# Penetration Testing of HTTP/3.0 Servers

Shaan Kumar, *19074015*,
Naresh Kumar, *19075047*,
Nikita, *19075048*

*Under the guidance of*
**Dr. Mayank Swarnkar, Assistant Professor**,
Department of Computer Science and Engineering
Indian Institute of Technology (BHU), Varanasi

November 22, 2022

# Table of Contents



## HTTP Protocols

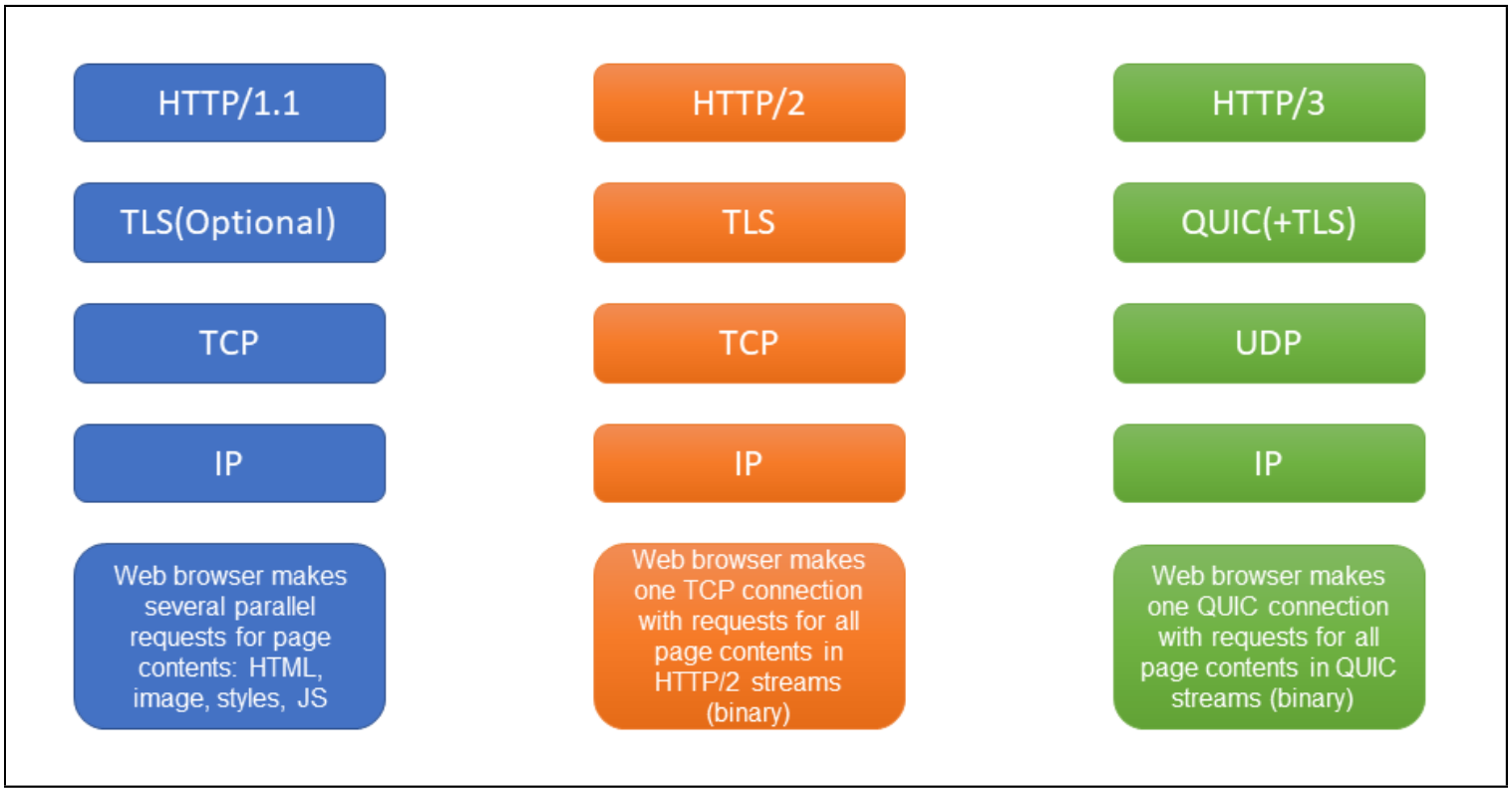



Figure: 1. HTTP Versions

# Introduction

## QUIC Transport Protocol

- fast, secure, evolvable generic transport protocol
- integrates with TLS
- uses connection IDs  frames
- reduced connection establishment time
- supports multiplexing
- solves head-of-line blocking problem in HTTP/2

# Literature Survey

| **Author et al** | **Proposed Work** | **Shortcoming of work** |
|---|---|---|
| Xudong Cao, Shangru Zhao, Yuqing Zhang, 2019 [1] | 0-RTT Attack and Defense of QUIC Protocol | Tests the security mechanism of QUIC, & proposes a new attack against the protocol, proves feasibility of attack through experiments. |
| Adam Langley, Alistair Riddoch, Alyssa Wilk, 2017 [3] | The QUIC Transport Protocol: Design and Internet-Scale Deployment | Layering enables modularity but often at the cost of performance. Squashing the layers of HTTPS in QUIC allows to weed out inefficiencies in the HTTPS stack. |
| Efstratios Chatzoglou1, Vasileios Kouliaridis1, Georgios Karopoulos2 Georgios Kambourakisa, 2015 [4] | Revisiting QUIC attacks: A comprehensive review on QUIC security and a hands-on study Quick is QUIC? | 1. In presence of attackers, QUIC maybe unable to attain 0-RTT connections. 2. Analysed the pitfalls of designing performance-driven secure protocols. |
| Robert Lychev, Samuel Jero, Alexandra Boldyreva, 2015 [4] | How Secure and Quick is QUIC? Provable Security and Performance Analyses | 1. In presence of attackers, QUIC maybe unable to attain 0-RTT connections. 2. Analysed the pitfalls of designing performance-driven secure protocols. |
| Igor Nogueira de Oliveira, Rafael Roque Aschoff, 2018 [6] | QUIC and TCP: A Performance Evaluation | Influence of RTT in the experiment was noticeable while packet loss ratio influence was inexpressive. |

# Literature Survey

| **Author et al** | **Proposed Work** | **Shortcoming of work** |
|---|---|---|
| Robin Marx, Joris Herbots Wim Lamotte, Peter Quax, 2020 [5] | Same Standards, Different Decisions: A Study of QUIC and HTTP/3 Implementation Diversity | Analysed behaviour of 15 different QUIC implementations based on features such as Flow Control, Congestion Control, Prioritization and 0-RTT etc |
| Mehdi Yosofie, Benedikt Jaeger, 2019 | Recent Progress on the QUIC Protocol | Discussed testing QUIC in production mode within Chrome/ Chromium on YouTube and other Google services by Google. |
| Sarah Cook, Bertrand Mathieu, Patrick Truong, 2017 [2] | QUIC: Better For What And For Whom? | QUIC outperforms HTTP/2 over TCP/TLS in unstable networks such as wireless mobile networks. |

# Motivation

- HTTP/3 a very recent HTTP protocol, eyecandy for hackers.
- HTTP/3 not allowed by many networks.
- Adoption of HTTP/3 is increasing so deciding which HTTP/3 implementation is better for use maybe helpful.
- To analyze features of HTTP/3 servers : faster connection set-up, less Head of Line blocking, connection migration.

# Work Done

- Learned about HTTP protocols:.
  - **HTTP/1.1**: Web browser makes several parallel requests for page contents: HTML, image, styles, JS.
  - **HTTP/2**: Web browser makes one TCP connection with requests for all page contents in HTTP/2 streams (binary). Head of Line blocking issue exists in HTTP/2.
  - **HTTP/3**: Web browser makes one QUIC connection with requests for all page contents in QUIC streams (binary).
- Read about the **Head of Line blocking** problem in HTTP/2.
- Studied behaviour of different QUIC supporting servers with packet loss, delay, network change and compared QUIC with earlier versions.

# Work Done

**Tools Used:**

- **Scapy**: Scapy is a powerful interactive packet manipulation program. Used this to manipulate the data of packets which were sent to the QUIC test servers.
- **Wireshark**: Wireshark is a free and open-source packet analyzer. Used this to track and analyze the packets sent to the QUIC test servers.

# Work Done

- Penetration testing performed to QUIC servers are as:

| Experiment Number | Parameters Changed |
|---|---|
| 1 | Reduce header length of UDP layer |
| 2 | Changed frag flag of IP layer |
| 3 | Changed the flag of IP layer and dport in UDP layer |
| 4 | Changed IP Version and header length in IP version |
| 5 | Changed ihl variable in IP layer, Reduced header length |
| | in UDP layer and changed IP's id |

# Work Done

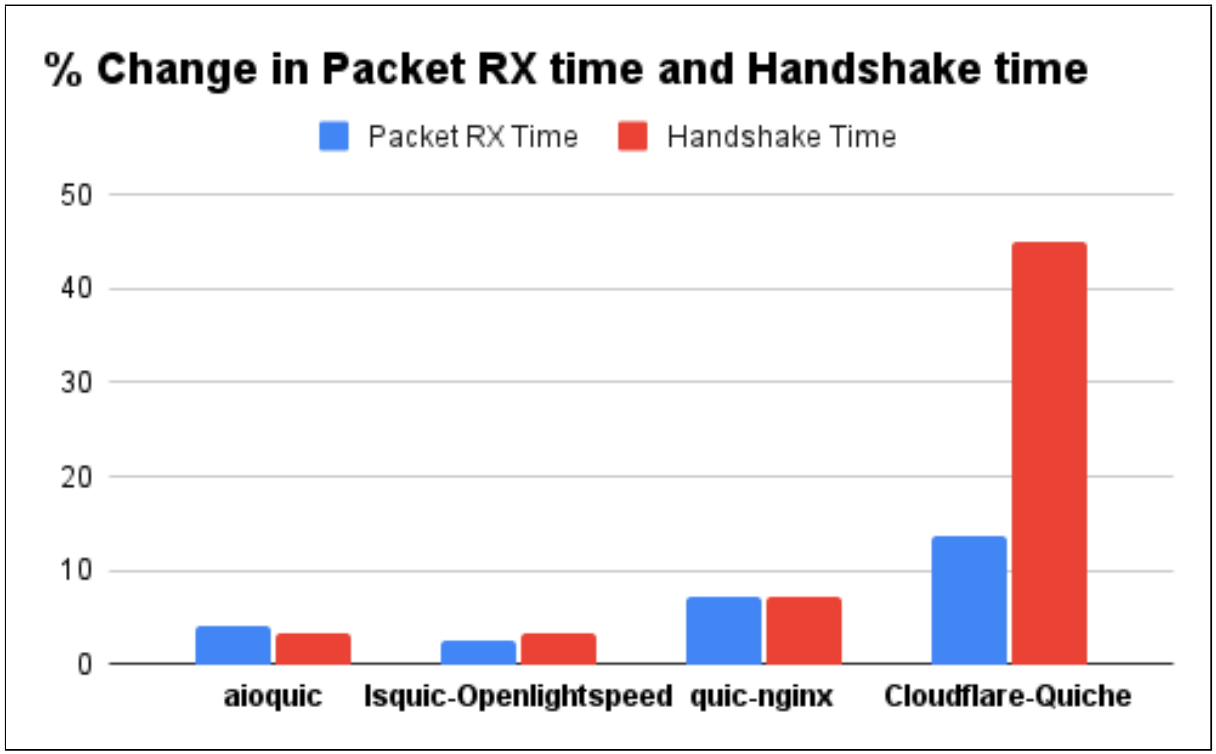
**Steps for performing experiment:**

- Start by taking a QUIC packet.
- Modify the data in packet using Scapy.
- Flood the test server by sending this QUIC packet.
- Measure PACKET RX time and HANDSHAKE time with the help of `http3check.net`
  - **PACKET RX** - Time between the first packet sent and the first packet received (measured in milliseconds).
  - **HANDSHAKE TIME** - Time between when the first packet is sent and when the handshake is completed (measured in milliseconds).

# Work Done

**Experiment 1**: Reduce header length of UDP layer.

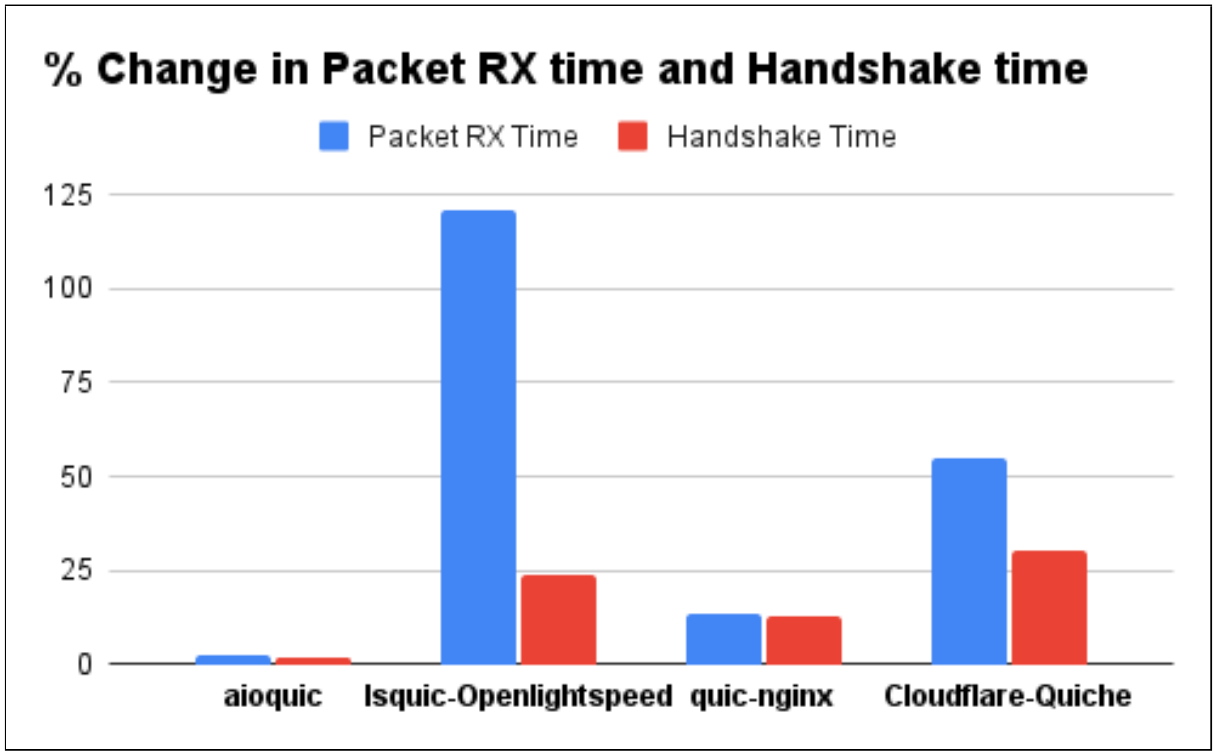
| TYPE | aioquic | lsquic-Openlightspeed | quic-nginx | Cloudflare-Quiche |
|---|---|---|---|---|
| PACKET RX | 185.07 | 8.89 | 171.08 | 7.43 |
| HANDSHAKE | 92.79 | 7.77 | 84.34 | 4.75 |



# Work Done

**Experiment 2**: Changed frag flag of IP layer.

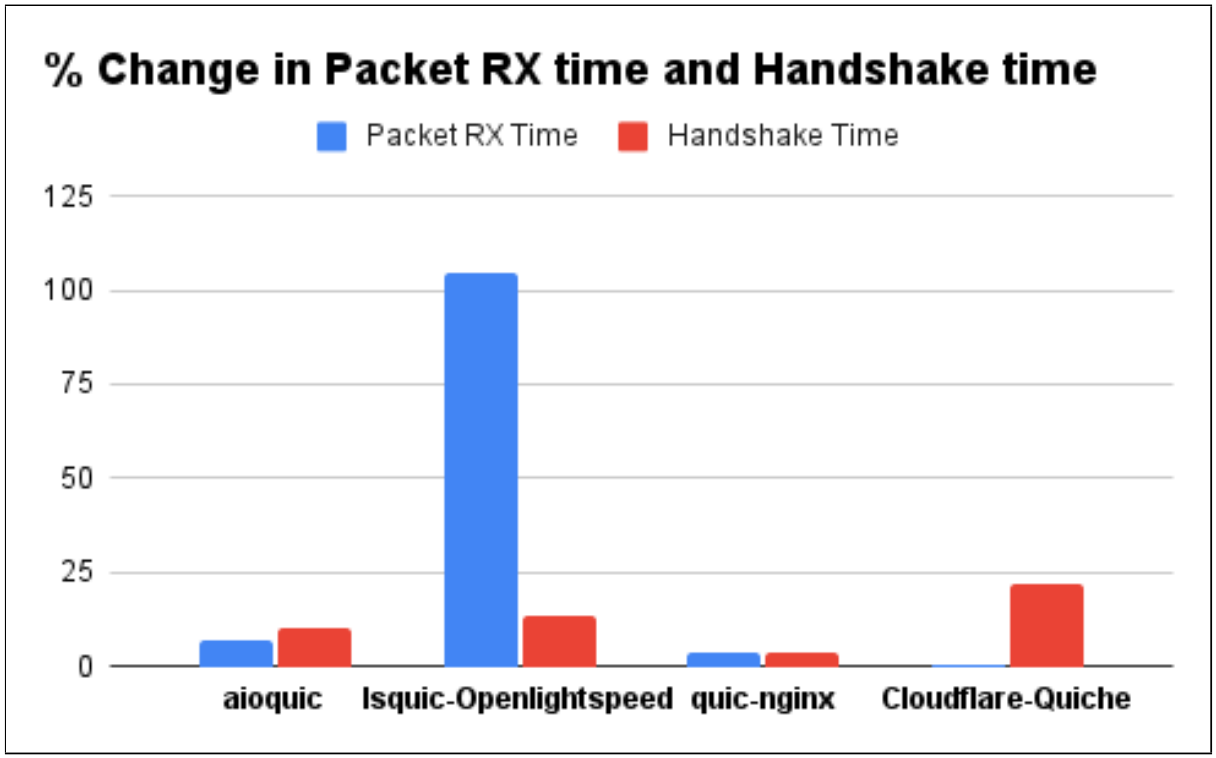
| TYPE | aioquic | lsquic-Openlightspeed | quic-nginx | Cloudflare-Quiche |
|---|---|---|---|---|
| PACKET RX | 182.25 | 20.16 | 180.58 | 10.10 |
| HANDSHAKE | 91.54 | 9.93 | 88.83 | 4.27 |



# Work Done

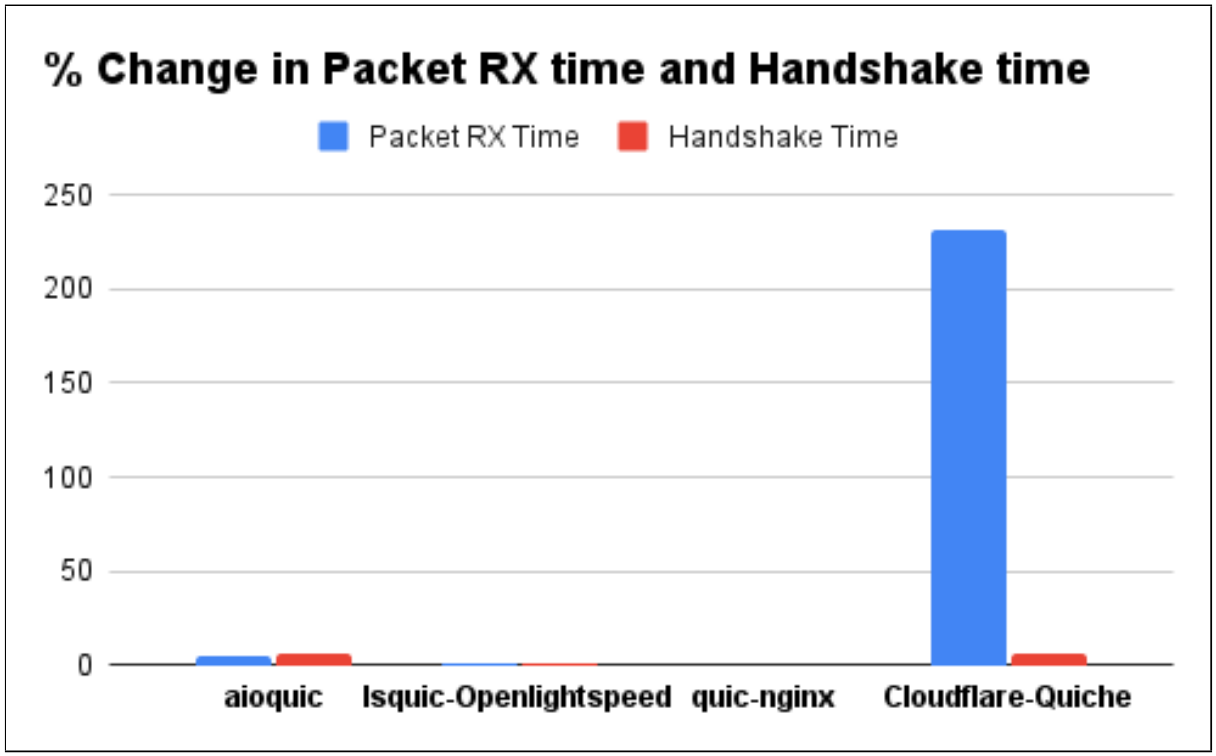
**Experiment 3**: Changed the flag of IP layer and dport in UDP layer.

| TYPE | aioquic | lsquic-Openlightspeed | quic-nginx | Cloudflare-Quiche |
|---|---|---|---|---|
| PACKET RX | 190.50 | 18.62 | 165.13 | 6.49 |
| HANDSHAKE | 99.40 | 9.14 | 81.45 | 2.55 |



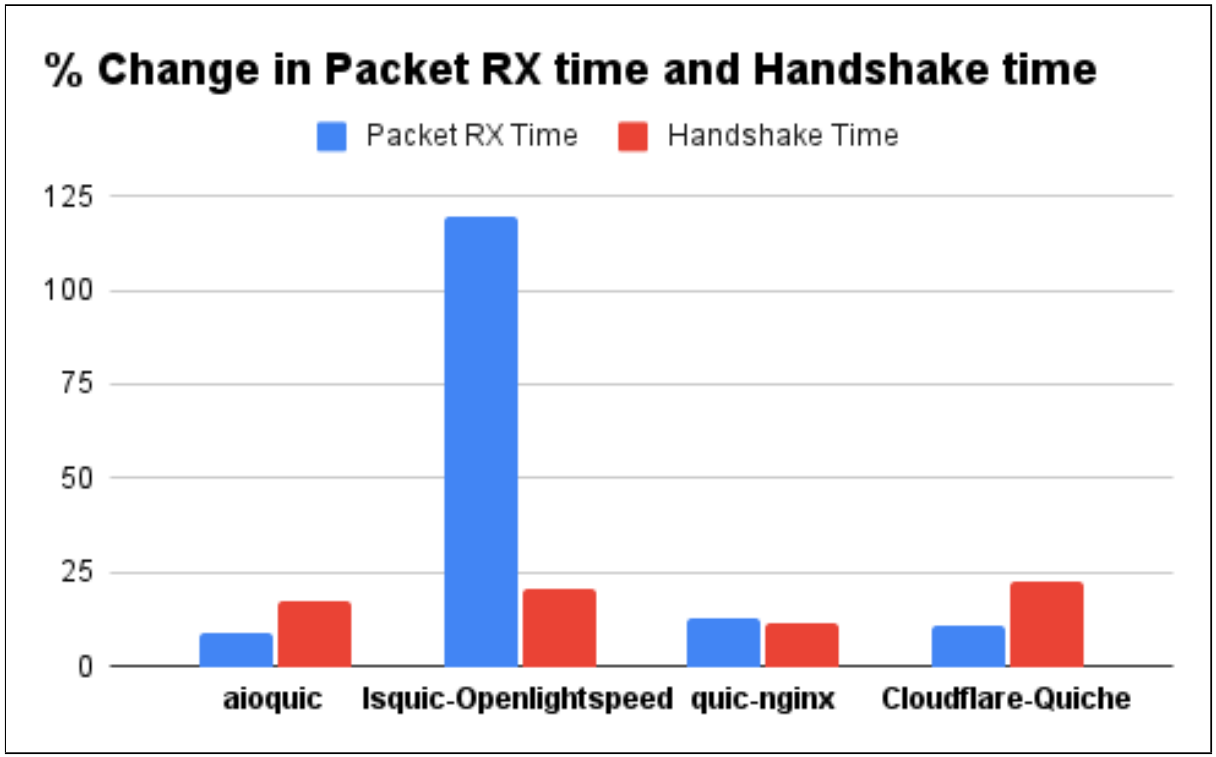
**Experiment 4**: Changed IP Version and header length in IP version.

| TYPE | aioquic | lsquic-Openlightspeed | quic-nginx | Cloudflare-Quiche |
|---|---|---|---|---|
| PACKET RX | 187.02 | 9.04 | 159.58 | 21.65 |
| HANDSHAKE | 95.57 | 7.93 | 78.54 | 3.48 |



**Experiment 5**: Changed ihl variable in IP layer, Reduced header length in UDP layer and changed IP's id.

| TYPE | aioquic | lsquic-Openlightspeed | quic-nginx | Cloudflare-Quiche |
|---|---|---|---|---|
| PACKET RX | 193.52 | 20.02 | 179.86 | 5.84 |
| HANDSHAKE | 105.38 | 9.69 | 87.65 | 2.53 |



# Proposed Work

- To try and compare implementations of QUIC supporting servers like **aioquic, nginx-quiche, openlitespeed, Cloudflare**.
- To study the behaviour of various servers implementing QUIC protocol by sending forgery packets.
- To compare different servers on the basis of performance i.e. HANDSHAKE TIME and PACKET RX time in handling forgery packets.

# Experimental Setup

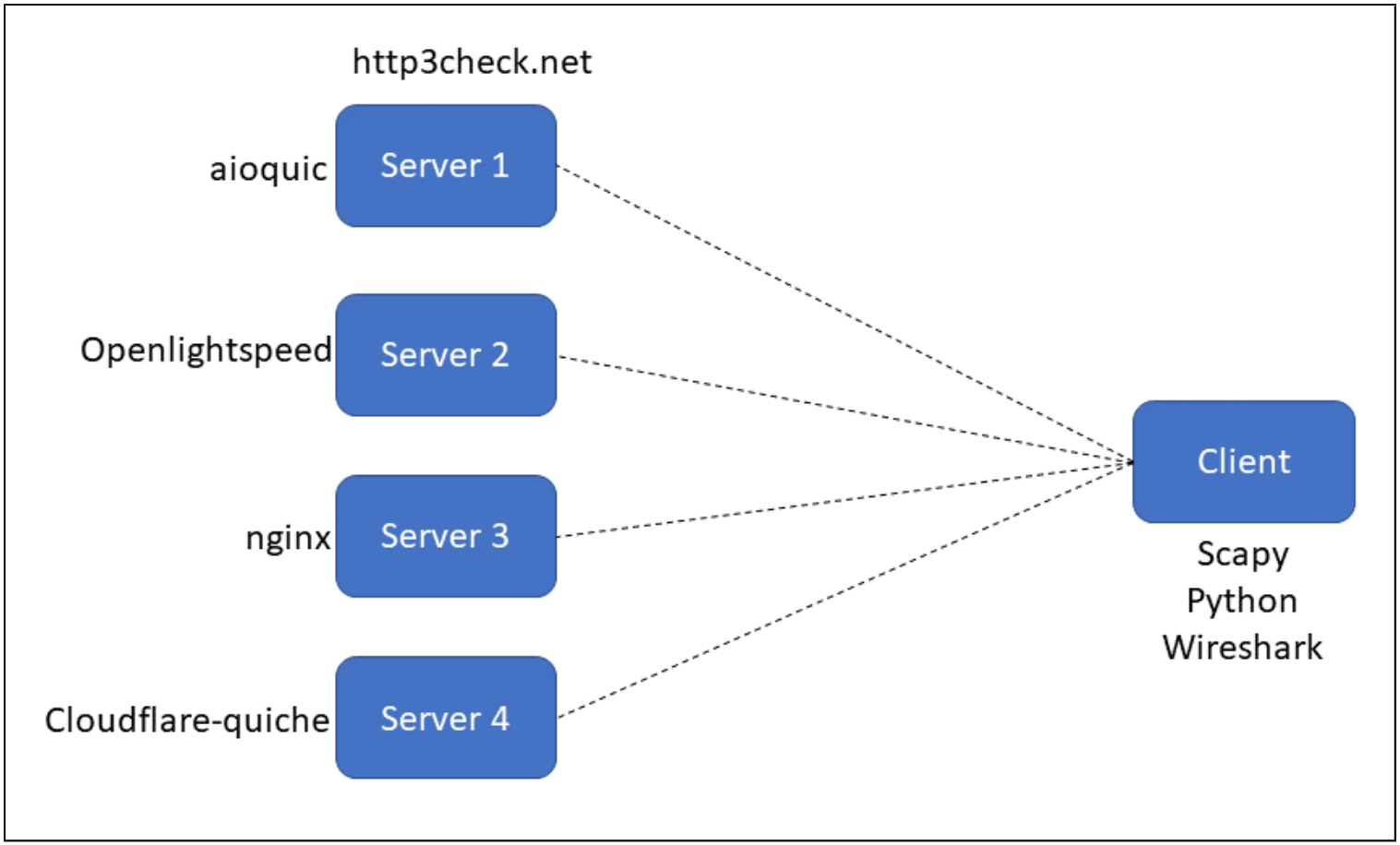



Figure: 2. Setup for Penetration Testing

# Experimental Results

## HANDSHAKE Time:

| Server name | Normal | No change | Test1 | Test2 | Test3 | Test4 | Test5 |
|---|---|---|---|---|---|---|---|
| aioquic | 89.9 | 92.701 | 92.794 | 91.548 | 99.401 | 95.574 | 105.384 |
| Isquic-Openlightspeed | 8.037 | 10.202 | 7.776 | 9.931 | 9.141 | 7.934 | 9.697 |
| Quic-nginx | 78.673 | 81.089 | 84.344 | 88.836 | 81.459 | 78.548 | 87.652 |
| Cloudflare-Quiche | 3.283 | 3.205 | 4.758 | 4.27 | 2.557 | 3.488 | 2.533 |

Table: Handshake Time for servers on different tests

| Server name | No change | Test1 | Test2 | Test3 | Test4 | Test5 |
|---|---|---|---|---|---|---|
| aioquic | 3.115 | 3.219 | 1.833 | 10.568 | 6.311 | 17.223 |
| Isquic-Openlightspeed | 26.937 | 3.247 | 23.566 | 13.736 | 1.281 | 20.654 |
| Quic-nginx | 3.070 | 7.208 | 12.918 | 3.541 | 0.158 | 11.413 |
| Cloudflare-Quiche | 2.375 | 44.928 | 30.063 | 22.113 | 6.244 | 22.844 |

Table: Percentage change in Handshake time

# Experimental Results

## PACKET RX Time:

| Server name | Normal | No change | Test1 | Test2 | Test3 | Test4 | Test5 |
|---|---|---|---|---|---|---|---|
| aioquic | 177.846 | 184.673 | 185.077 | 182.25 | 190.506 | 187.027 | 193.529 |
| Isquic-Openlightspeed | 9.118 | 19.862 | 8.894 | 20.163 | 18.627 | 9.046 | 20.023 |
| Quic-nginx | 159.575 | 164.084 | 171.081 | 180.58 | 165.136 | 159.584 | 179.861 |
| Cloudflare-Quiche | 6.54 | 8.061 | 7.431 | 10.108 | 6.499 | 21.65 | 5.847 |

Table: Packet RX Time for servers on different tests

| Server name | No change | Test1 | Test2 | Test3 | Test4 | Test5 |
|---|---|---|---|---|---|---|
| aioquic | 3.838 | 4.065 | 2.476 | 7.118 | 5.162 | 8.818 |
| Isquic-Openlightspeed | 117.832 | 2.456 | 121.134 | 104.288 | 0.789 | 119.598 |
| Quic-nginx | 2.825 | 7.210 | 13.163 | 3.484 | 0.005 | 12.712 |
| Cloudflare-Quiche | 23.256 | 13.623 | 54.556 | 0.626 | 231.039 | 10.596 |

Table: Percentage change in Packet RX Time

# Experimental Results

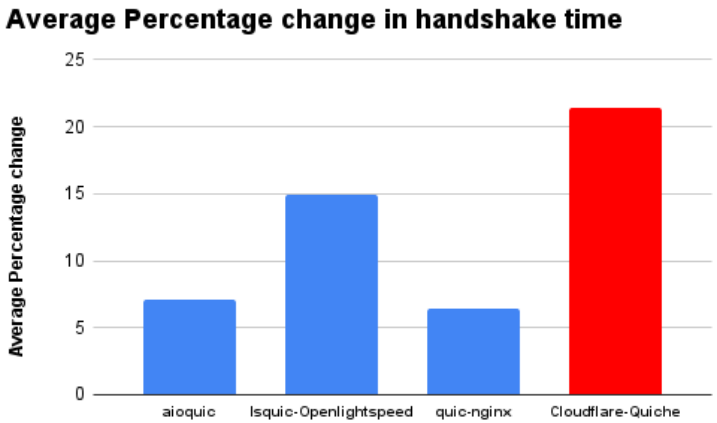



Figure: Average Handshake Time Graph

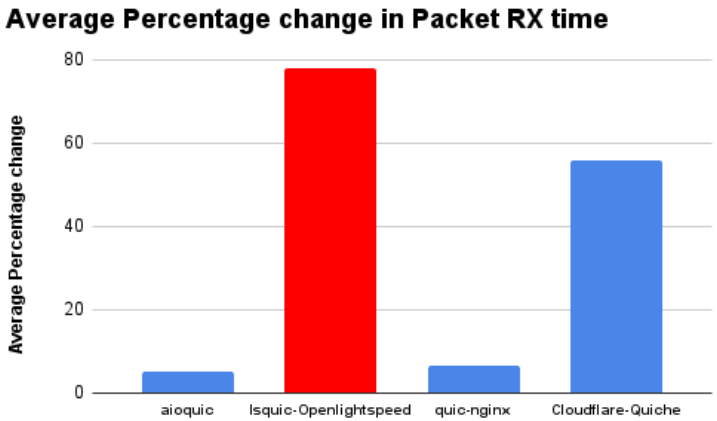



Figure: Average Packet RX Time Graph

# Conclusion

- HTTP/3 is basically a improvement in terms of performance, security and not a reform.
- Main issue with QUIC lies in the initial phase of handshake because the client hello packet is unencrypted.
- Our experiment of pentesting shows that the order for using these servers is:
  - On the basis of Handshake time: **nginx** > **aioquic** > **openlightspeed** > **cloudflare**
  - On the basis of Packet RX time: **aioquic** > **nginx** > **cloudflare** > **openlightspeed**

# References


[1] Xudong Cao, Shangru Zhao, and Yuqing Zhang. "0-RTT Attack and Defense of QUIC Protocol". In: *2019 IEEE Globecom Workshops (GC Wkshps)*. Dec. 2019, pp. 1–6. DOI: 10.1109/GCWkshps45667.2019.9024637.

[2] Sarah Cook et al. "QUIC: Better for what and for whom?" In: May 2017, pp. 1–6. DOI: 10.1109/ICC.2017.7997281.

[3] Adam Langley et al. "The QUIC Transport Protocol: Design and Internet-Scale Deployment". In: Aug. 2017, pp. 183–196. DOI: 10.1145/3098822.3098842.

[4] Robert Lychev et al. "How Secure and Quick is QUIC? Provable Security and Performance Analyses". In: *2015 IEEE Symposium on Security and Privacy*. 2015, pp. 214–231. DOI: 10.1109/SP.2015.21.

[5] Robin Marx et al. "Same Standards, Different Decisions: A Study of QUIC and HTTP/3 Implementation Diversity". In: Aug. 2020. DOI: 10.1145/3405796.3405828.


# Thank You!