oleg Bilovus

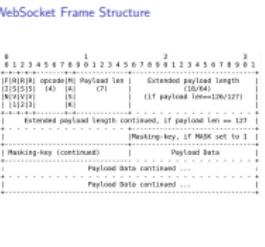
Background
HTTP polling
HTTP long polling
Streaming

WebSocket protocol
Definition
Handshake
Upgrade Request
Upgrade Response
Frame
API

Performance vs TCP Socket
Performance Evaluation
WebSocket sequence diagram
Network traffic
Handshake overhead
Frame overhead
Results
Data Transfer Time
Connection
Data

Conclusion

WebSocket
2024-06-03
└── WebSocket protocol
 └── Frame
 └── WebSocket Frame Structure



We will not go into the details because it is out of the scope of this presentation and, as mentioned earlier, the added overhead to the payload data is minimal.

Callback	Description
onopen	invoked when WebSocket session is established, signalizes that the protocol is ready to transfer payload data
onerror	invoked whenever an error occurs
onclose	invoked when one of the peers has terminated the session
onmessage	invoked when an incoming message from another peer has arrived

WebSocket API

The API is defined by its states of readiness, responses to a networking or messaging **event**.

Callback	Description
onopen	invoked when WebSocket session is established, signalizes that the protocol is ready to transfer payload data
onerror	invoked whenever an error occurs
onclose	invoked when one of the peers has terminated the session
onmessage	invoked when an incoming message from another peer has arrived

WebSocket
Oleg Bilovus

Background
HTTP polling
HTTP long polling
Streaming

WebSocket protocol
Definition
Handshake
Upgrade Request
Upgrade Response
Frame
API

Performance vs TCP Socket
Performance Evaluation
WebSocket sequence diagram
Network traffic
Handshake overhead
Frame overhead
Results
Data Transfer Time
Connection
Data

Conclusion

WebSocket
2024-06-03
WebSocket protocol
API
WebSocket API

Outline

Background

HTTP polling

HTTP long polling

Streaming

WebSocket protocol

Definition

Handshake

Frame

API

Performance vs TCP Socket

Performance Evaluation

WebSocket sequence diagram

Network traffic

Data Transfer Time

Conclusion

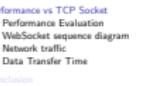


WebSocket

Performance vs TCP Socket

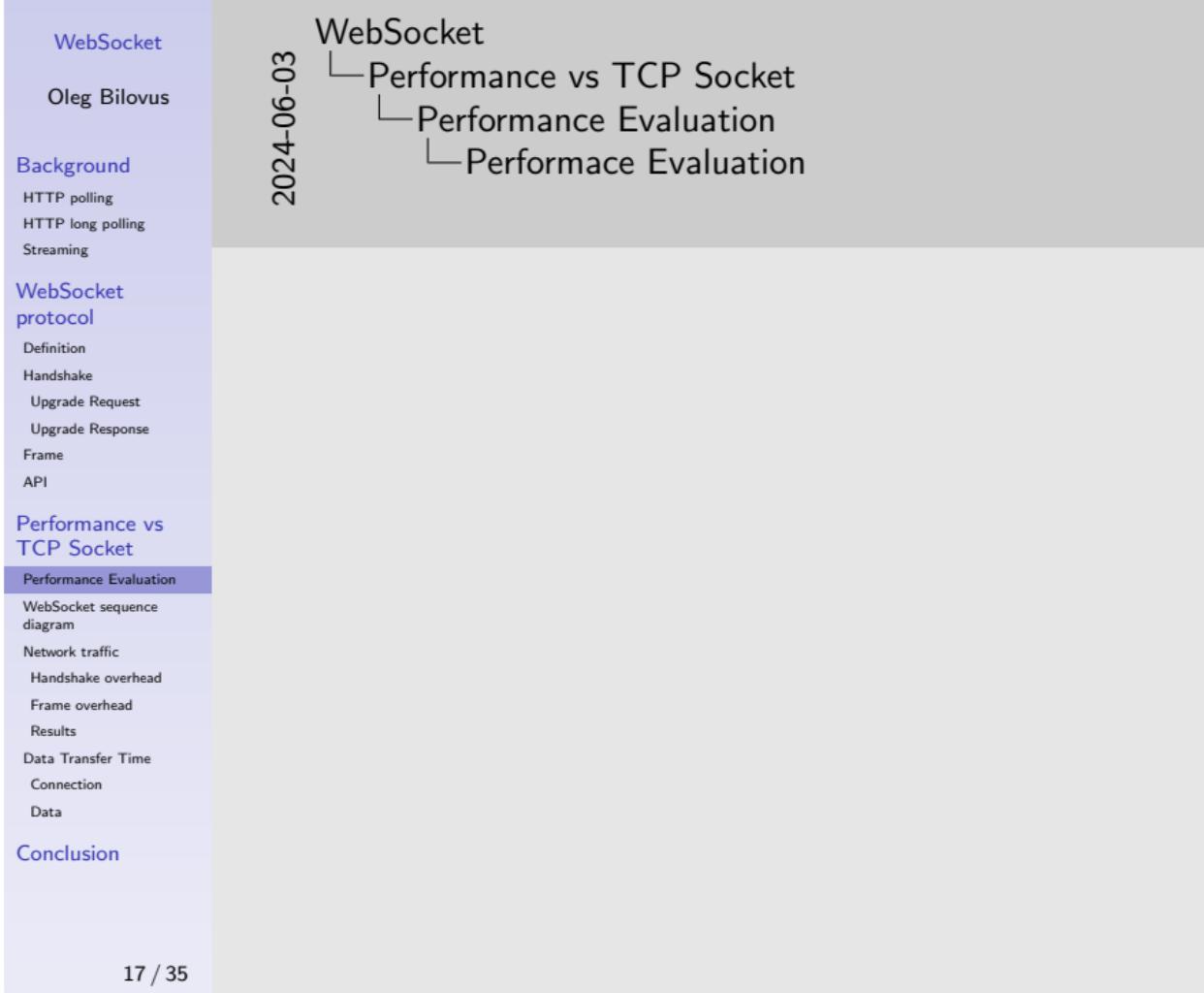
Outline

We will now try to evaluate the WebSocket performance compared to the raw TCP Socket.



Performance Evaluation

- ▶ Performance evaluation of the WebSocket and the TCP Socket protocol consists of:



Performance Evaluation

▶ Performance evaluation of the WebSocket and the TCP Socket protocol consists of:

Performance Evaluation

- ▶ Performance evaluation of the WebSocket and the TCP Socket protocol consists of:
 - ▶ Network traffic

WebSocket	
Oleg Bilovus	
Background	
HTTP polling	
HTTP long polling	
Streaming	
WebSocket protocol	
Definition	
Handshake	
Upgrade Request	
Upgrade Response	
Frame	
API	
Performance vs TCP Socket	
Performance Evaluation	
WebSocket sequence diagram	
Network traffic	
Handshake overhead	
Frame overhead	
Results	
Data Transfer Time	
Connection	
Data	
Conclusion	

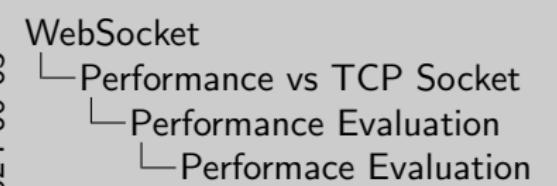
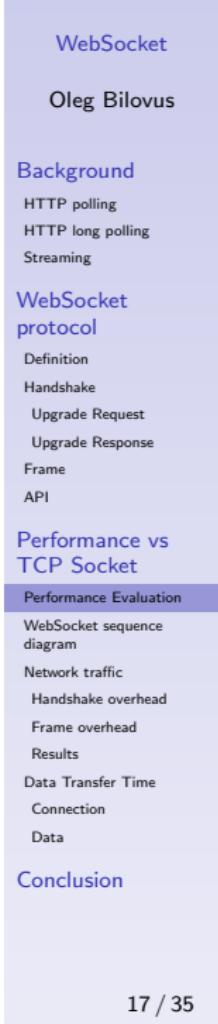
2024-06-03	WebSocket	
	Performance vs TCP Socket	
	Performance Evaluation	
	Performance Evaluation	

► Performance evaluation of the WebSocket and the TCP Socket protocol consists of:
▶ Network traffic

Performance Evaluation

Performance Evaluation

- ▶ Performance evaluation of the WebSocket and the TCP Socket protocol consists of:
 - ▶ Network traffic
 - ▶ Data transfer time



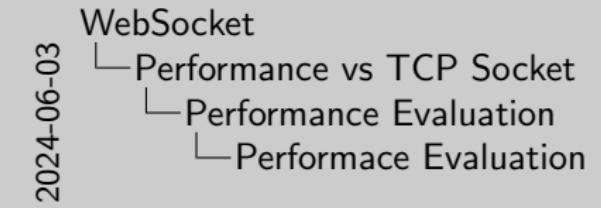
► Performance evaluation of the WebSocket and the TCP Socket protocol consists of:
▶ Network traffic
▶ Data transfer time

► Performance evaluation of the WebSocket and the TCP Socket protocol consists of:
▶ Network traffic
▶ Data transfer time

Performance Evaluation

- ▶ Performance evaluation of the WebSocket and the TCP Socket protocol consists of:
 - ▶ Network traffic
 - ▶ Data transfer time
- ▶ Network traffic is *evaluated analytically* using the protocol specifications.

WebSocket	Background	Performance vs TCP Socket	Conclusion
Oleg Bilovus	HTTP polling HTTP long polling Streaming	Performance Evaluation	
	WebSocket protocol	Performance Evaluation	
	Definition Handshake Upgrade Request Upgrade Response Frame API		
	Performance vs TCP Socket	Performance Evaluation	
	WebSocket sequence diagram Network traffic Handshake overhead Frame overhead Results Data Transfer Time Connection Data		
	Conclusion		

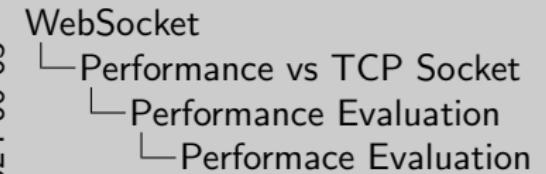


Performance Evaluation

- ▶ Performance evaluation of the WebSocket and the TCP Socket protocol consists of:
 - ▶ Network traffic
 - ▶ Data transfer time
- ▶ Network traffic is evaluated analytically using the protocol specifications.

Performance Evaluation

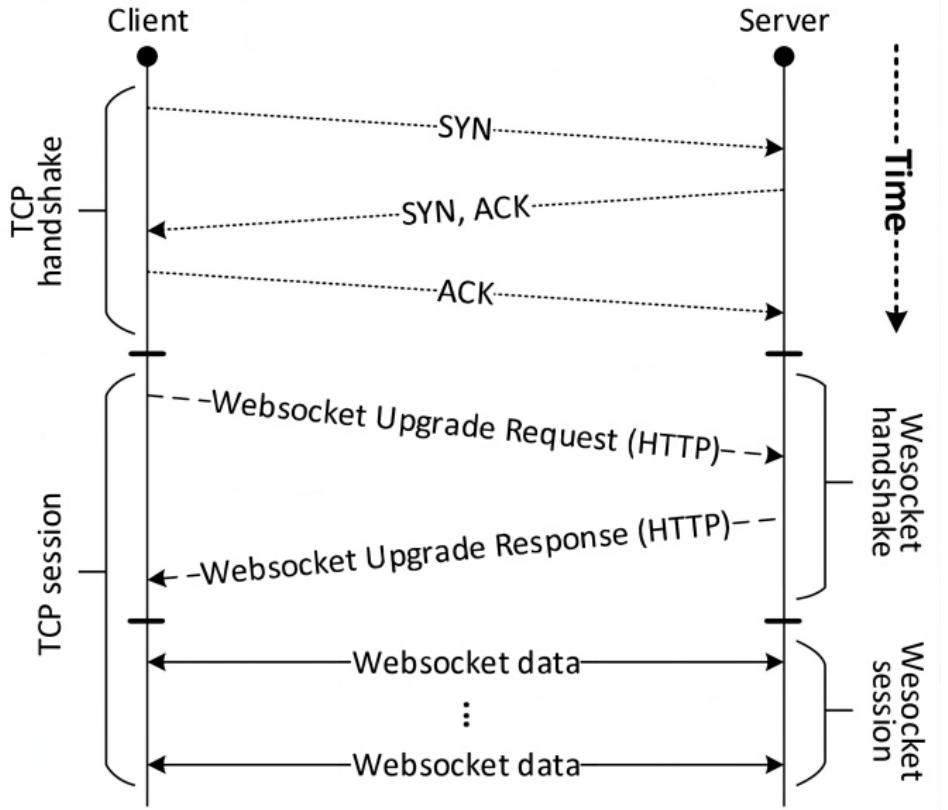
- ▶ Performance evaluation of the WebSocket and the TCP Socket protocol consists of:
 - ▶ Network traffic
 - ▶ Data transfer time
- ▶ Network traffic is *evaluated analytically* using the protocol specifications.
- ▶ Data transfer time is *evaluated experimentally* in a laboratory test bed.



Performance Evaluation

- ▶ Performance evaluation of the WebSocket and the TCP Socket protocol consists of:
 - ▶ Network traffic
 - ▶ Data transfer time
- ▶ Network traffic is evaluated analytically using the protocol specifications.
- ▶ Data transfer time is evaluated experimentally in a laboratory test bed.

WebSocket sequence diagram



WebSocket
Oleg Bilovus

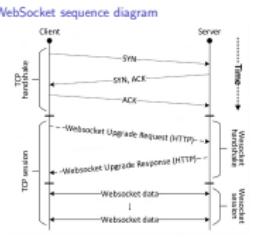
Background
HTTP polling
HTTP long polling
Streaming

WebSocket protocol
Definition
Handshake
Upgrade Request
Upgrade Response
Frame
API

Performance vs TCP Socket
Performance Evaluation
WebSocket sequence diagram
Network traffic
Handshake overhead
Frame overhead
Results
Data Transfer Time
Connection
Data

Conclusion

WebSocket
2024-06-03
Performance vs TCP Socket
WebSocket sequence diagram
WebSocket sequence diagram



WebSocket stay on top of TCP which means it will always add more overhead than the raw TCP Socket, but WebSocket is easier to use in a web environment.

Analytical Evaluation of Network Traffic

WebSocket

Oleg Bilovus

Background

- HTTP polling
- HTTP long polling
- Streaming

WebSocket protocol

- Definition
- Handshake
- Upgrade Request
- Upgrade Response
- Frame
- API

Performance vs TCP Socket

- Performance Evaluation
- WebSocket sequence diagram

Network traffic

- Handshake overhead
- Frame overhead
- Results
- Data Transfer Time
- Connection
- Data

Conclusion

WebSocket

Performance vs TCP Socket

Network traffic

Analytical Evaluation of Network Traffic

2024-06-03

Both protocols will have the lower level protocols fields overhead such as *Ethernet, IP and TCP header*.

- Both protocols will have the lower level protocols fields overhead such as *Ethernet, IP and TCP header*.

Analytical Evaluation of Network Traffic

- ▶ Both protocols will have the lower level protocols fields overhead such as *Ethernet, IP and TCP header*.
- ▶ For this reason, the analysis consider only the overhead the WebSocket incurs:

WebSocket

Oleg Bilovus

Background

HTTP polling
HTTP long polling
Streaming

WebSocket protocol

Definition
Handshake
Upgrade Request
Upgrade Response
Frame
API

Performance vs TCP Socket

Performance Evaluation
WebSocket sequence diagram

Network traffic

Handshake overhead
Frame overhead
Results
Data Transfer Time
Connection
Data

Conclusion

WebSocket

Performance vs TCP Socket

Network traffic

Analytical Evaluation of Network Traffic

2024-06-03

- ▶ Both protocols will have the lower level protocols fields overhead such as *Ethernet, IP and TCP header*.
- ▶ For this reason, the analysis consider only the overhead the WebSocket incurs:

Analytical Evaluation of Network Traffic

Analytical Evaluation of Network Traffic

WebSocket

Oleg Bilovus

Background

HTTP polling
HTTP long polling
Streaming

WebSocket
protocol

Definition
Handshake
Upgrade Request
Upgrade Response
Frame
API

Performance vs
TCP Socket

Performance Evaluation
WebSocket sequence
diagram

Network traffic

Handshake overhead
Frame overhead
Results
Data Transfer Time
Connection
Data

Conclusion

WebSocket

Performance vs TCP Socket

Network traffic

Analytical Evaluation of Network Traffic

- ▶ Both protocols will have the lower level protocols fields overhead such as *Ethernet, IP and TCP header*.
- ▶ For this reason, the analysis consider only the overhead the WebSocket incurs:
 - ▶ Handshake

Both protocols will have the lower level protocols fields overhead such as *Ethernet, IP and TCP header*.
For this reason, the analysis consider only the overhead the WebSocket incurs:

- ▶ Handshake

Analytical Evaluation of Network Traffic

Analytical Evaluation of Network Traffic

WebSocket

Oleg Bilovus

Background

HTTP polling
HTTP long polling
Streaming

WebSocket protocol

Definition
Handshake
Upgrade Request
Upgrade Response
Frame
API

Performance vs TCP Socket

Performance Evaluation
WebSocket sequence diagram

Network traffic

Handshake overhead
Frame overhead
Results
Data Transfer Time
Connection
Data

Conclusion

WebSocket

Performance vs TCP Socket

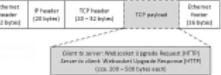
Network traffic

Analytical Evaluation of Network Traffic

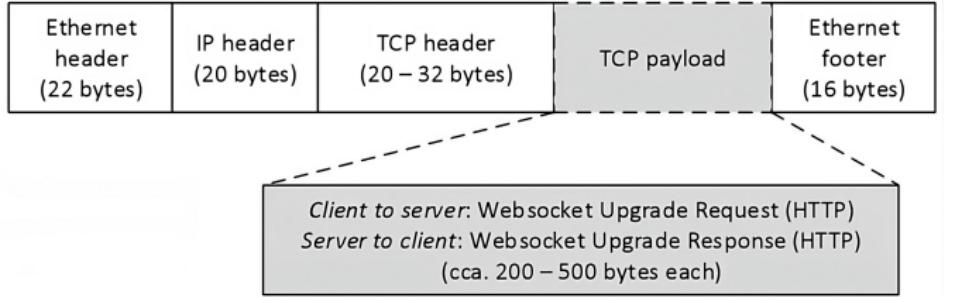
2024-06-03

- ▶ Both protocols will have the lower level protocols fields overhead such as *Ethernet, IP and TCP header*.
- ▶ For this reason, the analysis consider only the overhead the WebSocket incurs:
 - ▶ Handshake
 - ▶ Frame header for each frame

- ▶ Both protocols will have the lower level protocols fields overhead such as *Ethernet, IP and TCP header*.
- ▶ For this reason, the analysis consider only the overhead the WebSocket incurs:
 - ▶ Handshake
 - ▶ Frame header for each frame



Handshake overhead



WebSocket
Oleg Bilovus

Background
HTTP polling
HTTP long polling
Streaming

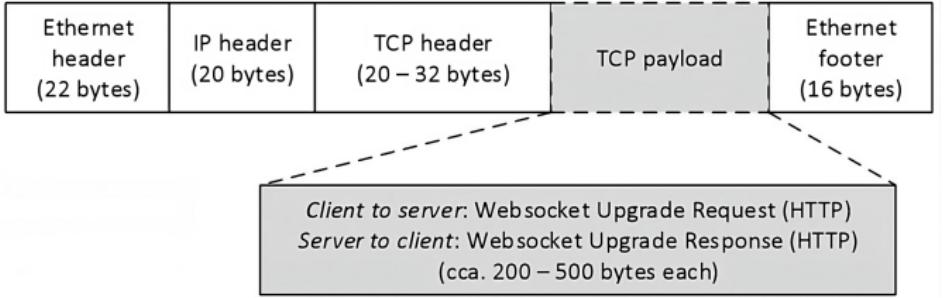
WebSocket protocol
Definition
Handshake
Upgrade Request
Upgrade Response
Frame
API

Performance vs TCP Socket
Performance Evaluation
WebSocket sequence diagram
Network traffic
Handshake overhead
Frame overhead
Results
Data Transfer Time
Connection
Data

Conclusion

WebSocket
2024-06-03
Performance vs TCP Socket
Network traffic
Handshake overhead

Handshake overhead



- The overhead is **fixed in length** and typically counts few hundreds of bytes.

WebSocket
Oleg Bilovus

Background

HTTP polling
HTTP long polling
Streaming

WebSocket protocol

Definition
Handshake
Upgrade Request
Upgrade Response
Frame
API

Performance vs TCP Socket

Performance Evaluation
WebSocket sequence diagram
Network traffic

Handshake overhead

Frame overhead
Results
Data Transfer Time
Connection
Data

Conclusion

WebSocket

- Performance vs TCP Socket
 - Network traffic
 - Handshake overhead

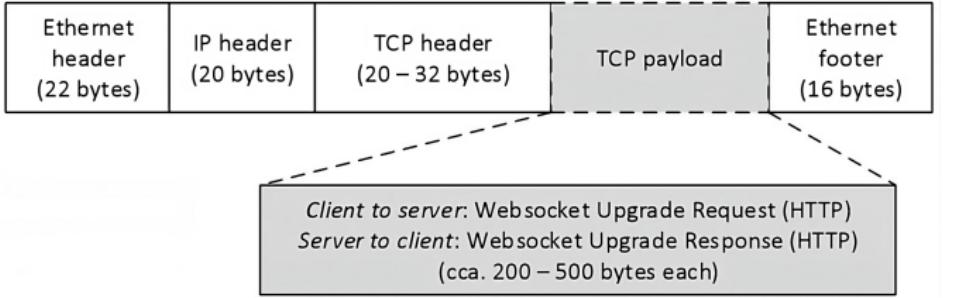
2024-06-03

Handshake overhead



► The overhead is **fixed in length** and typically counts few hundreds of bytes.

Handshake overhead



- ▶ The overhead is **fixed in length** and typically counts few hundreds of bytes.
- ▶ It is **performed only once** per session.

WebSocket
Oleg Bilovus

Background

HTTP polling
HTTP long polling
Streaming

WebSocket protocol

Definition
Handshake
Upgrade Request
Upgrade Response
Frame
API

Performance vs TCP Socket

Performance Evaluation
WebSocket sequence diagram

Handshake overhead

Frame overhead
Results
Data Transfer Time
Connection
Data

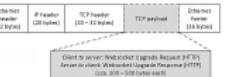
Conclusion

WebSocket

- Performance vs TCP Socket
 - Network traffic
 - Handshake overhead

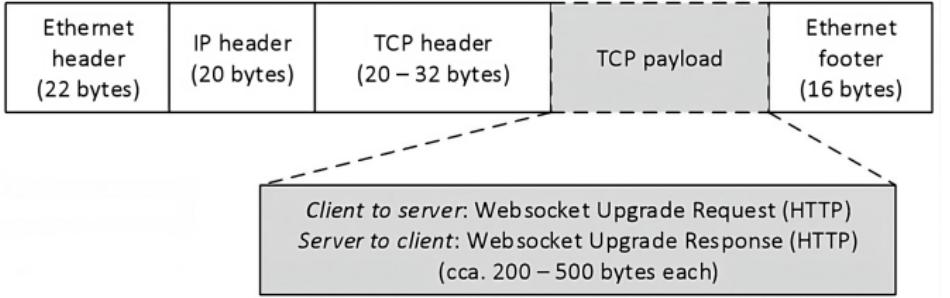
2024-06-03

Handshake overhead



- The overhead is **fixed in length** and typically counts few hundreds of bytes.
- It is **performed only once** per session.

Handshake overhead



- ▶ The overhead is **fixed in length** and typically counts few hundreds of bytes.
- ▶ It is **performed only once** per session.
- ▶ Its **significance decreases** with the increasing number of frames sent over the same session. Thus, the evaluation is focused on long-running sessions.

WebSocket
Oleg Bilovus

Background
HTTP polling
HTTP long polling
Streaming

WebSocket protocol
Definition
Handshake
Upgrade Request
Upgrade Response
Frame
API

Performance vs TCP Socket
Performance Evaluation
WebSocket sequence diagram
Network traffic
Handshake overhead
Frame overhead
Results
Data Transfer Time
Connection
Data

Conclusion

WebSocket
2024-06-03
Performance vs TCP Socket
Network traffic
Handshake overhead

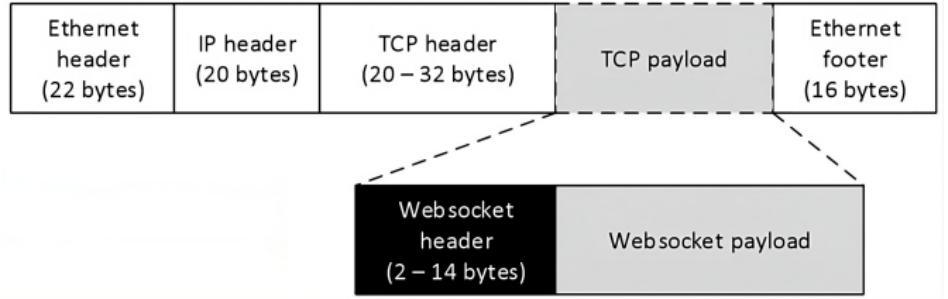
Handshake overhead

Ethernet header (12 bytes) | IP header (20 bytes) | TCP header (20 – 32 bytes) | TCP payload | Ethernet footer (16 bytes)

*Client to server: Websocket Upgrade Request (HTTP)
Server to client: Websocket Upgrade Response (HTTP)
(cca. 200 – 500 bytes each)*

- ▶ The overhead is **fixed in length** and typically counts few hundreds of bytes.
- ▶ It is **performed only once** per session.
- ▶ Its **significance decreases** with the increasing number of frames sent over the same session. Thus, the evaluation is focused on long-running sessions.

Frame overhead



WebSocket
Oleg Bilovus

Background

HTTP polling
HTTP long polling
Streaming

WebSocket protocol

Definition
Handshake
Upgrade Request
Upgrade Response
Frame
API

Performance vs TCP Socket

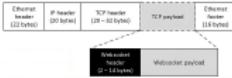
Performance Evaluation
WebSocket sequence diagram
Network traffic
Handshake overhead
Frame overhead
Results
Data Transfer Time
Connection
Data

Conclusion

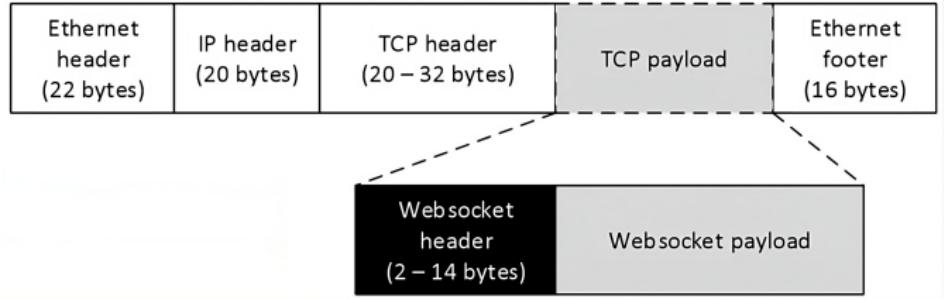
WebSocket
Performance vs TCP Socket
Network traffic
Frame overhead

2024-06-03

Frame overhead



Frame overhead



- The overhead counts **2 to 14 bytes** for each frame.

WebSocket
Oleg Bilovus

Background

HTTP polling
HTTP long polling
Streaming

WebSocket protocol

Definition
Handshake
Upgrade Request
Upgrade Response
Frame
API

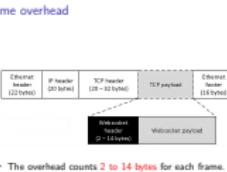
Performance vs TCP Socket

Performance Evaluation
WebSocket sequence diagram
Network traffic
Handshake overhead
Frame overhead
Results
Data Transfer Time
Connection
Data

Conclusion

WebSocket
Performance vs TCP Socket
Network traffic
Frame overhead

2024-06-03



Results

- When the data are transferred with TCP Socket, they are **directly embedded as TCP Payload**.

WebSocket	
Oleg Bilovus	
Background	
HTTP polling	
HTTP long polling	
Streaming	
WebSocket protocol	
Definition	
Handshake	
Upgrade Request	
Upgrade Response	
Frame	
API	
Performance vs TCP Socket	
Performance Evaluation	
WebSocket sequence diagram	
Network traffic	
Handshake overhead	
Frame overhead	
Results	
Data Transfer Time	
Connection	
Data	
Conclusion	

WebSocket	
Performance vs TCP Socket	
Network traffic	
Results	

Results
► When the data are transferred with TCP Socket, they are **directly embedded as TCP Payload**.

Results

- When the data are transferred with TCP Socket, they are **directly embedded as TCP Payload**.
- With WebSocket, the TCP Payload consists of both data and Frame header.

WebSocket	
Oleg Bilovus	
Background	
HTTP polling	
HTTP long polling	
Streaming	
WebSocket protocol	
Definition	
Handshake	
Upgrade Request	
Upgrade Response	
Frame	
API	
Performance vs TCP Socket	
Performance Evaluation	
WebSocket sequence diagram	
Network traffic	
Handshake overhead	
Frame overhead	
Results	
Data Transfer Time	
Connection	
Data	
Conclusion	

2024-06-03	WebSocket
	Performance vs TCP Socket
	Network traffic
	Results

Results

- ▶ When the data are transferred with TCP Socket, they are **directly embedded as TCP Payload**.
- ▶ With WebSocket, the TCP Payload consists of both data and Frame header.
- ▶ This relation can be written as:

$$P_{TCP} = data \quad (1)$$

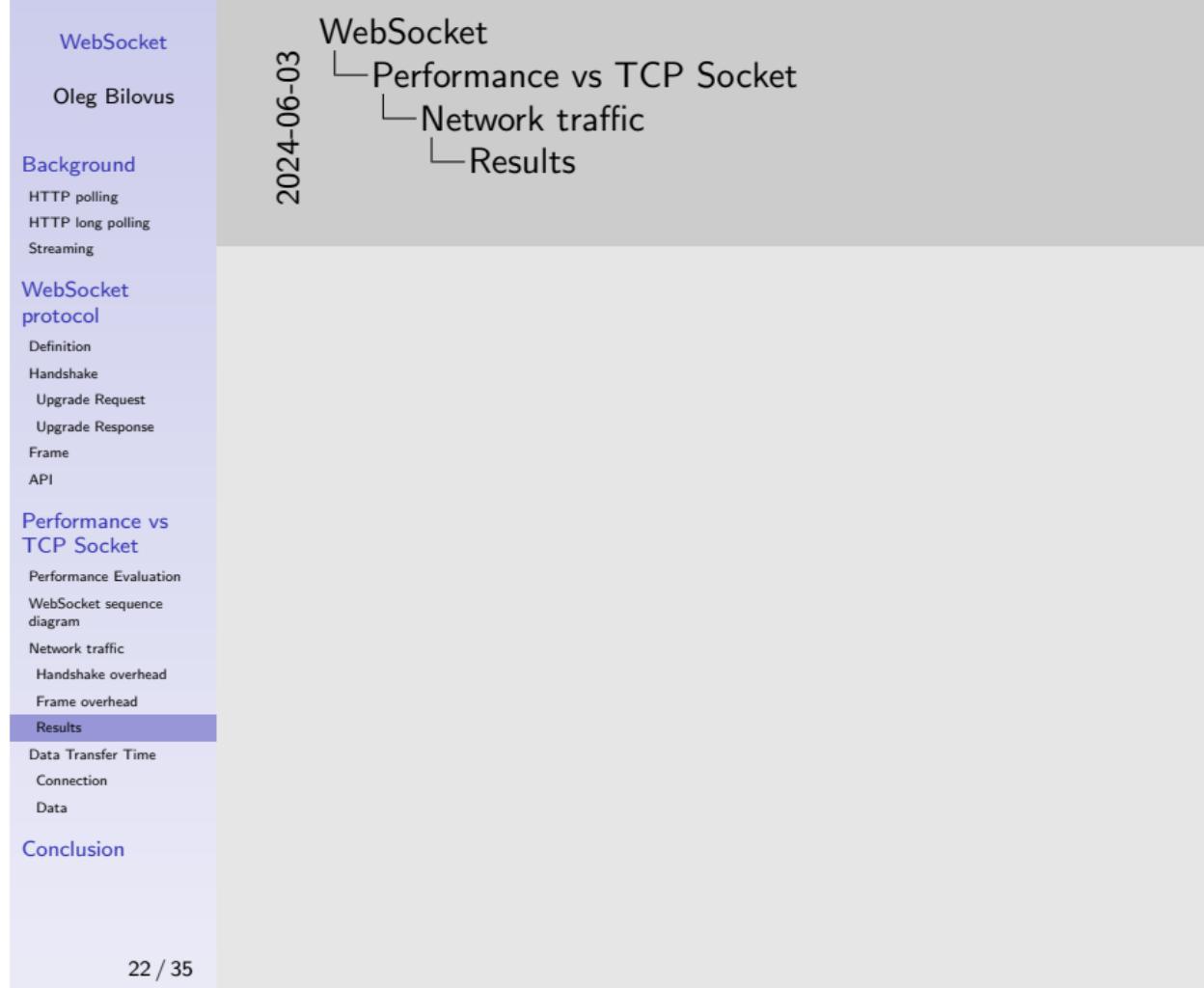
$$P_{WS} = data + H \quad (2)$$

where:

P = payload

$data$ = data to send

H = length of frame's header



Results

- ▶ When the data are transferred with TCP Socket, they are **directly embedded as TCP Payload**.
- ▶ With WebSocket, the TCP Payload consists of both data and Frame header.
- ▶ This relation can be written as:

where:

P = payload
 $data$ = data to send
 H = length of frame's header

$$P_{TCP} = data \quad (1)$$
$$P_{WS} = data + H \quad (2)$$

Results

- ▶ When the data are transferred with TCP Socket, they are **directly embedded as TCP Payload**.
- ▶ With WebSocket, the TCP Payload consists of both data and Frame header.
- ▶ This relation can be written as:

$$P_{TCP} = \text{data} \quad (1)$$

$$P_{WS} = \text{data} + H \quad (2)$$

where:

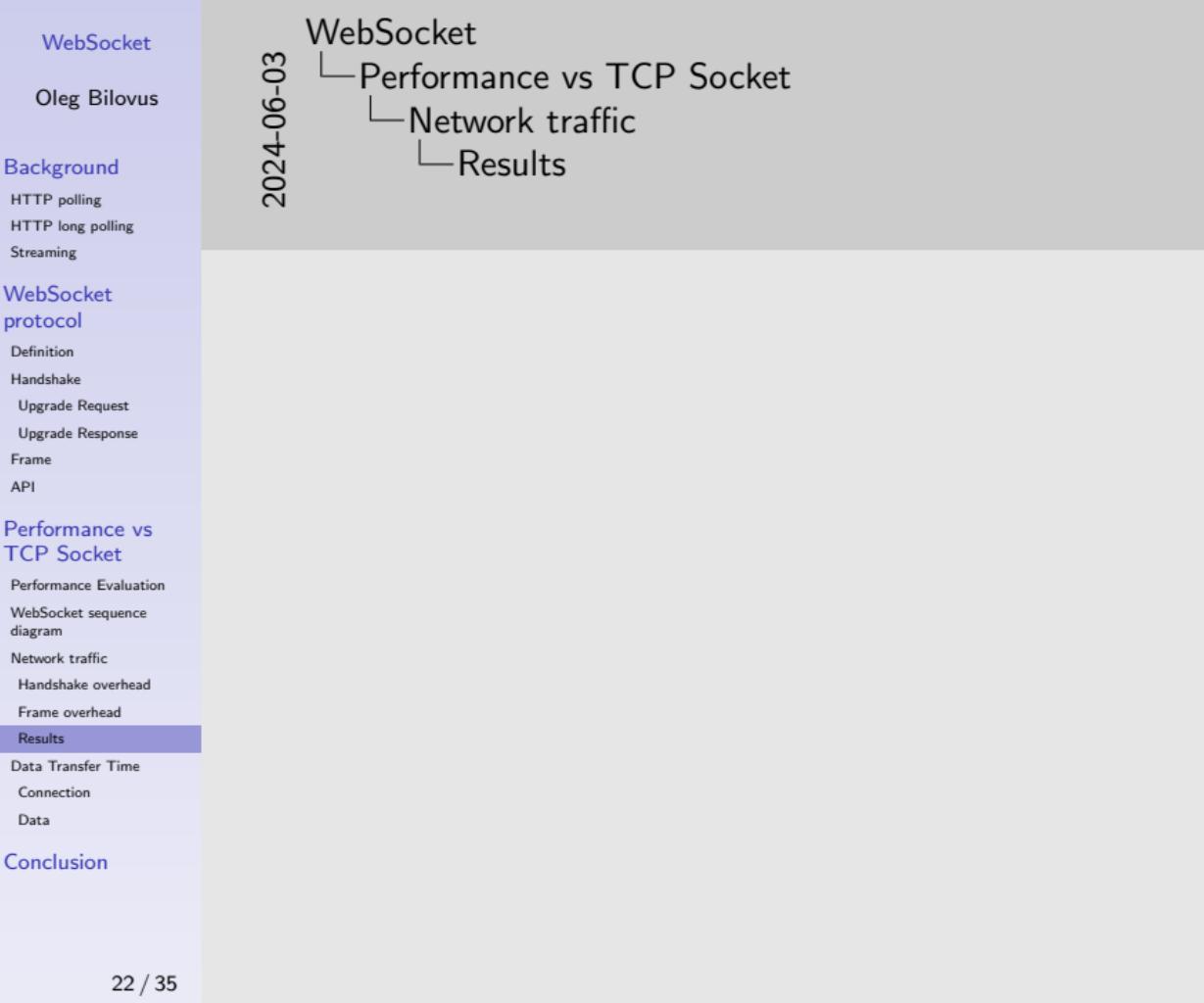
P = payload

data = data to send

H = length of frame's header

- ▶ We can now define the **network traffic overhead O_P** a WebSocket has over a TCP Socket:

$$O_P = \frac{P_{WS} - P_{TCP}}{P_{TCP}} \cdot 100\% \quad (3)$$



Results

- ▶ When the data are transferred with TCP Socket, they are **directly embedded as TCP Payload**.
- ▶ With WebSocket, the TCP Payload consists of both data and Frame header.
- ▶ This relation can be written as:

$$P_{TCP} = \text{data} \quad (1)$$
$$P_{WS} = \text{data} + H \quad (2)$$

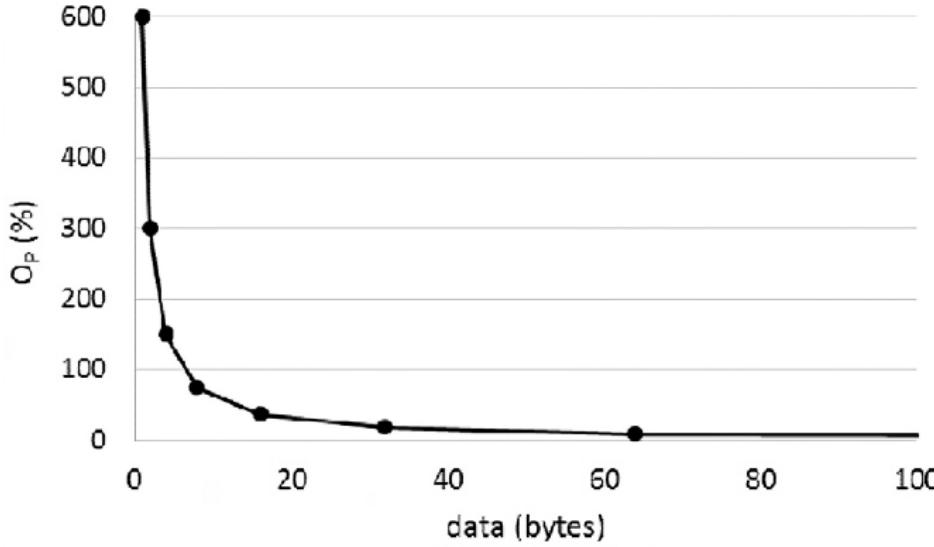
where:

- P = payload
- data = data to send
- H = length of frame's header

▶ We can now define the **network traffic overhead O_P** a WebSocket has over a TCP Socket:

$$O_P = \frac{P_{WS} - P_{TCP}}{P_{TCP}} \cdot 100\% \quad (3)$$

Results



WebSocket

Oleg Bilovus

Background

HTTP polling
HTTP long polling
Streaming

WebSocket protocol

Definition
Handshake
Upgrade Request
Upgrade Response
Frame
API

Performance vs TCP Socket

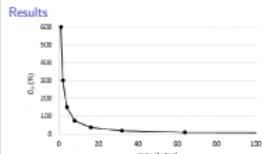
Performance Evaluation
WebSocket sequence diagram
Network traffic
Handshake overhead
Frame overhead
Results
Data Transfer Time
Connection
Data

Conclusion

WebSocket

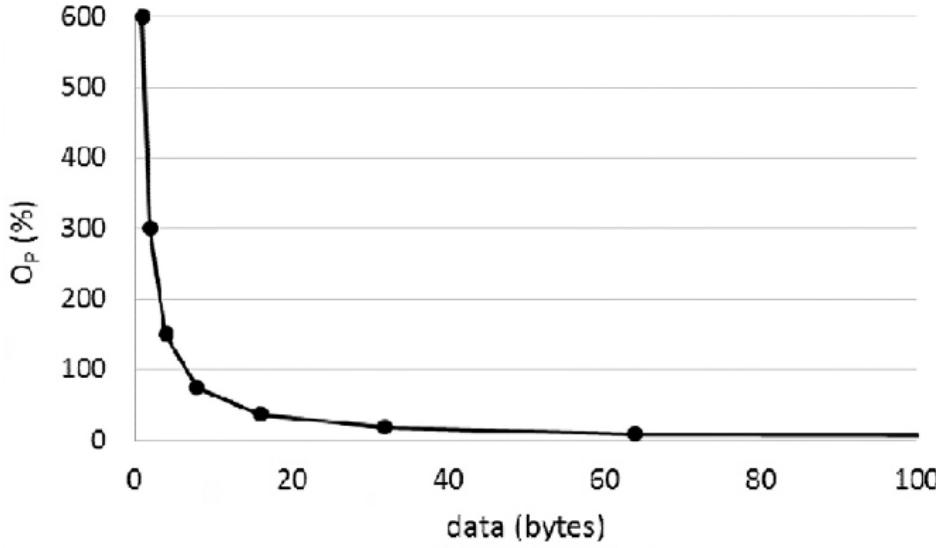
Performance vs TCP Socket
Network traffic
Results

2024-06-03



Because if I want to send 1 byte, with WebSocket it will add 14 bytes of frame header. The difference is smaller and smaller because 14 bytes of overhead on a 1 KB data is nothing. The difference is almost identical.

Results



- ▶ Significant difference in performance only for tiny data.

WebSocket

Oleg Bilovus

Background

- HTTP polling
- HTTP long polling
- Streaming

WebSocket protocol

- Definition
- Handshake
- Upgrade Request
- Upgrade Response
- Frame
- API

Performance vs TCP Socket

- Performance Evaluation
- WebSocket sequence diagram
- Network traffic
- Handshake overhead
- Frame overhead

Results

- Data Transfer Time
- Connection
- Data

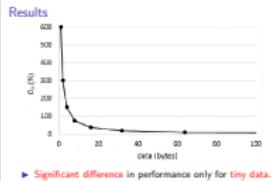
Conclusion

WebSocket

Performance vs TCP Socket

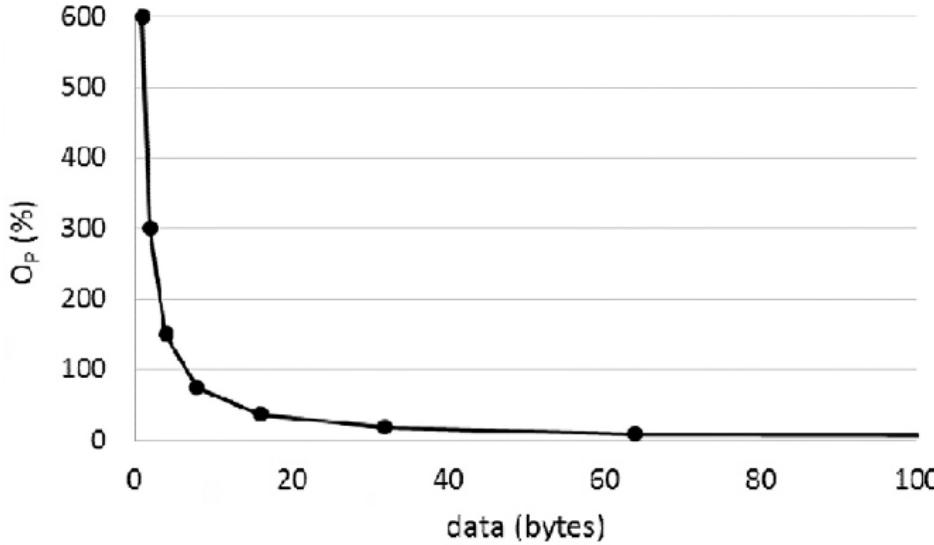
- Network traffic
- Results

2024-06-03



Because if I want to send 1 byte, with WebSocket it will add 14 bytes of frame header. The difference is smaller and smaller because 14 bytes of overhead on a 1 KB data is nothing. The difference is almost identical.

Results



- ▶ Significant difference in performance only for tiny data.
- ▶ For biggest messages, the WebSocket frame size converges very fast towards the TCP Socket size.

WebSocket
Oleg Bilovus

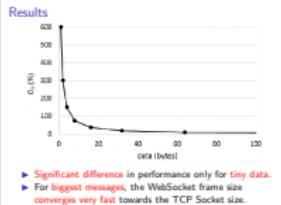
Background
HTTP polling
HTTP long polling
Streaming

WebSocket protocol
Definition
Handshake
Upgrade Request
Upgrade Response
Frame
API

Performance vs TCP Socket
Performance Evaluation
WebSocket sequence diagram
Network traffic
Handshake overhead
Frame overhead
Results

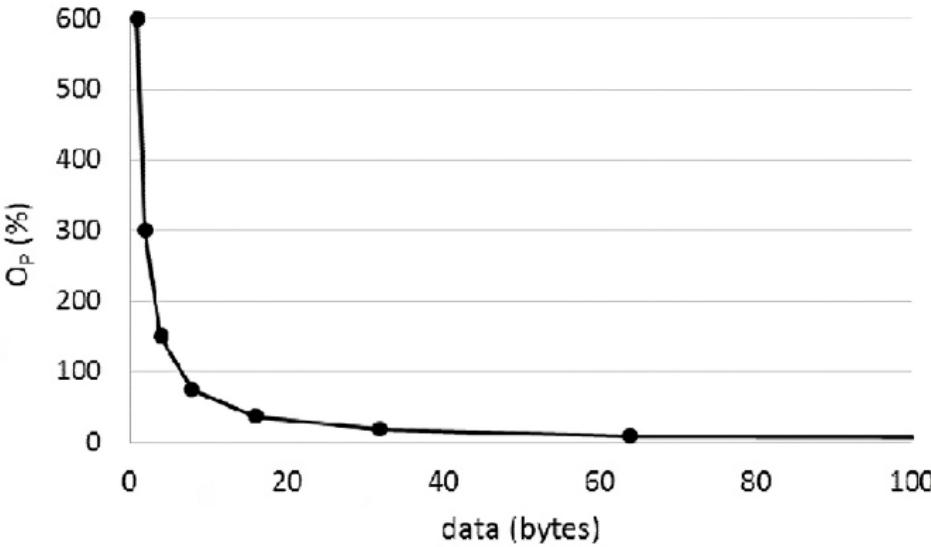
Conclusion

WebSocket
2024-06-03
Performance vs TCP Socket
Network traffic
Results

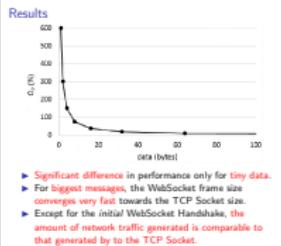


Because if I want to send 1 byte, with WebSocket it will add 14 bytes of frame header. The difference is smaller and smaller because 14 bytes of overhead on a 1 KB data is nothing. The difference is almost identical.

Results



- ▶ Significant difference in performance only for tiny data.
- ▶ For biggest messages, the WebSocket frame size converges very fast towards the TCP Socket size.
- ▶ Except for the *initial* WebSocket Handshake, the amount of network traffic generated is comparable to that generated by to the TCP Socket.



Because if I want to send 1 byte, with WebSocket it will could add 14 bytes of frame header. The difference is smaller and smaller because 14 bytes of overhead on a 1 KB data is nothing. The difference is almost identical.

Experimental Evaluation of Data Transfer Time

WebSocket

Oleg Bilovus

Background

- HTTP polling
- HTTP long polling
- Streaming

WebSocket
protocol

- Definition
- Handshake
- Upgrade Request
- Upgrade Response
- Frame
- API

Performance vs
TCP Socket

- Performance Evaluation
- WebSocket sequence diagram
- Network traffic
- Handshake overhead
- Frame overhead
- Results

Data Transfer Time

- Connection
- Data

Conclusion

WebSocket

Performance vs TCP Socket

Data Transfer Time

Experimental Evaluation of Data Transfer Time

2024-06-03

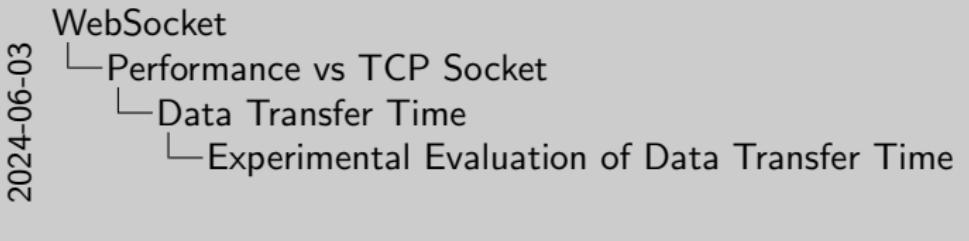
▶ Two host machines.

- ▶ Two host machines.

Experimental Evaluation of Data Transfer Time

- ▶ Two host machines.
- ▶ One playing the role of the server.

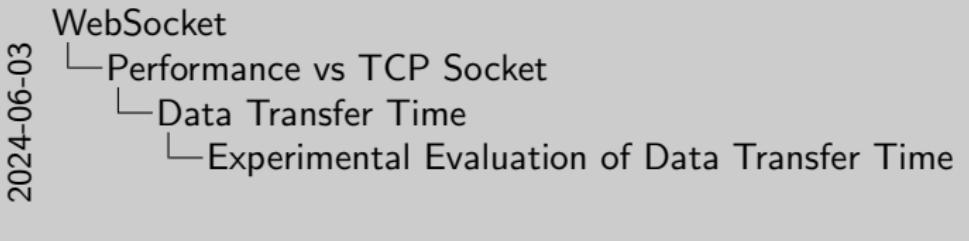
WebSocket	
Oleg Bilovus	
Background	
HTTP polling	
HTTP long polling	
Streaming	
WebSocket protocol	
Definition	
Handshake	
Upgrade Request	
Upgrade Response	
Frame	
API	
Performance vs TCP Socket	
Performance Evaluation	
WebSocket sequence diagram	
Network traffic	
Handshake overhead	
Frame overhead	
Results	
Data Transfer Time	
Connection	
Data	
Conclusion	



Experimental Evaluation of Data Transfer Time

- ▶ Two host machines.
- ▶ One playing the role of the server.
- ▶ While the other being a client.

WebSocket	
Oleg Bilovus	
Background	
HTTP polling	
HTTP long polling	
Streaming	
WebSocket protocol	
Definition	
Handshake	
Upgrade Request	
Upgrade Response	
Frame	
API	
Performance vs TCP Socket	
Performance Evaluation	
WebSocket sequence diagram	
Network traffic	
Handshake overhead	
Frame overhead	
Results	
Data Transfer Time	
Connection	
Data	
Conclusion	



Environment configuration

	Client	Server
Hardware	CPU: AMD Turion II P520 RAM: 6 GB	CPU: AMD Athlon X2 5000 RAM: 5 GB
OS	Windows 8 64-bit	Windows 8 64-bit
Network	1000BASE-T (Gigabit Ethernet, host machines directly connected using UTP Cat5 Ethernet cable)	
TCP implementation	<code>java.net.Socket</code> (Java JDK 1.7)	<code>java.net.Socket</code> (Java JDK 1.7)
WebSocket implementation	<code>websocket.client</code> (Jetty 9.1.0)	<code>websocket.servlet</code> (Jetty 9.1.0)

WebSocket
Oleg Bilovus

Background
HTTP polling
HTTP long polling
Streaming

WebSocket protocol
Definition
Handshake
Upgrade Request
Upgrade Response
Frame
API

Performance vs TCP Socket
Performance Evaluation
WebSocket sequence diagram
Network traffic
Handshake overhead
Frame overhead
Results

Data Transfer Time
Connection
Data

Conclusion

WebSocket
2024-06-03
Performance vs TCP Socket
Data Transfer Time
Environment configuration

	Client	Server
Hardware	CPU: AMD Turion II P520 RAM: 6 GB	CPU: AMD Athlon X2 5000 RAM: 5 GB
OS	Windows 8 64-bit	Windows 8 64-bit
Network	1000BASE-T (Gigabit Ethernet, host machines directly connected using UTP Cat5 Ethernet cable)	
TCP implementation	<code>java.net.Socket</code> (Java JDK 1.7)	<code>java.net.Socket</code> (Java JDK 1.7)
WebSocket implementation	<code>websocket.client</code> (Jetty 9.1.0)	<code>websocket.servlet</code> (Jetty 9.1.0)

TCP Connection and WebSocket session time

In the first experiment, it is measured the time required for client and server to establish a TCP connection and WebSocket session.

WebSocket

Oleg Bilovus

Background

- HTTP polling
- HTTP long polling
- Streaming

WebSocket protocol

- Definition
- Handshake
- Upgrade Request
- Upgrade Response
- Frame
- API

Performance vs TCP Socket

- Performance Evaluation
- WebSocket sequence diagram
- Network traffic
- Handshake overhead
- Frame overhead
- Results
- Data Transfer Time

Connection

Data

Conclusion

WebSocket

Performance vs TCP Socket

Data Transfer Time

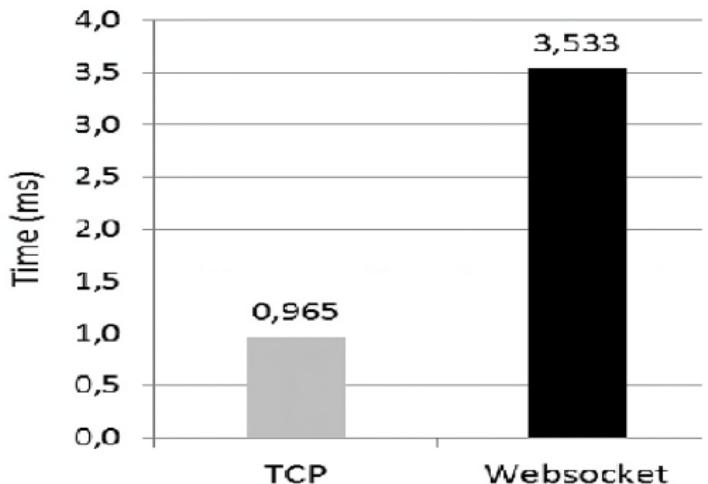
TCP Connection and WebSocket session time

2024-06-03

TCP Connection and WebSocket session time
In the first experiment, it is measured the time required for client and server to establish a TCP connection and WebSocket session.

TCP Connection and WebSocket session time

In the first experiment, it is measured the time required for client and server to establish a TCP connection and WebSocket session.



- ▶ WebSocket session lasts **3.7 times longer** than establishing a TCP connection.

WebSocket

Oleg Bilovus

Background

HTTP polling
HTTP long polling
Streaming

WebSocket protocol

Definition
Handshake
Upgrade Request
Upgrade Response
Frame
API

Performance vs TCP Socket

Performance Evaluation
WebSocket sequence diagram
Network traffic
Handshake overhead
Frame overhead
Results
Data Transfer Time

Connection

Data

Conclusion

WebSocket

Performance vs TCP Socket

Data Transfer Time

TCP Connection and WebSocket session time

2024-06-03

TCP Connection and WebSocket session time
In the first experiment, it is measured the time required for client and server to establish a TCP connection and WebSocket session.

A bar chart titled 'Time (ms)' on the y-axis (ranging from 0,0 to 4,0). It compares 'TCP' at 0,965 ms and 'Websocket' at 3,533 ms. A red arrow points from the text above to the 'Websocket' bar.

Protocol	Time (ms)
TCP	0,965
Websocket	3,533

► WebSocket session lasts **3.7 times longer** than establishing a TCP connection.

TCP Connection and WebSocket session time

- ▶ The reason for such slow performance of the WebSocket is the fact that the protocol is not a *transport protocol*.

WebSocket

Oleg Bilovus

Background

- HTTP polling
- HTTP long polling
- Streaming

WebSocket
protocol

- Definition
- Handshake
- Upgrade Request
- Upgrade Response
- Frame
- API

Performance vs
TCP Socket

- Performance Evaluation
- WebSocket sequence diagram
- Network traffic
- Handshake overhead
- Frame overhead
- Results
- Data Transfer Time

Connection

Data

Conclusion

WebSocket

Performance vs TCP Socket

Data Transfer Time

TCP Connection and WebSocket session time

2024-06-03

▶ The reason for such slow performance of the WebSocket is the fact that the protocol is not a *transport protocol*.

TCP Connection and WebSocket session time

TCP Connection and WebSocket session time

- ▶ The reason for such slow performance of the WebSocket is the fact that the protocol is not a *transport protocol*.
- ▶ WebSocket sits on top of TCP and uses HTTP for the handshake.

WebSocket

Oleg Bilovus

Background

- HTTP polling
- HTTP long polling
- Streaming

WebSocket
protocol

- Definition
- Handshake
- Upgrade Request
- Upgrade Response
- Frame
- API

Performance vs
TCP Socket

- Performance Evaluation
- WebSocket sequence diagram
- Network traffic
- Handshake overhead
- Frame overhead
- Results
- Data Transfer Time

Connection

Data

Conclusion

WebSocket

Performance vs TCP Socket

Data Transfer Time

TCP Connection and WebSocket session time

2024-06-03

- ▶ The reason for such slow performance of the WebSocket is the fact that the protocol is not a *transport protocol*
- ▶ WebSocket sits on top of TCP and uses HTTP for the handshake.

TCP Connection and WebSocket session time

We have already seen this previously.

TCP Connection and WebSocket session time

- ▶ The reason for such slow performance of the WebSocket is the fact that the protocol is not a *transport protocol*.
- ▶ WebSocket sits on top of TCP and uses HTTP for the handshake.
- ▶ Which means it first has to establish a TCP connection, allocate the resource for HTTP and then can establish a WebSocket handshake.

WebSocket	
Oleg Bilovus	
Background	
HTTP polling	
HTTP long polling	
Streaming	
WebSocket protocol	
Definition	
Handshake	
Upgrade Request	
Upgrade Response	
Frame	
API	
Performance vs TCP Socket	
Performance Evaluation	
WebSocket sequence diagram	
Network traffic	
Handshake overhead	
Frame overhead	
Results	
Data Transfer Time	
Connection	
Data	
Conclusion	

2024-06-03

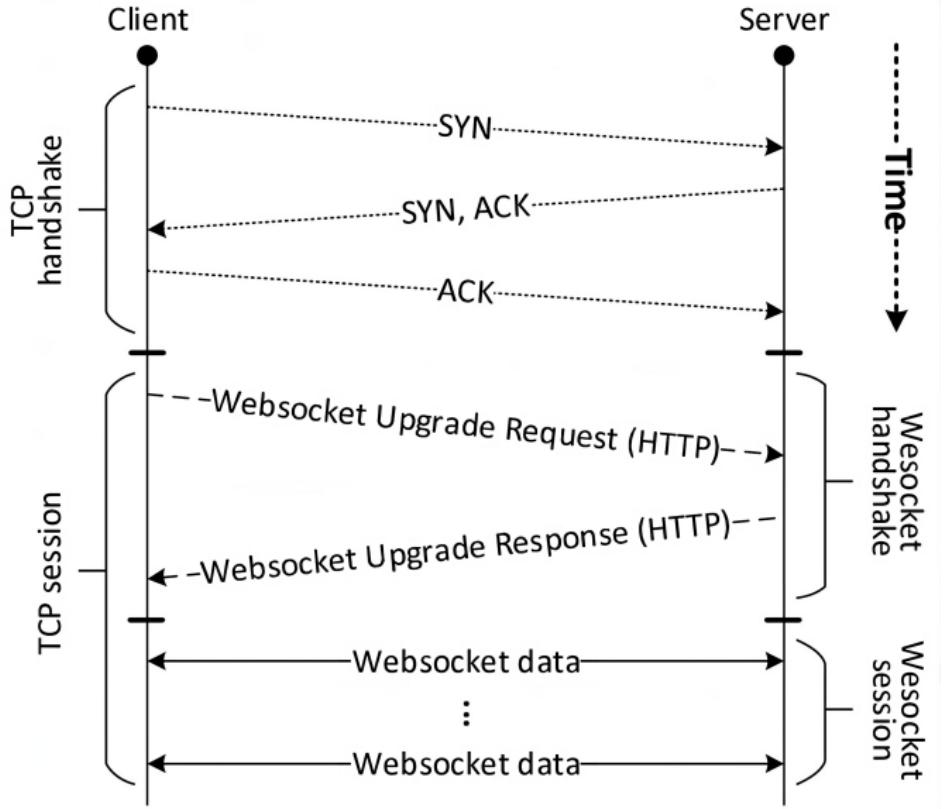
WebSocket
└ Performance vs TCP Socket
 └ Data Transfer Time
 └ TCP Connection and WebSocket session time

We have already seen this previously.

- ▶ The reason for such slow performance of the WebSocket is the fact that the protocol is not a *transport protocol*
- ▶ WebSocket sits on top of TCP and uses HTTP for the handshake.
- ▶ Which means it first has to establish a TCP connection, allocate the resource for HTTP and then can establish a WebSocket handshake.

TCP Connection and WebSocket session time

WebSocket sequence diagram



WebSocket
Oleg Bilovus

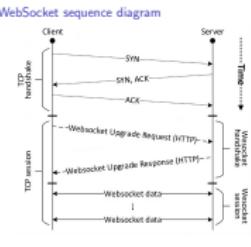
Background
HTTP polling
HTTP long polling
Streaming

WebSocket protocol
Definition
Handshake
Upgrade Request
Upgrade Response
Frame
API

Performance vs TCP Socket
Performance Evaluation
WebSocket sequence diagram
Network traffic
Handshake overhead
Frame overhead
Results
Data Transfer Time
Connection
Data

Conclusion

WebSocket
2024-06-03
Performance vs TCP Socket
Data Transfer Time
WebSocket sequence diagram



WebSocket stay on top of TCP which means it will always add more overhead than the raw TCP Socket, but WebSocket is easier to use in a web environment.

► In the second experiment, it is measured the data transfer time after the TCP connection and WebSocket session have been established.

Data transfer time after connection

- In the second experiment, it is measured the data transfer time after the TCP connection and WebSocket session have been established.

WebSocket	
Oleg Bilovus	
Background	
HTTP polling	
HTTP long polling	
Streaming	
WebSocket protocol	
Definition	
Handshake	
Upgrade Request	
Upgrade Response	
Frame	
API	
Performance vs TCP Socket	
Performance Evaluation	
WebSocket sequence diagram	
Network traffic	
Handshake overhead	
Frame overhead	
Results	
Data Transfer Time	
Connection	
Data	
Conclusion	

2024-06-03	WebSocket
	Performance vs TCP Socket
	Data Transfer Time
	Data transfer time after connection

- ▶ In the second experiment, it is measured the data transfer time after the TCP connection and WebSocket session have been established.
- ▶ The Client generates a given amount of data and sends them to the server.

Data transfer time after connection

- ▶ In the second experiment, it is measured the data transfer time after the TCP connection and WebSocket session have been established.
- ▶ The Client generates a given amount of data and sends them to the server.

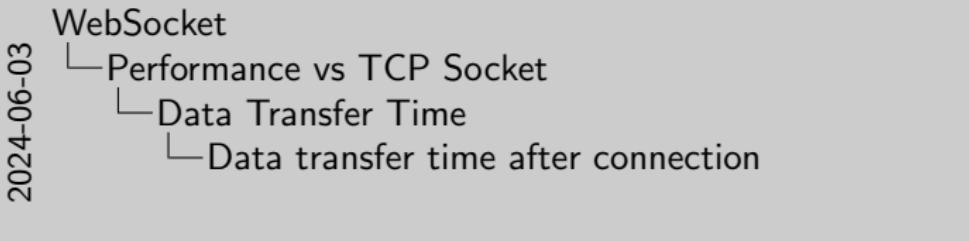
WebSocket	
Oleg Bilovus	
Background	
HTTP polling	
HTTP long polling	
Streaming	
WebSocket protocol	
Definition	
Handshake	
Upgrade Request	
Upgrade Response	
Frame	
API	
Performance vs TCP Socket	
Performance Evaluation	
WebSocket sequence diagram	
Network traffic	
Handshake overhead	
Frame overhead	
Results	
Data Transfer Time	
Connection	
Data	
Conclusion	

2024-06-03	WebSocket
	Performance vs TCP Socket
	Data Transfer Time
	Data transfer time after connection

Data transfer time after connection

- ▶ In the second experiment, it is measured the data transfer time after the TCP connection and WebSocket session have been established.
- ▶ The Client generates a given amount of data and sends them to the server.
- ▶ The Server echoes the same data back to the client.

WebSocket	
Oleg Bilovus	
Background	
HTTP polling	
HTTP long polling	
Streaming	
WebSocket protocol	
Definition	
Handshake	
Upgrade Request	
Upgrade Response	
Frame	
API	
Performance vs TCP Socket	
Performance Evaluation	
WebSocket sequence diagram	
Network traffic	
Handshake overhead	
Frame overhead	
Results	
Data Transfer Time	
Connection	
Data	
Conclusion	



Data transfer time after connection

- ▶ In the second experiment, it is measured the data transfer time after the TCP connection and WebSocket session have been established.
- ▶ The Client generates a given amount of data and sends them to the server.
- ▶ The Server echoes the same data back to the client.
- ▶ It is possible to define the **relative time overhead** O_T a WebSocket incurs over TCP as:

$$O_T = \frac{T_{WS} - T_{TCP}}{T_{TCP}} \cdot 100\% \quad (4)$$

where:

T = time to transfer data

WebSocket
Oleg Bilovus

Background
HTTP polling
HTTP long polling
Streaming

WebSocket protocol
Definition
Handshake
Upgrade Request
Upgrade Response
Frame
API

Performance vs TCP Socket
Performance Evaluation
WebSocket sequence diagram
Network traffic
Handshake overhead
Frame overhead
Results
Data Transfer Time
Connection
Data

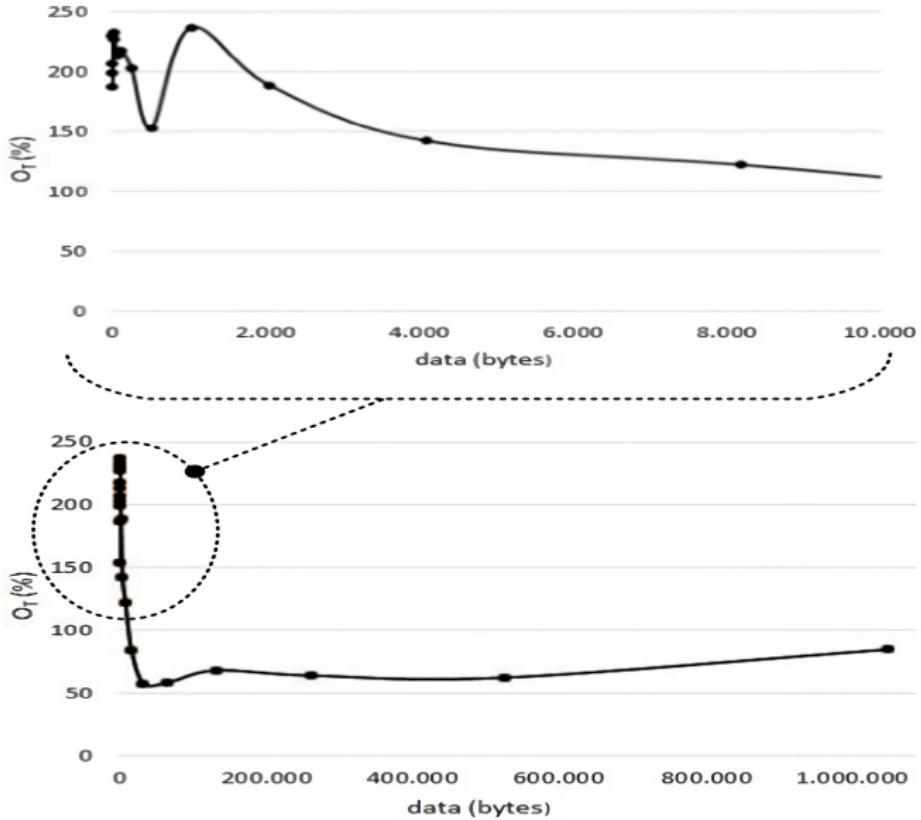
Conclusion

WebSocket
2024-06-03
Performance vs TCP Socket
Data Transfer Time
Data transfer time after connection

Data transfer time after connection
► In the second experiment, it is measured the data transfer time after the TCP connection and WebSocket session have been established.
► The Client generates a given amount of data and sends them to the server.
► The Server echoes the same data back to the client.
► It is possible to define the **relative time overhead** O_T a WebSocket incurs over TCP as:
$$O_T = \frac{T_{WS} - T_{TCP}}{T_{TCP}} \cdot 100\% \quad (4)$$

where:
 T = time to transfer data

Data transfer time after connection



WebSocket
Oleg Bilovus

Background

HTTP polling
HTTP long polling
Streaming

WebSocket protocol

Definition
Handshake
Upgrade Request
Upgrade Response
Frame
API

Performance vs TCP Socket

Performance Evaluation
WebSocket sequence diagram
Network traffic
Handshake overhead
Frame overhead
Results
Data Transfer Time
Connection
Data

Conclusion

WebSocket

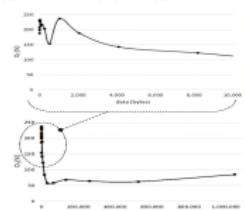
Performance vs TCP Socket

Data Transfer Time

Data transfer time after connection

2024-06-03

Data transfer time after connection



► The WebSocket performs slower than the TCP.

Data transfer time after connection

WebSocket
Oleg Bilovus

Background

HTTP polling
HTTP long polling
Streaming

WebSocket protocol

Definition
Handshake
Upgrade Request
Upgrade Response
Frame
API

Performance vs TCP Socket

Performance Evaluation
WebSocket sequence diagram
Network traffic
Handshake overhead
Frame overhead
Results
Data Transfer Time
Connection
Data

Conclusion

WebSocket

Performance vs TCP Socket

Data Transfer Time

Data transfer time after connection

2024-06-03

- The WebSocket performs slower than the TCP.

Data transfer time after connection

- ▶ The WebSocket performs slower than the TCP.
- ▶ The performance drop is more significant for small messages.

WebSocket	
Oleg Bilovus	
Background	
HTTP polling	
HTTP long polling	
Streaming	
WebSocket protocol	
Definition	
Handshake	
Upgrade Request	
Upgrade Response	
Frame	
API	
Performance vs TCP Socket	
Performance Evaluation	
WebSocket sequence diagram	
Network traffic	
Handshake overhead	
Frame overhead	
Results	
Data Transfer Time	
Connection	
Data	
Conclusion	

2024-06-03	WebSocket
	Performance vs TCP Socket
	Data Transfer Time
	Data transfer time after connection

► The WebSocket performs slower than the TCP.
► The performance drop is more significant for small messages.

Data transfer time after connection

- ▶ The WebSocket performs slower than the TCP.
- ▶ The performance drop is more significant for small messages.
- ▶ The overhead of WebSocket *fluctuates* between 150% and 250% for small messages.

WebSocket	
Oleg Bilovus	
Background	
HTTP polling	
HTTP long polling	
Streaming	
WebSocket protocol	
Definition	
Handshake	
Upgrade Request	
Upgrade Response	
Frame	
API	
Performance vs TCP Socket	
Performance Evaluation	
WebSocket sequence diagram	
Network traffic	
Handshake overhead	
Frame overhead	
Results	
Data Transfer Time	
Connection	
Data	
Conclusion	

2024-06-03	WebSocket
	Performance vs TCP Socket
	Data Transfer Time
	Data transfer time after connection

- ▶ The WebSocket performs slower than the TCP.
- ▶ The performance drop is more significant for small messages.
- ▶ The overhead of WebSocket fluctuates between 150% and 250% for small messages.

Data transfer time after connection

- ▶ The WebSocket performs slower than the TCP.
- ▶ The performance drop is more significant for small messages.
- ▶ The overhead of WebSocket *fluctuates* between 150% and 250% for small messages.
- ▶ For biggest messages, the overhead is more *stable* at 60-70%.

WebSocket	
Oleg Bilovus	
Background	
HTTP polling	
HTTP long polling	
Streaming	
WebSocket protocol	
Definition	
Handshake	
Upgrade Request	
Upgrade Response	
Frame	
API	
Performance vs TCP Socket	
Performance Evaluation	
WebSocket sequence diagram	
Network traffic	
Handshake overhead	
Frame overhead	
Results	
Data Transfer Time	
Connection	
Data	
Conclusion	

WebSocket	
Performance vs TCP Socket	
Data Transfer Time	
Data transfer time after connection	

- ▶ The WebSocket performs slower than the TCP.
- ▶ The performance drop is more significant for small messages.
- ▶ The overhead of WebSocket fluctuates between 150% and 250% for small messages.
- ▶ For biggest messages, the overhead is more stable at 60-70%.

Why WebSocket performs slower?

- ▶ WebSocket sits on top of TCP.

WebSocket

Oleg Bilovus

Background

- HTTP polling
- HTTP long polling
- Streaming

WebSocket
protocol

- Definition
- Handshake
- Upgrade Request
- Upgrade Response
- Frame
- API

Performance vs
TCP Socket

- Performance Evaluation
- WebSocket sequence
diagram
- Network traffic
- Handshake overhead
- Frame overhead
- Results
- Data Transfer Time
- Connection
- Data

Conclusion

WebSocket

Performance vs TCP Socket

Data Transfer Time

Why WebSocket performs slower?

2024-06-03

Why WebSocket performs slower?

▶ WebSocket sits on top of TCP.

Why WebSocket performs slower?

- ▶ WebSocket sits on top of TCP.
- ▶ WebSocket uses event-driven callback-based API to deliver data, which requires additional application data handling.

WebSocket	
Oleg Bilovus	
Background	
HTTP polling	
HTTP long polling	
Streaming	
WebSocket protocol	
Definition	
Handshake	
Upgrade Request	
Upgrade Response	
Frame	
API	
Performance vs TCP Socket	
Performance Evaluation	
WebSocket sequence diagram	
Network traffic	
Handshake overhead	
Frame overhead	
Results	
Data Transfer Time	
Connection	
Data	
Conclusion	

2024-06-03	WebSocket
	Performance vs TCP Socket
	Data Transfer Time
	Why WebSocket performs slower?

▶ WebSocket sits on top of TCP.
▶ WebSocket uses event-driven callback-based API to deliver data, which requires additional application data handling.

Why WebSocket performs slower?

Why WebSocket performs slower?

- ▶ WebSocket sits on top of TCP.
- ▶ WebSocket uses event-driven callback-based API to deliver data, which requires additional application data handling.
- ▶ In 2014, the WebSocket protocol emerged just a few years before with little production systems deployed. While TCP has been used in production for decades and has highly optimized libraries.

WebSocket

Oleg Bilovus

Background

HTTP polling
HTTP long polling
Streaming

WebSocket
protocol

Definition
Handshake
Upgrade Request
Upgrade Response
Frame
API

Performance vs
TCP Socket

Performance Evaluation
WebSocket sequence
diagram
Network traffic
Handshake overhead
Frame overhead
Results
Data Transfer Time
Connection
Data

Conclusion

WebSocket

Performance vs TCP Socket

Data Transfer Time

Why WebSocket performs slower?

2024-06-03

- ▶ WebSocket sits on top of TCP.
- ▶ WebSocket uses event-driven callback-based API to deliver data, which requires additional application data handling.
- ▶ In 2014, the WebSocket protocol emerged just a few years before with little production systems deployed. While TCP has been used in production for decades and has highly optimized libraries.

Why WebSocket performs slower?

Outline

Background

HTTP polling

HTTP long polling

Streaming

WebSocket protocol

Definition

Handshake

Frame

API

Performance vs TCP Socket

Performance Evaluation

WebSocket sequence diagram

Network traffic

Data Transfer Time

Conclusion

WebSocket
Oleg Bilovus

Background

HTTP polling

HTTP long polling

Streaming

WebSocket
protocol

Definition

Handshake

Upgrade Request

Upgrade Response

Frame

API

Performance vs
TCP Socket

Performance Evaluation

WebSocket sequence
diagram

Network traffic

Handshake overhead

Frame overhead

Results

Data Transfer Time

Connection

Data

Conclusion

WebSocket
Conclusion

2024-06-03

Outline

WebSocket

Conclusion

2024-06-03

WebSocket
Oleg Bilovus

Background

HTTP polling
HTTP long polling
Streaming

WebSocket protocol

Definition
Handshake
Upgrade Request
Upgrade Response
Frame
API

Performance vs TCP Socket

Performance Evaluation
WebSocket sequence diagram
Network traffic
Handshake overhead
Frame overhead
Results
Data Transfer Time
Connection
Data

Conclusion

- As expected, TCP slightly outperforms WebSocket because the WebSocket sits on top of it.

- ▶ As expected, TCP slightly outperforms WebSocket because the WebSocket sits on top of it.
- ▶ The amount of generated network traffic is almost the same.

WebSocket

Conclusion

2024-06-03

WebSocket
Oleg Bilovus

Background

HTTP polling
HTTP long polling
Streaming

WebSocket protocol

Definition
Handshake
Upgrade Request
Upgrade Response
Frame
API

Performance vs TCP Socket

Performance Evaluation
WebSocket sequence diagram
Network traffic
Handshake overhead
Frame overhead
Results
Data Transfer Time
Connection
Data

Conclusion

- ▶ As expected, TCP slightly outperforms WebSocket because the WebSocket sits on top of it.
- ▶ The amount of generated network traffic is almost the same.

- ▶ As expected, TCP slightly outperforms WebSocket because the WebSocket sits on top of it.
- ▶ The amount of generated network traffic is almost the same.
- ▶ The difference in data transfer time is remarkable.

WebSocket

Conclusion

2024-06-03

WebSocket
Oleg Bilovus

Background
HTTP polling
HTTP long polling
Streaming

WebSocket protocol
Definition
Handshake
Upgrade Request
Upgrade Response
Frame
API

Performance vs TCP Socket
Performance Evaluation
WebSocket sequence diagram
Network traffic
Handshake overhead
Frame overhead
Results
Data Transfer Time
Connection
Data

Conclusion

- ▶ As expected, TCP slightly outperforms WebSocket because the WebSocket sits on top of it.
- ▶ The amount of generated network traffic is almost the same.
- ▶ The difference in data transfer time is remarkable.

- ▶ As expected, TCP slightly outperforms WebSocket because the WebSocket sits on top of it.
- ▶ The amount of generated network traffic is almost the same.
- ▶ The difference in data transfer time is remarkable.
- ▶ The advantages of the WebSocket protocol is its alignment with the existing Web infrastructure, where low-level TCP protocol is not directly applicable.

WebSocket

Conclusion

2024-06-03

WebSocket
Oleg Bilovus

Background

HTTP polling
HTTP long polling
Streaming

WebSocket protocol

Definition
Handshake
Upgrade Request
Upgrade Response
Frame
API

Performance vs TCP Socket

Performance Evaluation
WebSocket sequence diagram
Network traffic
Handshake overhead
Frame overhead
Results
Data Transfer Time
Connection
Data

Conclusion

- ▶ As expected, TCP slightly outperforms WebSocket because the WebSocket sits on top of it.
- ▶ The amount of generated network traffic is almost the same.
- ▶ The difference in data transfer time is remarkable.
- ▶ The advantages of the WebSocket protocol is its alignment with the existing Web infrastructure, where low-level TCP protocol is not directly applicable.

References

-  Alexey Melnikov and Ian Fette, *The WebSocket Protocol*, RFC 6455, December 2011.
-  D. Skvorc, M. Horvat, and S. Srbljic, *Performance evaluation of websocket protocol for implementation of full-duplex web streams*, 2014 37th International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO), 2014, pp. 1003–1008.
-  Lijing Zhang and Xiaoxiao Shen, *Research and development of real-time monitoring system based on websocket technology*, Proceedings 2013 International Conference on Mechatronic Sciences, Electric Engineering and Computer (MEC), 2013, pp. 1955–1958.

WebSocket
Oleg Bilovus

Background

HTTP polling
HTTP long polling
Streaming

WebSocket protocol

Definition
Handshake
Upgrade Request
Upgrade Response
Frame
API

Performance vs TCP Socket

Performance Evaluation
WebSocket sequence diagram
Network traffic
Handshake overhead
Frame overhead
Results
Data Transfer Time
Connection
Data

Conclusion

WebSocket
Conclusion

2024-06-03

References

- References
-  Alexey Melnikov and Ian Fette, *The WebSocket Protocol*, RFC 6455, December 2011.
 -  D. Skvorc, M. Horvat, and S. Srbljic, *Performance evaluation of websocket protocol for implementation of full-duplex web streams*, 2014 37th International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO), 2014, pp. 1003–1008.
 -  Lijing Zhang and Xiaoxiao Shen, *Research and development of real-time monitoring system based on websocket technology*, Proceedings 2013 International Conference on Mechatronic Sciences, Electric Engineering and Computer (MEC), 2013, pp. 1955–1958.