Jevgeni Ulm

IMPROVING XAMKLAB NETWORK BY MEANS OF LINK AGGREGATION CONTROL PROTOCOL

Bachelor's thesis

Bachelor of Engineering

Information Technology

2023

South-Eastern Finland
University of Applied Sciences

Degree title	Bachelor of Engineering
Author(s)	Jevgeni Ulm
Thesis title	Improving Xamklab network by means of Link-Aggregation Control Protocol
Year	2023
Pages	41 pages, 8 pages of appendices
Supervisor	Heikki Brotkin

ABSTRACT

The study was inspired by two situations which hindered lab classes. The first situation was due to the lack of network connection in a classroom. The second situation was due to the lack of bandwidth.

The aim of the study was to improve the network without investing into new equipment. One way to do so was by means of Link-Aggregation Control Protocol. Improving the network lowered the possibility of similar incidents taking place again in the future. This was achieved due to added redundancy and increased bandwidth.

The study researched the link aggregation and its alternatives. The link aggregation was chosen due to being cheapest and the most environmentally friendly of the options. The network was tested, after which, the link aggregation was implemented, and the network was tested again. Test results were recorded and compared.

The study showed that the link aggregation was an effective way of adding redundancy and increasing the bandwidth without any costs resulting in minimal environmental impact.

Keywords: link aggregation, LACP, XCP-NG, switch, server

TABLE OF CONTENTS

1 INTRODUCTION

The decision to write a thesis about a link aggregation came from two situations which took place during my internship at South-Eastern Finland University of Applied Sciences. The first situation happened in the beginning of a lab class for the first-year students in a classroom MB316. The students were supposed to access Learn webpage to find instructions for the forementioned lab, however the internet connection was down. The reason for this was that a cable going from the server switch to the classroom switch got disconnected, thus leaving the classroom without the internet. Second situation happened in the same classroom with the same students while a different lab. In the lab the students were supposed to use two virtual machines which were prepared by the lecturer and stored on a shared network storage. However, when all the students tried to copy those virtual machines to classroom machines at the same time, it turned out that one Gigabit connection was not enough, as some students saw an estimated waiting times of over 30 minutes. This thesis is attempting to remove the possibility of similar situations happening in the future by means of the link aggregation.

The thesis consists of two parts. The first part involves information regarding the link aggregation and the supporting protocol, information regarding devices used in the link aggregation, and finally information regarding testing methods which are meant to measure the performance of the network connection before and after the link aggregation. The second part involves testing the network connection before the link aggregation, implementing the link aggregation to the test environment, then implementing it to the production environment, and finally, the network connection will be tested again after the link aggregation is complete. The test results will be documented and compared. Two implementations are needed for removing the possibility of breaking the production environment and lowering the amount of down time on the production environment.

The aim for the thesis is to improve network connection between the servers and the classroom MB316 by adding redundancy, and by increasing bandwidth by means of link aggregation. The main reason choosing the link aggregation over other alternatives is its environmental impact and cost. All devices used in the project support link aggregation according to their documentation. This way no new devices need to be acquired, resulting in zero costs and zero e-waste. The link aggregation is going to be implemented on four ports for each device, which is going to increase the number of links between the devices from one to four. This approach will provide multiple redundancy links and increase the bandwidth up to fourfold.

2 PREREQUISITES

This chapter introduces one of the network technologies called Ethernet and devices used in the link aggregation and in testing.

2.1 Ethernet

Ethernet is a network technology used for connecting various network devices to the same network, as well as connecting various networks. Ethernet is often referred to as a wired network technology, defined in IEEE 802.3, however it also includes a wireless network technology, defined in IEEE 802.11, commonly known as wireless LAN.

Ethernet was developed at Xerox PARC, nowadays just PARC which stands for Palo Alto Research Center. It was developed by Robert Metcalfe and David Boggs between 1973 and 1974. It was inspired by ARPANET and Aloha System. The creation of Ethernet was driven by two applications: a laser printer and early Internet access. Robert Metcalfe and other workers at the office wanted to have access from individual computers to the printer and to the Internet. Internet access was implemented by a big coaxial cable in the ceiling running through the corridors. Ethernet was implemented by tapping into the coaxial cable from individual computers. Since coaxial cable was located close to the ceiling, it was never referred to as coaxial, it was referred to as Ether. The name Ether was borrowed from physicists of the 19^{th} century, who thought that the Earth and the Sun had a medium, which was called luminiferous ether. In 1983 Ethernet was defined in IEEE 802.3. Through the 1980s Ethernet was competing with

TokenRing from IBM and other local area networks. However, by the end of 1980s Ethernet became the dominant local area network, because it adapted to market realities, for example, it shifted from coaxial to twisted pair cabling. Then, in 1993, when World Wide Web was released into public domain and people started buying personal computers to be on the Internet, Ethernet became more dominant. (The History of Ethernet 2006.)

As Robert Metcalfe said, “Ethernet today bears little resemblance to the CSMA/CD technology that David Boggs and I developed in 1973” (Ibid). He said it, because by 2006 Ethernet had evolved into larger networking technology than a few office workers wanting access to the printer and to the Internet from individual computers. Since 2006 Ethernet has evolved even more, nowadays Ethernet is used in local area networks but also in wide area networks, in metropolitan area networks, as well as by network carriers. Ethernet speed has also improved. It started at 2.94 Mbit/s and nowadays speeds reach up to 400Gbit/s. (Terabit Ethernet: The New Hot Trend in Data Centers, n.d.) Capabilities of Ethernet have improved from sending and receiving data to being able to also provide power, PoE or Power over Ethernet, defined in IEEE 802.3af is standard which allows to transmit power additionally to the data over a single Ethernet cable. PoE itself has evolved as well, as there is PoE+ standard, defined in IEEE 802.3at, and PoE++ standard, defined in IEEE 802.3bt. (Charlene, 2022.) This thesis focuses on the link aggregation and Link Aggregation Control Protocol, both are defined in IEEE 802.3ad.

2.2 Devices

As it was stated in the introduction, the goal of this project is to improve Xamklab network without buying new hardware and use what is currently available. Four devices are used in the current configuration. Two switches and two server machines. Each server is connected to the server switch by a single Ethernet cable and the server switch is connected to the classroom switch by a single Ethernet cable. This can be seen in Figure 1.

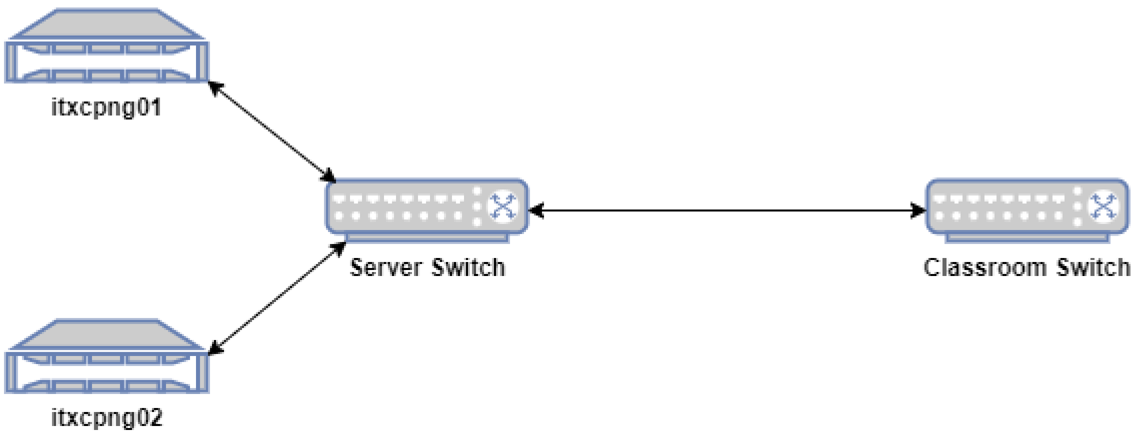

Figure 1. Network topology

Both servers are ProLiant DL380 Gen10, which have Intel Xeon Silver 4208 CPU, 64 GB of system memory, HPE 1 Gb 331i four port ethernet adapter, one ethernet management port and are running XCP-NG virtualization platform as an operating system.

Each server has one HPE 1 Gb 331i four port ethernet adapter, of which only one port is used to connect servers to the server switch. The plan for the project is to utilize all four ports. If in the future additional ethernet ports are required, ports could be added through PCI-E extension cards. Both servers have six slots for PCI-E extension cards: four half-length/full-height and two full-length/full-height. However, to have the expansion for all six slots, two expansion modules are needed. Currently servers have only one module which can accommodate two half-length/full-height and one full-length/full-height extension cards. By length is meant PCI-E port length, where full-length is PCI-E x16 port and half-length is PCI-E x8 port. By height is meant PCI-E card’s backplate height.

Both servers are running XCP-NG virtualization as an operating system. XCP-NG stands for Xen Cloud Platform - next generation and it is an open-source hypervisor based on XenServer. (XCP-NG, n.d.) XCP-NG supports three options for network bonding: Balance-SLB, Active/Backup and LACP. Balance-SLB bonding does not require switch configuration and does not use LACP or similar protocols. It increases the bandwidth as all links are active. If one or more fail, the traffic will be forwarded to remaining members of bond. Active/Backup bonding also does not require switch configuration. One of the bond members is active and utilized, another one is up only when the active member is down. LACP bonding requires the switch to be LACP aware and configured. The benefit is that LACP detects path failures even if the physical port does not go down. (Sands & Lambert 2022.)

Two switches, which are in use, are Aruba 2540-48G-4SFP+ Switch, model number JL355A, and HP 2530-48G Switch, model number J9775A. HP 2530-48G Switch has 48 RJ-45 autosensing 10/100/1000 ports, 4 fixed GbE SFP ports and one dual-personality serial console port, serial connection can be made through RJ-45 port or USB micro-B port. (Aruba 2530 switch series – specifications, n.d.) It is running on HP proprietary software, version YA.16.07.0002. Aruba 2540-48G-4SFP+ Switch has 48 RJ-45 autosensing 10/100/1000 ports, 4 full only SFP+ 1/10GbE ports and one dual-personality console port. It is running on HP proprietary software, version YC.16.10.0002. (Aruba 2540 switch series, 2020.)

As mentioned before, HP 2530-48G Switch (further referred to as classroom switch) has 48 Gigabit Ethernet ports, of which 37 are occupied: 34 ports for workstations, of which 33 are student computers and one is teacher’s computer, one for a wireless access point, one for a classroom printer and one for connection between classroom switch and server switch. This leaves 11 ports available for use. The plan for the project is to add 3 additional links from classroom switch to server switch, which will reduce the number of available ports from 11 to 8, which is believed to be enough for any future projects and/or additional devices.

Aruba 2540-48G-4SFP+ Switch (further referred to as server switch) has 48 Gigabit Ethernet ports as well, however only 20 ports are occupied: 11 ports for production servers, 5 ports for project servers, one port for gateway, one port for connection between server switch and classroom switch, one port for connection between server switch and project server switch, and one port for connection to Protopaja. This means 28 ports are available for use. The plan for the project is to add 3 additional links from each server to the server switch and 3 additional links from server switch to classroom switch, which will reduce the number of available ports from 28 to 19, which is believed to be enough for any future project and/or additional devices.

Both switches have 4 SFP ports, which could have been used for the connection between them, however, due to the classroom switch having only SFP ports rather than SFP+, that connection would be limited to one Gigabit per cable, which can also be achieved by using Ethernet ports on the switches. I believe that SFP+ ports on the server switch can be utilized better, for example improving the connection between the server room and a switch closet (or other classrooms), as they would support connection speed up to 10 Gigabit, also fiber optic cables can do greater distances than ethernet cables.

Other devices affected by the link aggregation are devices located in the classroom MB316. Those devices do not participate directly in the link aggregation, but they would benefit from it. There are 34 computers in the classroom, 33 of them are for students and one is for the teacher. Each machine has one Gigabit network card, which is used to connect them to the classroom switch. These devices are running on Windows 11 operating system.

3 LINK AGGREGATION

Link aggregation combines multiple network connections to work as a single link to achieve redundancy and in some cases also higher bandwidth. The link aggregation for Ethernet is defined in IEEE 802.3ad. There are multiple link aggregation types which fall into static and dynamic categories. In this project dynamic type of the link aggregation is used, as well as it is going to utilize Link Aggregation Control Protocol, also known as LACP, as it is vendor-independent, therefore supporting devices from various manufacturers. There are vendor specific protocols as well, such as Cisco’s EtherChannel and PAgP which stands for Port Aggregation Protocol.

3.1 Static

In the static type of the link aggregation, ports are added to Link Aggregation Group (LAG), manually configured, and do not communicate with another end device. This way each device is configured standalone and has no information about other end device link settings. (Link Aggregation Overview, n.d.)

The fact that devices do not communicate can be beneficial in a situation if only one of the devices supports link aggregation and failover/load-balancing is necessary. However, this approach is more difficult to monitor, especially if links are configured between multiple devices and in series. If two devices have the static link aggregation configured between them and a failure occurs, which might lead to a situation, in which one device will shut down a port and another will not, resulting in lost packets. (Ibid.)

It is crucial for the classroom to have an Internet connection as well as the connection to the servers for most of the classes, it would be impactful if the failure occurred. It would be difficult to troubleshoot the link aggregation related issues if it were configured statically, considering it is not only part of the Xamklab network which could cause issues. In case of the link aggregation causing the failure, the first difficulty would be detecting the failed device and second difficulty would be finding the reason for the failure. Finding the failed device would require accessing all the devices until the faulty one is found.

3.2 Dynamic

In the dynamic type of link aggregation, ports are added to Link Aggregation Group and are manually configured to use Link Aggregation Control Protocol. Then, ports use Link Aggregation Control Protocol Data Units (LACPDU) to negotiate settings between two connected devices. (Link Aggregation Overview, n.d.)

LACP helps to prevent misconfiguration of LAG settings, as it will not allow a link to be established and will set the link status as “down”. Since members are sending LACPDUs to make sure both ends are up, meaning that when one of the member links stops sending packets, it is removed from LAG to minimize packet loss. As soon as the link is back up, it is added back to the LAG automatically. One of the advantages of the dynamic link aggregation over the static one is their ability to monitor the connection using LACPDUs, and in case of a failure, both devices will shut down the corresponding port on their end, eliminating the possibility of lost packets. Another advantage is ease of configuration and maintenance compared to the static configuration. (Ibid.)

As mentioned in the previous chapter, connections are crucial for the proper operation of classes in the classroom. The dynamic link aggregation, in the project topology, allows to check the status of all links from one device, which is server switch. In case of a failure related to the link aggregation, checking the status of LAG on server switch will help to identify the faulty device by showing which part of the link has failed.

3.3 LACP

Link Aggregation Control Protocol is a protocol which allows ports participating in the link aggregation to exchange information using LACPDUs. This allows for easier link aggregation setup and easier maintenance. The port which initializes the communication is called an actor and the port which is answering the communication is called a partner. LACPDU consists of the information about the actor port and the partner port. The information about the partner port is the actor’s current view of its partner’s parameters. The following parameters are embedded in LACPDU:

- Port number
- System ID
- Key
- Status

The status information parameter consists of the following six flags:

- LACP_Activity
- LACP_Timeout
- Aggregate
- Synchronization
- Collecting
- Distributing

The LACP_Activity flag demonstrates participant's intent to transmit LACPDUs periodically to setup and maintain the aggregation. The flag can be set to Active LACP or Passive LACP. An active participant can be the actor or the partner, meaning it can initiate the packet or receive. A passive participant can only be the partner, meaning it will never send the packet first, it will only reply. The LACP_Timeout flag indicates the period between the packets. The flag can be set to Short Timeout or Long Timeout. (Seaman 1999.) When set to short, LACPDU will be sent every second and the timeout value is three seconds. When set to long, LACPDU will be sent every 30 seconds and the timeout value is 90 seconds. (Sun Ethernet Fabric Operating System, 2012.) If the timeout value runs out and partner did not reply, the ports are removed from LAG until the communication is established again. The Aggregate flag has two values. It can be set to Aggregatable or Individual state. The Aggregatable state is granted when the System ID and the Key parameters of multiple port match, allowing them to be part of the same aggregate link. The Individual state is set if the previously mentioned parameters do not match, resulting in port not being used in the aggregation. The Synchronization flag is responsible for monitoring the System ID and the Key parameters and making sure they are synchronized. The flag has two states In Sync and Out of Sync. The Collecting flag indicates that the participant's collector, the reception component of the aggregation, is definitely on. The Distributing flag indicates that the participant's distributor, the distributing component of the aggregation, is not definitely off. (Seaman 1999.)

3.4 Alternatives

As an alternative to the link aggregation, to increase the bandwidth between the devices, SFP+ 10Gb fiber optic could be used. However, only one of the devices in this project has SFP+ ports, which is the server switch. It would require extra expenses to be able to create 10Gb fiber optic connection between two devices. The first expense would be SFP+ modules for server switch, the second would be a PCI-E SFP+ network cards for servers, as well as SFP+ modules for it, or a completely new switch to replace a classroom switch, as well as SFP+ modules for it.

Another alternative could be to upgrade both switches to newer models, which could support 10Gb over copper cabling. Servers would need a capable 10Gb network card as well. However, that would increase the expenses. It is likely that this implementation would run into other hardware bottlenecks, which would prevent servers achieving full bandwidth. One of those bottlenecks could be storage read and write speeds, when accessing the shared network server. This could lead to chasing a goal of eliminating bottlenecks, which in turn would lead to additional expenses.

The forementioned alternatives increase the bandwidth but do not include redundancy unless the link aggregation is used. The current device setup allows, on hardware and software levels, to use the link aggregation to improve the bandwidth and include redundancy without any extra costs.

4 TESTING METHODS

This chapter is going to describe the methods for potentially seeing the difference between initial and updated configuration by means of various tests. The plan is to test the bandwidth by doing two tests and test the redundancy by doing a single test. The first bandwidth test is copying the same file from the server to sixteen classroom machines at the same time using a script. The next bandwidth test is running network speed test application on sixteen classroom machines at the same time. The last test is imitating a connection failure by disabling the port on the server switch, while there is an active ping from the classroom machine to one of the servers.

4.1 Script

A script is a set of commands that are executed by an operating system or an application (script, n.d.). In the project, Windows batch files with .bat file extension are used. Batch file consists of operating system commands that are carried out one after the other. The operating system interprets the commands and turns them into machine language executed by the CPU. Usually, batch files are used to perform a series of routine file management operations such as making backups and launching applications. (batch file, n.d.) However, in this case, the main reason for using batch files is to make testing consistent.

The main test for testing bandwidth is a simulation of multiple students copying a VM file from network storage at the same time. It is implemented by a script which logs start time, copies a file from network storage to local storage and logs end time. This way it will be possible to record the time spent transferring the file, as well as to calculate the speed by using file size and time spent. The script will be embedded in a scheduled task which should allow the script to run on sixteen classroom machines at the same time.

4.2 OpenSpeedTest

OpenSpeedTest is described as a cross-platform internet speed test application. It is built using JavaScript and HTML5. This allows the application to run on various web browsers within different operating systems without any additional apps. OpenSpeedTest is designed for detecting stable speed, not the top speed. It can be hosted locally by downloading source code and configuring own web server, by using provided Docker image, or by downloading source code and configuring server. (About openSpeedtest.com, n.d.)

The plan for the project is to install OpenSpeedTest server locally on a virtual machine. VM will be hosted on one of the servers. Then, a scheduled task will be created which will run the OpenSpeedTest on sixteen classroom devices at same time to apply load on the network connection. Individual test results for each device will be displayed on the device and will be gathered manually.

4.3 Ping

Ping stands for Packet Internet or Inter-Network Groper. It is a basic Internet program used to verify that a specific destination IP address exists and the machine having it can accept requests on the network. Ping is often used diagnostically to ensure that a destination machine is operating. Any operating system with networking capability can use ping. It works by sending ICMP echo request packet to a specified destination machine, which on arrival of request packet sends back echo reply packet. On the successful arrival of the echo reply packet back to the initial machine, two values are returned. The first value is a verification of a successful round trip and the second one is a round-trip time, which defines how long time it took from sending the echo request packet to receiving echo reply packet. (Zola 2021.)

This test consists of one classroom computer pinging the file share server. Then, on the server switch one port connected to the classroom switch will be disabled to imitate the port failure. Before the link aggregation is implemented, the result is expected to be a loss of connection, meaning ping will fail, because of a lack of link from the classroom computer to the server. After the link aggregation implementation, it is expected that no packets will be dropped, even if the port sending ping packets will be shut down, other ports should take over and keep up the link and ping packets.

4.4 Alternatives

One alternative to OpenSpeedTest could be a network connection speed test tool called iPerf. It is an open-source command line tool for network throughput testing between two hosts. The iPerf tool can generate TCP and UDP traffic. (iPerf - The ultimate speed test tool for TCP, UDP and SCTP, n.d.) This tool could have been used for measuring the maximum achievable bandwidth, however OpenSpeedTest application was chosen due to being easier to set up and perform tests with. iPerf has two versions, iPerf2 and iPerf3. iPerf2 is an older version which has known bugs and issues. There was an attempt to fix those issues, but in the end, it was decided to build a new, simpler tool. This is how iPerf3 development started. These two versions are not backward compatible. (iperf3 FAQ, 2017.) Both versions require an installation on the server host and an executable on the client host. This is the first reason choosing OpenSpeedTest over iPerf, as OpenSpeedTest requires only a browser on client machine, which is already installed. Secondly, iPerf3 does not support multi-threading, which means that it might be a limiting factor and maximum bandwidth would not be reached. Thirdly, iPerf3 does not support more than one client machine, which is out of scope.

Other options are netperf, which also requires server and client installation, and NetCPS is a Windows Command Line utility, which uses very few system resources during monitoring but also requires the server and client installation.

5 IMPLEMENTATION

This chapter introduces the implementation of the link aggregation between the devices, explaining in detail how devices are configured. To implement the link aggregation between MB316 classroom and the servers, four devices need to be configured: two switches, classroom switch and server switch, and two servers.

5.1 Test implementation

Before implementing the link aggregation on the production environment, it was decided to test it out on a project environment. The switches stayed the same, however, only one server was used, and it has different hardware from the servers introduced before. The configuration of the server used for test implementation will not be discussed as it is unnecessary, the only parts worth mentioning are that it runs the same operating system, which is XCP-NG, and that it has only two network ports, which is the minimum amount to create the link aggregation.

For the test implementation, two links had to be created. One between the server switch and the classroom switch, and another between the server switch and the test server. It was decided to create a link between the similar devices first. As both of the switches are essentially from HPE, the command structure for their configuration is identical. Both (switches') consoles were accessed through the serial connection at first, then their configurations were slightly changed to have access through SSH.

To configure LACP ports on HP switch, the first step is to log into the console of the switch and enter the configuration mode. Next, the interfaces used in the link aggregation need to be identified and then configured by following command: *trunk 1 trk1 lacp*, where at first is defined that the port should be set to trunk mode, then which port is set to trunk, then trunk group is defined and then, the last part defines that the trunking protocol is set as LACP. (Kuchenski 2013.) These instructions were followed to configure the link aggregation on both switches.

Next step was to configure the link between the server switch and the test server. As instructions for configuring the switch were clear. It was decided to configure the ports on the server switch first. However, that resulted in the loss of connection to the test server because server ports were not configured yet and the connection could not be established. To revert the made changes on the switch, the following command was used: *no trunk 37-38*, which removed selected ports from trunk mode (Ibid.). Graphical user interface was used to configure the link aggregation on the test server. From the graphical user interface New->Network option was chosen. Next, it had to be specified, on which pool the new network will be created, then the type of the network. Next, network parameters suck as interface(s), name, description, MTU, and bond mode. After the test server was configured, it was time to configure the server switch again. The test server configuration can be seen in Figure 2.

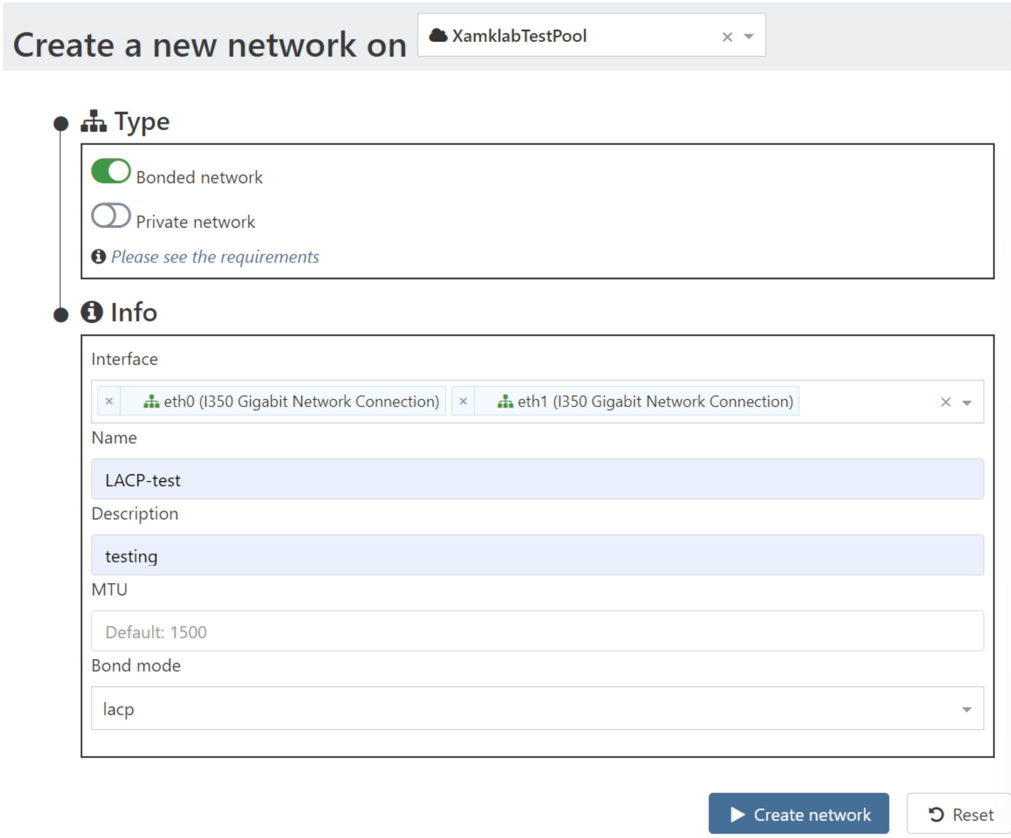

Figure 2. The test server LACP configuration

The first way to test if the link aggregation had any effect on the devices was to check the test server's GUI and see if it reports one or two Gigabit connection. It reported two. Next, to confirm that it actually is working, OpenSpeedTest application was installed on one of the VMs running on the test server. It was installed as a docker container because it was a quick way to do it. After the application was installed, the speed test was run simultaneously from two of the classroom machines and both achieved full one Gigabit per second speeds, meaning that the link aggregation was working, and it can be implemented to the production environment.

5.2 Xamklab implementation

For the production implementation, three links had to be created. One between the server switch and the classroom switch, and two between the server switch and each server. The switch configuration was identical to the test implementation. Four ports were configured on the classroom switch using the following command: *trunk 45-48 trk1 lacp*, and for the server switch the following command was used: *trunk 21-24 trk1 lacp*.

From the experience of test implementation, the next step was to configure the servers. An attempt was made to use GUI to configure the interfaces, however, an error occurred. There is no clear indicator what caused the error, but it is possible it was due to the pool having two servers, which lost the ability to communicate when set to trunk because the switch was not configured yet, and they reverted to the previous settings. It was decided to configure the servers through CLI. SSH was used to access the servers. The first step was to create a new network on the first server. It was created using the following command: *xe network-create name-label=LACP*. Then, the bond was created using this command: *xe bond-create mode=lacp network-uuid=<network-uuid> pif-uuids=<pif-uuids>*, where network-uuid is the universally unique identifier of the newly created network named LACP and the pif-uuids are the identifiers of the physical network interfaces of the server. (LACP Bonding in XenServer - Configuration and Troubleshooting, 2014.) On the second server, the configuration was the same but the step with creating the new network was skipped, because the network was created on the first server already.

The server switch CLI was accessed to finish configuring the link between the servers and the server switch. In the configuration mode, two following commands were entered to configure the ports connected to the servers:

- *trunk 13,15,17,19 trk2 lacp*
- *trunk 14,16,18,20 trk3 lacp*

As the dynamic link aggregation is simpler to create and maintain, only one command was needed to make sure that all the LACP links are active. The command and the output can be seen in Figure 3 below.

```
OpenSSH SSH client

sw-mb317-01# sh lacp

                              LACP

         LACP      Trunk    Port               LACP      Admin   Oper
  Port   Enabled   Group    Status   Partner   Status    Key     Key
  -----  -------   -------  -------  -------   -------   ------  ------
  13     Active    Trk2     Up       Yes       Success   0       533
  14     Active    Trk3     Up       Yes       Success   0       534
  15     Active    Trk2     Up       Yes       Success   0       533
  16     Active    Trk3     Up       Yes       Success   0       534
  17     Active    Trk2     Up       Yes       Success   0       533
  18     Active    Trk3     Up       Yes       Success   0       534
  19     Active    Trk2     Up       Yes       Success   0       533
  20     Active    Trk3     Up       Yes       Success   0       534
  21     Active    Trk1     Up       Yes       Success   0       532
  22     Active    Trk1     Up       Yes       Success   0       532
  23     Active    Trk1     Up       Yes       Success   0       532
  24     Active    Trk1     Up       Yes       Success   0       532

sw-mb317-01#
```

Figure 3. Server switch 'show lacp' command output

In the server switch CLI, *show lacp* command was issued. The output showed that all LACP links are active, meaning that the implementation is done. From the output can be seen that ports are up and the LACP Status is set as Success on all of the ports.

6 TESTING

This chapter introduces tests, testing of the configuration before and after the link aggregation, and comparison of the results.

To test the configuration before and after, three tests were created. The first test is the script test. The script for testing can be seen in Figure 4.

```
set start=%time:~0,8%
copy "\\itfile1\Public\Security Fundamentals VMs\Ubuntu_SecFun2.ova"^
 "C:\Users\student\Desktop\Ubuntu_SecFun2.ova"
set end=%time:~0,8%
echo %computername%,%start%,%end%>>^
 "\\itfile1\Public\Jevgeni\Thesis\Script_results.txt"
del "C:\Users\student\Desktop\Ubuntu_SecFun2.ova"
```

Figure 4. Testing script

The first line is creating a variable called start which records starting time, which is limited to seconds. The second line is copying the VM file from the shared network storage to the classroom machine. The third line is creating a variable called end which records ending time, which is limited to seconds. The fourth line is writing three variables, computer name, start time and end time to a predefined file to store the results. The predefined file is stored on the shared network storage for all machines to have an ability to access it. The last line deletes the VM file from the classroom machine to return the machine to the stage before the test was run.

The second test is the network speed test using OpenSpeedTest. A virtual machine was created to be able to run the OpenSpeedTest application on the server. Itxcpng02 was chosen to host the VM, as it runs less VMs. Therefore, it has more available resources, as well as less interference with the important VMs. Ubuntu Server 22.04 operating system was chosen for the VM, as it consumes less resources than Windows and it is simple to operate. Instead of compiling the application from the sources an easier solution was used. This way, installation of a separate web server and configuration of it was avoided.

It was decided to download an installation file for OpenSpeedTest server application. Because Ubuntu Server was chosen as the OS, it lacked graphical user interface, meaning that everything had to be done through command line interface. This created a small problem, downloading the file directly to the VM was not successful. Trying to download the file using curl or wget utilities resulted in HTTP 403 error code, which means that the client is forbidden from accessing the resource. Then, the file was downloaded to a separate workstation and transferred to the VM using Secure Copy Protocol. After the file was on the VM, OpenSpeedTest server was installed to the VM. When trying to run the newly installed server application, it gave an error of missing the display component. To resolve it, a display manager and desktop environment were installed. As a display manager lightdm was chosen and as for desktop environment ubuntu-desktop was chosen. After installing the required software and rebooting the VM, it was possible to launch the server application. The views of launched server application can be seen in Figure 5 and 6 below. Figure 5 shows the server view and Figure 6 shows the client view.

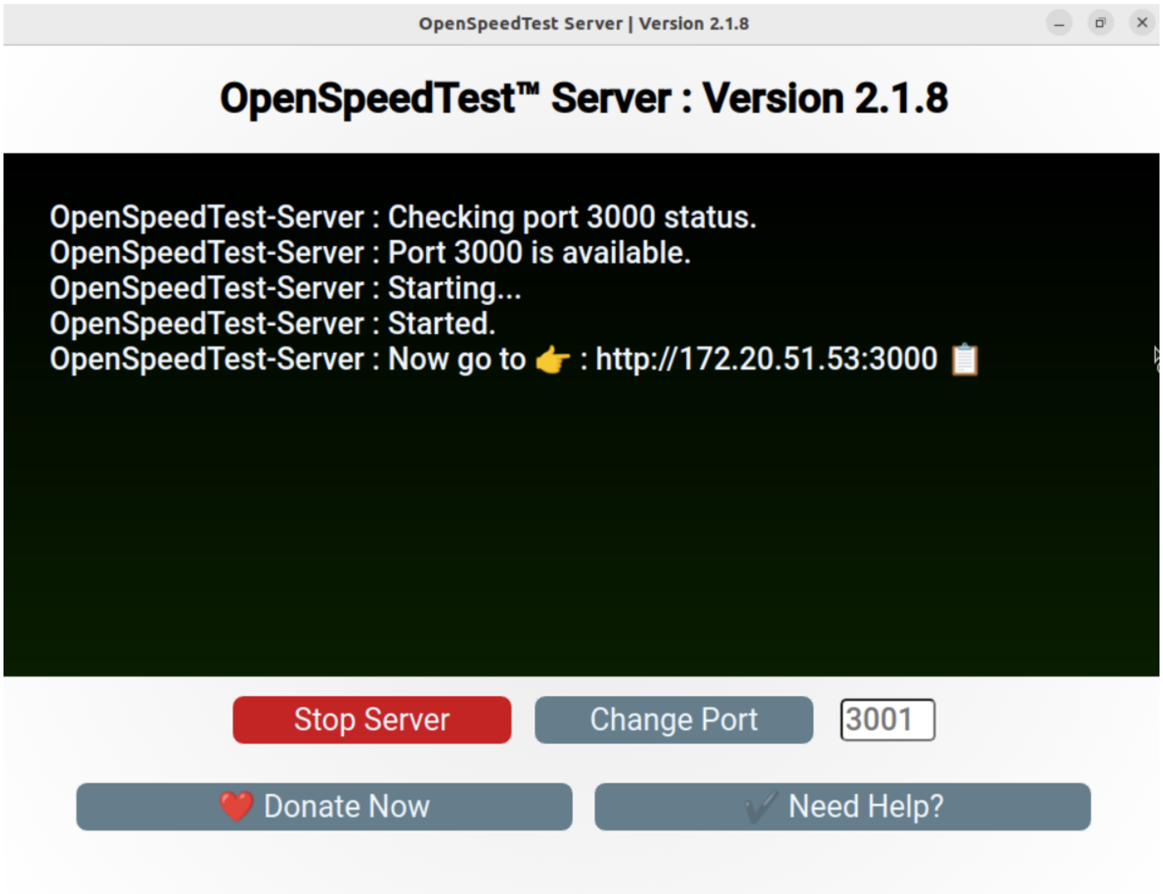

Figure 5. OpenSpeedTest server view

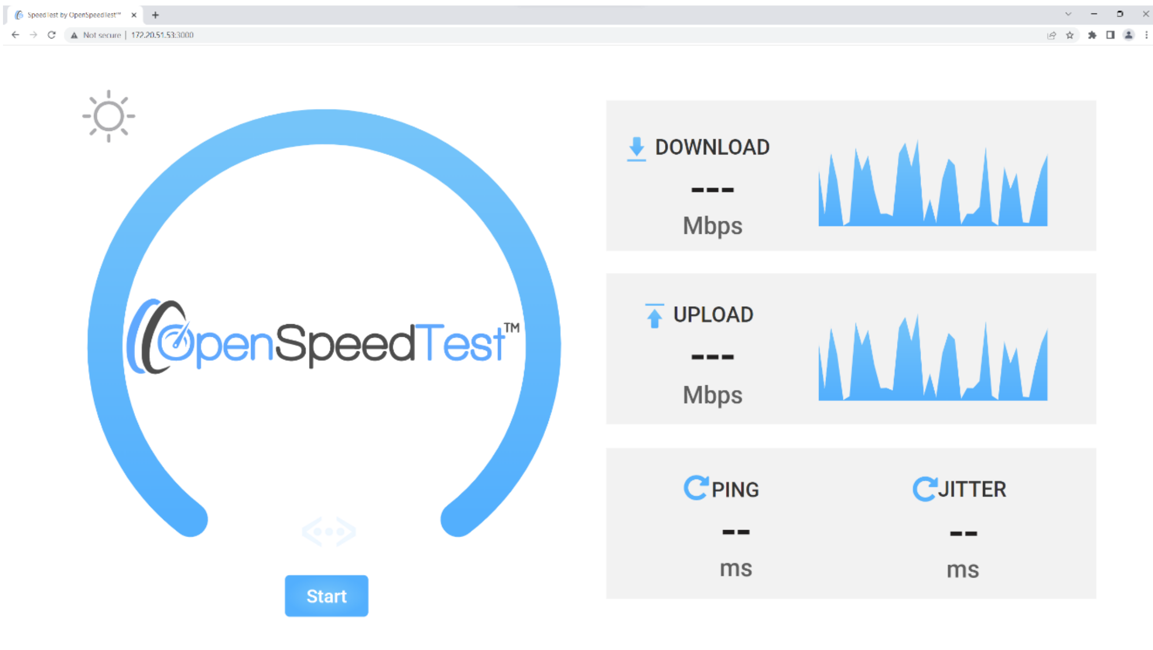

Figure 6. OpenSpeedTest client view

The last test is the ping test which is testing the redundancy of the connection. It does not need any additional steps as everything is already built into the operating systems.

6.1 Initial configuration testing

Initial configuration testing took place on 27th of February 2023. This date was chosen because it was Monday of a holiday week. This means that no students were present, and the classroom was empty. To run the first test on the classroom machines simultaneously a scheduled task was created. The scheduled task was set to run the script at the set time on 16 classroom machines. The set time for the task was at three at night. This time was chosen for two reasons. Firstly, the classroom machines were on at this time after they had completed installing updates and secondly, to verify that the scheduled tasks and the script work. Both, the script and the scheduled task, were tested beforehand, however, double-checking is a good practice. The idea was that if the test does not work the way it was intended, there would be time to fix it and run it on the same day multiple times. However, the test ran successfully, and it was decided to run it twice more, to get more consistent results. The second and the third test were run at 11:30 and 12:30 on the same day. Additionally, after running each test, a single machine test was run to get reference values for what the single machine can achieve in a condition with no load.

After the first test was run three times, and additional single machine tests, the results were moved from the text file to the excel file using ‘Get Data from text file’ option. After that, a few variables were added, and a few were calculated to the table. Those variables were Duration, Duration in seconds, File size in MB, Average transfer speed in MB/s, and Average transfer speed in Mbit/s. The tables containing the values for each machine for each run can be found in Appendix 1, and the table containing the values for the single machine can be found in Appendix 1 as well. The averages for the values for each run can be seen in Table 1.

Table 1. Script test result averages before the link aggregation

Computer Name	Duration	Duration in seconds	File Size (MB)	Average speed (MB/s)	Average speed (Mbit/s)
Average 1st run:	0.13.45	824.50	7057	8.73	70
Average 2nd run:	0.13.33	813.12	7057	9.30	74
Average 3rd run:	0.13.34	814.38	7057	9.29	74
Average single host:	0.01.12	71.67	7057	98.47	788

As can be seen from Table 1, the time to transfer the VM file from the shared network storage to 16 classroom machines at the same time takes significantly longer than for a single machine. On average, it took around thirteen and half minutes to get the file transferred. It is twelve minutes longer than the single machine time. An interesting result can be found from Appendix 2 and Appendix 3. MB316-01 machine had completed the file transfer in less than half of the average time. This means that the bandwidth was not spread equally.

In the situation which inspired the thesis, 2 VM files had to be transferred. On average they were the same size as the VM file in the script test. This means that for sixteen students to start that lab, it would take close to 30 minutes of waiting for VM files to be transferred. That time is a third of the lab time.

The second test, which the idea of is to complement the first test was run later, on the same day. It was also run through the scheduled task, however, for it to work, the user running the task had to be logged in to see the results on the screen. This scheduled task was run with a command to run a command in command prompt which opens chrome in incognito mode and immediately runs the speed test application for five minutes in each direction, meaning five minutes download speed test and five minutes upload speed test. The command used looked like this: *start chrome --incognito "http://172.20.51.53:3000?R&S=L"*. Start command is responsible for starting an application, the application is in this instance chrome, incognito parameter is chosen due to OpenSpeedTest recommending using the mode, and last comes the link which should be opened. The link is the same link as the OpenSpeedTest server application is showing, however two arguments are added. The first argument 'R' is responsible for running the test as soon as the link is accessed, bypassing the start button. The second argument 'S=L' is responsible for running the test for five minutes each direction. S stands for stress and L stands for low.

After the second test was run, the results were recorded manually by transferring the results from the screens of the machines to an Excel file. The table with results contains Computer Name, Download Speed, Upload speed, Ping, and Jitter columns. The table contains each machine results, the average values for forementioned parameters, total download and upload values, as well as the single machine results. The results can be seen in Table 2.

Table 2. Openspeedtest results before the link aggregation

Computer Name	Download speed (Mbit/s)	Upload speed (Mbit/s)	Ping (ms)	Jitter (ms)
MB316-01	59.0	66.9	2.0	0.7
MB316-02	62.7	57.5	18.0	2.0
MB316-03	67.2	59.9	6.0	40.0
MB316-04	57.4	66.5	6.0	0.4
MB316-05	58.0	64.3	6.0	0.4
MB316-12	52.9	64.8	7.0	0.8
MB316-13	65.9	59.7	8.0	0.6
MB316-14	50.8	57.8	17.0	0.8
MB316-15	58.8	63.0	3.0	0.5
MB316-16	57.9	70.9	8.0	2.0
MB316-23	62.3	60.4	7.0	1.0
MB316-24	61.6	60.0	1.0	0.4
MB316-25	57.2	65.3	2.0	0.6
MB316-26	50.3	65.4	7.0	2.0
MB316-27	56.2	59.5	6.0	0.5
MB316-34	46.7	52.1	1.0	0.3
Average:	57.8	62.1	6.6	3.3
Total:	924.9	994.0		
Single host	984.9	981.8	2.0	0.5

As can be seen in Table 2, the average download speed is slightly less than average file download speed in the previous test. That is due to test running for a certain time on all test machines rather than in previous test some machines were done before others meaning leftover machines had higher speed. The total download speed and the total upload speed is in a margin of error with the single host speeds. The bandwidth of the connection can be calculated by combining download or upload speeds of the individual machines. This indicates that full one Gigabit bandwidth was used in both cases, and the VM running the speed test application was not bottlenecking the connection.

The last test was run after the second one. For this test, the console of the server switch was accessed to be able to disable the port connecting the server switch and the classroom switch. Then, from one of the classroom machines, ping was initiated to the shared network storage server with the following command: *ping itfile1 -t*. This command starts a ping and sends ping packets until it is stopped. The results can be seen in Figure 7.

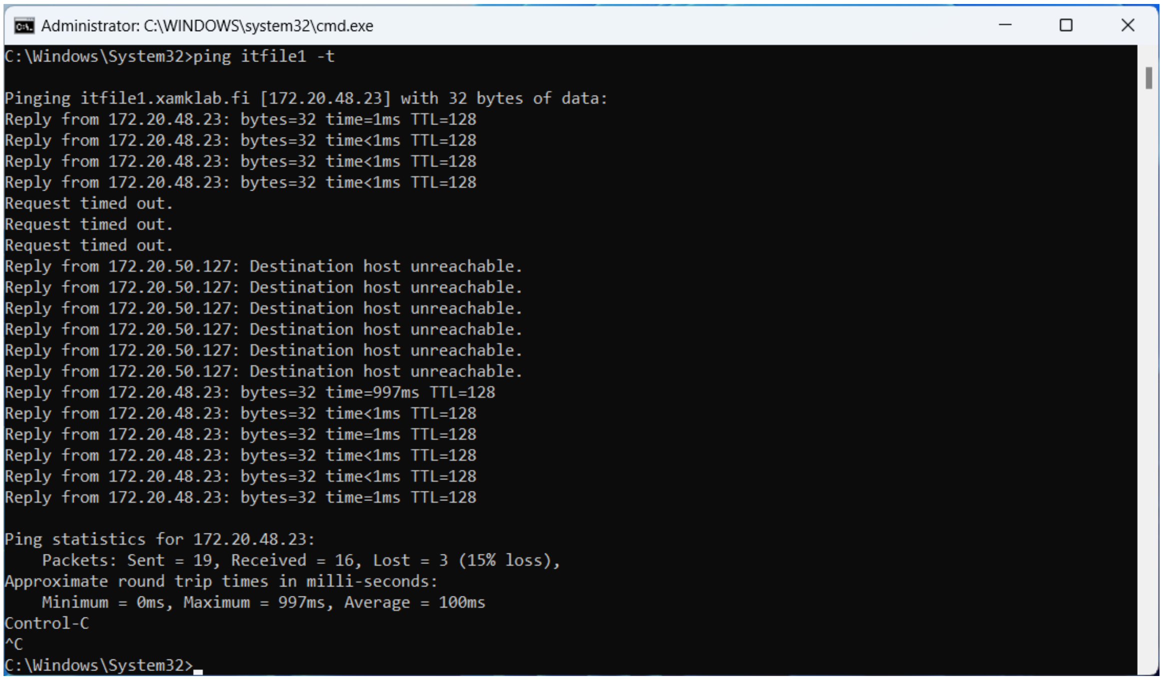

Figure 7. Ping test before the link aggregation

After the ping test was run, the result was as expected. Disconnecting the single connection between the classroom switch and the server switch resulted in the loss of connection, therefore ping packets were dropped. As soon as the connection was reset, the pings were successful again.

6.2 Updated configuration testing

Updated configuration testing took place on 2nd March 2023. This date was chosen because it was the next day after the implementation and it was ready for testing. This was still part of the holiday week, meaning that no students were present, and the classroom was empty. This should give the most accurate comparison with the least variables changed between the initial and updated configuration tests. To run the first test on the classroom machines simultaneously a scheduled task was recreated. As before, the scheduled task was set to run the script at the set time on 16 classroom machines. The set time for the task was at ten in the noon. As can be seen, it does not match the time when the initial configuration test was run. The reason for choosing this time over three at night was to be present while the test was running to be able to monitor the process. The test ran successfully, and it was decided to run it two times more, to confirm the results. The second and the third test were run at 11:30 and

12:00 on the same day. Just as with the initial configuration testing, single machine tests were run as well to have a baseline for a single device's performance and to see if there are any changes.

As before, after the first tests were run, the results were handled in an identical way to the initial configuration test results. The tables containing the values for each machine for each run can be seen in Appendix 2. It should be noted that Table 10 is missing MB316-16 results and Table 11 is missing MB316-15 results. It is certain that they ran the test, as the test was overseen, and the results are close to the first run. The most probable reason for this situation is that two machines edited the file at the same time, but only one of the edits was saved. The table containing the values for the single machine can be seen in Appendix 2. Averages for the values for each run can be seen in Table 3.

Table 3. Script test result averages after the link aggregation

Computer Name	Time Difference	Time Difference in seconds	File Size (MB)	Average speed (MB/s)	Average speed (Mbit/s)
Average 1st run:	0.05.01	301.00	7057	23.56	188
Average 2nd run:	0.05.13	313.40	7057	22.53	180
Average 3rd run:	0.05.08	307.73	7057	22.94	184
Average single host:	0.01.12	72.00	7057	98.03	784

As can be seen from the table above, the time to transfer the VM file from the shared network storage to 16 classroom machines at the same time still takes longer than for a single machine. However, now, on average, it took around five minutes to get the file transferred. It is four minutes longer than the single machine time.

In the situation which inspired the thesis, 2 VM files had to be transferred. On average they were the same size as the VM file in the script test. This means that for sixteen students to start that lab, it would take close to ten minutes of waiting for VM files to be transferred. It might be still a bit long, but the time could be used to read through the instructions to understand the scope of the lab.

As with the initial configuration testing, the second test was run on the same day after the first test. It was run using the same method of running the command to start the test using the scheduled task. The results were recorded using the manual method as well. The variables in the table are identical to the initial configuration testing. The results can be seen in Table 4.

Table 4. Openspeedtest result after the link aggregation

Computer name	Download speed (Mbit/s)	Upload speed (Mbit/s)	Ping (ms)	Jitter (ms)
MB316-01	276.3	167.8	2.0	0.2
MB316-02	296.3	216.7	1.0	0.5
MB316-03	215.8	164.2	9.0	1.0
MB316-04	318.8	216.9	8.0	1.0
MB316-05	251.4	245.4	3.0	0.8
MB316-12	269.7	191.3	4.0	0.9
MB316-13	106.2	136.4	11.0	1.0
MB316-14	238.1	163.9	2.0	0.5
MB316-15	302.6	213.3	12.0	2.0
MB316-16	240.4	149.8	3.0	0.3
MB316-23	235.6	171.5	8.0	1.0
MB316-24	153.8	221.9	6.0	3.0
MB316-25	352.4	216.2	7.0	1.0
MB316-26	261.5	208.6	23.0	3.0
MB316-27	229.6	220.7	2.0	0.2
MB316-34	194.9	174.4	1.0	0.3
Average	246.5	192.4	6.4	1.0
Total	3943.4	3079.0		
Single host	986.8	996.1	2.0	0.3

As can be seen from the table above, the speed results have improved. The total bandwidth of the connection can be calculated by combining download or upload speeds of the individual machines. This test shows that the bandwidth has increased to four Gigabits as can be seen from the total value of download speeds. However, the upload speed total value is only a bit over three Gigabits. This is presumably due to the Openspeedtest VM not having enough resources to cope with the test. As the download speed part is more client heavy and the upload speed part is more server heavy. The single machine test is limited to one Gigabit bandwidth since the classroom computers have one Gigabit capable network adapters.

The last test was run after the second one. The preparation for the test was identical to the initial configuration test. The console of the server switch was accessed, and from one of the classroom machines, ping was initiated to the shared network storage server. During this test multiple ports were disabled to test various configurations.

After the test, the results were different from the initial configuration test. It was expected that no ping packets would be lost, and it was true for the case if the port which got disabled was not transmitting the packets. In case the port was transmitting the packets, then one of the ping packets was dropped, however then, as expected, other ports took over the transmission and ping continued successfully. In this configuration, three ports could be disabled, and the network connection would be available to the classroom, although at lower bandwidth. The results of one ping packet being dropped in case of disabling the transmitting port can be seen in Figure 8.

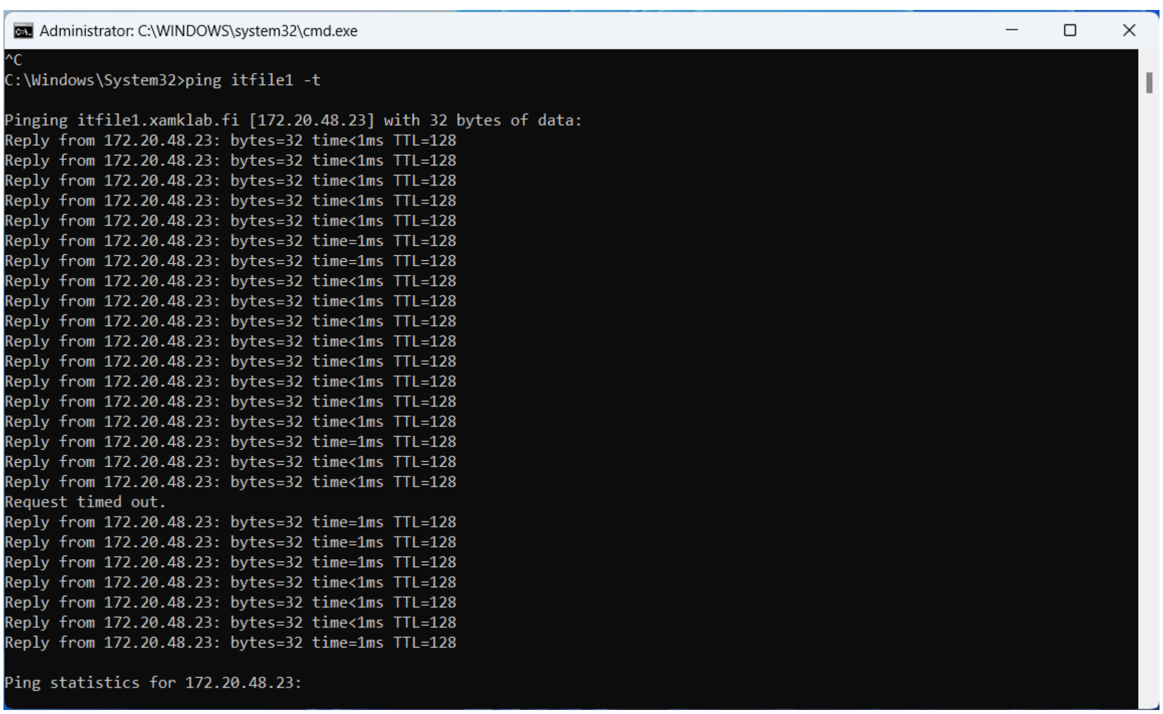

Figure 8. Ping test after the link aggregation

The expected result for the ping test was that no packets will be dropped, however in one of the cases, that is not true. The results are not as good as expected but the fact that the dropout is just for a single packet is the second-best result.

6.3 Comparison of the results

Three tests were run on both configurations. The first test was testing bandwidth and was run three times to get accurate results. The second test was complimentary to the first one and was run once. The third test was testing redundancy rather than bandwidth and was run once.

6.3.1 Comparison of the script results

All results for the script test can be found in Appendices 1 and 2. From the tables in Appendices 1 and 2 averages of each machine of three runs were used to compose graphs with the average duration in seconds and the average transfer speed in Mbit/s. These graphs can be seen in Figures 9 and 10. Individual graphs for each run before and after the link aggregation can be seen in Appendices 3 and 4.

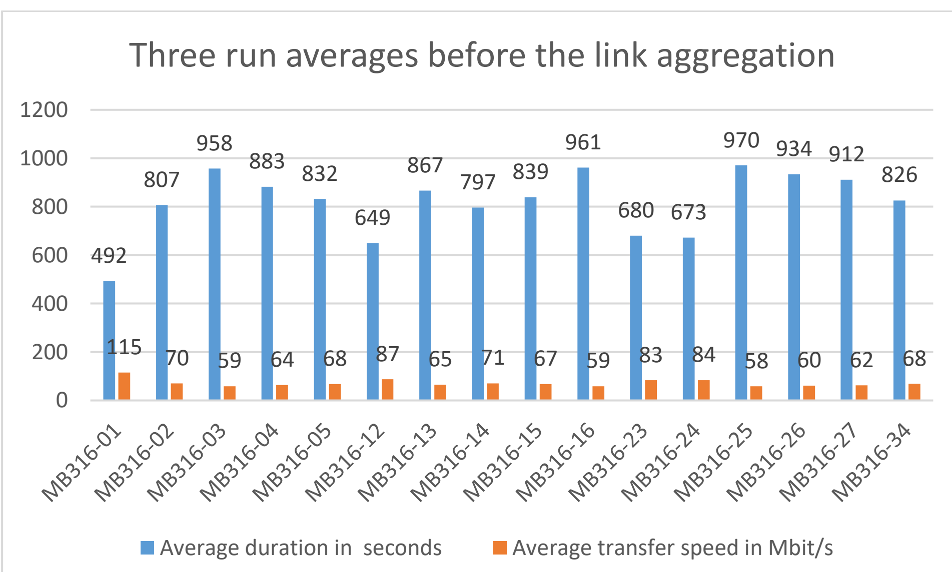

Figure 9. Script test three run averages before the link aggregation

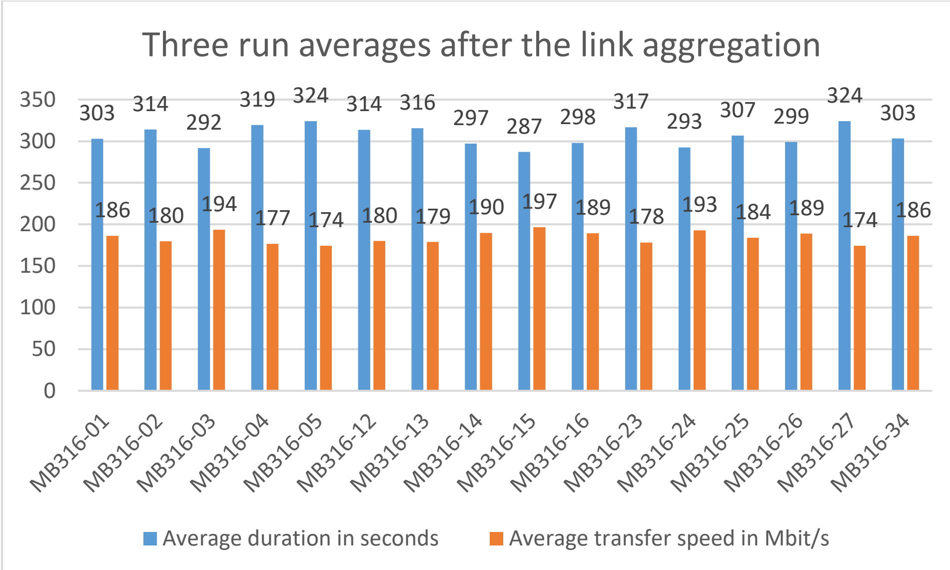

Figure 10. Script test three run averages after the link aggregation

In Figures 9 and 10, it can be seen that the average transfer speed increased, and the average transfer time decreased, as well as the delta of lowest and highest values decreased after the implementation of the link aggregation. The fastest average transfer time before the link aggregation was 492 seconds and the slowest was 970 seconds, resulting in a delta of 478 seconds. After the link aggregation, the fastest average transfer time was 287 seconds and the slowest was 324 seconds, resulting in a delta of 37 seconds. The average highest transfer speed before the link aggregation was 115 Mbit/s and the lowest was 58 Mbit/s, resulting in a delta of 57 Mbit/s. After the link aggregation, the average highest transfer speed was 324 Mbit/s and the lowest was 292 Mbit/s, resulting in a delta of 32 Mbit/s.

The first test was also run on a single machine to get a baseline before and after the link aggregation. The comparison of a single machine script test results can be seen in Figure 11.

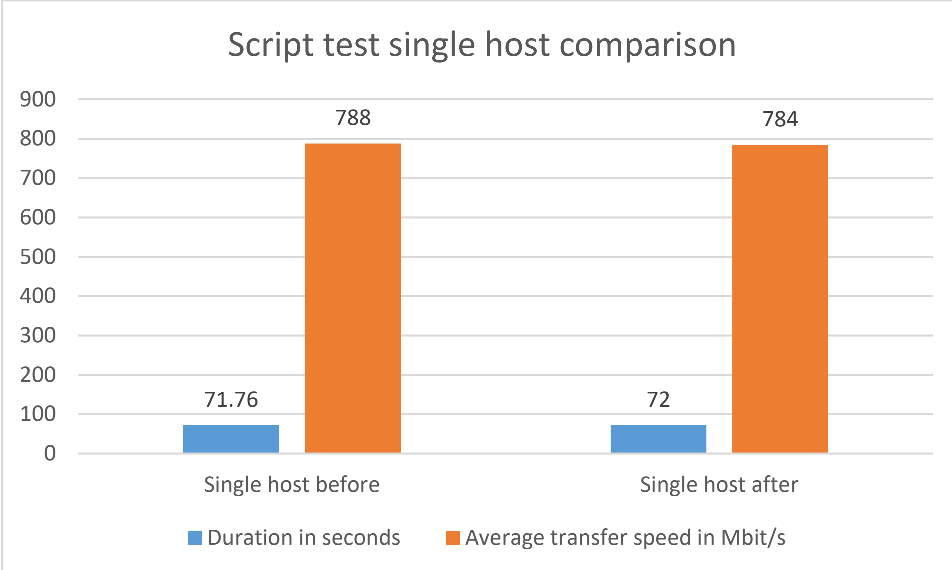

Figure 11. Script test single host comparison

As the single machine was not utilising the full bandwidth on this test even before the link aggregation. It was expected that results will stay same.

6.3.2 Comparison of the OpenSpeedTest results

The second test results from Tables 2 and 4 were transformed into multiple graphs. Two first graphs are download speed results before and after the link aggregation and two graphs after those are upload speed results before and after the link aggregation. Those four graphs can be seen in Figures 12-15.

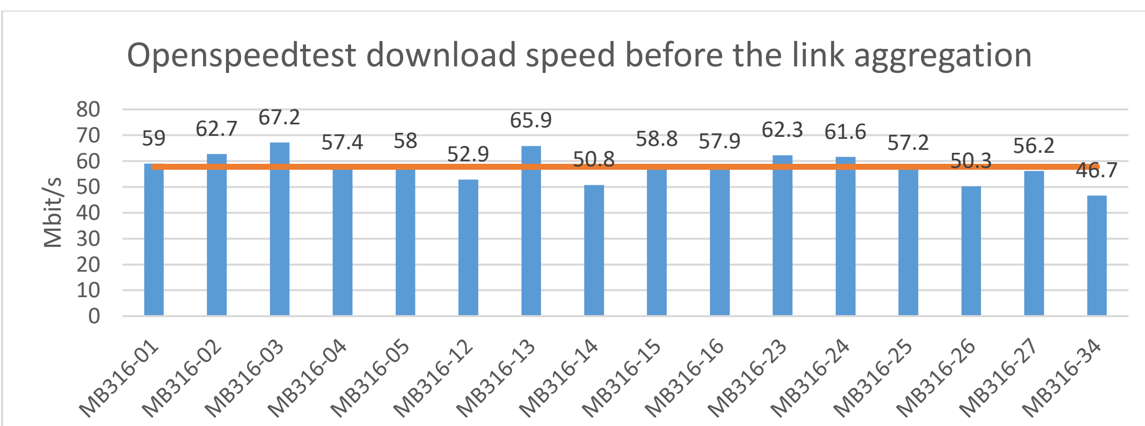

Figure 12. Openspeedtest download speed before the link aggregation

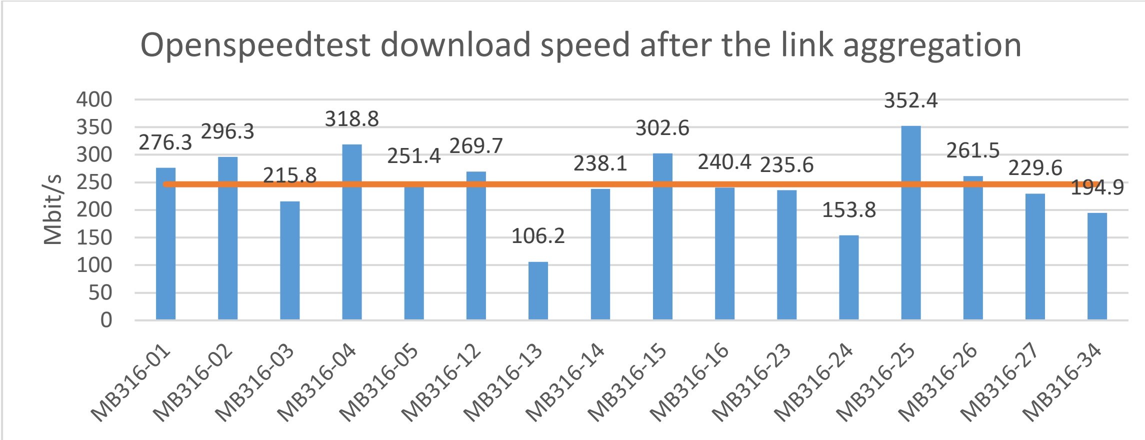

Figure 13. Openspeedtest download speed after the link aggregation

In Figures 12 and 13, it can be seen that the download speed increased. If before the link aggregation the lowest speed was 46,7 Mbit/s, the highest speed was 67,2 Mbit/s, the delta was 20,5 Mbit/s, and the average speed was 57,8 Mbit/s, then after the lowest was 106,2 Mbit/s, the highest speed was 352,4 Mbit/s, the delta was 246,2 Mbit/s, and the average speed was 246,5 Mbit/s. Overall, it is a good outcome, as the download speed increased and the lowest of the after results is higher than the highest of the before results. However, one part of the test in which before results look better is the delta. This result contradicts the script test results where the link aggregation improved the consistency of the performance between different machines. The difference could be from the script test being more of a real-world test, which represented an action, which is more likely to be performed by users. Whereas OpenSpeedTest created a synthetic load. However, the before results contradict this somewhat. Another reason for the increase in delta could be increase of load on the VM hosting the OpenSpeedTest application, which could have been configured with less resources than necessary.

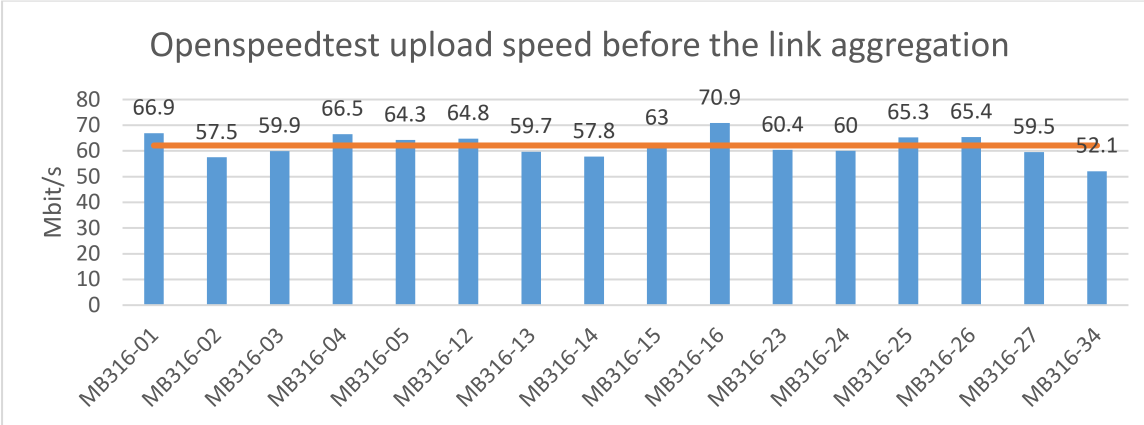

Figure 14. Openspeedtest upload speed before the link aggregation

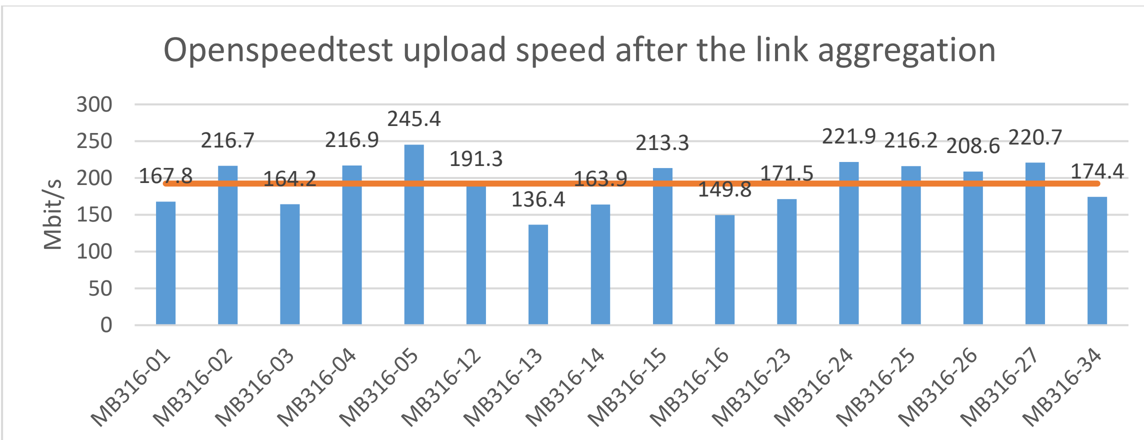

Figure 15. Openspeedtest upload speed after the link aggregation

In Figures 14 and 15, it can be seen that the upload speed increased as well. If before the link aggregation the lowest speed was 52,1 Mbit/s, the highest speed was 70,9 Mbit/s, the delta was 18,8 Mbit/s, and the average speed was 62,1 Mbit/s, then after the lowest was 136,4 Mbit/s, the highest speed was 245,4 Mbit/s, the delta was 109 Mbit/s, and the average speed was 192,4 Mbit/s. Overall, the results look in general same as the download speed results. The lowest of the after results is higher than the highest of the before, and the only value where before test results are better is the delta. Another finding is that the difference between the download and upload averages is higher for the test performed after the implementation. However, as it was mentioned before, that is presumably due to the VM lacking the resources, which also supports the explanation for the higher delta values.

The link aggregation had no effect on the ping time nor the jitter time. The mentioned values can be seen in Tables 2 and 4. There was no recorded difference between the ping or the jitter values. That is the reason they were not discussed before, nor will be discussed later.

Overall, the first and second test results showed that the link aggregation improved the network by successfully increasing the bandwidth. Higher bandwidth resulted in higher download and upload speeds when multiple machines were loading the network. However, when a single machine performance stayed the same due to limitation of the machine's hardware.

6.3.3 Comparison of the ping results

The ping test revealed that before the link aggregation, if one of the ports in the chain between the classroom machine and the server was disabled, then the classroom machine would lose the network connection. However, after the link aggregation, if one of the four ports in the link aggregation was disabled, the others would take over the transmission. The ping test results show that the link aggregation had a positive effect on the network connection as well. It added redundancy.

6.3.4 Discussion

This subchapter is a quick overview of the performed tests, results of the tests, and how they could be improved.

The first test was a script test, in which the idea was to copy a file from a network storage to the local storage and record start and end times. That was automated using a script and a scheduled task. One part which should be changed if it is replicated is the file where the results are stored. Each machine should create its own file to remove the possibility of two machines writing to the same file at the same time and one of the results not being recorded.

The second test was performed using OpenSpeedTest application. Using a scheduled task and URL parameters, each machine performed the test at the same time and with the same parameters. One part which could be improved is, if it is supported by the environment, running the application in a docker container. It was possible to run the docker container in a VM, but that would create another layer of virtualization in the current environment. It is rumored that XCP-NG will have container support in the future. Another part is adding testing to see how many resources OpenSpeedTest application needs to eliminate possibility of application being the bottleneck.

The third test was a ping test. The test was performed by sending ping packets from the classroom machine to the network shared storage server and disabling port(s) on the server switch to imitate the port failure. To improve the test, the port(s) could be disabled from the classroom switch or the server hosting the VM, to make sure that the link aggregation works on all links. Another improvement could be sending ping packets from more than one machine.

7 CONCLUSION

The aim of the thesis was to improve the Xamklab network by means of the link aggregation. The link aggregation was meant to increase the bandwidth and add redundancy. Link-Aggregation Control Protocol was used as planned as all the devices supported it. The aim was achieved according to the results of multiple tests. The redundancy between the classroom machines and the servers was achieved, and the bandwidth was increased nearly four times. No new devices were bought resulting in zero e-waste.

Three tests were run to measure the bandwidth and the redundancy of the network. Two tests were testing the bandwidth and one was testing the redundancy. All three tests showed that the link aggregation had improved the aspects of testing. However, none of the tests were perfect and could be improved.

The Xamklab network can be continued to be improved by adding the link aggregation to other classrooms. The study itself can be continued by adding Windows Server Update Service which helps to manage distribution of updates for Microsoft product. It downloads and stores the updates locally, which can be distributed at any time. This solution combined with the link aggregation would make the updating classroom machines and servers easier and quicker.

REFERENCES

About openSpeedtest.com. n.d. OpenSpeedTest. Web page. Available at: https://openspeedtest.com/about-speed-test [Accessed 20 February 2023].

Aruba 2540 switch series. 2020. Aruba. PDF document. Available at: https://www.arubanetworks.com/assets/ds/DS_2540SwitchSeries.pdf [Accessed 20 February 2023].

Aruba 2530 switch series – specifications. n.d. Hewlett Packard Enterprise. Web page. Available at: https://support.hpe.com/hpesc/public/docDisplay?docId=emr_na-c03624915#N113F7 [Accessed 20 February 2023].

batch file. n.d. The Computer Language Co Inc. Web page. Available at: https://www.pcmag.com/encyclopedia/term/batch-file [Accessed on 17 March 2023].

Charlene. 2022. Understanding PoE Standards and PoE Wattage. Web page. Available at: https://community.fs.com/blog/understanding-poe-standards-and-poe-wattage.html [Accessed on 14 March 2023].

iperf3 FAQ. 2017. ESnet. Web page. Available at: https://software.es.net/iperf/faq.html [Accessed 13 March 2023].

iPerf - The ultimate speed test tool for TCP, UDP and SCTP. n.d. iPerf. Web page. Available at: https://iperf.fr/ [Accessed 13 March 2023].

Kuchenski, D. 2013. LACP TRUNK ON HP PROCURVE SWITCH. Web article. Available at: https://thenetworkstack.com/lacp-trunk-on-hp-procurve-switch/ [Accessed on 3 April 2023]

LACP Bonding in XenServer - Configuration and Troubleshooting. 2014. Citrix. Web page. 08 August 2018. Available at: https://support.citrix.com/article/CTX135690/lacp-bonding-in-xenserver-configuration-and-troubleshooting [Accessed on 3 April 2023]

Link Aggregation Overview. n.d. Cisco. Web page. Available at: https://www.cisco.com/assets/sol/sb/Switches_Emulators_v2_3_5_xx/help/250/index.html#page/tesla_250_olh/aggregating_ports.html [Accessed 20 February 2023].

Zola, A. 2021. ping. Blog. July 2021. Available at: https://www.techtarget.com/searchnetworking/definition/ping [Accessed on 17 March 2023].

Sands, J. & Lambert, O. 2022. Networking bonds in XCP-ng. Blog post. 13 September 2022. Available at: https://xcp-ng.org/blog/2022/09/13/network-bonds-in-xcp-ng/ [Accessed 20 February 2023].

script. n.d. The Computer Language Co Inc. Web page. Available at: https://www.pcmag.com/encyclopedia/term/script [Accessed on 17 March 2023].

Seaman, M. 1999. Link Aggregation Control Protocol. PDF document. Available at: https://www.ieee802.org/3/ad/public/mar99/seaman_1_0399.pdf [Accessed on 17 March 2023].

Sun Ethernet Fabric Operating System. 2012. Oracle. PDF document. Available at: https://docs.oracle.com/cd/E19285-01/pdf/E21709.pdf [Accessed on 20 March 2023].

Terabit Ethernet: The New Hot Trend in Data Centers. n.d. Lanner Electronics Canada. Web page. Available at: https://www.lanner-america.com/blog/terabit-ethernet-new-hot-trend-data-centers/ [Accessed 14 March 2023].

The History of Ethernet. 2006. NetEvents.tv. Video clip. Available at: https://ghostarchive.org/varchive/g5MezxMcRmk [Accessed 20 February 2023].

XCP-NG. n.d. Development teams. Web page. Available at: https://xenproject.org/developers/teams/xcp-ng/ [Accessed 20 February 2023].

Table 5. Script test first run before the link aggregation

Computer Name	Start Time	End Time	Duration	Duration in seconds	File Size (MB)	Average speed (MB/s)	Average speed (Mbit/s)
MB316-12	3.00.00	3.11.04	0.11.04	664	7057	10.63	85
MB316-02	3.00.00	3.11.07	0.11.07	667	7057	10.58	85
MB316-13	3.00.00	3.11.15	0.11.15	675	7057	10.45	84
MB316-04	3.00.00	3.11.34	0.11.34	694	7057	10.17	81
MB316-14	3.00.01	3.12.24	0.12.23	743	7057	9.50	76
MB316-01	3.00.00	3.12.52	0.12.52	772	7057	9.14	73
MB316-23	3.00.00	3.13.06	0.13.06	786	7057	8.98	72
MB316-24	3.00.00	3.13.13	0.13.13	793	7057	8.90	71
MB316-15	3.00.00	3.13.28	0.13.28	808	7057	8.73	70
MB316-16	3.00.00	3.14.35	0.14.35	875	7057	8.07	65
MB316-05	3.00.00	3.14.44	0.14.44	884	7057	7.98	64
MB316-03	3.00.00	3.15.20	0.15.20	920	7057	7.67	61
MB316-34	3.00.01	3.15.40	0.15.39	939	7057	7.52	60
MB316-27	3.00.01	3.16.11	0.16.10	970	7057	7.28	58
MB316-25	3.00.01	3.16.40	0.16.39	999	7057	7.06	57
MB316-26	3.00.00	3.16.43	0.16.43	1003	7057	7.04	56
Average:			**0.13.45**	**824.50**	**7057**	**8.73**	**70**

Table 6. Script test second run before the link aggregation

Computer Name	Start Time	End Time	Duration	Duration in seconds	File Size (MB)	Average speed (MB/s)	Average speed (Mbit/s)
MB316-01	11.30.00	11.35.52	0.05.52	352	7057	20.05	160
MB316-24	11.30.00	11.40.16	0.10.16	616	7057	11.46	92
MB316-23	11.30.01	11.40.43	0.10.42	642	7057	10.99	88
MB316-12	11.30.00	11.40.45	0.10.45	645	7057	10.94	88
MB316-34	11.30.00	11.42.42	0.12.42	762	7057	9.26	74
MB316-14	11.30.00	11.42.55	0.12.55	775	7057	9.11	73
MB316-05	11.30.01	11.43.26	0.13.25	805	7057	8.77	70
MB316-15	11.30.00	11.43.44	0.13.44	824	7057	8.56	69
MB316-27	11.30.00	11.44.32	0.14.32	872	7057	8.09	65
MB316-02	11.30.00	11.44.58	0.14.58	898	7057	7.86	63
MB316-26	11.30.00	11.45.38	0.15.38	938	7057	7.52	60
MB316-25	11.30.01	11.45.52	0.15.51	951	7057	7.42	59
MB316-13	11.30.02	11.46.09	0.16.07	967	7057	7.30	58
MB316-03	11.30.00	11.46.19	0.16.19	979	7057	7.21	58
MB316-04	11.30.00	11.46.20	0.16.20	980	7057	7.20	58
MB316-16	11.30.00	11.46.44	0.16.44	1004	7057	7.03	56
Average:			**0.13.33**	**813.12**	**7057**	**9.30**	**74**

Table 7. Script test third run before the link aggregation

Computer Name	Start Time	End Time	Duration	Duration in seconds	File Size (MB)	Average speed (MB/s)	Average speed (Mbit/s)
MB316-01	12.30.01	12.35.54	0.05.53	353	7057	19.99	160
MB316-24	12.30.01	12.40.10	0.10.09	609	7057	11.59	93
MB316-23	12.30.02	12.40.13	0.10.11	611	7057	11.55	92
MB316-12	12.30.02	12.40.41	0.10.39	639	7057	11.04	88
MB316-34	12.30.01	12.42.58	0.12.57	777	7057	9.08	73
MB316-05	12.30.02	12.43.28	0.13.26	806	7057	8.76	70
MB316-02	12.30.01	12.44.17	0.14.16	856	7057	8.24	66
MB316-26	12.30.01	12.44.21	0.14.20	860	7057	8.21	66
MB316-14	12.30.01	12.44.33	0.14.32	872	7057	8.09	65
MB316-15	12.30.01	12.44.45	0.14.44	884	7057	7.98	64
MB316-27	12.30.01	12.44.54	0.14.53	893	7057	7.90	63
MB316-13	12.30.02	12.46.00	0.15.58	958	7057	7.37	59
MB316-25	12.30.02	12.46.02	0.16.00	960	7057	7.35	59
MB316-03	12.30.01	12.46.15	0.16.14	974	7057	7.25	58
MB316-04	12.30.01	12.46.15	0.16.14	974	7057	7.25	58
MB316-16	12.30.01	12.46.45	0.16.44	1004	7057	7.03	56
Average:			**0.13.34**	**814.38**	**7057**	**9.29**	**74**

Table 8. Script test single host runs before the link aggregation

Computer Name	Start Time	End Time	Duration	Duration in seconds	File Size (MB)	Average speed (MB/s)	Average speed (Mbit/s)
Single host	9.16.16	9.17.28	0.01.12	72	7057	98.01	784
Single host	10.51.32	10.52.44	0.01.12	72	7057	98.01	784
Single host	10.59.02	11.00.13	0.01.11	71	7057	99.39	795
Average:			**0.01.12**	**71.67**	**7057**	**98.47**	**788**

Table 9. Script test first run after the link aggregation

Computer Name	Start Time	End Time	Time Difference	Time Difference in seconds	File Size (MB)	Average speed (MB/s)	Average speed (Mbit/s)
MB316-24	9.30.00	9.34.31	0.04.31	271	7057	26.04	208
MB316-03	9.30.01	9.34.36	0.04.35	275	7057	25.66	205
MB316-15	9.30.01	9.34.37	0.04.36	276	7057	25.57	205
MB316-14	9.30.00	9.34.37	0.04.37	277	7057	25.48	204
MB316-16	9.30.00	9.34.48	0.04.48	288	7057	24.50	196
MB316-26	9.30.00	9.34.48	0.04.48	288	7057	24.50	196
MB316-34	9.30.00	9.34.51	0.04.51	291	7057	24.25	194
MB316-25	9.30.00	9.34.56	0.04.56	296	7057	23.84	191
MB316-01	9.30.01	9.34.57	0.04.56	296	7057	23.84	191
MB316-12	9.30.01	9.35.13	0.05.12	312	7057	22.62	181
MB316-02	9.30.00	9.35.13	0.05.13	313	7057	22.55	180
MB316-13	9.30.01	9.35.17	0.05.16	316	7057	22.33	179
MB316-23	9.30.00	9.35.21	0.05.21	321	7057	21.98	176
MB316-04	9.30.00	9.35.26	0.05.26	326	7057	21.65	173
MB316-27	9.30.00	9.35.30	0.05.30	330	7057	21.38	171
MB316-05	9.30.00	9.35.40	0.05.40	340	7057	20.76	166
Average:			**0.05.01**	**301**	**7057**	**23.56**	**188**

Table 10. Script test second run after the link aggregation

Computer Name	Start Time	End Time	Time Difference	Time Difference in seconds	File Size (MB)	Average speed (MB/s)	Average speed (Mbit/s)
MB316-15	11.30.01	11.34.59	0.04.58	298	7057	23.68	189
MB316-24	11.30.00	11.35.01	0.05.01	301	7057	23.45	188
MB316-03	11.30.00	11.35.02	0.05.02	302	7057	23.37	187
MB316-26	11.30.01	11.35.04	0.05.03	303	7057	23.29	186
MB316-14	11.30.00	11.35.11	0.05.11	311	7057	22.69	182
MB316-34	11.30.00	11.35.11	0.05.11	311	7057	22.69	182
MB316-01	11.30.00	11.35.12	0.05.12	312	7057	22.62	181
MB316-25	11.30.00	11.35.14	0.05.14	314	7057	22.47	180
MB316-02	11.30.00	11.35.18	0.05.18	318	7057	22.19	178
MB316-04	11.30.00	11.35.20	0.05.20	320	7057	22.05	176
MB316-13	11.30.00	11.35.20	0.05.20	320	7057	22.05	176
MB316-12	11.30.00	11.35.22	0.05.22	322	7057	21.92	175
MB316-23	11.30.00	11.35.22	0.05.22	322	7057	21.92	175
MB316-27	11.30.00	11.35.23	0.05.23	323	7057	21.85	175
MB316-05	11.30.00	11.35.24	0.05.24	324	7057	21.78	174
Average:			**0.05.13**	**313.40**	**7057**	**22.53**	**180**

Table 11. Script test third run after the link aggregation

Computer Name	Start Time	End Time	Time Difference	Time Difference in seconds	File Size (MB)	Average speed (MB/s)	Average speed (Mbit/s)
MB316-03	12.00.01	12.04.59	0.04.58	298	7057	23.68	189
MB316-01	12.00.01	12.05.02	0.05.01	301	7057	23.45	188
MB316-14	12.00.01	12.05.05	0.05.04	304	7057	23.21	186
MB316-24	12.00.01	12.05.07	0.05.06	306	7057	23.06	184
MB316-23	12.00.01	12.05.08	0.05.07	307	7057	22.99	184
MB316-26	12.00.02	12.05.08	0.05.06	306	7057	23.06	184
MB316-12	12.00.02	12.05.09	0.05.07	307	7057	22.99	184
MB316-34	12.00.01	12.05.09	0.05.08	308	7057	22.91	183
MB316-05	12.00.01	12.05.09	0.05.08	308	7057	22.91	183
MB316-16	12.00.01	12.05.09	0.05.08	308	7057	22.91	183
MB316-25	12.00.01	12.05.11	0.05.10	310	7057	22.76	182
MB316-02	12.00.01	12.05.12	0.05.11	311	7057	22.69	182
MB316-13	12.00.01	12.05.12	0.05.11	311	7057	22.69	182
MB316-04	12.00.01	12.05.13	0.05.12	312	7057	22.62	181
MB316-27	12.00.01	12.05.20	0.05.19	319	7057	22.12	177
Average:			**0.05.08**	**307.73**	**7057**	**22.94**	**184**

Table 12. Script test single host runs after the link aggregation

Computer Name	Start Time	End Time	Time Difference	Time Difference in seconds	File Size (MB)	Average speed (MB/s)	Average speed (Mbit/s)
Single host	10.09.58	10.11.11	0.01.13	73	7057	96.67	773
Single host	11.51.44	11.52.55	0.01.11	71	7057	99.39	795
Single host	12.48.47	12.49.59	0.01.12	72	7057	98.01	784
Average:			**0.01.12**	**72.00**	**7057**	**98.03**	**784**

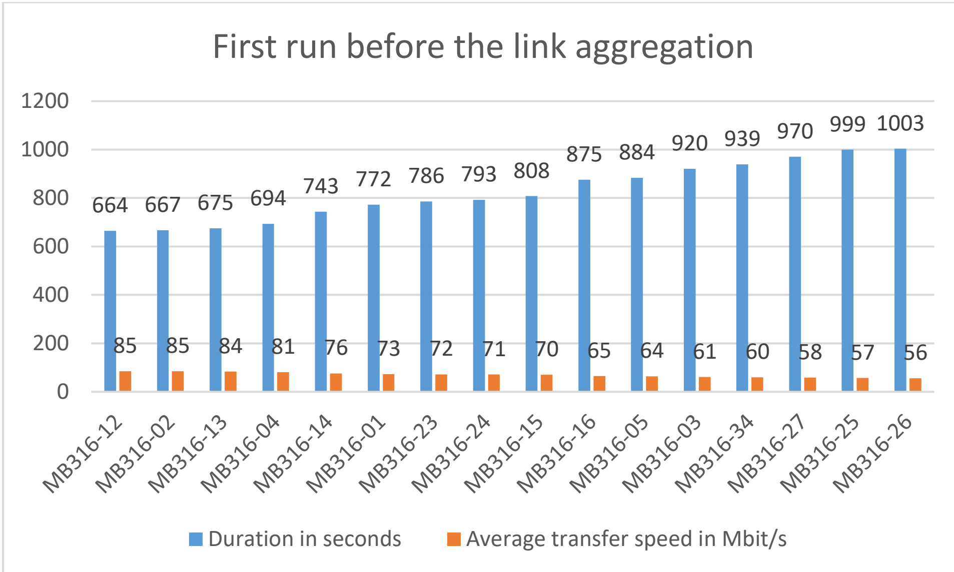

Figure 16. Script test first run before the link aggregation

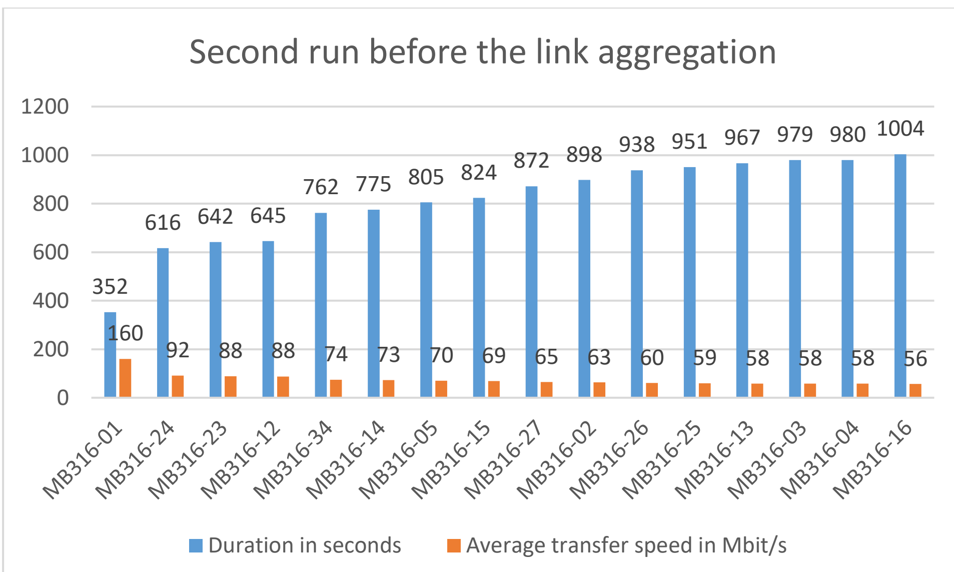

Figure 17. Script test second run before the link aggregation

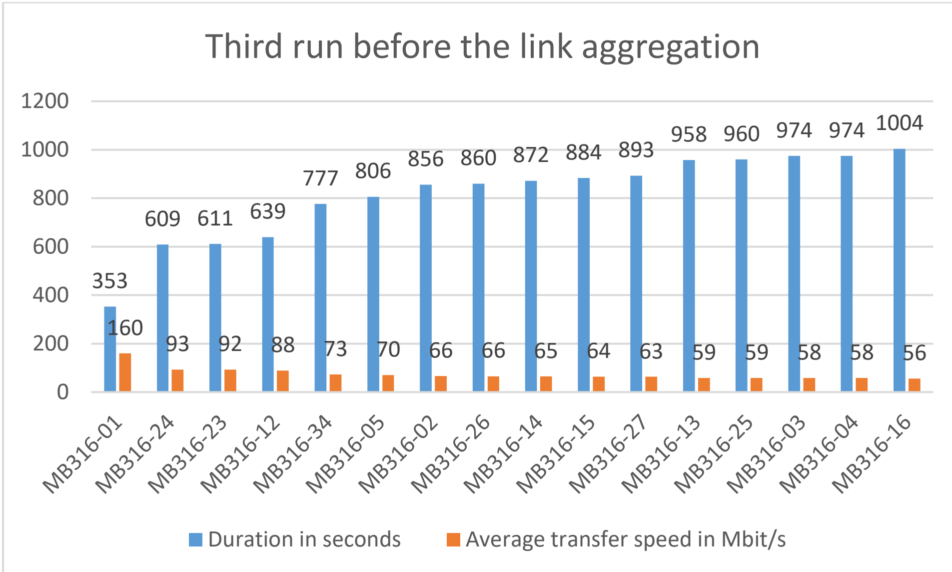

Figure 18. Script test third run before the link aggregation

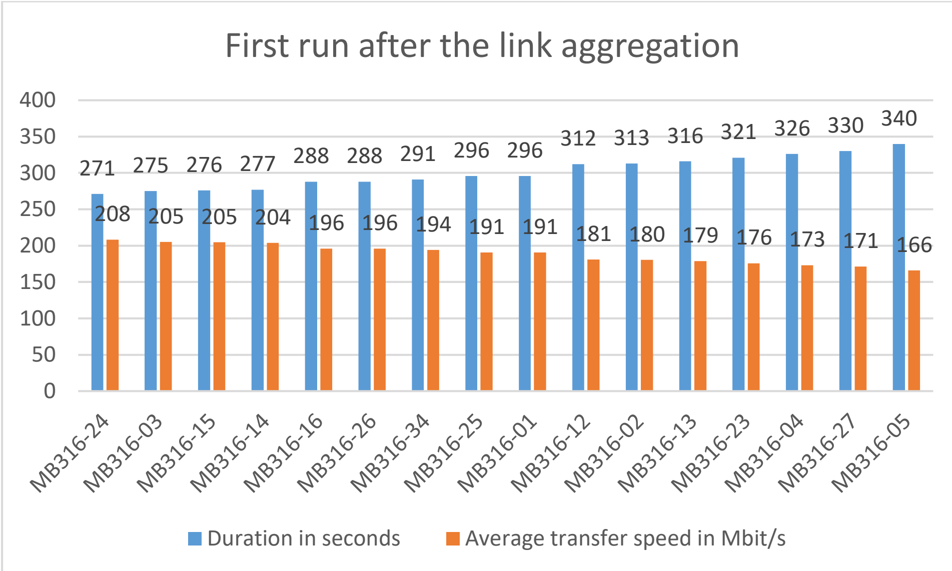

Figure 19. Script test first run after the link aggregation

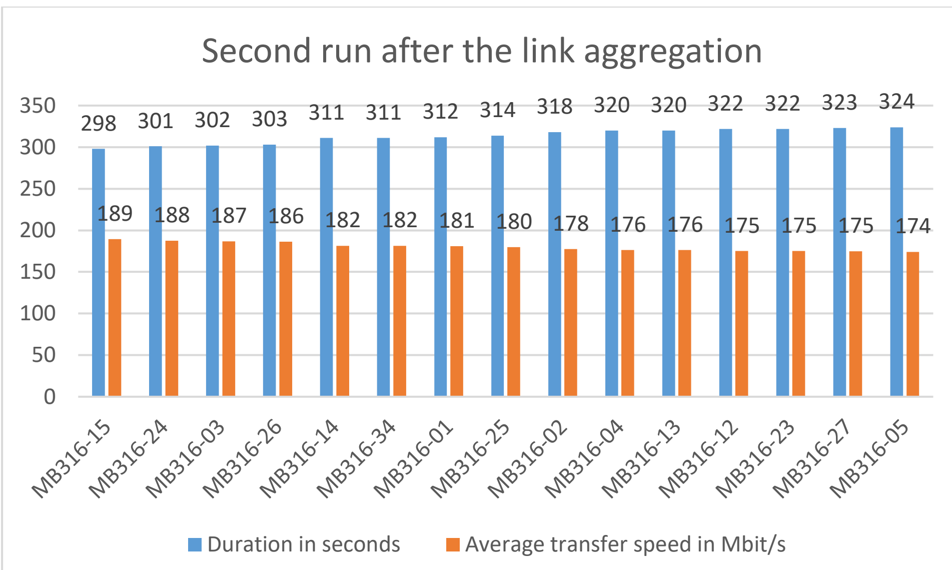

Figure 20. Script test second run after the link aggregation

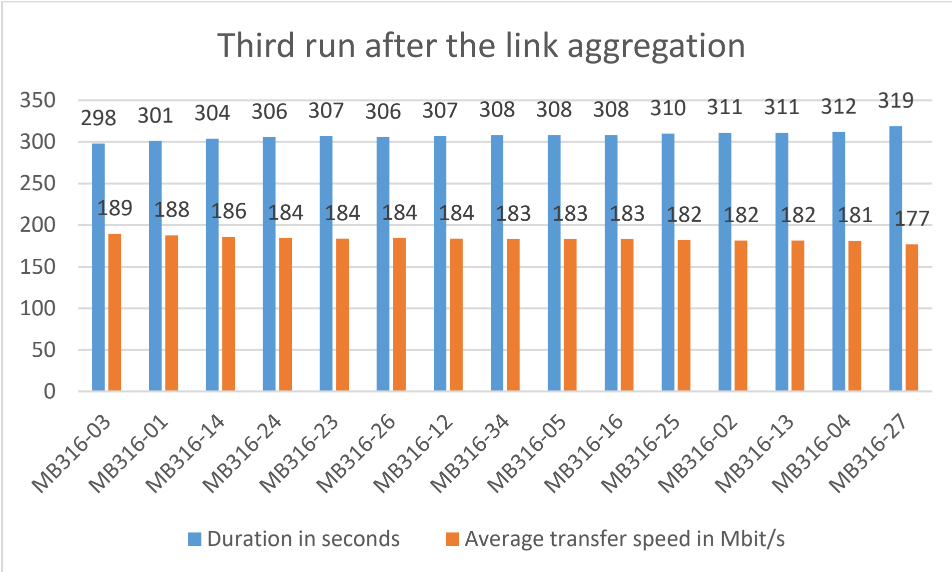

Figure 21. Script test third run after the link aggregation