dHealth 2024
D. Hayn et al. (Eds.)

doi:10.3233/SHTI240023

Comparison of WebSocket and Hypertext Transfer Protocol for Transfer of Electronic Health Records

Nikola KIRILOV[a,1]

[a] *Institute of Medical Informatics, Heidelberg University Hospital, Germany*

Abstract. Background: Electronic health records (EHR) emerged as a digital record of the data that is generated in the healthcare. Objectives: In this paper the transfer times of EHRs using the Hypertext Transfer Protocol and WebSocket in both local network and wide area network (WAN) are compared. Methods: A python web application to serve Fast Health Interoperability Resources (FHIR) records is created and the transfer times of the EHRs over both HTTP and WebSocket connection are measured. 45000 test Patient resources in 20, 50, 100 and 200 resources per Bundle transfers are used. Results: WebSocket showed much better transfer times of large amount of data. These were 18 s shorter in the local network and 342 s shorter in WAN for the 20 resource per Bundle transfer. Conclusion: RESTful APIs are a convenient way to implement EHR servers; on the other hand, HTTP becomes a bottleneck when transferring large amount of data. WebSocket shows better transfer times and thus its superiority in such situations. The problem can be addressed by developing a new communication protocol or by using network tunneling to handle large data transfer of EHRs.

Keywords. electronic health record, fhir, http, websocket, rest

1. Introduction

Electronic health records (EHR) emerged as a digital record of the data that is generated in the healthcare. They offer interoperability and immediate access to important information [1]. Nowadays EHRs are based on web servers and use databases, which makes transferring medical information from one facility to another easier and furthers the creation of health information exchange networks among medical organizations [2].

The recently developed Fast Health Interoperability Resources (FHIR) standard is gaining popularity and adoption by companies and institutions. The architecture of FHIR is organized by resources, which are predefined profiles of concepts in healthcare (e. g. Patient, Encounter, Condition, Procedure etc.) [3]. This makes the representational state transfer (REST) protocol suitable for the implementation of their application programming interfaces (APIs). The APIs are based on the Hypertext Transfer Protocol (HTTP), although in the Roy Fielding's dissertation HTTP was never mentioned as the only protocol for REST. He also mentioned the inefficiency of HTTP

[1] Corresponding Author: Nikola Kirilov, Institute of Medical Informatics, Heidelberg University Hospital, Germany, E-Mail: Nikola.Kirilov@med.uni-heidelberg.de

concerning its single request/response per connection behavior [4]. When requesting FHIR resources from the server these are usually provided in Bundles [5]. The number of resources per Bundle is limited (by default 20) by the server and in order to retrieve large number of resources, multiple requests need to be made. This makes such transactions time consuming.
WebSockets have been recently introduced to bring real-time capabilities to the web and to solve multiple problems [6]. The connection is established using HTTP GET request with an upgrade header and allows long-lived exchange of messages [7]. HTTP has already been identified as a bottleneck when transferring large data using REST [8].

2. Methods

To compare the transfer time of the EHRs from the server using HTTP requests and the transfer time of the same data using a WebSocket connection a python web application is created. The application was implemented using the Hypercorn library serving the resources from a HL7 application programming interface (HAPI) PostgreSQL database with the psycopg driver. The Hypercorn library offers a support for WebSocket connection over HTTP. The execution time of the database queries was similar for both HAPI FHIR and the test python implementation.
The test dataset consisted of 45000 test FHIR Patient resources. The 45000 resources were transfered using 20, 50, 100 and 200 resources per Bundle, which resulted in 2250, 900, 450 and 225 HTTP requests and WebSocket messages respectively. The test was carried out while the client was connected to the local network and then while the client was connected to the wide area network (WAN) (Figure 1).
Asynchronous JavaScript And XML was used for the execution of the HTTP requests and JavaScript for the WebSocket messages. Requests and messages were sent serially to avoid server performance issues which might alter the results. The total execution time was logged using the HTTP Archive files for each request.

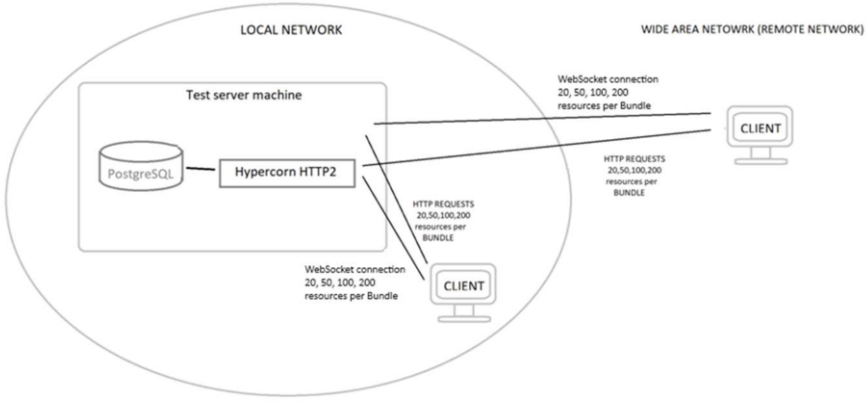

Figure 1. Graphical depiction of the test setup: server – client. Requests/Messages from WAN and in the local network.

3. Results

In the first test where the resources were transferred in the local network using HTTP requests there was a reduction of the transfer time as the number of resources per Bundle increased starting from 56.88 s for 20 resources per Bundle and dropping to 10.42 s for 200 resources per Bundle. Using WebSocket messages 20 resources per Bundle were transferred in 38.03 s and 200 resources per Bundle for 7.33 s. (Table 1, Figure 2).

Table 1. Transfer time of EHR resources using HTTP requests and WebSocket messages in the local network for 20, 50, 100 and 200 resources per bundle.

resources per Bundle	HTTP Requests	WebSocket Messages
20	56.78 s	38.03 s
50	37.95 s	17.89 s
100	18.24 s	10.80 s
200	10.42 s	7.33 s

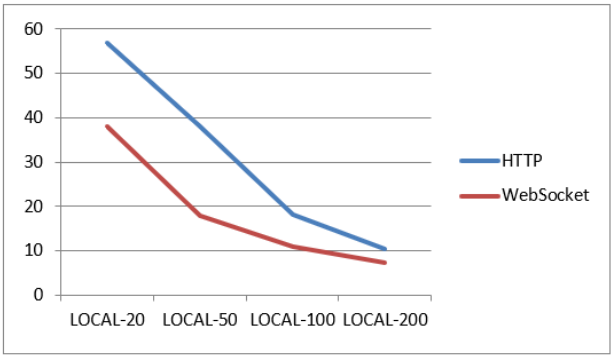

Figure 2. Transfer time of the EHR resources HTTP against WebsSocket in the local network.

The second test was carried out with a connection of the client from the WAN. The same reduction of transfer time was observed as in the local network starting from 509.51 s for 20 resources per Bundle and dropping to 60.22 s for 200 resources per Bundle using HTTP Requests. The observed transfer times using WebSocket messages started from 167.14 s for 20 resources per Bundle and fell to 24.22 s for 200 resources per Bundle. (Table 2, Figure 3).

Table 2. Transfer time of EHR resources using HTTP requests and WebSocket messages in the remote network for 20, 50, 100 and 200 resources per bundle.

resources per Bundle	HTTP Requests	WebSocket Messages
20	509.51 s	167.14 s
50	205.35 s	74.54 s
100	103.88 s	39.15 s
200	60.22 s	24.22 s

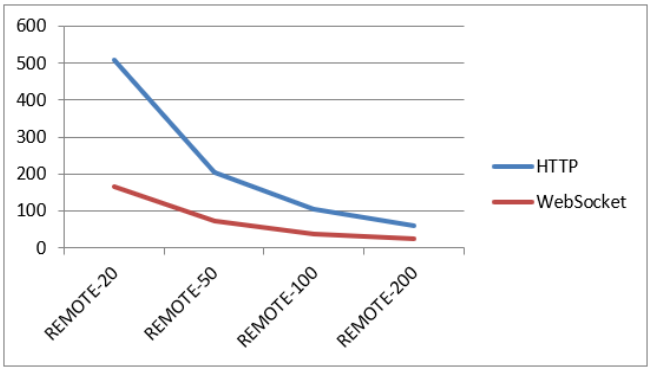

Figure 3. Transfer time of the EHR resources HTTP against WebSocket in the WAN.

An example related to the healthcare domain could be given when transferring data between two healthcare institutions. Each institution has a private network and the data needs to pass across the WAN. A possible solution to the problem where the better performance of WebSocket could be taken advantage of and the HTTP connection protocol could still be utilized is to use tunneling. Tunneling is the process of transmitting data across the network by encapsulating a packet and protocol into another packet with different protocol. A proxy server could be placed in the demilitarized zone (DMZ) of the private network of the healthcare institution where the FHIR server storing the data is located. This proxy can be used to tunnel the HTTP requests over WebSocket. A WebSocket client located in the second institution establishes a secure connection over the WAN and requests the data over this WebSocket connection tunneling HTTP. (Figure. 4)

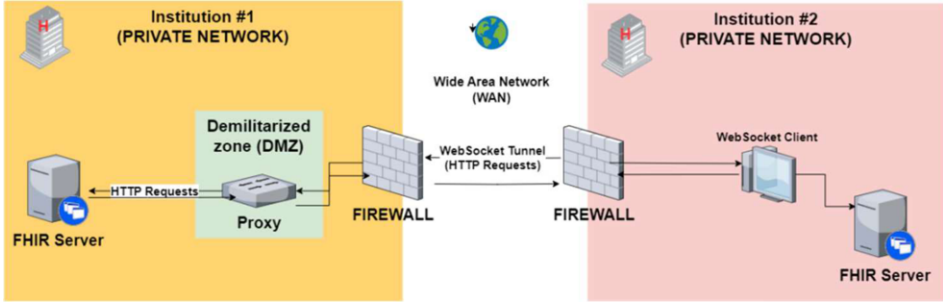

Figure 4. Clinical example: transfer of EHR resources using WebSocket tunnel.

4. Discussion

The results show that the most important factor for the transfer time of the data is the number of resources per Bundle. Increasing this number leads to reduction of the requests or messages needed. The requests needed to transfer 45000 Resources using 20 resources per Bundle are 2250 and for 200 resources per Bundle these are 225. With the HTTP connection the client makes a request and after the server response the

connection is closed. This process is repeated for each Bundle until all data is received. In the local network the transfer time decreases almost fivefold as the resources per Bundle increase to 200. In the remote network where each HTTP connection suffers from high-latency this decrease is over 9 times. The use of WebSocket connection improves the transfer times dramatically. The duration is 18 s shorter in the local network and the staggering 342 s shorter in the WAN for the 20 resources per Bundle transfer. The difference is becoming less prominent with the reduction in the number of requests. In our work we do not consider the time needed for establishing the WebSocket connection, which starts with an HTTP request with an upgrade header. This could be considered as a limitation of the work.

5. Conclusion

HTTP-based RESTful APIs are a convenient way to implement EHR servers; on the other hand, HTTP becomes a bottleneck when transferring large amount of data. WebSocket shows much better transfer times and thus its superiority in such situations. This problem can be addressed by developing a new communication protocol or by using network tunneling to handle large data transfer of EHRs.

References

[1] Cowie MR, Blomster JI, Curtis LH, Duclaux S, Ford I, Fritz F, Goldman S, Janmohamed S, Kreuzer J, Leenay M, Michel A, Ong S, Pell JP, Southworth MR, Stough WG, Thoenes M, Zannad F, Zalewski A. Electronic health records to facilitate clinical research. Clin Res Cardiol. 2017 Jan;106(1):1-9. doi: 10.1007/s00392-016-1025-6.

[2] Evans RS. Electronic Health Records: Then, Now, and in the Future. Yearb Med Inform. 2016 May 20;Suppl 1(Suppl 1):S48-61. doi: 10.15265/IYS-2016-s006. PMID: 27199197; PMCID: PMC5171496

[3] Ayaz M, Pasha MF, Alzahrani MY, Budiarto R, Stiawan D. The Fast Health Interoperability Resources (FHIR) Standard: Systematic Literature Review of Implementations, Applications, Challenges and Opportunities. JMIR Med Inform. 2021 Jul 30;9(7):e21929. doi: 10.2196/21929. Erratum in: JMIR Med Inform. 2021 Aug 17;9(8):e32869. PMID: 34328424; PMCID: PMC8367140.

[4] Fielding, Roy Thomas (2000). "Architectural Styles and the Design of Network-based Software Architectures". Dissertation. University of California, Irvine..

[5] https://build.fhir.org/bundle.html

[6] V. Wang, F. Salim, and P. Moskovits, The Definitive Guide to HTML5 Websocket. New York, NY: Apress, 2013.

[7] 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), Opatija, Croatia, 2014, pp. 1003-1008, doi: 10.1109/MIPRO.2014.6859715.

[8] R. K. L. Ko, M. Kirchberg, B. S. Lee and E. Chew, "Overcoming Large Data Transfer Bottlenecks in RESTful Service Orchestrations," 2012 IEEE 19th International Conference on Web Services, Honolulu, HI, USA, 2012, pp. 654-656, doi: 10.1109/ICWS.2012.107.