An Empirical Evaluation of Video Conferencing Systems Used in Industry, Academia and Entertainment in 2020

Jim Cuijpers*
j.j.c.j.cuijpers@vu.nl

Kelvin Elsendoorn*
k.elsendoorn@student.vu.nl

Ean-Dan Tjon-Joek-Tjien*
e.tjonjoektjien@student.vu

Riccardo Iesari*
r.iesari@student.vu.nl

Federico Casenove*
f.casenove@student.vu.nl

Jesse Donkervliet
Support cast
j.j.r.donkervliet@vu.nl

Alexandru Iosup
Support cast
a.iosup@vu.nl

* These authors have contributed equally to this paper.

***Abstract*—Video Conferencing Systems have become an important part in our everyday activities. We use it for our team meetings at work, attending lectures for our studies, even for get-togethers with friends to play an online video game. The Corona pandemic of 2020 has accelerated the use of these systems, but how much do we really know about these systems, and how much do we know about the behavior of these systems in different environments? In this paper we design and conduct real-world experiments measuring system and network utilization for different video conferencing scenarios that can occur. We find that there are indeed significant differences between these systems and their behavior in different environments, one such example is the significantly higher CPU utilization of Discord compared to Zoom and Microsoft Teams.**

I. Introduction

Video conferencing software is at the time of writing one of the most important tools to upkeep communication between people in both industry and academia. In a survey performed by Forbes Insight in 2017 80% of the respondents say that video conferencing is fast becoming the norm for internal teams at Microsoft [1]. A Gartner HR Survey in 2020 reports that 86% of the recruitment interviews are done over video call [2], and in a recent survey by Zoom they indicate that 88% of the respondents believe that video conferencing software will help more people get advanced degrees and lower the dropout rate by 73% [3]. Additionally, Microsoft CEO Satya Nadella reported in 2020 that Microsoft Teams usage has increased to more than 75 million daily active users during the Covid-19 pandemic [4]. However, we do not understand how these video conferencing systems behave in the real world.

Nowadays some of the most famous and used Video Conferencing Tools are: Zoom, Cisco Webex, Google Meet, Skype Meet and Microsoft (MS) Teams. All of these systems offer different features and every one of these systems has a premium plan with different costs, which allows subscribers to access more features, generally useful in the *business world* [5].

Performance evaluation of existing video conferencing software has been researched from different perspectives, Jansen et al. [6] have studied the performance of WebRTC based video conferencing with the main focus being on the Google Congestion Control (GCC) algorithm. Results of their experiments have shown that WebRTC is sensitive to variation in the round-trip time (RTT) of packets and packet losses. Bursty packet losses and retransmissions over long RTTs lead to poor video performance. Vasconcelos et al. [7] have studied different virtualization platforms to deploy conference systems. They have evaluated its virtual performance under a real-world workload and concluded that video conferencing software are well suited for virtualization platforms. Schreiber, Joopari, and Rashid [8] evaluated the performance of video conferencing over Asynchronous Transfer Mode (ATM) and Gigabit Ethernet, showing that Gigabit Ethernet outperforms ATM significantly. Meanwhile, Xiong et al. [9] have designed and evaluated a real-time video conferencing environment to support teaching and focuses more on the possibilities of video conferencing software from a societal perspective. It becomes apparent that research has been done in the underlying technologies of such systems, or their applications. However, performance evaluations of contemporary (at the time of writing) video conferencing systems such as Zoom or Google Meet have not been researched.

Now that video conferencing becomes increasingly apparent in everyday activities it is important that everyone has access to this. Not everyone has the same personal computer and not everyone enjoys an internet infrastructure that provides fast internet speeds as they have in Taiwan or Singapore [10]. Therefore, it is important that we study the CPU utilization and bandwidth usage, and the behavior of these systems when connected to different internet technologies. Video Conferencing Systems are often used to conduct meetings with one other person, or with larger groups. We refer to meetings between two people as one-on-one video sessions.

In this paper, we design and conduct experiments to evaluate contemporary video conferencing systems under different real-world scenarios. Such experiments include the evaluation of bandwidth and CPU utilization in one-on-one

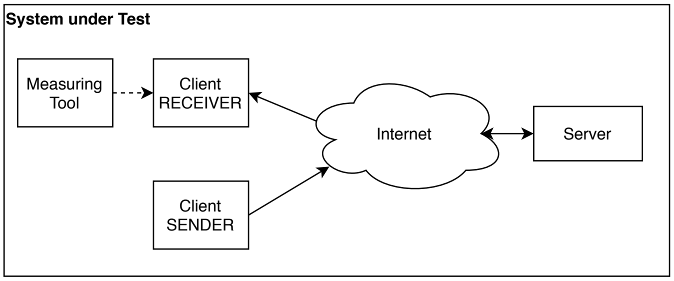

Fig. 1. Overview of the system under test.

video sessions and their performance under sessions with more people involved, like for example, university lectures. Additionally, we evaluate the performance of these systems when they are used with different network technologies such as Gigabit Ethernet, 2.4GHz Wi-Fi and 5GHz Wi-Fi.

The main contributions of this paper are:

1) The design and implementation of a conceptual tool that conducts experiments measuring system and network related metrics. (Section III)
2) A set of real world experiments targeting CPU utilization and metrics associated with network traffic. (Section IV)
3) An empirical evaluation of different Video Conferencing Software in different scenarios and an evaluation on the behavior of such systems on different types of networks (Ethernet vs. Wi-Fi 4 vs. Wi-Fi 5). (Section V)
4) An in-depth discussion on the results of our conceptual experiments. (Section VI)

II. Background

During the Covid-19 pandemic the usage of Video Conferencing Systems has increased drastically. Video Conferencing Systems allows two or more individuals to interact with each other using audio and video. Generally, these individuals are geographically distributed. Those systems provide an infrastructure to enable this form of communication and a client application to access it, and in some cases the possibility to access this technology by web. Between them we have Zoom, MS Teams and Discord.

Zoom is a cloud-based video conferencing service that enables meeting by video, audio and a live chat. It is also possible to record the meeting, screen-sharing and much more. It allows large groups of people in a single meeting and users do not need an account to attend this meeting, these features made Zoom, during the pandemic, an ideal platform for university lectures, business meetings, and entertainment. In fact, a lot of artists, mainly musicians, performed live concerts for crowdfunding and also organized online party to cheer up people facing a lockdown [11]. In addition, Zoom is a cloud platform and differentiates itself by its connection process. Whenever someone attempts to access the Zoom platform there is an optimized path to Zoom's geographically distributed infrastructure. First, the Zoom client contacts the Zoom web infrastructure to receive the metadata required to access the meeting. The Zoom web infrastructure sends back a packet with the optimal Zoom Meeting Zones. Second, the Zoom client performs a network test with every Zoom meeting zone. After having found the optimal zone, the client requests for the optimal media router and uses this to connect to the session. Finally, Zoom creates a connection for each type of media and attempts to use Zoom's own protocol [12].

MS Teams is part of the Microsoft Office 365 suite. It is cloud-based and this video conferencing software presents a lot of features designed to facilitate team collaboration. MS Teams makes use of peer-to-peer media communication. It does this by using standard IETF Interactive Connectivity Establishment (ICE) procedures. When two people want to connect to each other using MS Teams, they have to connect to the audio/video/screen sharing (VBSS) conferencing server which is part of Microsoft 365. STUN connectivity check messages from both users are used to find the best media path, which is then selected. This media path is then used to transfer media. Additionally, MS Teams has a different flow when using the Public Switched Telephone Network (PSTN), however, we will not evaluate this. Therefore, PSTN is out of scope in this paper [13].

Discord is a VoIP application originally created for the gaming community, thanks to its low latency and high stability. Apart from the features a lot of other Video Conferencing Systems offer, Discord allows people to create their own groups in which people can interact with each other, and is mainly focused on entertainment. Discord takes advantage of WebRTC. WebRTC is a specification for real-time communication composed of media components, to implement their audio and video features. However, the Discord desktop client uses a media engine built on top of the WebRTC native library, which allows them to customize the communication protocol in their native application. First, as Discord is using the WebRTC native library they can make use of the low level API from WebRTC 'webrtc::Call' to create both send and receive streams. Secondly, Discord does not make use of ICE (which WebRTC uses), because every client already connects to their media relay server, which is also their own implementation in C++. Then finally, because Discord is using the WebRTC native library they can implement their own transport layer. This allows them to use the faster Salsa20 encryption [14] instead of datagram transport layer security/secure real-time transport protocol [14]. The backend services that the clients are communicating through are written in Elixir [14]. When the client is online it maintains a websocket connection to the Discord gateway. All the messages and audio related streams are going through the Discord gateway [14].

There are lots of details that distinguish Teams, Zoom and Discord infrastructures between them and other Video Conferencing Systems. For example, Zoom uses its own data

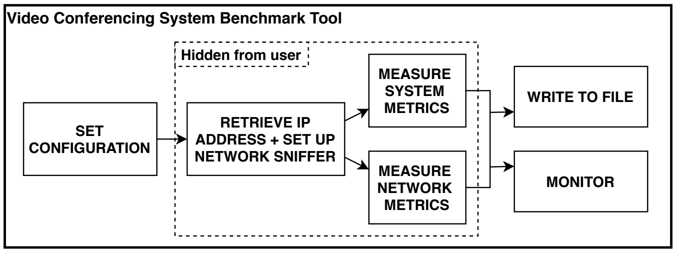

Fig. 2. Overview of Video Conferencing System performance evaluation tool.

centers for paid customers and redirect the traffic to AWS data centers for the others [15], whereas MS Teams only uses servers one of their own data centers, ensuring performance using their SDN-based architecture [16]. MS Teams selects this server based on the geographical location of the first joiner of the meeting. Based on this assumption, we made an abstraction that makes it easy to understand how we tested these three Video Conferencing Systems. Figure 1 represents the system under test in a one-to-one experiment, each machine has the Video Conferencing Client running. In a one-to-one session we consider one machine to be the sender of the workload, and the other machine is the receiver. We measure the system and network metrics from the machine on the receiving end. Latency and jitter will not be measured because, since we are not able to make measurements from the server side.

III. Design and Implementation of a Video Conferencing Performance Evaluation Tool

To evaluate the performance of Video Conferencing Systems, we need a tool that measures system metrics like CPU utilization, but we are also interested in network metrics such as the number of packets that we receive and how large these packets are. Furthermore, we want to change the environment by changing the available bandwidth to see how the Video Conferencing System reacts. Therefore, from the tool is required that on the the Linux operating it can:

1) Measure CPU Utilization as a percentage where each CPU can be utilized up to 100%.
2) Measure the number of packets received from the Video Conferencing System per second.
3) Measure the average size of the packets received from the Video Conferencing System per second.
4) Measure the bandwidth received from the Video Conferencing System.
5) Support Video Conferencing Systems Zoom, MS Teams, and Discord.
6) The ability to throttle the bandwidth for the Video Conferencing System dynamically.

As we deem the ease of use a top priority, we do not need the user to track down the IP address and (multiple) process IDs of the system it wants to measure. However, we do need the user to be aware of its own network interface. Additionally, we ask the user to verify if the automatically generated configuration is correct.

When the user has given the configuration, the IP address of the Video Conferencing System gets determined by capturing the packet data for 30 seconds with PyShark [17], which is a Python wrapper for Tshark [18]. The source IP address that is most common amongst all the captured packets is chosen as the IP address that belongs to the Video Conferencing System. Again, the user will be asked to verify this. Then a separate thread will measure the CPU utilization every second with the help of Psutil, which is a cross-platform library for retrieving information on running processes and system utilization [19]. While the tool is running, the user gets shown a monitor in the terminal, enabling them to follow the measurements real-time. At the end of the benchmarking session all the data gets written to a comma-separated CSV file. A high-level overview of the flow is shown in Figure 2.

IV. Experiment Design and Implementation

Running a Video Conferencing System client on your computer needs resources, one of them is the CPU. While having a video or voice call with someone else, there is a continuous stream of UDP packets over the network. Data transfer rates differ per country, based on the quality of their internet infrastructure. In a report by Panda Security we observe that some countries have very fast internet such as Taiwan or Singapore, with 85.02 and 70.86 mean download speed in MBps respectively, while other countries such as Uruguay or Jamaica have significantly lower mean download speeds with 9.16 and 9.08 MBps respectively [10]. Especially, countries with lower mean download speeds benefit from Video Conferencing Systems that use the network efficiently. Therefore, not only do we want to measure the CPU utilization of those systems, we are also interested in the number of packets they send per second and the size of those packets, knowing these metrics, we can also determine the number of bytes that we receive. Table IV gives an overview of the experiments, including a description and the priority. The requirements are created to compare different Online Video Conferencing systems.

A. System and Network metrics with different Workloads

To understand the normal behaviour of these systems we test three different Video Conferencing Systems: Zoom, MS Teams, and Discord with different workloads:

1) Audio and Video one-on-one call with near real-world video feed and real-world audio feed.
2) Audio-only one-on-one call with a real-world audio feed.
3) Video-only one-on-one call with near real-world video feed.

Depending on the experiment, we either change the workload in an environment that stays the same, or we keep the workload the same and observe the behavior in different environments.

	Requirement	Description	Priority
A	Measure system and network metrics under different workloads.	The experiment is performed to understand the performance of these systems by recording each metric.	Must
B	Measure system and network metrics while limiting the bandwidth.	This experiment captures the effect of limiting bandwidth on these systems.	Should
C	Measure system and network metrics while limiting CPU resources.	This experiment captures the effect of limiting CPU resources on these systems.	Could
D	Measure system and network metrics on different networks.	System and Network metrics are captured on Ethernet, Wifi 4, Wifi 5, and 4G.	Could
E	Measure system and network metrics in a classroom environment.	A classroom experiment with more participants, which captures System and Network metrics.	Could

TABLE I
OVERVIEW OF MAIN REQUIREMENTS FOR THE EXPERIMENT DESIGN.

In these experiments we measure the CPU utilization, number of packets per second, average size of packet per second, and the received bytes per second. We repeat this experiment for every system 50 times. This is done by taking 1-minute samples from a benchmark that we run for 50 minutes. This allows us to observe the variability of each of the metrics. We will observe the variability by taking the population mean, median, 25th empirical quartile (lower quartile), 75th empirical quartile (upper quartile), and the minimum and maximum value. The minimum and maximum values are denoted by the lower quartile $-1.5 \times IQR$ (Interquartile range, the difference between the lower quartile value and upper quartile value) and the upper quartile $+1.5 \times IQR$. This is a controlled experiment and although we have a near real-world workload. It does not capture the behaviour of these systems with a complete random video feed, or multiple people in a video and/or voice call.

*B. System and Network Metrics while Limiting Bandwidth (**BONUS**)*

Not every person has the same quality of internet connection, this experiment observes the changes in both the network and system metrics when the bandwidth gets limited. For the three different Video Conferencing Systems we will measure the number of packets received, the average size of those packets, and the CPU utilization for a call of 10 minutes where the first minute will be non-limited, then the second minute the bandwidth limit starts at a download speed of 500 kBps (kilobytes per second) and will subsequently lowered with 50 kBps every minute till the end of the duration of the experiment. We will look at the operational data of these systems and observe how the metrics change over time.

C. Difficulties Limiting CPU Resources

Not only have we tried limiting the bandwidth. We have also tried limiting the CPU resources. However, this was problematic. The following issues have been observed:

1) CPU resource throttling is done in a best effort approach, this resulted in a lot of measurements where the CPU was actually not throttled enough to be under the threshold that we want to observe in the experiment.
2) Some of the Video Conferencing Systems uses more than a single process to do its job. We run again in the same issue that not every process was throttled enough to be under the threshold that we want to observe. Additionally, we were not able to give the whole set of processes an upper bound of resource usage, so that it can determine on its own how it wants to use those resources.

Both issues made us conclude that these experiments will be highly inaccurate and not representative to the real-world. Therefore, we have chosen to not include these experiments in this paper.

*D. Ethernet vs. Wi-Fi 4 vs. Wi-Fi 5 vs. 4G (**BONUS**)*

The experiment we conduct is an experiment in which we observe the same workload for Zoom when the machine is connected to different network technologies. We will evaluate the performance of Zoom on a KPN NetwerkNL network with 100 Mbps download speed and 10 Mbps upload speed. We differentiate by connecting to this network with Ethernet, Wi-Fi 4 (802.11n), Wi-Fi 5 (802.11ac), and 4G LTE. The media feed in this experiment is both audio and video simultaneously.

In this experiment, we measure the CPU utilization, number of packets per second, average size of packet per second, and the received bytes per second over a duration of 10 minutes. The data will be sampled per minute, and we compute the variability of the performance as described in Experiment IV-A.

*E. Classroom Experiment (**BONUS**)*

Finally, we conduct an experiment in a classroom environment. The key difference in this experiment is that there are more participants in the video conference call compared to the design of the previous experiments.

In this experiment, we measure the CPU utilization, number of packets per second, average size of packet per second, and the received bytes per second over a duration of 45 minutes. The data will be sampled per minute, and we compute the variability of the performance as described in Experiment IV-A. The classroom experiment will only be conducted in Zoom, as it will be based on the lectures for the course, Distributed Systems.

V. EXPERIMENT RESULTS

This Section presents the results of the various experiments described in the previous Section. We present the system and network utilization of Experiment IV-A first, showing

	Version Number
Zoom	5.5.3
MS Teams	1.3.00.25560
Discord	72814

TABLE II
VERSION NUMBERS PER VIDEO CONFERENCING CLIENT.

	avg. bandwidth (kBps)	avg. CPU cores
Zoom	131.26	0.131
MS Teams	109.30	0.141
Discord	418.37	1.108

TABLE III
AVERAGE BANDWIDTH AND CPU CORES USED PER VIDEO CONFERENCING SYSTEM.

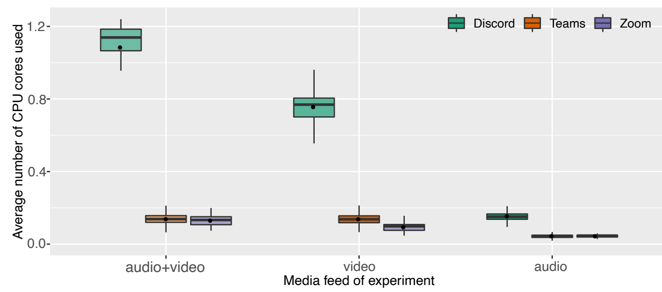

Fig. 3. The average number of CPU cores used for Zoom, MS Teams, and Discord in an one-on-one audio and video call.

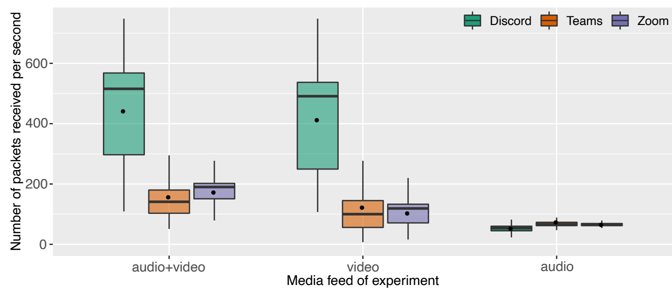

Fig. 4. The average number of network packets received for Zoom, MS Teams, and Discord in an one-on-one audio and video call.

how these systems behave in an one-on-one setting, with different media workloads. Then we show the results of Experiment IV-B in which we adjust the environment by limiting the download speed. We conclude with two more bonus experiments. We observe the behavior of Zoom when we connect to a different type of network, and we observe the behavior of Zoom in a classroom setting. Our main findings are:

- A Discord has the lowest bandwidth usage during audio streaming.
- A Zoom has the lowest bandwidth usage during combination of video and audio streaming.
- A Discord uses significantly more CPU resources than Zoom and MS Teams.
- B Both MS Teams as Zoom do not have higher CPU utilization when the available bandwidth is low.
- E Streamed video of a constant view still increases the used network resources significantly.

Experimental Setup

We use different workloads for our experiments described in Section IV. The workloads for Experiment IV-A can be subdivided into three different categories. These categories are a video [20] with a duration of 2:36 minutes played in a continuous loop, an audio stream from the microphone generated by turning on NPO2 radio, and a combination of the two (i.e. a workload with both audio and video). What the three workloads have in common is that they are conducted in a one-on-one setting. The client sending the video and audio feed is assumed not to be the bottleneck. For Experiment IV-B and Experiment IV-D the workload consists of a video with a duration of 2:36 minutes played in a continuous loop. The duration of the workload is 10 minutes in total. The workload of the classroom experiment is a real-world workload in which we measure a lecture on Distributed Systems with at least 20 people attending for 45 minutes. In this workload one person is continuously making use of the screen sharing functionality with the webcam on. The rest of the participants are most of the time muted with their webcam off. The chat functionality is frequently used.

For all experiments, we use a single machine to measure the system utilization and network traffic. This machine has a Ryzen 7 2700X processor, 16GB DDR4 memory, and has an Ethernet connection over a KPN network with 100 Mbps download speed and 10 Mbps upload speed. The version numbers of the clients that have been used in these experiments are denoted in Table II. The full screen resolution for the system that ran the experiments was 1920×1080. For all three systems the default resolution settings are used. For Zoom, Zoom HD is not used and both MS Teams and Discord do not have the option to change the resolution.

For Experiment IV-A, IV-D, and IV-E this environment stays the same over the course of the workloads. However for Experiment IV-B we change the environment over time. The first minute of the experiment the machine on which the benchmark is conducted is identical to every other experiment. However, after 1 minute the download speed gets limited to 500 kBps. Every subsequent minute the download speed gets further limited by 50 kBps till the download speed is at 0 kBps.

A. System and Network metrics with different Workloads

To understand the differences between the video conferencing software, each is measured during three different workloads as described in the experimental setup. In this experiment we measure the CPU utilization in the number of cores, number of packets received per second, average packet length per second and the bandwidth of the Video Conferencing Software during three workloads. With

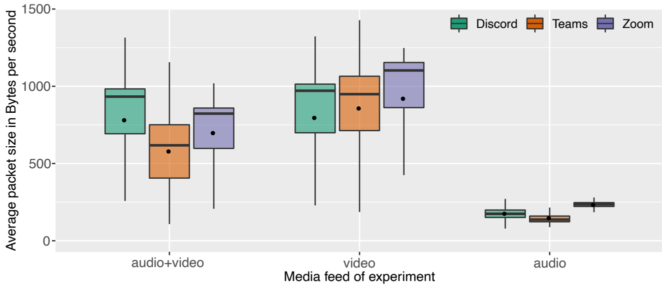

Fig. 5. The average packet size in bytes for Zoom, MS Teams, and Discord in an one-on-one audio and video call.

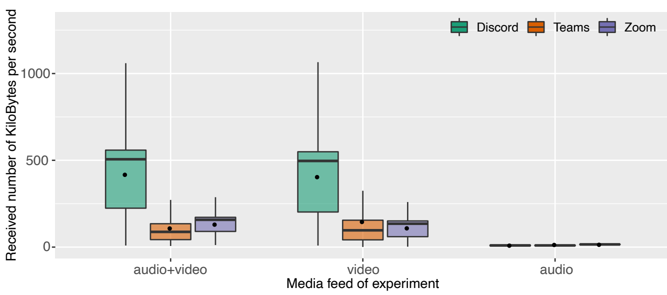

Fig. 6. The number of bytes received for Zoom, MS Teams, and Discord in an one-on-one audio and video call.

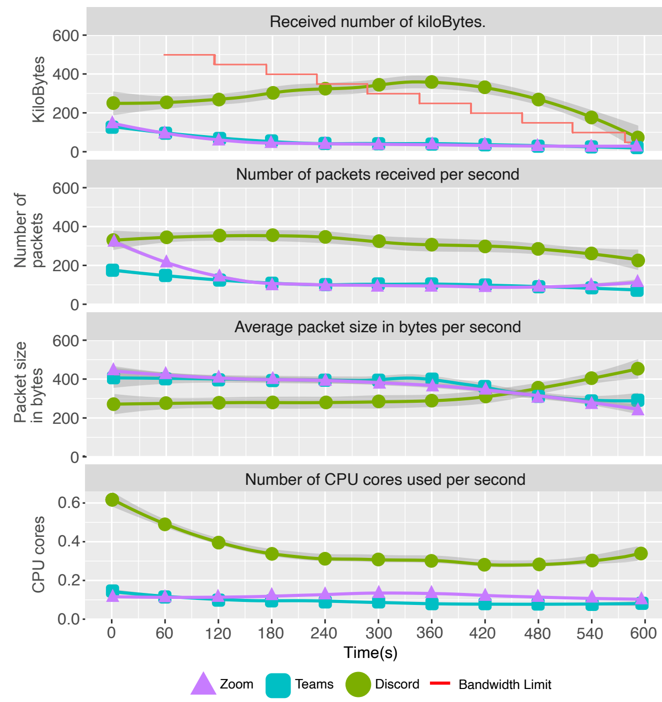

Fig. 7. The number of packets, size of packets, CPU cores and bandwidth during bandwidth throttling for Zoom, MS Teams, and Discord. The grey band shows the variation.

these measurements these Video Conferencing Software can be compared. The Video Conferencing Software performs well in our experiment if both the bandwidth and CPU utilization is low. Figures 3, 4, 5, and 6 shows the results of performance during different workloads. For an explanation of the used box plots, see section IV-A.

The results of the audio streaming show that it is not a heavy load both the bandwidth and CPU utilization. It shows that of all three Video Conferencing Software use small packages for the audio streaming. This small package size for audio also reduces the average packet size in the results of the combined experiment, simultaneous audio and video streaming. Zoom has the largest bandwidth during the audio workload, closely followed by MS Teams. Discord has the lowest bandwidth when only audio is streamed. Discord is built for voice interaction between players of a game. Therefore, they obviously made this feature efficient.

During video streaming, Discord uses approximately 500 kBps, significantly more bandwidth compared with both Zoom and MS Teams. Remarkably, Discord also uses significantly more CPU utilization, which does not seem to be used for data compression. Zoom and MS Teams both use less bandwidth and have less CPU utilization. MS Teams uses less and smaller packets than Zoom, hence its bandwidth is lower. The use of smaller packet sizes means there are more software interrupts, but gives the ability for the conferencing client software to process more work per unit of time. But the ratio between the packet payload and packet header is lower, therefore, more individual packets are sent compared to large packets. Large packets have the disadvantage that they fill up the connection sooner. Hence, the higher use of CPU utilization of MS Teams compared to Zoom might be caused by continuously handling smaller data packets or because of the stronger data compression ratio being used by MS Teams. For the three different systems we have denoted the average received kilobytes per second and the average CPU cores used in Table III. Due to inaccurate measurements the combination of audio and in the measurements of Zoom use less bandwidth than when streaming only video.

B. System and Network metrics while Limiting Bandwidth

To understand the ability of the conferencing software to adapt to environment changes, we limited the bandwidth at the receiving client. We expect the adaptive conferencing software to change the data compression ratio. This will increase the CPU utilization but decrease the bandwidth.

In a stream-oriented communication codecs play a key role to achieve high-quality communication. Codecs is a portmanteau of *coder-decoder*. It encodes and then decodes the digital data, to and from a format with certain properties respectively. The digital data is communicated between clients and servers in this format. Example of properties of formats are data quality and compression ratio.

Zoom uses the codec H.264 Annex G(SVC) [21]. The

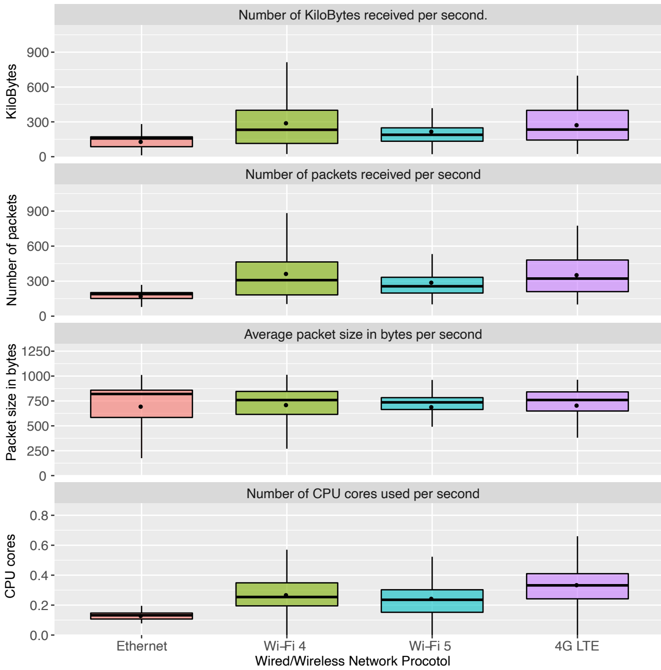

Fig. 8. The number of packets, size of packets, CPU cores and bandwidth for Zoom connected with different network technologies.

key concept behind a Scalable Video Codec (SVC) is the use of multiple layers for a single stream. Each layer is dependent on the previous one, except for the base layer. The base layer is the minimum video quality while each subsequent layer adds improvements to the video quality. Once a stream is received , it needs to be decoded which requires CPU utilization. When these streams are transported on a network their number of layers is adjusted based on the network capacity. As can be seen in Figure 7 the CPU utilization is very constant. We did expect the CPU utilization to decrease when bandwidth throttling is applied. A layer with less quality will be streamed, resulting in less data to be decoded. During the bandwidth throttling we do not see a difference in ratio between package rate and package length.

MS Teams uses the H.264 Advanced Video Codec (AVC) [22], also known as MPEG-4 Part 10. The main difference between SVC and AVC is that AVC use the single bit-rate for a stream while SVC uses multiple layers for managing quality. Therefore we expect that the CPU utilization decreases when the bandwidth is throttled as the Codec will perform the same algorithm, but this has to be done on less data. It is important to note that this AVC approach could cause jitters or latency problems in a limited network environment during the bit-rate adaptation phase. The results of MS Teams in Figure 7 reflects this trend of using less CPU utilization. Similar to Zoom the ratio between package rate and package length does not change during bandwidth throttling.

Codecs are not only about Video but also Audio. The Audio Codec used by MS Teams and Zoom are SILK [22] and Opus [23] respectively. Since they both use the same technology for streaming audio frames we did expect the same results in terms of audio-only workload for the CPU utilization. Our findings confirm our hypothesis as shown in Figure 3.

The bandwidth throttling does not work for Discord. Therefore, this experiment could not be done. More about this in the Discussion.

D. *Ethernet vs. Wi-Fi 4 vs. Wi-Fi 5 vs. 4G (**BONUS**)*

We performed Experiment V-A over Wi-Fi 4, Wi-Fi 5 and 4G. Because the maximum transmission unit for the 802.11 protocol used over the wireless connections is higher than for Ethernet we expect larger packet sizes. The wireless connections easily exceed the KPN NetwerkNL network bandwidth of 100 Mbps that is used during the experiment, hence for the bandwidth we expect no difference between each other and the Ethernet connection. This means that with the larger size packets, the number of packets is expected to decrease. By the same reasoning, KPN NetwerkNL being the bottleneck, we expect no big differences between Wi-Fi 4, Wi-Fi 5 and 4G.

The results do not confirm our expectations. As can be seen in Figure 8, the bandwidth slightly increased, the number of packets increased and the packet size seems to be decreased. The increase in CPU utilization might be caused by the additional encryption layer when being connected over a wireless connection, although we expect the decryption to happen on a different process ID and therefore not being registered by our tool. The packet size over Wi-Fi 5 is significantly more constant during the experiment, also this surprised us. We even observe that Wi-Fi 4 performs better than Wi-Fi 5. We hypothesise that this is due to the fact that the speed of the network does not play a role in our metrics, however, the improved range of Wi-Fi 4 could have had a positive effect on the results.

E. *Classroom Experiment (**BONUS**)*

During this experiment we compare the load when simulated to a lecture using Zoom Video Conferencing Software. There were twenty six people in the meeting, with no people joining or leaving. One teacher streamed a powerpoint presentation, webcam with virtual background and sound. A second teacher streamed a webcam with a virtual background. During the lecture there are three video streams involved, compared to one during the simulation, this would suggest a higher bandwidth use. But, both teachers and the power point have less changing/moving parts than our video stream. Therefore, we predict the bandwidth to be fairly equal on average. We do expect more fluctuations in both bandwidth and CPU utilization because of students asking or answering questions, hence increasing the audio

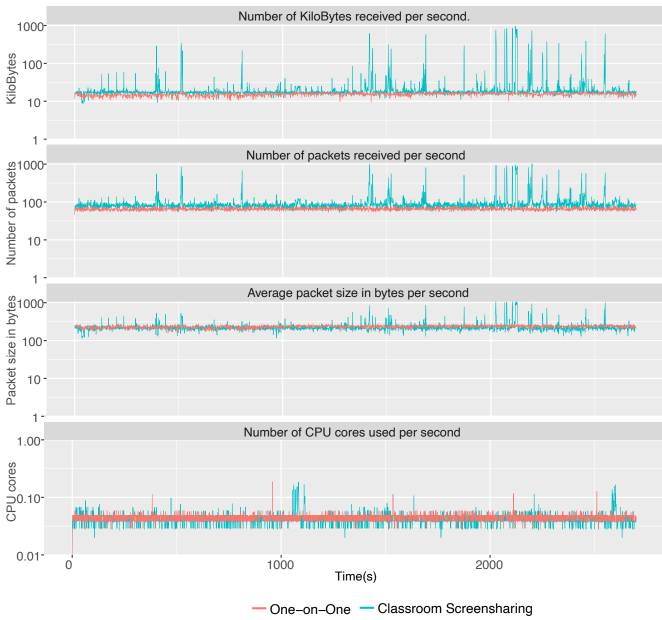

Fig. 9. The number of packets, size of packets, CPU cores and bandwidth for Zoom during experiments V-A in red and during lecture in blue.

bandwidth, and changes of powerpoint slides. Because the we expect more fluctuations in the bandwidth, we naturally expect more fluctuations in the number of packets per second and average size of the packets per second.

The results of the classroom experiment are presented in Figure 9. The metrics are presented over time to show that as expected, the metrics have a higher variation. The video used in simulated workload switches rapidly between different videos, whereas the streamed webcams should record less changing/moving parts. The average used bandwidth is higher compared to the simulated workload. This shows that even a video with not many changing/moving parts significantly increases the used network resources. The CPU utilization of the classroom experiment does not fluctuate as much as expected and is very similar to the CPU utilization of Experiment V-A presented by the red curve in Figure 9.

VI. Discussion

Our experiments show that Zoom uses the least network and system resources during video streaming. MS Teams uses slightly more system resources, but comes very close and even out performs Zoom when streaming both audio and video. Both show slightly different network properties. Zoom uses on average more, but smaller packets compared to MS Teams. Discord shows a significant difference compared to MS Teams and Zoom, with almost triple the number of packets received, packet sizes 5 times larger, and a significant use of CPU where it uses more than a full core when conducting a call with both audio and video with Discord. The high CPU usage seems to be a reoccurring problem for Discord with a lot of complaints from the gaming community across online forums. A Discord employee on the online forum Reddit explained that Discord is not fully utilizing the hardware in the most efficient way to encode the video data [24].

Audio streaming does not have a large impact on the resource. Discord uses the least bandwidth while Zoom and MS Teams follow with not much more bandwidth uses. Discord has compared to MS Teams and Zoom almost twice as much CPU utilization. This CPU utilization seems to be a recurring problem for Discord.

Just like online gaming, video conferencing is strongly related to network Quality of Service (QoS). User's satisfaction with the system determines if the user wants to use these systems, and we observe that online gaming and video conferencing are very much alike. As shown by Chen [25] unsatisfactory network conditions discourage users from continuing the online game. Chen proposes three improvements on user's perception of QoS. Improving user satisfaction by designing the system that it automatically adapts to network quality in real-time in order to improve user satisfaction, optimizing network infrastructure, and network troubleshooting to efficiently measure network conditions. Unsatisfactory network conditions experienced in a video conferencing system discourages users as it affects their QoS, and they will look for a replacement. We identify that the video conferencing system provider should also focus on Chen's three improvements to ensure a high QoS to its users.

For our experiments we have a video feed [20] that loops, but it is not representative for a real-world video feed. However, we chose it for reproducibility and simplicity. The same applies for using an external radio to generate audio feed. We choose to do multiple repetitions in one session instead of multiple sessions throughout the day for efficiency sake. However, the experiments could have been conducted anytime of the day, which could have captured scenarios in which the server of the system tested has been under lower or higher load.

To measure bandwidth, number of packets, and package length, a continuous loop collects this information for each second and aggregates this. Sampling per second is done in a best effort approach, but there is a slight discrepancy between the time we aggregate the data and persist the data, these measurements vary by two hundredths of a second. We observe in the graphs of Experiment IV-B that we fail to stay under the specified bandwidth limit when the experiment nears the end of its duration. This is especially visible in the experiment done with Discord. The reason for this is the following, Wondershaper [26] has been used to throttle our bandwidth and it does so by utilizing a queue that captures the network packets. It makes sure to dequeue several packets that satisfies our limit on bandwidth throughput. However, Wondershaper starts to show undefined behavior when seemingly the queue overflows. We hypothesize that Wondershaper flushes the queue. Resulting in higher download speeds than we limit the system to. In our experiments we only consider local measurements. We made sure the sender had both good hardware and connection. Limitations of the sender-side were not considered. Expanding these experiments to a more distributed setting could reveal interesting findings.

In our experiments we were not able to measure latency and video quality. Software conferencing software can have low CPU utilization and low bandwidth by trading latency and video quality. Future work on this topic should extend our experimental design with these metrics.

VII. Conclusion

We have performed empirical experiments on the Video Conferencing Software Zoom, Discord and MS Teams where we consider the software to be a black box. As our results demonstrate, it is possible to compare Video Conferencing Software this way. We have shown that Discord uses the least bandwidth during audio streaming, Zoom uses the least bandwidth during video streaming while MS Teams uses the least resource during both audio and video streaming. Furthermore, we have shown that Discord uses significantly more CPU utilization and bandwidth compared to MS Teams and Zoom during video streaming.

Performing these experiments on these complex distributed software was hard, these video conferencing systems take for example many precautions to reduce the bandwidth, making it hard to create a good workload. We see our work as work in process, future research should include latency and video quality for a better comparison.

References

[1] Forbes Insights Team, "Optimizing team performance: How and why video conferencing trumps audio," October 2017. [Online]. Available: https://bit.ly/2W6gx92

[2] M. Baker, "Gartner hr survey shows 86 of organizations are conducting virtual interviews to hire candidates during coronavirus pandemic," April 2020. [Online]. Available: https://gtnr.it/33Zr9uM

[3] S. Ann Earon, "The value of video communications in education." [Online]. Available: https://bit.ly/2IDpN1i

[4] T. Warren, "Microsoft teams jumps 70 percent to 75 million daily active users," April 2020. [Online]. Available: https://bit.ly/2IIoe2i

[5] Nate Drake and Brian Turner, "7 best video conferencing tools of 2020." [Online]. Available: https://bit.ly/3gzU4L8

[6] B. Jansen, T. Goodwin, V. Gupta, F. A. Kuipers, and G. Zussman, "Performance evaluation of webrtc-based video conferencing," *SIGMETRICS Perform. Evaluation Rev.*, vol. 45, no. 3, pp. 56–68, 2017. [Online]. Available: https://doi.org/10.1145/3199524.3199534

[7] P. R. M. Vasconcelos, G. A. de Araújo Freitas, and T. G. Marques, "Virtualization technologies in web conferencing systems: A performance overview," in *11th International Conference for Internet Technology and Secured Transactions, ICITST 2016, Barcelona, Spain, December 5-7, 2016*. IEEE, 2016, pp. 376–383. [Online]. Available: https://doi.org/10.1109/ICITST.2016.7856734

[8] K. Nisar, N. I. Sarkar, and Y. Dole, "Performance studies of voice and video conferencing over ATM and gigabit ethernet backbone networks," *Int. J. Technol. Diffusion*, vol. 3, no. 1, pp. 22–32, 2012.

[9] C. Xiong, J. Ge, Q. Wang, and X. Wang, "Design and evaluation of a real-time video conferencing environment for support teaching: an attempt to promote equality of K-12 education in china," *Interact. Learn. Environ.*, vol. 25, no. 5, pp. 596–609, 2017. [Online]. Available: https://doi.org/10.1080/10494820.2016.1171786

[10] Panda Security, "Working abroad: What countries have the fastest internet?" December 2019. [Online]. Available: https://bit.ly/2JSJ7Is

[11] C. Ahlgrim, "Artists are playing live concerts from their homes amid the coronavirus outbreak — here's how to watch the best ones," May 2020.

[12] Zoom Video Communications, Inc., "Zoom security white paper," October 2020. [Online]. Available: https://bit.ly/376ycUj

[13] Microsoft Corporation, "Microsoft teams call flows," August 2018. [Online]. Available: https://bit.ly/3768WOe

[14] J. Vass, "How discord handles two and half million concurrent voice users using webrtc," September 2018. [Online]. Available: https://bit.ly/3qM2vHR

[15] P. Judge, "Most of zoom runs on aws, not oracle - says aws," April 2020.

[16] T. Arbuthnot, "Where in the world will my microsoft teams meeting be hosted?" June 2019.

[17] KimiNewt, "pyshark." [Online]. Available: https://bit.ly/3nbD5kz

[18] Wireshark, "Tshark man page." [Online]. Available: https://bit.ly/3nlZrQo

[19] Giampaolo Rodola, "psutil." [Online]. Available: https://bit.ly/2Ltk9Qv

[20] RoyaltyFree Videos for content creators, "Royaltyfree channel trailer free to use stock footage no copyright," 2018. [Online]. Available: https://bit.ly/3a7NB97

[21] Vikram Sachdeva, "Zoom — video conf app at scale," May 2018. [Online]. Available: https://bit.ly/3gEgcDT

[22] Rodolfo Castro, "Is your company's network ready for microsoft teams?" January 2020. [Online]. Available: https://bit.ly/3m52kE5

[23] Zoom. [Online]. Available: https://bit.ly/3m3uz5W

[24] ReallyAmused, "Discord uses 100% of cpu when screensharing," 2018. [Online]. Available: https://bit.ly/3oIurup

[25] K. Chen, P. Huang, and C. Lei, "How sensitive are online gamers to network quality?" *Commun. ACM*, vol. 49, no. 11, pp. 34–38, 2006. [Online]. Available: https://doi.org/10.1145/1167838.1167859

[26] Magnific0, "Wondershaper, tool for bandwidth throttling." [Online]. Available: https://bit.ly/345EFgt