School of Computer Science and Informatics, Cardiff University
CMT400: 60 CREDITS

Author: Sophie Wells (C124353)
Supervised by: Dr Andrew Hood

Date of Completion: 20/09/2022

Abstract

With the increase of public cloud environments like Microsoft Azure, there is the greater need for testing cloud security against attacks such as denial of service. Public cloud providers have restrictions in place which prevent the testing of attacks such as denial of service. Private cloud environments are able to be created using private extendable cyber ranges often kept offline. These allow complete control of the cloud environment being created and therefore allow testing of the impact of attacks such as denial of service to be carried out on the cloud environment.

This project will capture the impact denial of service attacks have on a physical, private cloud and public cloud environment. This will allow the real data produced from the physical environment to be compared with the two sets of simulated data generated from the two cloud environments. The aim of this is to identify any differences and potential reasonings for it. The method is designed and implemented and the results of which are presented in this project. The results of the experiment showed the differences in speed and capacity of a denial of service attack across the three environments. Finally, some future work from the research was identified.

Table of Contents

1.Introduction

1.1. Research Justification

As the popularity of cloud environments continues to increase, there is an increasing need to understand the issues and challenges associated with its performance. The use of private cloud environments as a form of testing has become increasingly common [1]. However, there is limited amount of published research that evaluates how similar private cloud environments are to public cloud environments. By answering this, the reliability of private cloud environments experiment results used to evaluate public cloud environments can be assessed and any differences can be presented. The simulated data generated from the private and public cloud environments will also be compared to the real data produced from the physical environment. The purpose of this is to identify any differences and potential reasonings for it.

Therefore, the aim of this paper is to compare the effects a denial of service attack has on a physical network, a private cloud environment and a public cloud environment. The private cloud environment will be created using a cyber range and the services offered by a commercial cloud provider will be utilised for the public cloud environment. The purpose of this will be to compare the results of the same attack carried out on a private and public cloud environment as well as a physical environment.

1.2. Aim and Objectives

This research will aim to answer the following research question:

> *To what extent does the effect of a denial of service attack on a public cloud environment differ against a private cloud environment and a physical environment?*

The aim of thesis shall be answered through completing the following objectives:

1. Perform extensive and systematic research in the literature review on denial of service attacks, virtualisation, cloud environments and denial of service attacks in cloud environments.
2. Propose a method of performing a denial of service attack on a physical network, a private cloud environment and a public cloud environment.
3. Demonstrate and compare the effects of a denial of service attack on a physical network, a private cloud environment and a public cloud environment.
4. Review the difference in the three different environments.

1.3. Scope of the Research Project

This research will focus on comparing the effects of a denial of service attack across multiple types of environments.
Chapter 1 Introduction: Aims to justify and summarise the need for the research area and topic. It will also detail the structure and scope of this research paper.

Chapter 2 Background: Consists of a literature review that critically analyses existing literature about denial of service attacks and cloud environments. It aims to find gaps in research and contradicting studies as well as gain a greater understanding of the topic. It outlines the proposed solution to answer some of the ambiguities presented through gaps in research and contradiction.

Chapter 3 Approach: Outlines the process in which the project was completed and justifies the tools that were used.

Chapter 4 Design and Methodology: Outlines the design of the experiment and the reasoning for network structure in each environment.

Chapter 5 Experiment: Details how the design of the experiment was put into practice. It records any problems occurred and how they were overcome. These experiments will allow me to provide an answer to the research question which is the aim of this project.

Chapter 6 Results and Evaluation: Contains the evaluation of the results obtained during the experiment undertaken in the previous chapter. The main focus of this chapter will be to present the results for each environment, compare them and provide potential reasons for the outcome.

Chapter 7 Conclusion and Future Work: Concludes the research paper by providing a summary of the steps taken to answer the research question as well as plans for future work to improve this experiment.

2. Background

The Background chapter presents the relevant research that has allowed the completion of this paper and provides additional context.

2.1 Denial of Service Attacks

A denial of service attack is the name given to an attack that attempts to restrict legitimate users of the service from using the desired resource. Common methods of approaching this is to “flood” a network to prevent genuine network traffic or to disrupt connections between machine and thus preventing access to the service. These disruptions can be targeted at specific users or services within a system or can aim to disrupt the entire system. There are several types of denial of service attacks including TCP SYN Flood, UDP Flood and Distributed Denial of Service attacks [2] [3].

TCP SYN flooding is an attack for Internet Protocol based networks. Its aims to block the victim’s machine from receiving legitimate requests by exploiting the TCP three-way handshake. The attack replies on a server, upon receiving the initial SYN (synchronise/start) packet, sending a SYN/ACK (synchronise/acknowledge) packet back in return and waits for an ACK (acknowledgement) to be sent back to initiate the exchange [2] [3]. It works by sending TCP (Transmission Control Protocol) connection requests that contain spoofed source addresses to the victim’s device [4]. The victim machine will send a SYN/ACK (synchronise/acknowledge) packet in return and wait for an acknowledgement that will never come because it is a spoofed address. This depletes the machines resources because it continually sends SYN/ACK (synchronise/acknowledge) packets to spoof addresses and will prevent genuine requests from being received. A diagram showing the three way handshake can be found in figure 1.

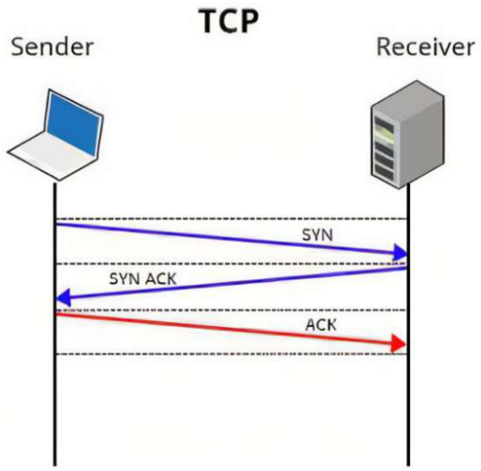


Figure 1: Diagram detailing how TCP works [5]

UDP Flood attacks are based on UDP (User Datagram Protocol) echo and character generator services [2] [3]. They are classed as high rate flood attacks because the aim is to consume all the available network bandwidth between two machines [6], as shown in figure 2. It sends packets to the target destination but is not concerned with whether it reaches it before sending another packet. Therefore, if a packet fails to send, another packet is immediately sent. UDP echo is used because it records the time it takes to reach the victim's machine and come back and records any packet loss. UDP character generator services works by causing random data to return to the attacker's machine for every datagram received [7].

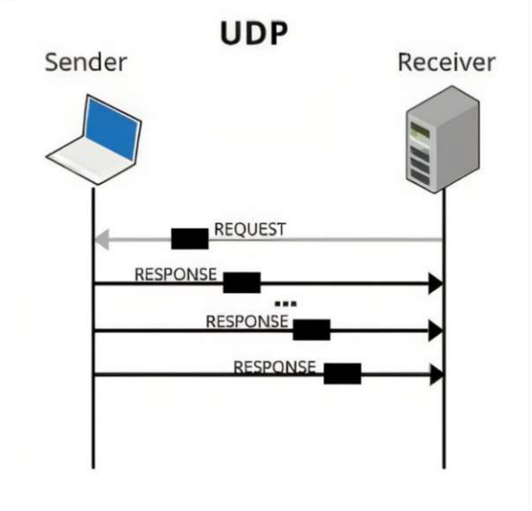


Figure 2: Diagram detailing how UDP works [5]

Distributed Denial of Service attacks rely on the methods of attacks, such as the above, but the attacks are deployed across multiple machines simultaneously. An attacker takes control of a master machine, usually by infecting it with malware, and uses it infect other machines, often called botnets. When the attacker is satisfied with the amount of infected machines, the master machine instruct them all to perform a type of denial of service attack simultaneously, such as UDP flood attack [2] [3]. This is diagrammatically shown in figure 3. The aim of a distributed denial of service attack remains the same but can be more difficult to trace. Therefore, this coordinated attack poses a major threat against the availability in the Internet. The severity of this threat has been documented in a range of different papers that have analysed distributed denial of service attacks [8] [9] [10].

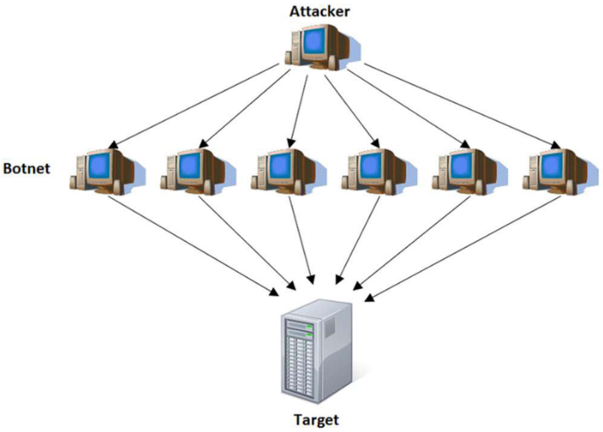


Figure 3: Diagram showing how a distributed denial of service attack works [11]

A survey published by KASPERSKY LAB in 2014, collated data from 3900 companies across 27 countries. It showed the average cost of a Denial of Service incident cost small to medium sized companies was $52,000 and $444,000 for large companies [12]. As well as the significant financial cost, the survey also reported that 38% of respondents believed the attack damaged their reputation. Surveys cannot always be a reliable source of knowledge, but the size of this survey gives a good indication on the impact of a Denial of Service can have on companies ranging in size. To correlate this information, the work of Arora, K., Kumar, K., Sachdeva, M. (2011) also reported that Denial of Service attacks can cause irreparable damage to companies but states that for this reason it is common for companies not to report when attacks have occurred [13]. This can make it extremely difficult to get accurate metrics for the frequency and severity of Denial of Service attacks.

In 2016, Solomon, B. and Fox-Brewster, T. reported a massive distributed denial of service attack that took several major websites offline. This websites included Netflix, Spotify, Twitter, Reddit, Amazon, Yelp and The New York Times. The attack was coordinated on a major Domain Name System (DNS) called Dyn and caused the websites to be down or only partially functional for several hours. The impact of such attack was so severe it warranted a statement from the United States White House press secretary at the time, Josh Earnest, who reported that the Department of Homeland Security were monitoring the malicious attacks [14]. The unavailability of these systems could cause reputational damage as well as significant financial costs to both recover from the attack and prevent a similar attack occurring in the future [13].

An article released by Radware (2015), stated that in 2014 Boston Children’s Hospital were the first health care organisation that was victim of a distributed denial of service attack by a hacktivist group. An emergency response team were quickly contacted to mitigate the effects of the attack but they identified the potential impacts that could have been fatal. It would prevent prescriptions to be to electronically routed to pharmacies, email supported critical processes in some departments and therefore email downtime could be disastrous, access to electronic health records would be stopped. The hospital also used the same Internet Service Provider (ISP) as seven other health care institutions and therefore the attack had the potential to bring down multiple areas of the Boston health care infrastructure [15]. This shows the importance of denial of service prevention and mitigation techniques and how they can prevent deadly repercussions of such attacks.

2.2 Virtualisation

Virtualisation creates a simulated computing environment, that enables user’s access to multiple machines of varying specifications from one single machine. This simulation allows a user to use a different operating system to the one their computer has. This concept was originally developed in the 1960s by IBM [16]. A paper that investigated the performance, advantages and options of virtual machines and networks-installation, showed that virtual machine technology has since progressed and now provides a range of benefits including isolation and resource sharing [16] which can be utilised for greater return on hardware investment and minimal downtime due to easier resource management [17].

Virtualisation is made possible through the use of a hypervisor. A hypervisor is an additional layer between the hardware and the operating system that allows physical hardware resources, such as memory and storage, to individual guest operating systems or applications running on a virtualised environment [16]. A study published by Che, J., Shi, C., Yu, Y. and Lin, W., stated that the efficiency of the hypervisor will largely impact the performance of the entire system [18]. A comparison between the architecture structure of a physical and virtual machine can be found in figure 4.

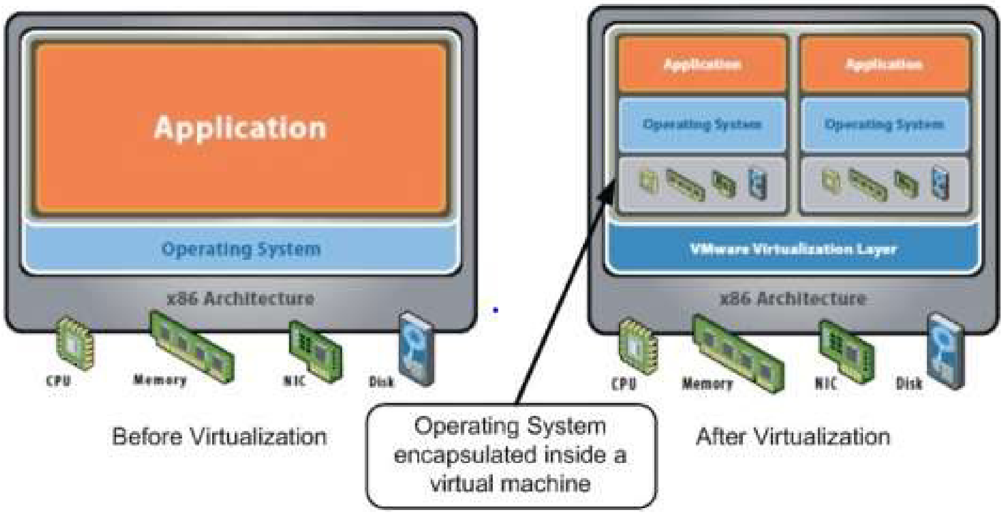


Figure 4: Diagram to show the difference between the architecture of a physical and virtual machine [19]

However, Vaughan-Nichols, S.J. (2008) recorded that the use of a hypervisor can bring extra vulnerabilities to a system because it has more access to hardware resources than typical resources especially if it has root level access to the system [20]. Bazargan, F., Yeun, C.Y. and Zemerly, M.J., (2012) therefore concluded that the secureness of a virtualised environment depends mainly on the level of protection of the hypervisor [21], in their paper analysing the security threats of virtualisation.

The use of virtualisation has a range of benefits such as high resource utilisation rate, easy IT infrastructure management and power savings that are caused by creating multiple virtual machines on a single machine. Virtualisation provides a high level of reliability because it maintains functionality and availability by providing high isolation between virtual machines. This also means that if a virtual machine is infected with malware, it does not necessary effect the physical machine and allows the other virtual machines to remain in use. However, if there is a hardware failure it will bring down all of the virtual machines unless there is a redundant system in place with the same system specifications and configurations. The CPU processing power needs to be considered before virtualised environments are created, in order to maintain a machine’s performance [21].

2.3 Cloud Environments

Virtualisation is a technology that is used in cloud environments. In 2011, Carroll, M., Kotzé, P. and Merwe, A.V.D. published a study that showed that cloud environments build on the capabilities of virtualisation by enabling multi-tenancy, scalability and resource pooling [22]. Cloud computing allows on-demand delivery of resources, such as applications, severs and data storage, over the internet. These resources can be used without installing and maintain them in your local system. IBM reported that there are different types of cloud computing deployment models including public cloud and private cloud [23].

Public cloud environments are usually provided by a commercial cloud provider, such as AWS [24] or Microsoft Azure [25]. A survey published by Fernandes, D.A., Soares, L.F., Gomes, J.V., Freire, M.M. and Inácio, P.R. (2014) reported that public cloud environments provide an off-premise data centre that has an on-demand elastic operation. This ability for growth and shrinkage of data stores allows a pay as you go model to be implemented which means you only pay for what you are using [26]. A paper by Qian, L., Luo, Z., Du, Y. and Guo, L. that investigated cloud computing indicated that it allows users' access to the cloud using a web browser interface but is often stated to be less secure compared to other cloud models because it is more susceptible to malicious attacks [27].

A study into the opportunities and challenges of cloud computing in 2014 addressed the function of sharing resources in cloud environments. It is a process that allows users data to be spread across different physical machines. For example, if a user had two virtual machines, they could be stored on two different physical machines, which also allows multiple users to store their data on the same machine. If the user needs more space, they can store their data in a section of another machine and if they need less space, their data can be taken off that particular machine and someone else can utilise it [28].

Private cloud environments differ from public cloud because the applications and resources are not managed by the service provider as with public cloud but instead managed by the organisation itself. They are internal enterprise data centre that pool together services and make them available for users at an organisational level. The benefits of using private cloud is it is easier to maintain, is more secure and provides more control over deployment and use [27].

2.4 Cyber Ranges

A cyber range is a simulation platform that enables organisations to replicate existing or proposed systems to test and develop user skills and system operations [29]. A survey produced by Chouliaras, N., Kittes, G., Kantzavelou, I., Maglaras, L., Pantziou, G. and Ferrag, M.A., in 2021 showed that main uses of cyber ranges were for research, training and security exercises [30]. The technology can be used to test how different attacks performed on cloud environments affect the system. The survey showed that cyber ranges were commonly used for educational courses that investigated cloud security [30]. However, during my extensive research I did not find any reference to the similarities between a private cloud environment produced using a cyber range and a public, commercial cloud environment. This is important to find out because public cloud environments are more accessible because they use a service provider that does everything on your behalf, rather than setting it up within an organisational or household [27].

A study published in 2020 by Yamin, M.M., Katt, B. and Gkioulos, V. showed a significant increase in cyber range use from 2005 to 2017 especially in red teams [1]. This suggests more experiments and tests are being run using on a private cloud environment using a cyber range. This method can be preferable to researchers because when using a cyber range the researcher has complete control of the system and how the cloud environment works, making it an easier to control and manipulate than when using a public cloud environment [27]. Although, easier to undertake, it is imperative to find out whether these experiments are producing results that can be reliably related to public cloud environments.

The DIATEAM cyber range allows users to create topologies which is a set of virtual equipment that can be connected using virtual cables to interact like a network. A screenshot of some of the topologies available on the cyber range can be seen in figure 5. A topology gate is a key feature of the DIATEAM cyber range because it allows the connection to multiple topologies [31]. This is useful when a repeated function is needed, such as connecting the cyber range to the Internet.

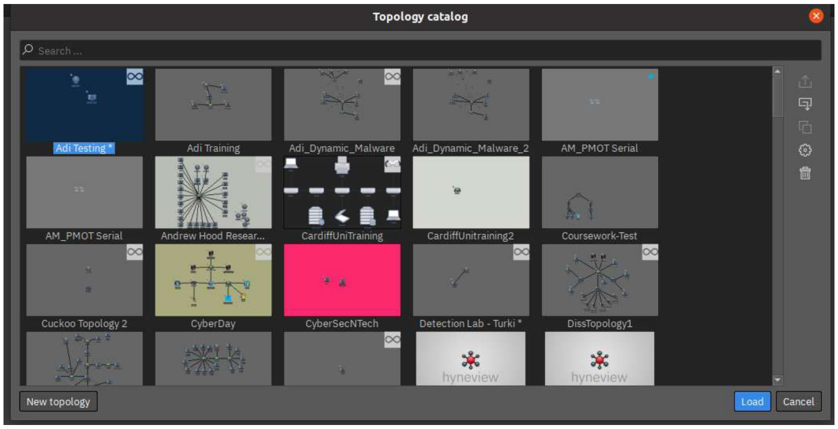


Figure 5: Topology catalogue in the cyber range

2.5 Networking Stack In Physical and Cloud Environments

Data in a physical environment is transferred through the TCP/IP networking stack very differently to a cloud environment. A physical environment uses hardware meaning that data in the application layer can be transferred to the physical layer, goes through an ethernet cable to the physical layer of a different machine and up to its application layer [32]. This process is shown in figure 6.

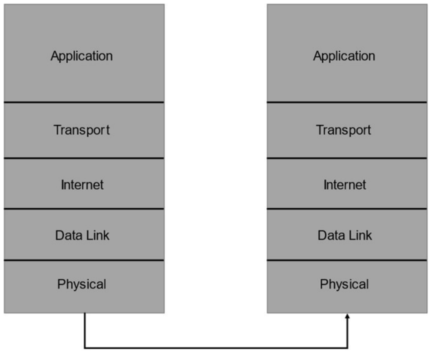


Figure 6: TCP/IP network stack in a physical environment

However, data transferring in a cloud environment cannot follow this simple process and requires additional steps to transfer its data. This is because, there is no hardware involved and purely relies on software. This results in data having to flow down from the application layer to the physical layer of the machine, to then be transferred to the physical layer of an immediate system within the machine. This decides which virtual machine the data is going to by going up to its network/Internet layer and back down to the physical layer. From here, the data is transferred to the physical layer of the immediate system link for the virtual machine the data is going to. The data goes up to its network/Internet layer before it return to the physical layer to be transferred to the physical layer of the relevant virtual machine. Finally, the data is processed to the application layer of the virtual machine, where it can be used by the user. The data transfer process through the TCP/IP networking stack is diagrammatically shown in figure 7.

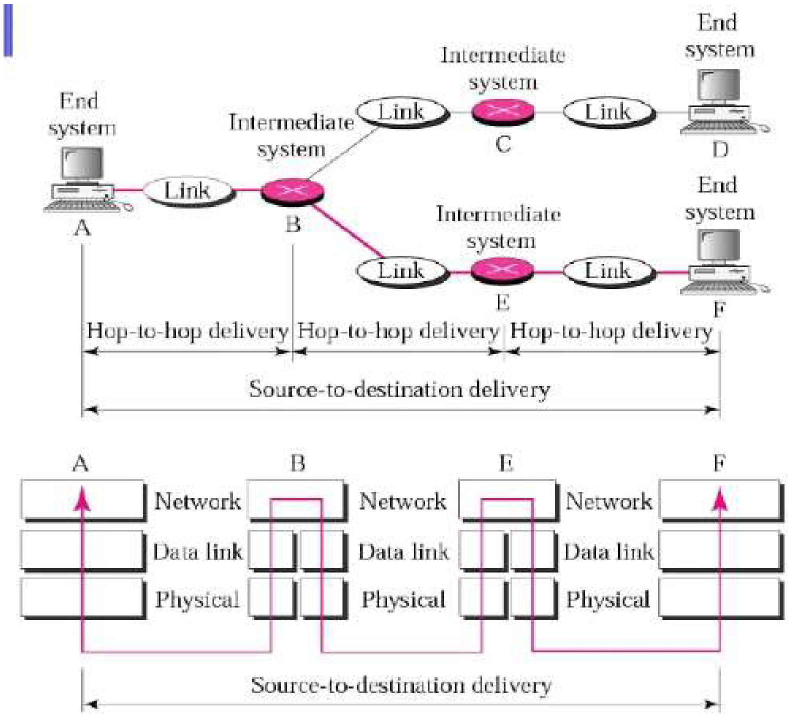


Figure 7: TCP/IP network stack in a cloud environment [33]

2.6 Real and Simulated Data

Simulated data is produced by mirroring real world conditions to predict what would happen if the same process was followed with real hardware. Such data can be produced using an array of platforms including cloud environments. Using a cloud environment to simulate data has the benefit of scalability [22]. This allows large scale networks to be deployed and tested without the overhead cost of physically creating the network.

However, there an limitations to simulated data that could result in vastly different results between the real and simulated data. This is because, the process required to obtain real and simulated data can be very different. An example of this is the data transfer method through the TCP/IP network stack, as shown above. An experiment published by Niznan, J., Papousek, J. and Pelánek, R. explored the role small differences in simulated data had in predictive accuracy in student modelling. It showed that these small differences in simulated data had important impacts on the behaviour of the education systems ability to adapt. The experiment found that simulated data was useful but should be followed by the use of real data [34].

2.7 Denial of Service Attacks in Cloud Environments

Research into distributed and economic denial of service attacks published by Somani, G., Gaur, M.S. and Sanghi, D., (2015), has shown cloud environments are still vulnerable to such attacks but can affect the environment different [35]. In physical environments denial of service attacks works to restrict legitimate users of the service from using the desired resource. In a cloud environment denial of service attacks can cause heavy downtime and economic loss, similar to physical environments. Economic denial of service can also be utilised to attack cloud environments. They work by sending packet requests at a slower speed to avoid detection. It relies on pay as you go billing, which many commercial providers use, with the aim of financially draining the company by increasing the amount of cloud storage they are paying for. An experiment compared the effects of Distributed denial of service and Economic denial of service attacks on a single physical server with multiple virtual machines and a scalable cloud experiment. It showed almost all components of cloud architecture were affected by the distributed denial of service attack. The economic denial of service attack showed that cloud features such as auto-scaling, isolation and multi-tenancy multiplies the impact of the attack. The paper determined cloud resource allocation architecture should be thoroughly investigated to protect it from these attacks [35].

A similar experiment done by Ficco, M. and Rak, M., (2014) investigated the affect denial of service attacks have on cloud environments using a slowly increasing polymorphic denial of service attack strategy recorded similar results. This stealthy attack strategy avoids detection by slowly increasing its intensity and conforming to the detection mechanisms' service arrival rates. It showed that it exploited the flexibility of the cloud service which forced services to scale up and therefore consume more resources than needed, leading to mainly financial based impacts on the customer rather than service availability [36].

2.8 Industry Tools Used In Network Analysis

Wireshark is the most widely used network protocol analyser across the globe. It captures network traffic in real time, that can be analysed offline and in depth. Data can be read from a range of sources, including Ethernet, IEEE, Bluetooth and USB. The output of a network capture is coloured coded toa allow for quick and intuitive analysis of different packet types [37]. An example of Wireshark being used during this project can be found in figure 12.

Python has an inbuilt package called socket which enables the connection between a client and a server, and therefore makes it possible for the two to interact with each other. Some of the main functionality of this package is as follows [38]:

- socket() is the function used to specify whether the connection is UDP or TCP.
- connect() allows the client to connect with the server.
- send() is used to send data from both sides, when it is required
- recv() allows data to be received from the client or server, when it is required

Conclusions

During this chapter, former research on all aspects that make up this project has been presented and critically analysed. This has facilitated the discovery of useful studies and a gap in research which this project will attempt to fill. In the next chapter, the strategy used to undertaken the experiment will be explained along with the justification for this.

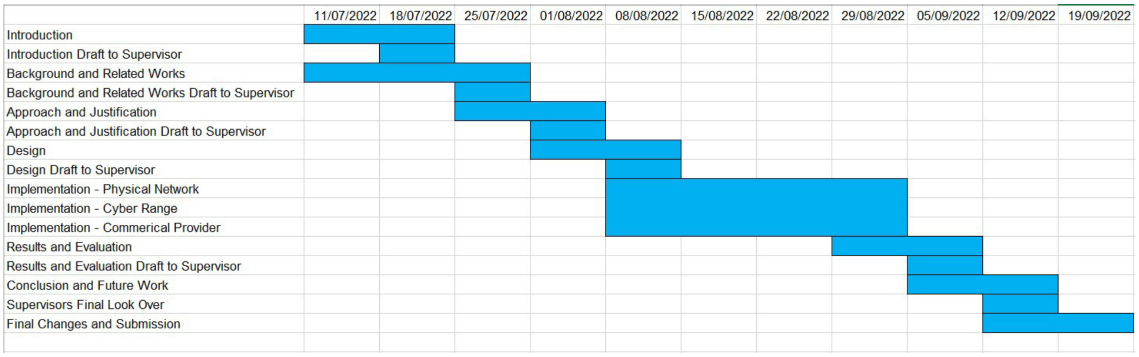
3. Approach

In this chapter, the approach taken to complete this paper will be explained. It will include a detailed justification of the methodology of tools chosen to use as well as a plan for the timings of the project.

3.1. Agile Development Methodology

Throughout this project an Agile development methodology [39] will be followed. An iterative approach allows sufficient time for detailed research to be carried out in order to learn and develop the necessary skills to complete this project to a high standard as well as provide a flexible timeframe to overcome any problems that may arise. This in turn will facilitate a range of denial of service attacks to be tested on the different environments simultaneously. An agile approach also allows documentation to be done concurrently with the testing. This ensures that milestones and events are documented when they occur in order to avoid crucial details and processed being missed in the documentation.

This project will be split into clear sections of work with the aim of making it less overwhelming and more manageable. Each section will contain milestones that have an estimated completion date to ensure work is completed promptly in order to remain on track. The breakdown of these sections, their milestones and completion dates are shown in the Gantt chart, figure 8. This work plan has been discussed with the project supervisor and will be regularly reviewed in order to resolve any potential issues that could affect the timely completion of all sections.



	11/07/2022	18/07/2022	25/07/2022	01/08/2022	08/08/2022	15/08/2022	22/08/2022	29/08/2022	05/09/2022	12/09/2022	19/09/2022
Introduction	■	■									
Introduction Draft to Supervisor		■									
Background and Related Works	■	■	■								
Background and Related Works Draft to Supervisor			■								
Approach and Justification			■	■							
Approach and Justification Draft to Supervisor				■							
Design				■	■						
Design Draft to Supervisor					■						
Implementation - Physical Network					■	■	■	■			
Implementation - Cyber Range					■	■	■	■			
Implementation - Commerical Provider					■	■	■	■			
Results and Evaluation								■	■		
Results and Evaluation Draft to Supervisor									■		
Conclusion and Future Work									■	■	
Supervisors Final Look Over										■	
Final Changes and Submission										■	■

Figure 8: Gantt chart

3.2. Development Solution

The decision to test a denial of service attack on a physical network, a private cloud environment and public cloud environment was based on a desire to get the most accurate results as well as an invaluable learning opportunity that would be challenging and provide exposure to new technologies.

To justify the tools used in this paper, extensive research was carried out to evaluate their functionality and usability for the purpose they were needed for. Microsoft Azure [25] was chosen for the public cloud environment because it is a popular cloud provider. It also offers free credits for thirty days which allows the experiments to be completed without causing major financial expense to the researcher. To create the private cloud environment the DIATEAM cyber range [29] that is available at Cardiff

University will be used. Cyber ranges are being used more frequently in recent years [1] and therefore are a useful resource to test as it grows in popularity.

The experiment itself will involve performing a UDP and TCP denial of service attack on the three different environments. These attacks were chosen because they are frequently used [2] [3] and therefore are key attacks to analyse in different environments. Unfortunately, a distributed denial of service will not be part of the scope of this project due to lack of resources required to perform it. It was included in the previous chapter (Background) because it is important to understand how a distributed denial of service attack differs from a denial of service attack. In the future, performing a distributed denial of service attack in each of the environments would be extremely insightful and add a greater depth to this area of research.

Conclusions

During this chapter, the approach to fulfil the project aim as well the methodology has been presented as well as the software and technology being used. The next chapter will describe how this software and technology will be used to design and implement the experiment.

4. Design and Methodology

In this chapter, a record of the process undertaken in order to design the experiment will be given. This was done using the software and technology justified in the previous chapter (Approach).

4.1. Environment Design

The three environments have different designs due to the resources available for this project and the nature of the environments.

4.1.1. Physical Environment

For the physical experiment, multiple network designs were created. One of which used a monitoring switch and an additional laptop to create a network with a monitoring laptop external to the attack. Another one used two laptops and were connected using an ethernet cable. This network design was chosen because of the resources available for this project. Both were Dell laptops, with an Intel Core i5-8250U processor and eight gigabytes of memory, that had an Ubuntu operating system. One of the laptops was designed to act as the attacker and monitor and the other to act as the victim. The physical network diagram can be found in figure 9.

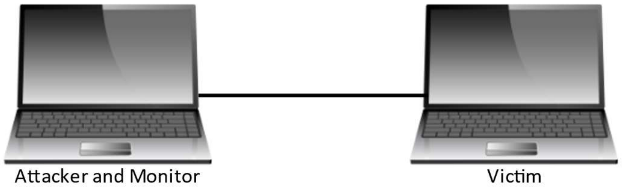


Figure 9: Physical network diagram

4.1.2. Private Cloud Environment

The resources available for the private cloud network were more readily available because the environment allowed for the easy creation of the resources needed. This resulted in the more complicated network being designed with a mirroring switch and an external monitoring machine. A topology gate was added to the design of this environment to enable connection to the Internet, as mentioned in the Background chapter. A diagram of the network produced in the private cloud environment can be found in figure 10.

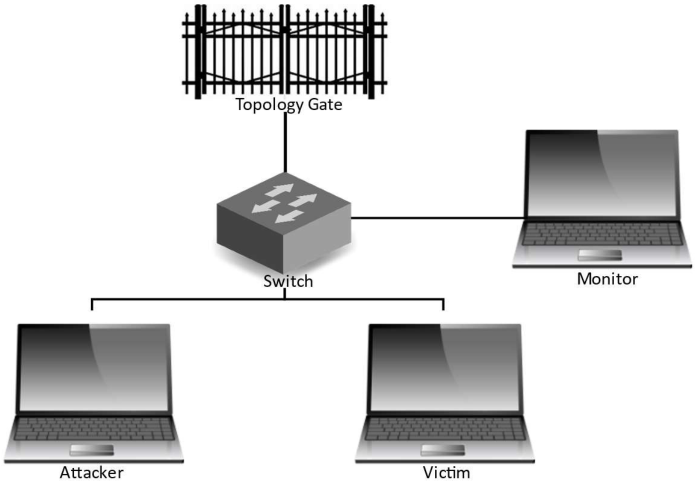


Figure 10: Private cloud network diagram

4.1.3. Public Cloud Environment

The public cloud environment took a similar network structure to the physical environment because of the restrictions and credit limit of the free subscription used on Microsoft Azure. Two virtual machines were designed with an Ubuntu operating system. The network diagram for the public cloud environment can be found in figure 11.

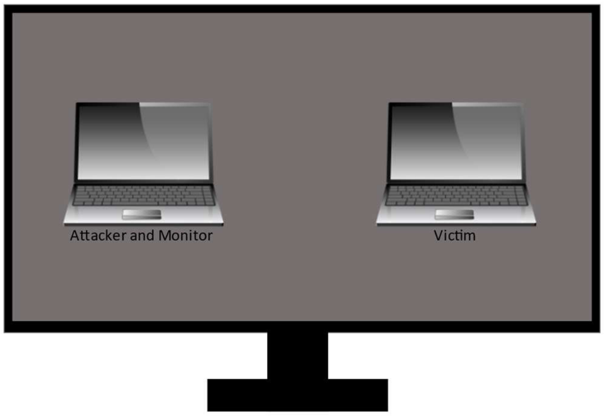


Figure 11: Public cloud network diagram

4.3 Denial of Service Attack Design

Python scripts will be used to perform the UDP denial of service attack and the TCP denial of service attack. This is because, there are packages within python that allow UDP and TCP denial of service attack scripts to be created with minimal amounts of code. This means more time can be spent carrying out the experiments and analysing the data rather than creating the complicated scripts that do not utilise existing packages. Python can also be easily executed in Linux terminal, making it ideal to use across all three environments.

Conclusion

This chapter has shown and justified the network diagrams for the three environments being created for the experiments. The next chapter (Experiment), will show how these network diagrams were used to carry out the experiments.

5. Experiment

The experiment chapter documents how the experiment design presented in the Design and Methodology chapter was implemented. This chapter provides a detailed account of the scripts used to perform the denial of service attacks discussed in the Background chapter and the setup of the physical, private cloud and public cloud environments used in this experiment.

5.1. Developing Python Scripts

Three python scripts were developed for this experiment. Two performed a UDP denial of service attack, one of which used time to dictate the length of the experiment and the other used the amount of packets sent. The other script implemented a TCP denial of service attack for a specific amount of time. The packet based UDP attack was set to send ten million packets in order to obtain a large dataset, that would allow for patterns in the network traffic to be presented and analysed.

The study conducted by Sharma, M. was used to determine the appropriate length of time to perform the UDP and TCP time based denial of service attacks. It showed that in the first half of 2021, the average distributed denial of service attack was performed for six minutes [40] and therefore by running the UDP time based attack for six minutes, it creates a more accurate representation of attacks in industry. However, due to the way UDP and TCP work, shown in figure 1 and 2, it did not make logical sense to only run the TCP attack for six minutes. This is because, it takes longer to send a TCP packet due to the three way handshake. This is opposed to a UDP packet, which sends a packet and is not affected if it does not reach its target destination and just sends another one immediately after the latter has been sent. Sharma, M.'s study was helpful to get a guideline of the how long to run an attack for, but it was based on a distributed denial of service attack, which are significantly more powerful due to the volume of machines being used in the attack. Based on this, the decision was made to run the TCP denial of service attack for four hours. This would allow the network capture returned to have a higher volume of useful data, that could be analysed. The three scripts referenced can be found in Appendix A.

5.1.1. UDP Denial of Service Python Scripts

The UDP denial of service python scripts prompt the user to enter the target IP address and port. The message sent within the packet was set to 'sophie wells' to make it easily visible when reviewing the Wireshark data post experiment. Both scripts imported the socket package and used *AF_INET* and *SOCK_DGRAM* to perform the attack. *AF_INET* sets the address family which designates that the type of address the python script will be communicating with, is an Internet Protocol v4 address. *SOCK_DGRAM* specifies that the script will be communicating via UDP packets. One of the scripts uses a for loop to send ten million packets and the other uses a while loop to send packets for 6 minutes (360 seconds).

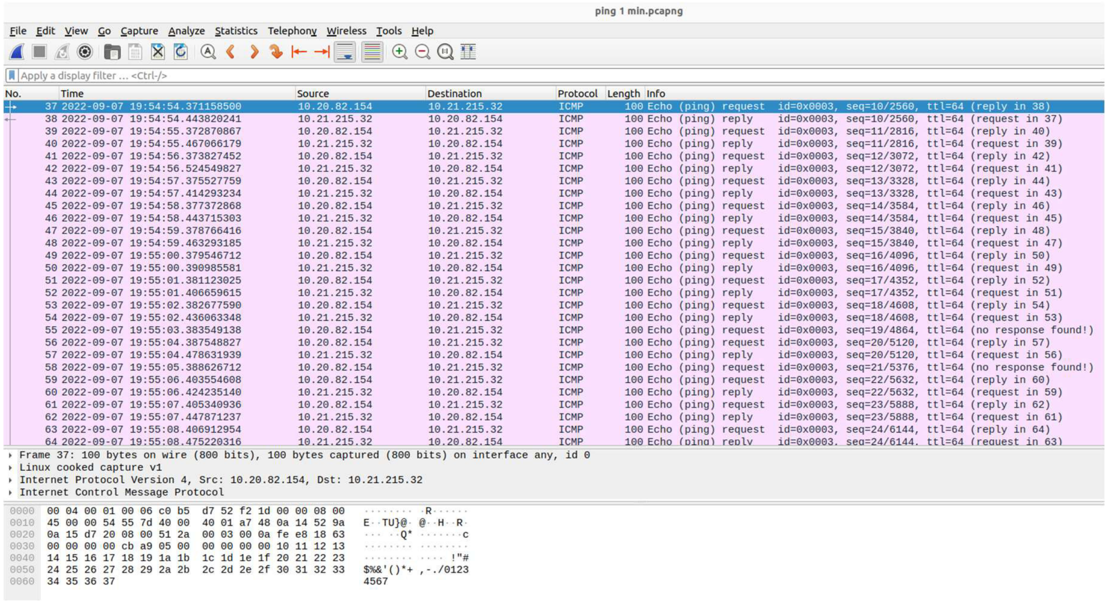
5.1.2. TCP Denial of Service Python Script

The TCP denial of service python script prompts the user to enter the target IP address and port. 'sophie wells' was set as the message sent in each packet to make it easily visible during the Wireshark review post experiment. The buffer size of the TCP packets being sent was set to 1024. *AF_INET* and *SOCK_STREAM* from the socket package was used to perform the attack. *SOCK_STREAM* specifies that the script will be sending and receiving TCP packets. The script used a for loop to send packets for 4 hours (14400 seconds).

5.2. Physical Experiment

As presented in the previous chapter (Design and Methodology), the network in the physical experiment had to be simplified due to resource limitations and time constraints. Two laptops, both running Ubuntu, were connected using an ethernet cable. One laptop was designated to be the attacker and monitoring machine and the other the victim machine. The victim machine was left in the state it was acquired in. However, the attacker/monitoring machine needed Wireshark to be installed and the python scripts containing the attacks to be downloaded.

Once the attacker/monitoring machine was set up, each machine was used to ping the other to ensure they were properly connected. Wireshark captured a one minute ping cycle from the attacker/monitoring machine to the victim machine, shown in figure 12. The purpose of this was to compare the time it took for the attacker machine to ping the victim machine in each environment.


ping 1 min.pcapng
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
Apply a display filter ... <Ctrl-/>

```
No.  Time                           Source         Destination    Protocol Length Info
  37 2022-09-07 19:54:54.371158500  10.20.82.154   10.21.215.32   ICMP     100 Echo (ping) request  id=0x0003, seq=10/2560, ttl=64 (reply in 38)
  38 2022-09-07 19:54:54.443820241  10.21.215.32   10.20.82.154   ICMP     100 Echo (ping) reply    id=0x0003, seq=10/2560, ttl=64 (request in 37)
  39 2022-09-07 19:54:55.372870867  10.20.82.154   10.21.215.32   ICMP     100 Echo (ping) request  id=0x0003, seq=11/2816, ttl=64 (reply in 40)
  40 2022-09-07 19:54:55.467066179  10.21.215.32   10.20.82.154   ICMP     100 Echo (ping) reply    id=0x0003, seq=11/2816, ttl=64 (request in 39)
  41 2022-09-07 19:54:56.373827452  10.20.82.154   10.21.215.32   ICMP     100 Echo (ping) request  id=0x0003, seq=12/3072, ttl=64 (reply in 42)
  42 2022-09-07 19:54:56.524549827  10.21.215.32   10.20.82.154   ICMP     100 Echo (ping) reply    id=0x0003, seq=12/3072, ttl=64 (request in 41)
  43 2022-09-07 19:54:57.375527759  10.20.82.154   10.21.215.32   ICMP     100 Echo (ping) request  id=0x0003, seq=13/3328, ttl=64 (reply in 44)
  44 2022-09-07 19:54:57.414293234  10.21.215.32   10.20.82.154   ICMP     100 Echo (ping) reply    id=0x0003, seq=13/3328, ttl=64 (request in 43)
  45 2022-09-07 19:54:58.377372868  10.20.82.154   10.21.215.32   ICMP     100 Echo (ping) request  id=0x0003, seq=14/3584, ttl=64 (reply in 46)
  46 2022-09-07 19:54:58.443715303  10.21.215.32   10.20.82.154   ICMP     100 Echo (ping) reply    id=0x0003, seq=14/3584, ttl=64 (request in 45)
  47 2022-09-07 19:54:59.378766416  10.20.82.154   10.21.215.32   ICMP     100 Echo (ping) request  id=0x0003, seq=15/3840, ttl=64 (reply in 48)
  48 2022-09-07 19:54:59.463293185  10.21.215.32   10.20.82.154   ICMP     100 Echo (ping) reply    id=0x0003, seq=15/3840, ttl=64 (request in 47)
  49 2022-09-07 19:55:00.379546712  10.20.82.154   10.21.215.32   ICMP     100 Echo (ping) request  id=0x0003, seq=16/4096, ttl=64 (reply in 50)
  50 2022-09-07 19:55:00.390985581  10.21.215.32   10.20.82.154   ICMP     100 Echo (ping) reply    id=0x0003, seq=16/4096, ttl=64 (request in 49)
  51 2022-09-07 19:55:01.381123025  10.20.82.154   10.21.215.32   ICMP     100 Echo (ping) request  id=0x0003, seq=17/4352, ttl=64 (reply in 52)
  52 2022-09-07 19:55:01.406659615  10.21.215.32   10.20.82.154   ICMP     100 Echo (ping) reply    id=0x0003, seq=17/4352, ttl=64 (request in 51)
  53 2022-09-07 19:55:02.382677590  10.20.82.154   10.21.215.32   ICMP     100 Echo (ping) request  id=0x0003, seq=18/4608, ttl=64 (reply in 54)
  54 2022-09-07 19:55:02.436063348  10.21.215.32   10.20.82.154   ICMP     100 Echo (ping) reply    id=0x0003, seq=18/4608, ttl=64 (request in 53)
  55 2022-09-07 19:55:03.383549138  10.20.82.154   10.21.215.32   ICMP     100 Echo (ping) request  id=0x0003, seq=19/4864, ttl=64 (no response found!)
  56 2022-09-07 19:55:04.387548827  10.20.82.154   10.21.215.32   ICMP     100 Echo (ping) request  id=0x0003, seq=20/5120, ttl=64 (reply in 57)
  57 2022-09-07 19:55:04.478631939  10.21.215.32   10.20.82.154   ICMP     100 Echo (ping) reply    id=0x0003, seq=20/5120, ttl=64 (request in 56)
  58 2022-09-07 19:55:05.388626712  10.20.82.154   10.21.215.32   ICMP     100 Echo (ping) request  id=0x0003, seq=21/5376, ttl=64 (no response found!)
  59 2022-09-07 19:55:06.403554608  10.20.82.154   10.21.215.32   ICMP     100 Echo (ping) request  id=0x0003, seq=22/5632, ttl=64 (reply in 60)
  60 2022-09-07 19:55:06.424235140  10.21.215.32   10.20.82.154   ICMP     100 Echo (ping) reply    id=0x0003, seq=22/5632, ttl=64 (request in 59)
  61 2022-09-07 19:55:07.405340936  10.20.82.154   10.21.215.32   ICMP     100 Echo (ping) request  id=0x0003, seq=23/5888, ttl=64 (reply in 62)
  62 2022-09-07 19:55:07.447871237  10.21.215.32   10.20.82.154   ICMP     100 Echo (ping) reply    id=0x0003, seq=23/5888, ttl=64 (request in 61)
  63 2022-09-07 19:55:08.406912954  10.20.82.154   10.21.215.32   ICMP     100 Echo (ping) request  id=0x0003, seq=24/6144, ttl=64 (reply in 64)
  64 2022-09-07 19:55:08.475220316  10.21.215.32   10.20.82.154   ICMP     100 Echo (ping) reply    id=0x0003, seq=24/6144, ttl=64 (request in 63)
```

```
Frame 37: 100 bytes on wire (800 bits), 100 bytes captured (800 bits) on interface any, id 0
Linux cooked capture v1
Internet Protocol Version 4, Src: 10.20.82.154, Dst: 10.21.215.32
Internet Control Message Protocol
```

```
0000  00 04 00 01 00 06 c0 b5  d7 52 f2 1d 00 00 08 00   ········ ·R······
0010  45 00 00 54 55 7d 40 00  40 01 a7 48 0a 14 52 9a   E··TU}@· @··H··R·
0020  0a 15 d7 20 08 00 51 2a  00 03 00 0a fe e8 18 63   ··· ··Q* ·······c
0030  00 00 00 00 cb a9 05 00  00 00 00 00 10 11 12 13   ········ ········
0040  14 15 16 17 18 19 1a 1b  1c 1d 1e 1f 20 21 22 23   ········ ···· !"#
0050  24 25 26 27 28 29 2a 2b  2c 2d 2e 2f 30 31 32 33   $%&'()*+ ,-./0123
0060  34 35 36 37                                        4567
```

Figure 12: Snapshot of the Wireshark capture of one minute ping cycle in the physical environment

When the connection between the machines had been established, the python scripts were ran. This included the six minute UDP attack, the ten million UDP packets attack and the four hour TCP attack. All of these attacks were executed three times in the attempt to establish a pattern.

There were no issues with the UDP attacks but it was realised that in order to carry out the TCP attack for four hours, both laptops needed to be on and not go into sleep mode. This was overcome by changing the laptop settings to never turn off while they were plugged into a power source.

5.3. Private Cloud Experiment

The cyber range facilitates at Cardiff University were utilised to create the private cloud environment. Remote access to the cyber range was not available with the use of a university laptop with strict access control restrictions. Therefore, the use of an ethernet cable was used to directly connect the cyber range to a laptop. From there, a topology could be created which followed the network diagram shown in the previous chapter (Design and Methodology). The topology required a topology gate to link to another topology on the cyber range that granted access the Internet. This is because, unlike a physical or public cloud environment, the cyber range is isolated and therefore does not automatically connect to the WiFi connection set on the laptop being used.

As shown in figure 9, the resources available in this environment allowed there to be three machines, an attacker, a victim and a monitoring machine, all of which were connected using a mirroring switch. All three machines used Kali Linux because of the pre-installed tools on this operating system. The mirroring switch allowed the monitoring machine to capture the network traffic between the attacker and victim machine, using Wireshark, despite not being part of the attack. The topology created can be shown in figure 13.

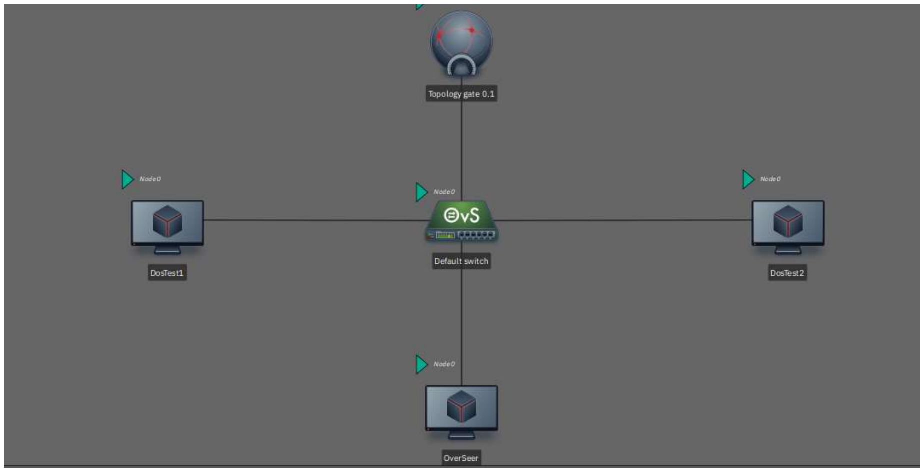
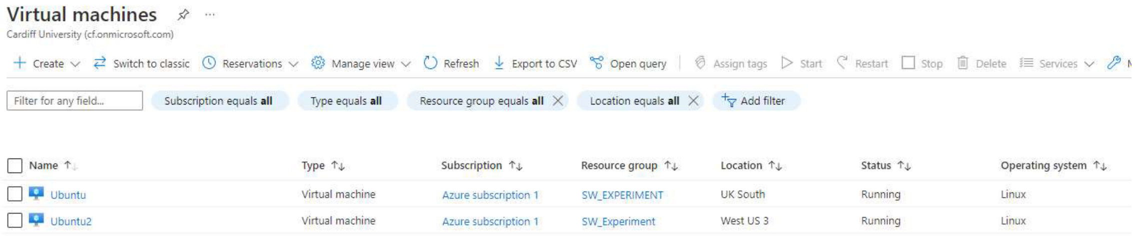


Figure 13: Private cloud environment topology

When the network topology was correctly set up, a one minute ping cycle was captured in Wireshark on the monitoring machine, from the attacker machine to the victim machine. The six minute UDP attack and ten million UDP packets attack, python scripts were both executed three times. However, during the experimental stage of this project, the cyber range at Cardiff University broke down and therefore was put out of service. This resulted in the TCP attacks not being able to be run on this environment. Unfortunately, no other cyber range facility was available in order to complete the data collection process in this environment.

5.4. Public Cloud Experiment

To set up the public cloud experiment, the Microsoft Azure Virtual Machines service was utilised. The free subscription used allowed the creation of two virtual machines. Both of which were Ubuntu machines; the virtual machine named Ubuntu in figure 14 was set up to act as the attacker and the monitor and the virtual machine named Ubuntu2 in figure 14 was set to be the victim. Wireshark was installed and the python scripts were downloaded onto the Ubuntu virtual machine. The virtual machines were connected to the same resource group, SW_Experiment, to create a network structure, shown in figure 14. By being in the same resource group, the machines were able to connect with the others private IP address without any further intervention.


Virtual machines

Cardiff University (cf.onmicrosoft.com)

Subscription equals **all** | Type equals **all** | Resource group equals **all** | Location equals **all**

Name	Type	Subscription	Resource group	Location	Status	Operating system
Ubuntu	Virtual machine	Azure subscription 1	SW_EXPERIMENT	UK South	Running	Linux
Ubuntu2	Virtual machine	Azure subscription 1	SW_Experiment	West US 3	Running	Linux

Figure 14: Virtual machines set up in Microsoft Azure

After the virtual machines were created, a remote desktop connection needed to be established for the attacker/monitoring machine to allow Wireshark and the python scripts to run simultaneously. This could not be done using bash or PowerShell, which were the preliminary methods of communicating with the virtual machines. This is because, both bash and PowerShell do not have the processing functionality to allow Wireshark to run and initiate the python script at the same time. Therefore, research was done and the Microsoft guidelines were used to understand how to launch a remote desktop connection [41].

Once the remote desktop connection was set up for the attacker/monitoring machine, a one minute ping cycle from the attacker/monitoring machine to the victim machine was captured on Wireshark. After this was successful, the three python scripts were ran but all of which crashed the remote desktop connection. This will be discussed further in the subsequent chapters. The timing and the amount of packets sent in the attacks were adapted to prevent the remote desktop connection from crashing before any data was captured on Wireshark.

Conclusion

In this chapter, a record of how the experiment design in the previous chapter (Design and Methodology) was put into practice was presented. The problems that were encountered during these experiments were also documented along with how it was overcome. In the next chapter (Results and Evaluation), the results from these experiments will be presented and discussed in the attempt to provide an answer to the research question which was the aim of this project.

6. Results and Evaluation

In this chapter, the results of the network traffic captured during the attacks on the three environments will be presented and the potential reasons for these results will be discussed.

Some of the results of the experiments run will be presented in a graphical form to visually show the network traffic presented as the python scripts were ran. To do this, the Wireshark data captures were processed to produce I/O graphs along with Excel derived graphs, and were used to compare the data from the three environments.

6.1. Results and Analysis of Physical Experiment

The results of the one minute ping cycle showed a consistent pattern of the ICMP packets being sent and returned within a second but with eight instances where no packet was returned. This is shown in figure 15.

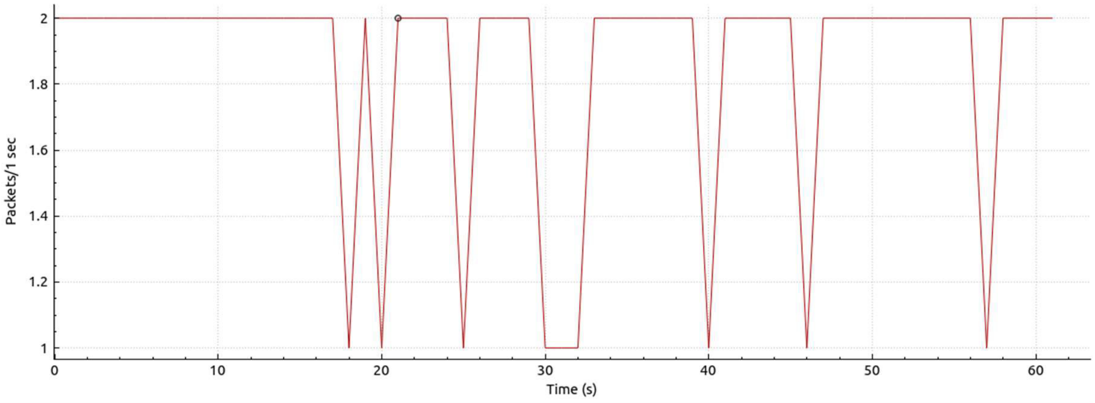


Figure 15: I/O graph of the one minute ping cycle in the physical environment

The three experiments ran with the six minute UDP attack python script showed very similar results. They all have peaks and troughs but stay consistently in the range of 25000 to 36000 packets per second. Because of this, the amount of packets sent in this specific timeframe were similar. Figure 16 shows the network traffic flow of all three UDP six minute attacks to show how similar the results were. Figures 17 to 19 shows the network traffic of the individual experiments.

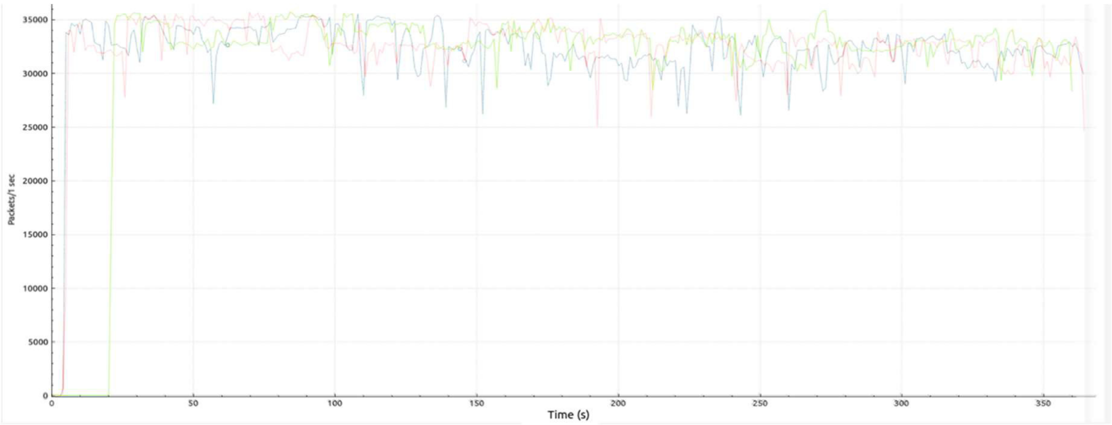


Figure 16: I/O graph comparing the three six minute UDP attacks in the physical environment.

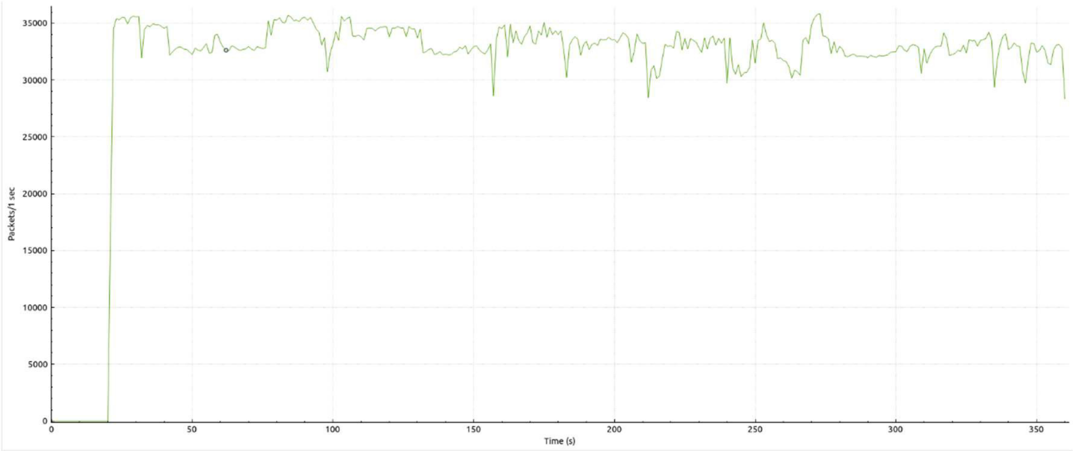


Figure 17: I/O graph showing the results of the first six minute UDP packets experiments

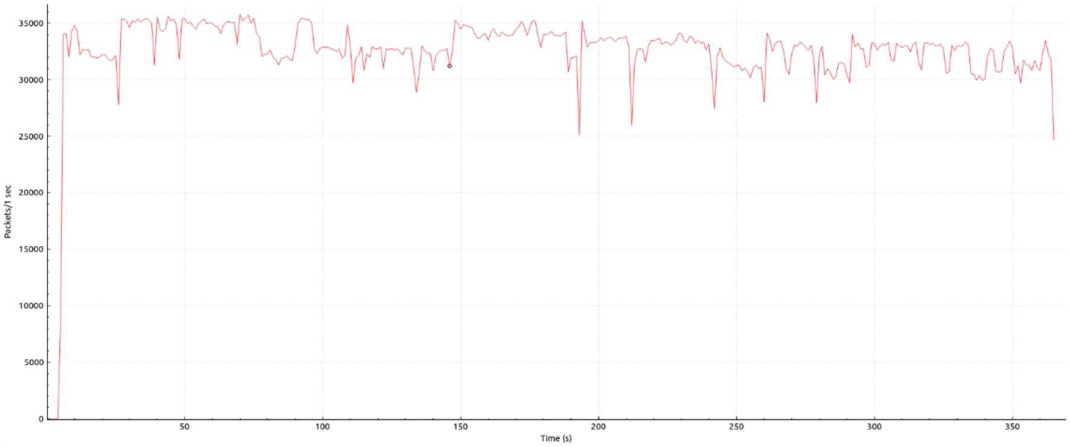


Figure 18: I/O graph showing the results of the second six minute UDP packets experiments

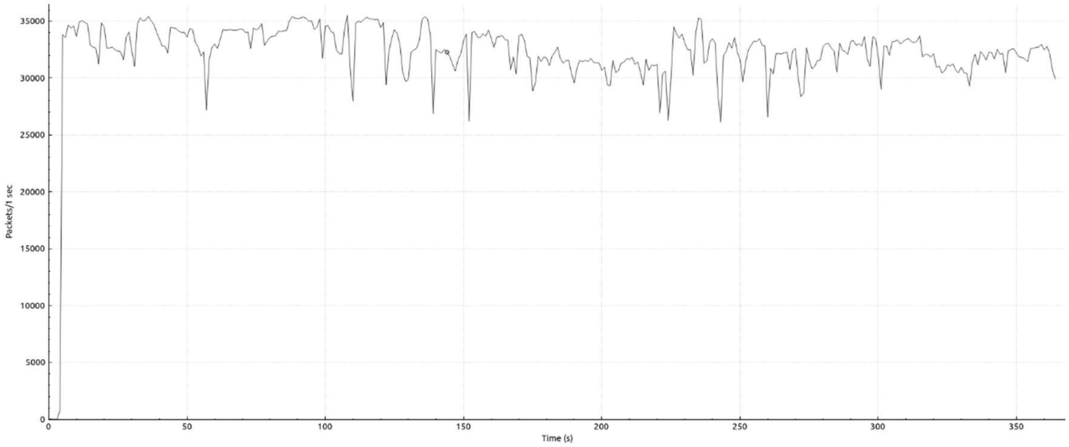


Figure 19: I/O graph showing the results of the third six minute UDP packets experiments

The next three experiments ran the ten million packets UDP attack python script. As with the six minute UDP attack, the results produced from the three experiments were very similar, all of which sending ten million packets in around five minutes. Figures 20 to 22 show the three experiments.

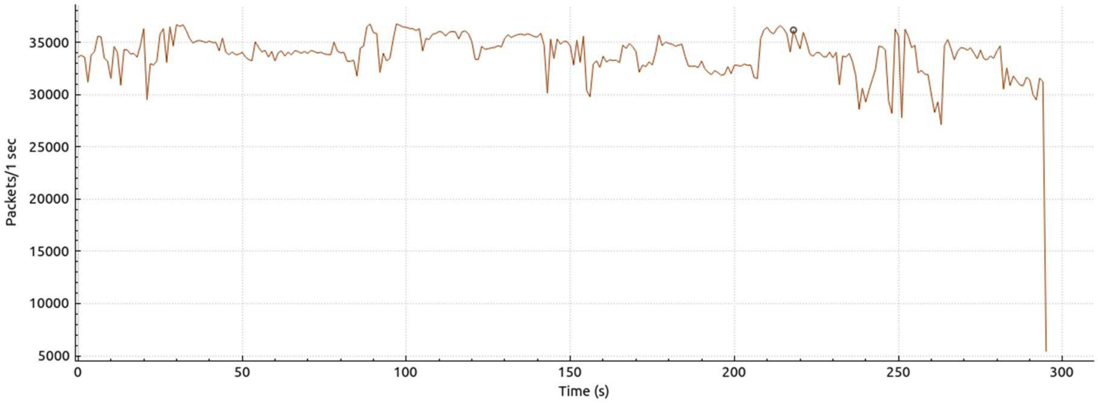


Figure 20: I/O graph showing the results of the first ten million UDP packets experiments

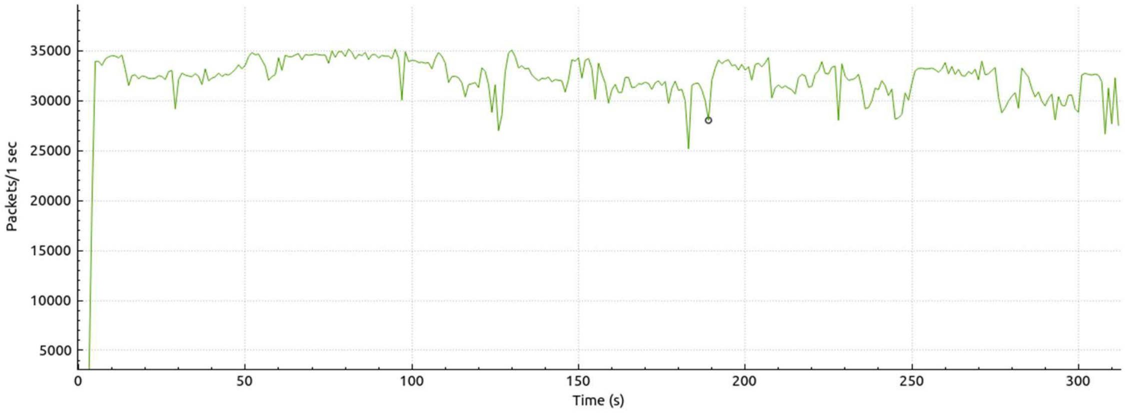


Figure 21: I/O graph showing the results of the second ten million UDP packets experiments

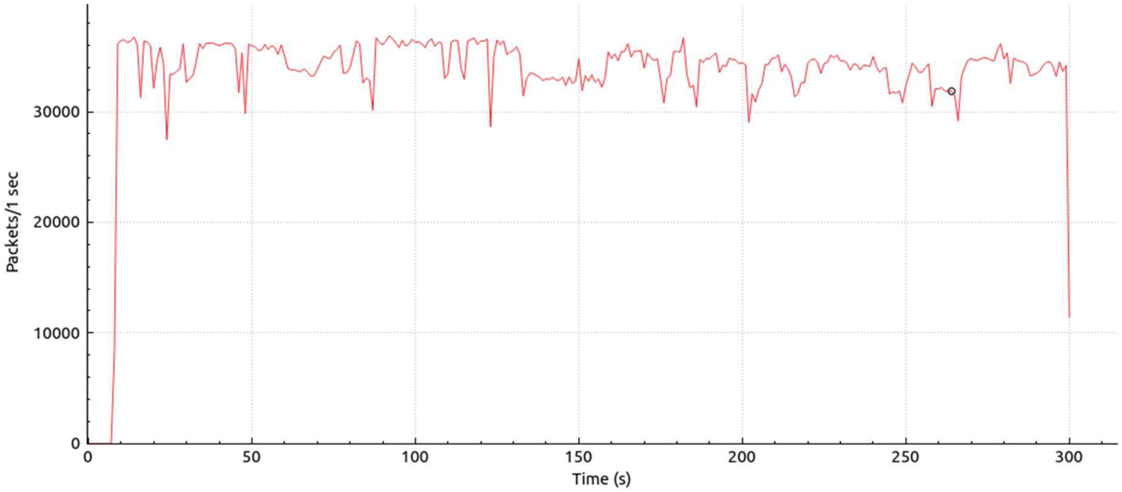


Figure 22: I/O graph showing the results of the third ten million UDP packets experiments

The first TCP attack experiment sent 714 packets in four hours. Figure 23 shows one major spike towards the end of the four hour attack that sent 125 packets per second. It also shows that there were four other notable spikes that sent between 60 and 90 packets per second. These spikes occurred roughly every hour with the exception of the smallest of the four spikes. The major spike also aligns with this pattern.

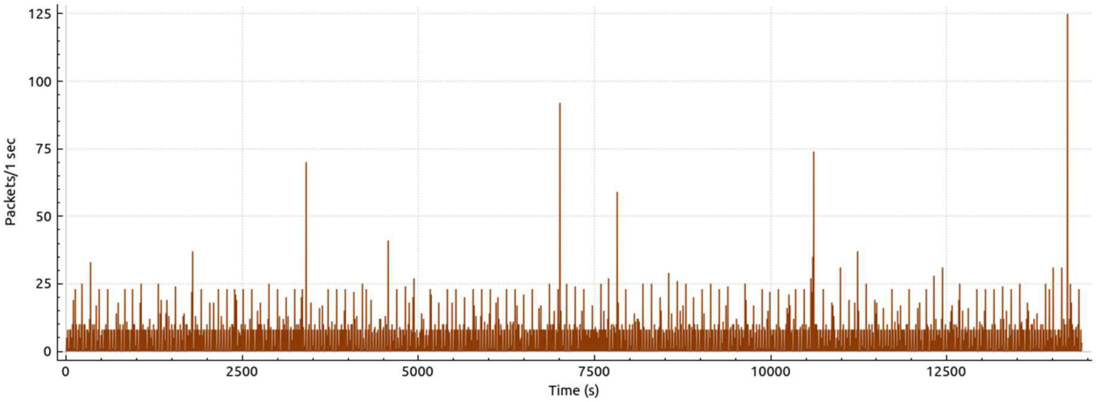


Figure 23: I/O graph presenting the network traffic of the first TCP attack on the physical environment

The second TCP attack experiment sent 717 packets in the four hours. Figure 24 shows four major spikes during this time that varied from 115 to 140 packets per second. As with the first experiment, these spikes occurred roughly every hour.

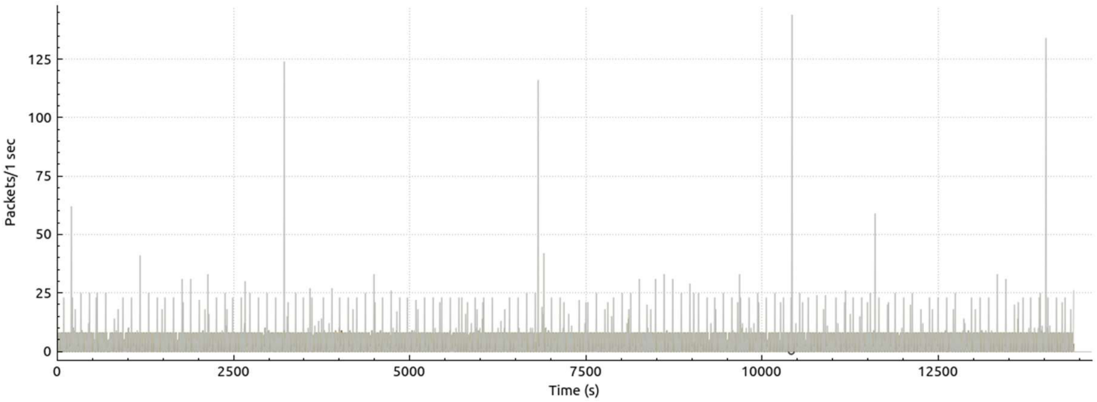


Figure 24: I/O graph presenting the network traffic of the second TCP attack on the physical environment

The third TCP attack experiment also sent 717 packets during the four hour attack. Figure 25 shows there was one major spike that sent 1200 packets in a second which is significantly more, than seen in the other two experiments. There were also four smaller but notable spikes, all approximately 125 packets per second, which followed the same pattern and occurred roughly every hour.

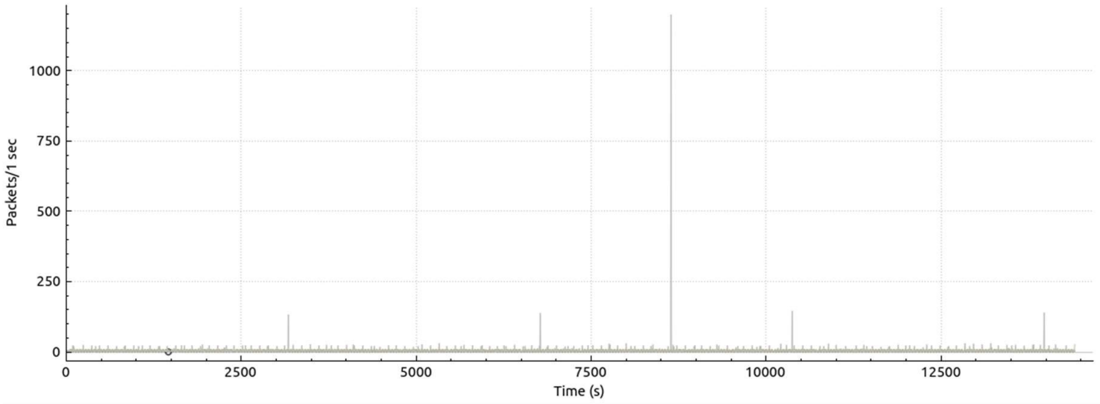


Figure 25: I/O graph presenting the network traffic of the first TCP attack on the physical environment

The three experiments that ran the four hour TCP attack all sent similar amount of packets and had a comparable pattern to their spikes. However, all of the network captures showed spikes in the amount of TCP packets being sent. This could be because of the activity going on in the background, for example, the operating system searching for updates to do, and therefore when there was no background activity, more TCP packets were able to be sent. However, the three experiments showed a varying degree of frequency and severity of these spikes. Experiment three showed a lower rate of packets per second for the majority of the four hour period except for one huge spike and smaller spikes that were comparable to the ones found in the previous two experiments. The results from experiments one and two were more similar but the spikes in experiment two were more consistent whereas the spikes in experiment one varied more in packets per second.

6.2. Results and Analysis of Private Cloud Experiment

The results of the ping cycle showed a pattern of the ICMP packets being sent and returned every second of the minute captured. This is shown in figure 26.

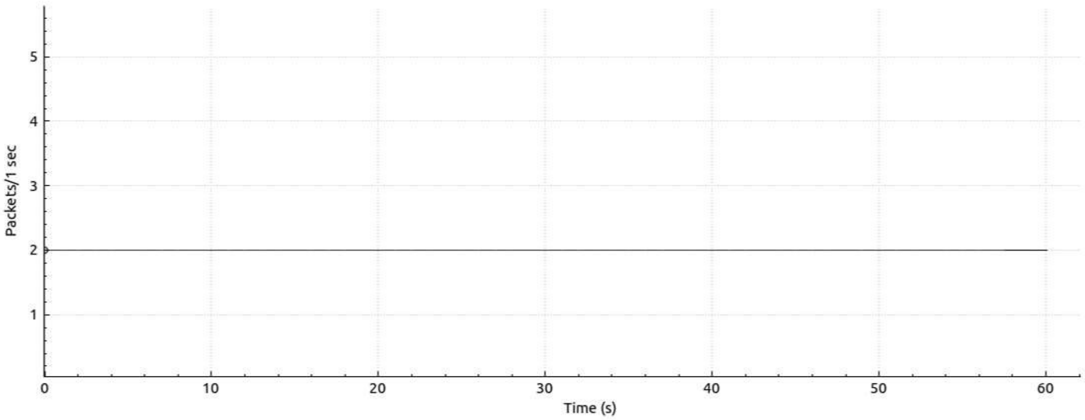


Figure 26: I/O graph of the one minute ping cycle in the private cloud environment

Three experiments were run to investigate the network traffic produced during a six minute UDP attack. All of these experiments produced very similar results. All three experiments consistently sent between 12000 and 15000 packet per second but all had several spikes which sent between 21000 and 24000 packets per second. The first experiment sent 5881096 packets, the second 5571940 packets and the third 6308093 packets. Figure 27 shows the similar pattern between these three experiments. Figure 28 to 30 shows the network traffic captures from the three experiments individually.

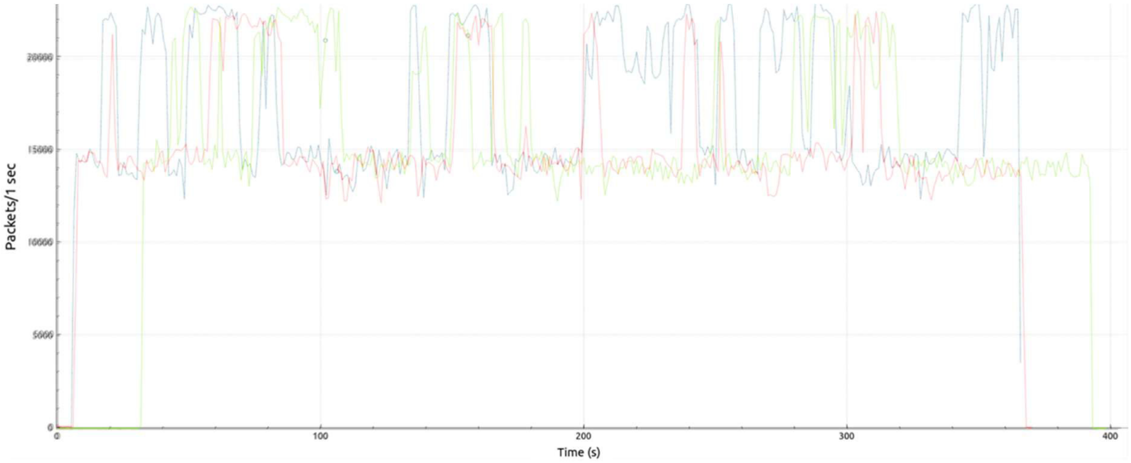


Figure 27: I/O graph comparing the three six minute UDP attacks in the private cloud environment.

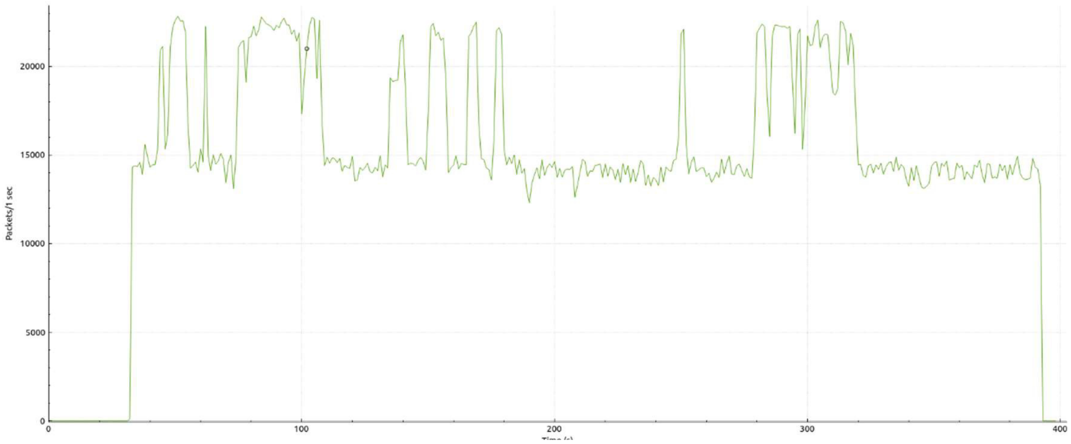


Figure 28: I/O graph showing the first six minute UDP attack in the private cloud environment.

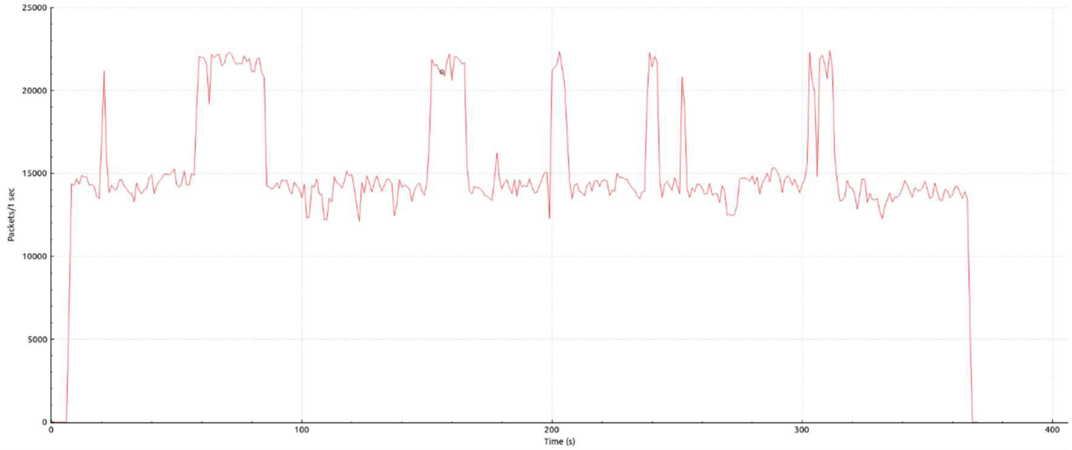


Figure 29: I/O graph showing the second six minute UDP attack in the private cloud environment.

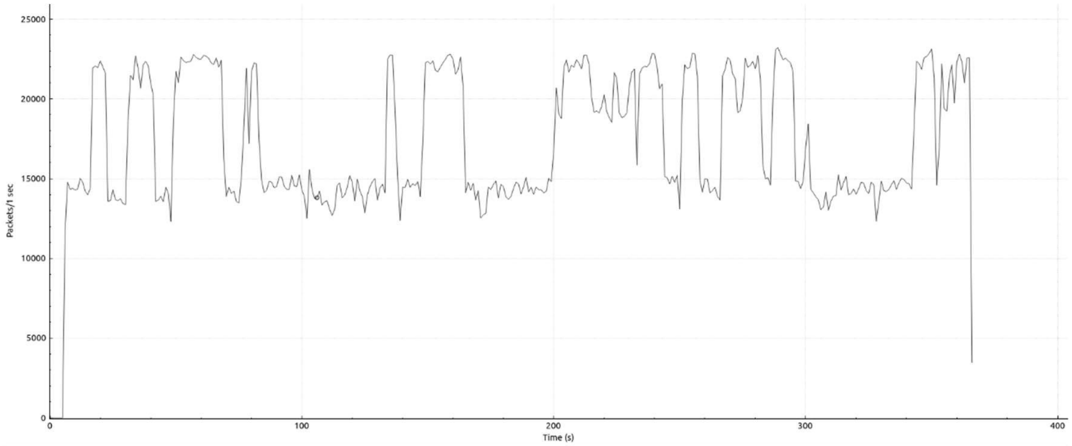


Figure 30: I/O graph showing the third six minute UDP attack in the private cloud environment.

The ten million packet UDP attack python script was ran three times. The first two experiments produced similar patterns and sent ten million UDP packets in five and a half minutes and six minutes. The similarities in both of these experiments can be found in figure 31. The third experiment using the ten million UDP attack python script produced a similar pattern to the other two experiments because the amounts of packets per second were consistently around 11000 to 16000, with the spikes in packets being between 20000 and 23000. The time was also similar and took just over five minutes. This can be shown in figure 32. The I/O graph produced for the third experiment could not be compared to the other two because the initiation of the Wireshark capture was immediate and did not have a delay like the first two experiments.

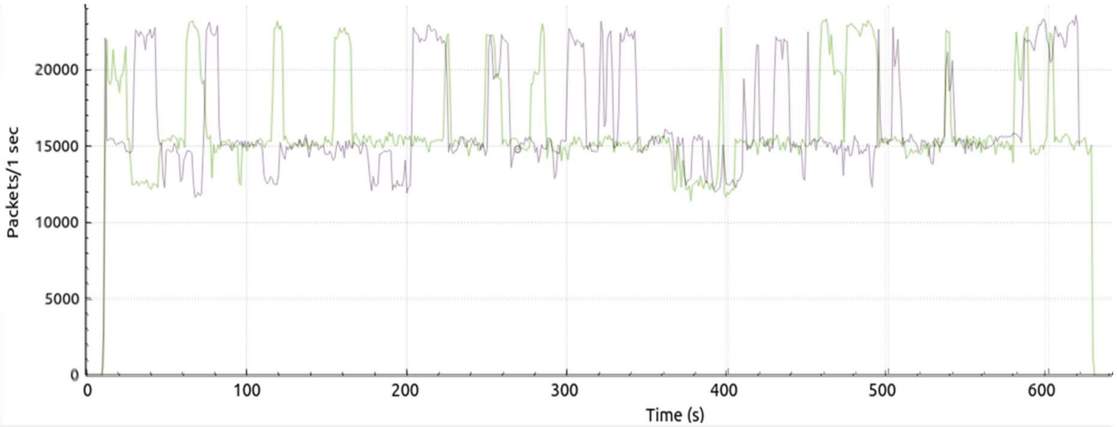


Figure 31: I/O graph comparing the first two ten million packet UDP attacks in the private cloud environment.

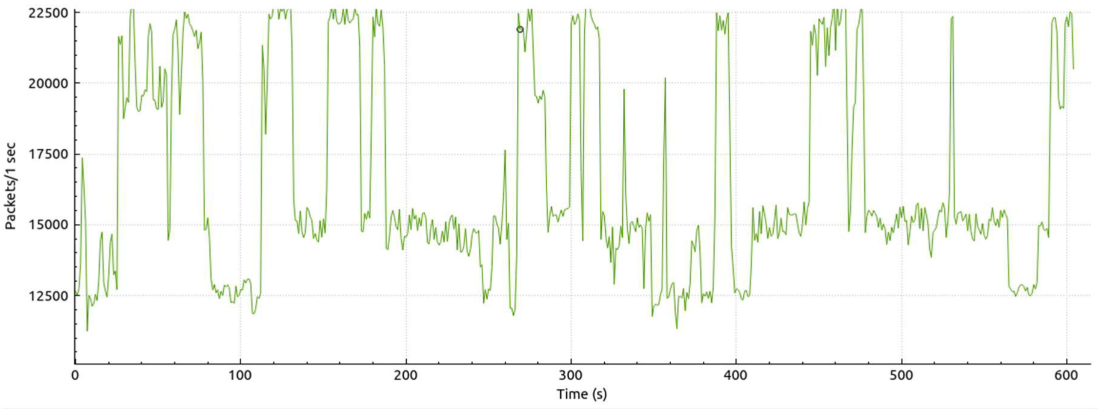


Figure 32: I/O graph showing the third ten million packet UDP attack in the private cloud environment.

Unfortunately, due to the unforeseen circumstance of the cyber range used to create the private cloud experiment being out of service during experimentation, the four hour TCP attack was not able to carried out. Based on how the private cloud environment performed during the UDP attacks and when compared with the physical environment results, it can be predicted that the TCP attack would have sent fewer TCP packets than in the physical environment. This assumption can also be drawn when evaluating the amount of steps through the network stack required to send a packet in a physical environment versus a virtualised environment. The use of hardware and software to send packets requires less steps through the network stack as a opposed to purely using software to send packets. On the basis of the four hour TCP attack performed on the physical environment, it can be anticipated that there would be spikes in the amount of TCP packets sent per second. This can be seconded by multiple research papers, including one by Ohsita, Y., Ata, S. and Murata, M., that analysed TCP packets, a study by Haggerty, J., Shi, Q. and Merabti, M. that established the need for early detection for denial of service attacks and a paper by Alekseev, I.V. that determined patterns from large scale distributed denial of service attacks. All of these studies also witnessed spikes in TCP packets during a TCP denial of service attack [42] [43] [44].

6.3. Results and Analysis of Public Cloud Experiment

The one minute ping cycle between the virtual machines produced the results shown in figure 33. It shows that there is a consistent pattern of a packet being sent and returned every second apart from one instance.

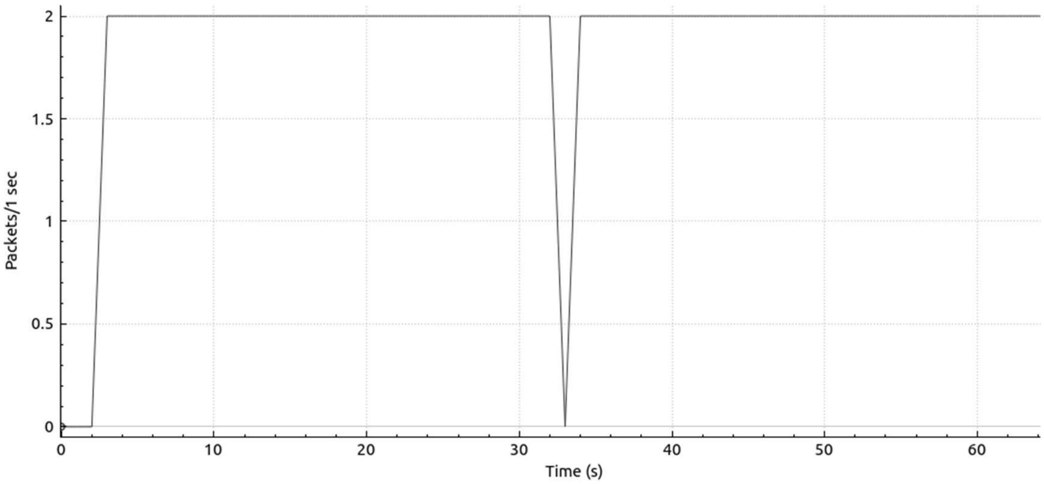
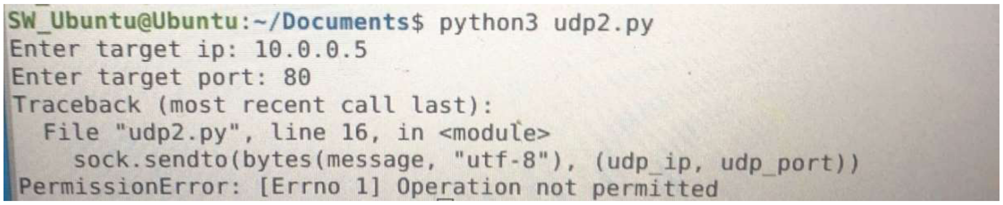


Figure 33: I/O graph of the one minute ping cycle in the public cloud environment

Running the same python scripts used in the other two environments was not possible in the Azure virtual machines. When the UDP six minute attack python script was ran, it caused the entire virtual machine to crash and the error message shown in figure 34 to be produced. A similar outcome was shown when the TCP attack was attempted.



```
SW_Ubuntu@Ubuntu:~/Documents$ python3 udp2.py
Enter target ip: 10.0.0.5
Enter target port: 80
Traceback (most recent call last):
  File "udp2.py", line 16, in <module>
    sock.sendto(bytes(message, "utf-8"), (udp_ip, udp_port))
PermissionError: [Errno 1] Operation not permitted
```

Figure 34: Permission error produced in public cloud UDP attack

To overcome this, the amount of UDP packets sent in the python script was altered to 2500. This caused the virtual machine to slow down significantly but allowed

Wireshark to capture the network traffic. Figures 35 to 37 show the three experiments ran sending 2500 UDP packets to victim virtual machine.

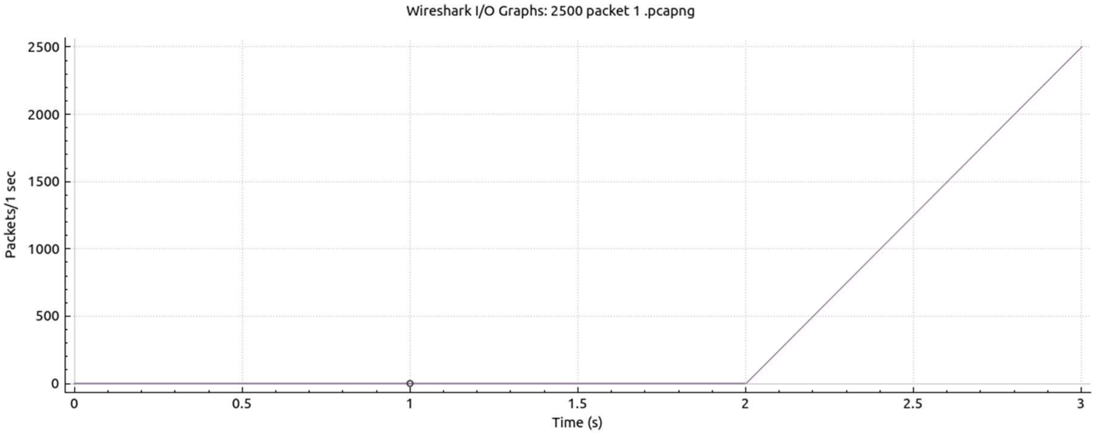


Figure 35: I/O graph of the first 2500 packet UDP attack in the public cloud environment

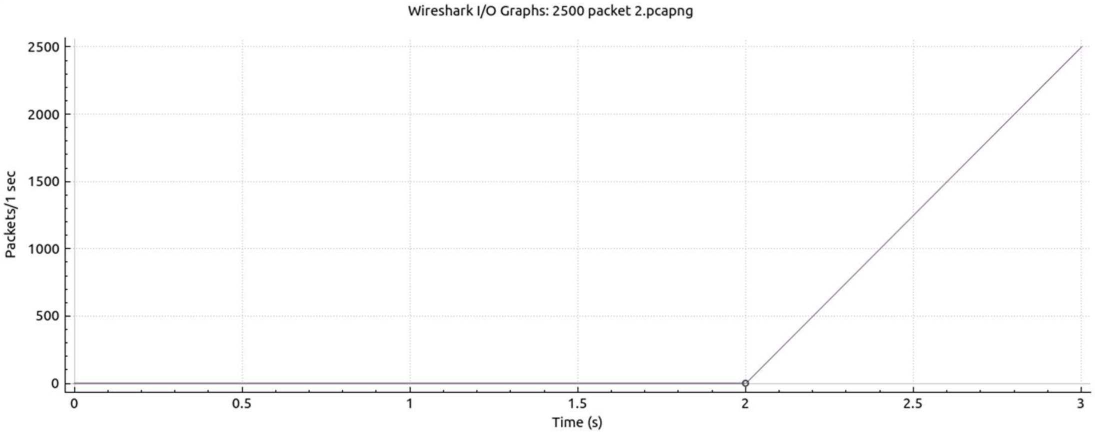


Figure 36: I/O graph of the second 2500 packet UDP attack in the public cloud environment

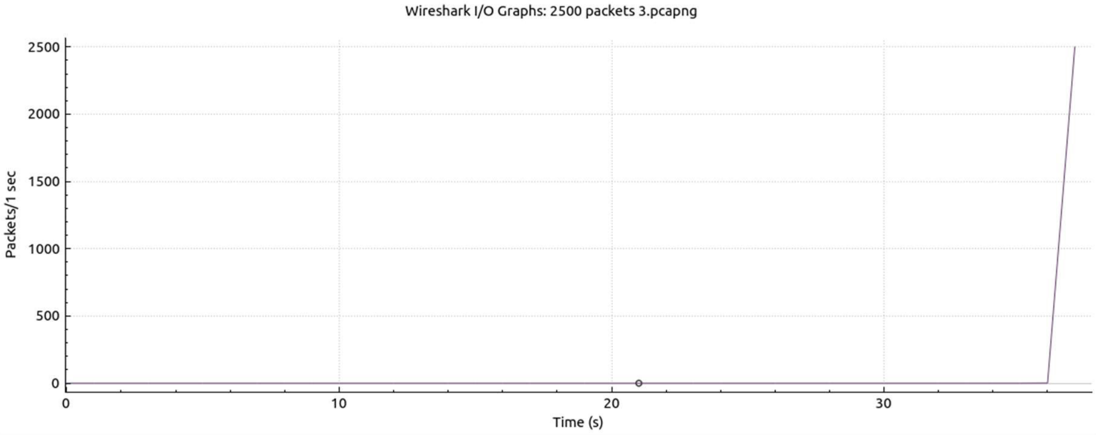
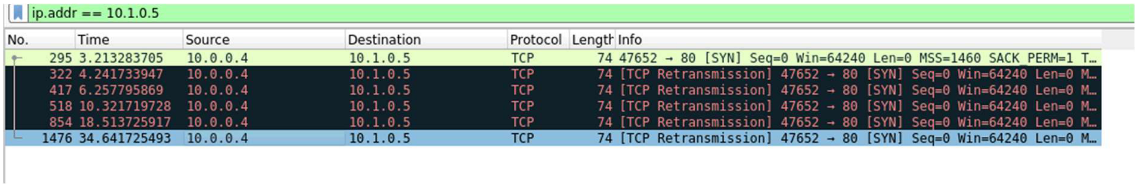


Figure 37: I/O graph of the third 2500 packet UDP attack in the public cloud environment

Figures 35 and 36 show very similar results, in both experiments the UDP packets started to send at the two second mark and finished at three seconds. However, figure 37 showed packets didn't start sending until 36 seconds but was similar to figures 35 and 36 by finishing a second later at 37 seconds. The delay in sending the packets does not appear to be a result of the attack but potentially caused by a delay in the execution of the python script.

The same solution was attempted when performing the TCP attack. However, when the amount of packets was lowered to two, the virtual machine immediately stopped any kind of TCP packet transfer. Figure 38 shows the error that was shown on the Wireshark capture. This meant no results could be produced for a TCP attack in the public cloud environment.



ip.addr == 10.1.0.5

No.	Time	Source	Destination	Protocol	Length	Info
295	3.213283705	10.0.0.4	10.1.0.5	TCP	74	47652 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK PERM=1 T…
322	4.241733947	10.0.0.4	10.1.0.5	TCP	74	[TCP Retransmission] 47652 → 80 [SYN] Seq=0 Win=64240 Len=0 M…
417	6.257795869	10.0.0.4	10.1.0.5	TCP	74	[TCP Retransmission] 47652 → 80 [SYN] Seq=0 Win=64240 Len=0 M…
518	10.321719728	10.0.0.4	10.1.0.5	TCP	74	[TCP Retransmission] 47652 → 80 [SYN] Seq=0 Win=64240 Len=0 M…
854	18.513725917	10.0.0.4	10.1.0.5	TCP	74	[TCP Retransmission] 47652 → 80 [SYN] Seq=0 Win=64240 Len=0 M…
1476	34.641725493	10.0.0.4	10.1.0.5	TCP	74	[TCP Retransmission] 47652 → 80 [SYN] Seq=0 Win=64240 Len=0 M…

Figure 38: Wireshark capture of TCP attack on the public cloud environment

If the four hour TCP attack experiments were able to ran without error, it can be believed that the same outcome would be produced as the one discussed for the private cloud environment. Comparisons between the data captured across the environments, warrants the assumption that a four hour TCP attack on a public cloud environment would send fewer packets than in the same attack on a physical environment. This is potentially due to the way hardware and software are used in each environment and how they interact with the network stack. However, the research in this project allows the belief that a TCP attack on a public cloud environment would have spikes in the amount of packets sent per second [42] [43] [44], which is the same as the pattern of a physical environment.

The results that were produced in this environment are clearly differing to the other environments because of the difficulties performing the attacks and capturing the network traffic. However, the result produced allowed rough comparisons to the physical and private cloud environment, which can be found in the next section.

6.4. Environment Comparison and Discussion

The one minute ping cycles performed on each environment showed that the private cloud environment was the most reliable as it sent and returned a packet every second for the entire 60 seconds, but the public cloud closely followed with one instance not returning the packet sent. The physical network was the least reliable as there were eight instances where no packet was returned. This is shown in figure 39.

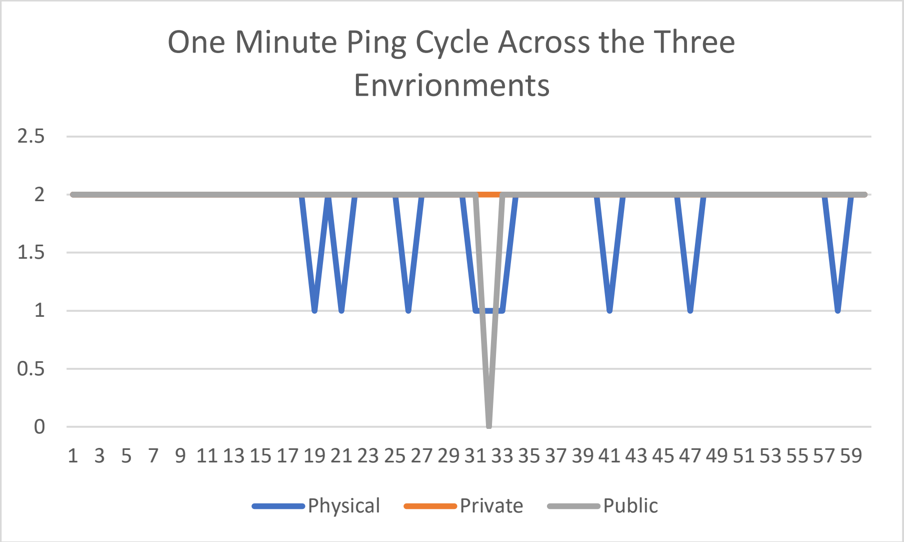


Figure 39: Comparison of a one minute ping cycle across the three environments

There are restrictions in place that limit the amount of network traffic that can occur at a given time when using in Microsoft Azure virtual machines. The restriction is implemented because resources are shared on public cloud environments. Therefore, when a UDP denial of service attack is executed on the virtual machines created on Microsoft Azure for this project, it does not affect just the designated victim machine but any virtual machine sharing the same physical machine with the same processor.

This restriction of Microsoft Azure caused the results captured in the public cloud environment to be limited. The public cloud environment was only able to send 2500 packets and therefore, the comparison between the three environments needed to be altered. The average time it took to send 2500 packets was calculated for each environment. This is not an ideal solution because 2500 packets does not create an accurate representation of a realistic UDP denial of service attack but it does allow the three environments to be compared of a small scale.

This comparison showed that the physical network was significantly quicker at sending 2500 UDP packets than the cloud environments. Figure 40 shows the average time it took to send 2500 UDP packets in each environment. To do this, the average time taken to send 2500 UDP packets in the physical and private cloud environments were first calculated and the time it took to send 2500 UDP packets was used in the public cloud environment.

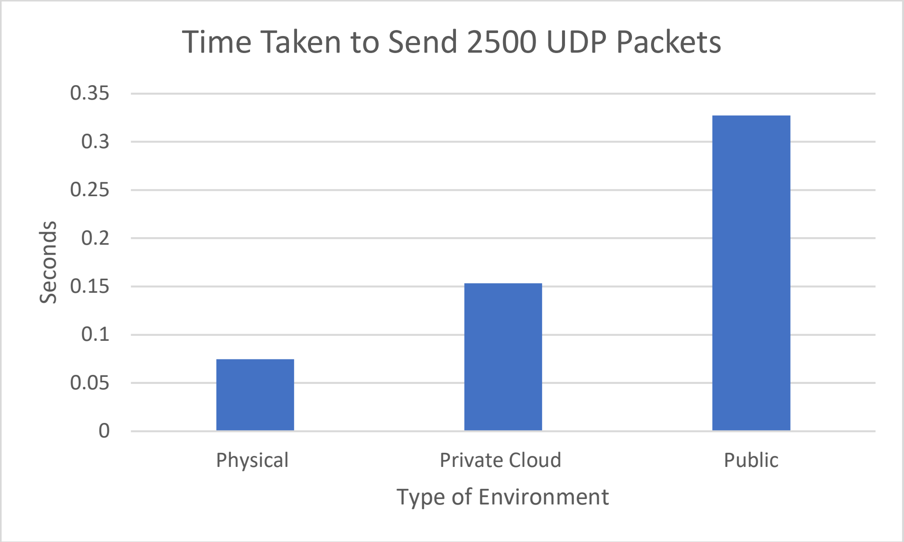


Figure 40: Comparison of the average time it took the three environments to send 2500 packets.

The public cloud environment was shown to be significantly slower than the other two environments. However, this may be due to the unusual result produced from the third experiment on the public cloud environment. This would have dramatically brought up the average time to send 2500 UDP packets. This result could be an outlier but would need further experimentation to prove this. Figure 41 shows the average time taken to send 2500 UDP packets in the physical and private cloud environments and the time it took to send 2500 UDP packets in the public cloud environment for each experiment.

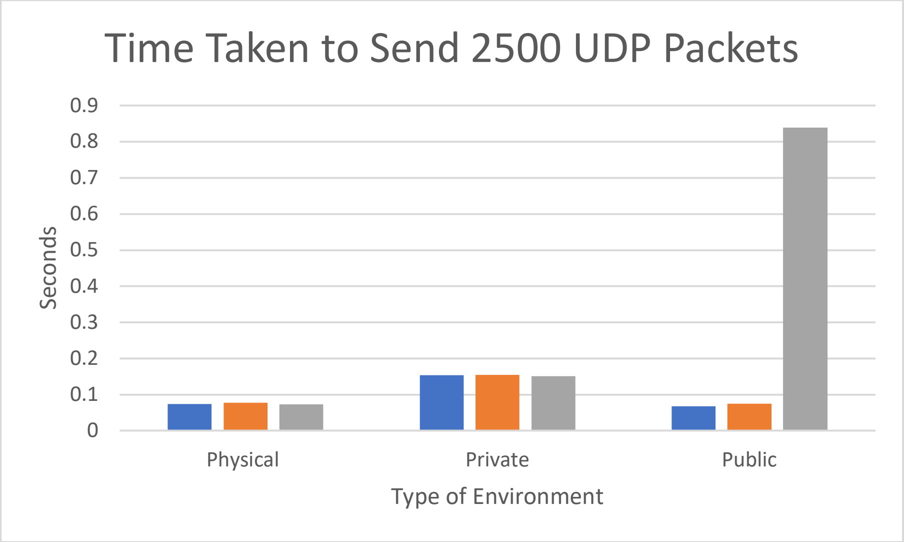


Figure 41: Comparison between the time taken to send 2500 packets in each environment

The results from the six minute UDP denial of service attack performed on the physical and private cloud environment showed a significant difference, see figure 42. The average amount of packets sent over the three experiments for the physical environment was 11842390 whereas it was 5920376.333 in the private cloud environment. This significant difference is potentially caused by the role hardware and software have in each environment. The physical environment uses the traditional method of flowing through the network stack because it uses hardware and software to process and transfer its data. However, the private cloud environment purely uses software to process and transfer its data and consequently has an alternative flow through the network stack, as shown in the Background chapter. The different approach of going through the network stack could have significant time implications.

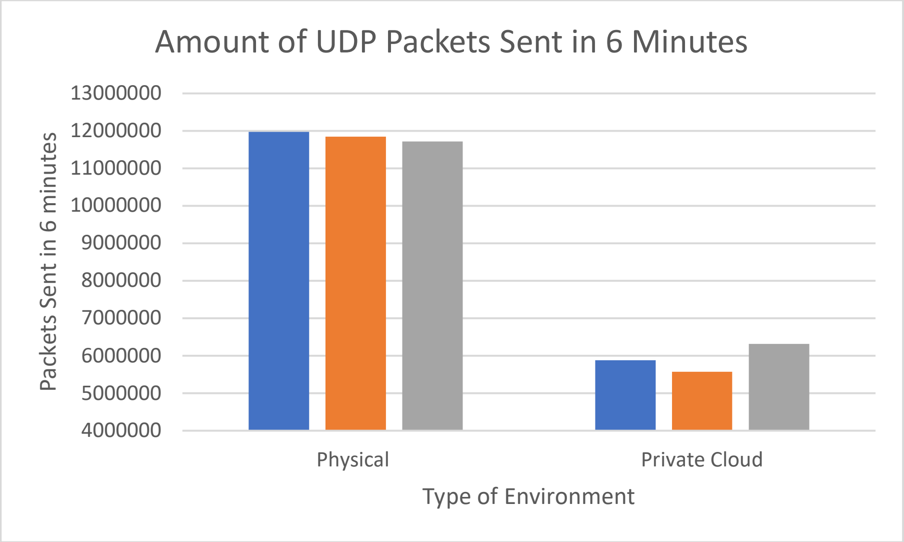


Figure 42: Comparison between the amount of UDP packets sent in six minutes in the physical and private cloud environment

When designing a denial of service attack, the environment needs to be considered. This is because, if a cloud environment is sending approximately half the amount of packets than a physical environment over a six minute period, the length of the attack needs to be increased. Alternatively, the attacker could set the amount of packets they wish to send, as done in the ten million packet UDP attack. However, when taking this approach, the attacker needs to be aware that it will take considerably longer than in a physical environment, as shown in figure 43.

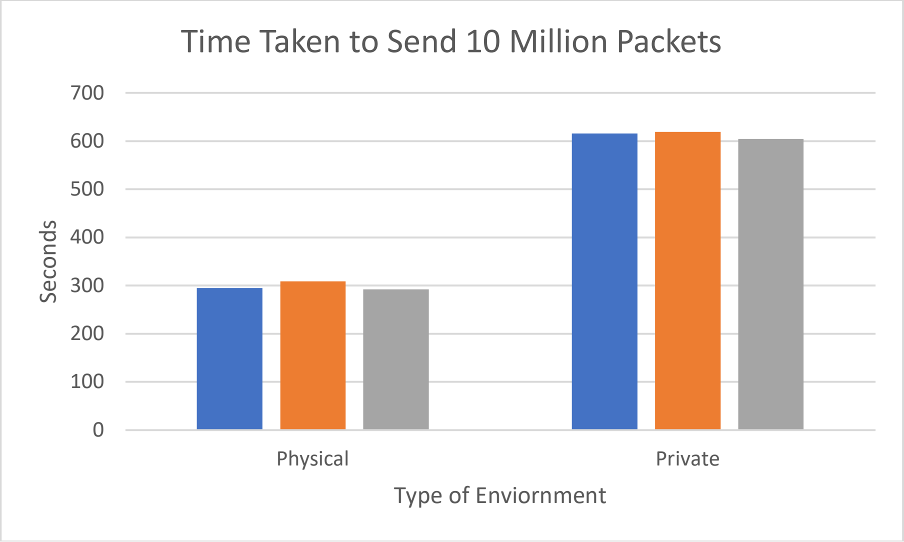


Figure 43: Comparison of the amount of time it takes to send ten million UDP packets in the physical and private cloud environment

Conclusion

During this chapter, the data captured during the three variations of denial of service attacks were presented and analysed. The results from each environment were compared and potential reasonings for their differences were articulated. In the next chapter (Conclusion and Future Work), the aims and objective of this project will be evaluated to determine whether they have been met and state any future work that would build upon the outcome of this project.

7. Conclusion and Future Work

Based on the content in the previous chapter, this chapter will review the aims and objectives of this project to determine whether they have been met and to what extent. Any future work and improvement that would be ideally done to improve this the outcome and reliability of this project will also be presented.

7.1. Research Overview

The research of this project is aimed at comparing different denial of service attacks in a physical, private cloud and public cloud environment. The research is focused on how these environments differ and the impact this has on the way a denial of service effects the environment. The captured data from demonstrating UDP and TCP denial of service attacks was used to compare the effects they had across the three environments. This data was graphically represented using I/O and Excel derived graphs where possible. The outcome from this research would aid in understanding the differences between public cloud and private cloud environments as well as physical environments. By understanding this, it could then be used to implement the relevant security measures.

This research has highlighted that the internal traffic management restrictions in place within public cloud environments prevent the testing of cloud security within these environments. Although it is sensible to have these restrictions in place when resources are being shared, it is also crucial that testing is able to occur on cloud environments to ensure their security. To overcome this, private cloud environments, created in cyber ranges, need to be utilised and shared across researchers to allow the testing of cloud security against attacks such as denial of service. Alternatively, public cloud providers could publish their results when they test the environments resilience against attacks and have a third party conduct the same tests to verify them.

This project aimed to test whether there was a difference between simulated and real datasets when the same attack was executed. When tested, the private cloud environment produced almost half the amount of UDP packets than the physical environment in the same amount of time. It also took almost twice as long to send ten million UDP packets in the private cloud environment compared to the physical environment. This is a significant difference in the datasets across the two environments and raises the concern about scalability. If these experiments were done on a larger scale, would the private cloud environment still take twice as long or send half the amount of packets? This could be a major limitation to the use of simulated data for testing.

Despite the differences in the simulated data produced in cloud environments and the real data produced in physical environments, cloud environments are growing in popularity because of the benefits discussed throughout this project. This raises the question of whether it matters that the simulated data is contrasting to the real data if there are other benefits of using cloud platforms. It also requires investigation as to the extent of the limitations of simulated data and whether it is a deeper issue than just slower timings. These are all important follow up questions from this project that become more important to answer the more we rely on the use of cloud based services and simulated data.

7.2. Aims and Objectives

The primary aim of this thesis was to answer the research question which was formulated from the literature review and the research objectives to be achieved:

> *To what extent does the effect of a denial of service attack on a public cloud environment differ against a private cloud environment and a physical environment?*

The following research objectives were achieved to answer this research question:

Objective One: Perform extensive and systematic research in the literature review on denial of service attacks, virtualisation, cloud environments and denial of service attacks in cloud environments.
To meeting this objective, an extensive literature review was conducted to cover each topic in this project. The research covered frequently used denial of service attacks, how they work and the impact they have had in real life scenarios. The technology surrounding cloud environments, both private and public, were covered in depth and compared to the technology used in physical environments. Research was also carried out around the differences between simulated and real data across different industries. Studies about denial of service attacks within cloud environments and physical environments were also evaluated. Network analysis tools, that proved to be crucial in this project, were also researched to aid with the development of the experiment.

Objective Two: Propose a method of performing a denial of service attack on a physical network, a private cloud environment and a public cloud environment.
Based on the extensive research that was carried out, this objective was met by proposing the use of python scripts to perform the UDP and TCP denial of service attacks. This was a compatible method of attack within all three of the environments being tested and allowed for easy deployment in a Linux terminal. Three python scripts were proposed for creation which included a six minute UDP denial of service attack, a ten million packet denial of service attack and a four hour denial of service attack. These scripts were adapted when problems arose during the public cloud environment part of the experiment which allowed some results to still be captured.

Network diagrams were also produced during the planning of the physical, private cloud and public cloud environments. These diagrams were created based on the resources available during this project.

Objective Three: Demonstrate and compare the effects of a denial of service attack on a physical network, a private cloud environment and a public cloud environment.
A UDP attack was performed on the three environments to complete this objective. It allowed a comparison to be made across all the environments. Significant differences between the physical and private cloud environment were found from both of the UDP denial of service attacks. The private cloud environment recorded sending almost half the amount of UDP packets during the six minute UDP attack and took approximately twice as long to send ten million packets, when compared to the physical environment.

The public cloud environment only allowed approximately 2500 packets to be sent before an error appeared and the virtual machine crashed. In order to compare all three environments during a UDP denial of service attack, the average time it took to send 2500 UDP packets in the physical and private cloud environment was calculated. This showed that physical environment was significantly less time than both cloud environments. However, there was a potential outlier in the public cloud environment results which would've considerably increased the average time across the three experiment. Further testing on this environment would be needed to determine whether this result was an outlier.

A TCP attack was also intended to be performed on all three environments, to further meet this objective, but was only able to be completed in the physical environment. The unforeseen closure of the cyber range and the internal traffic management restrictions on Microsoft Azure prevented the experiments to be carried out on the private and public cloud environments. However, assumptions of the expected outcomes were recorded based on the UDP denial of service comparison and external research.

Objective Four: Review the difference in the three different environments.
To meet this objective, the results of the UDP attack experiments were used to establish the differences between the environments. The limitations and problems that were raised during the experimental phase of this project were also used to review the differences across the three environments. It was used to recommend alternative methods of testing attacks, like denial of service, within cloud environments.

The key differences between simulated and real data were discussed in depth. The potential issue of vastly contrasting results between simulated and real data when experiments of this kind are hugely scaled was reviewed. Further questions were also raised to consider whether this difference would matter with the increased use of cloud based services.

7.3. Future Work

There are a number of potential future works and improvements identified for this research which are as follows:

7.3.1. Perform the four hour TCP attack in the Private Cloud Environment

The unfortunate event of the cyber range at Cardiff University being put out of service half way through the experiments resulted in no data being captured for the four hour TCP attack that had been created. It would be extremely helpful to gain the network traffic during the TCP attack because it could be compared to the results produced from the physical environment. It would provide an idea of how it effects cloud systems in general because it was the TCP attack was shut down immediately by Microsoft Azure when it was ran on the public cloud environment. Without unrestricted access to the Microsoft Azure system, the private cloud environment would be the closest representation of a TCP attack occurring on a public cloud environment. Therefore, making this experiment even more imperative in future work.

7.3.2. Performing distributed denial of service attacks across all of the environments

Distributed denial of service attacks were included as part of the literature review in the Background chapter because it was important to understand the differences a distributed denial of service attack had in comparison with a standard denial of service attack. However, it was not possible to carry out a distributed denial of service attack in the three environments because of the physical and financial resources available throughout this project. To build upon the experiments carried out, a distributed denial of service should be incorporated into future work, to further understand how different environments react to different variations of denial of service attacks.

7.3.3. Delay packets being sent

In an attempt to overcome the internal traffic management restrictions implemented in Microsoft Azure virtual machines, the packets should have a delay in being sent. During this project, it has been discovered that Microsoft Azure allows approximately 2500 UDP packets to be sent before it crashes and produces an error. The physical and private cloud environments were sending ten million packets to establish patterns in the network traffic captures. In the effort to get a comparable amount of packets from the public cloud environment, it would be worth trying to manipulate the packet timings in the attempt to avoid detection from the Azure system. This would be done by sending an amount of packets under the detection threshold, like 2000 packets and then pausing for a set amount of time before sending another 2000 packets. This loop could overcome the problem faced in this project but would need to be tested to confirm that it works. This would also build on the work completed by Aad, I., Hubaux, J.P. and Knightly, E.W., in 2008 that determined that a delay in packet sending resulted in it being harder for victim machines to detect an attack.

7.3.4. Consistently timed Wireshark initiation

If this experiment was reproduced, the delay between starting the Wireshark capture and starting the attack should be consistent to produce more accurate and comparable results. In this project, some of the results produced looked significantly different to the results from the other experiments using the same attack. This is thought to be because of the time when the Wireshark capture began. All of the relevant data was captured but there were at times big delays in the capture which made it more difficult to compare the datasets.

References

[1] Yamin, M.M., Katt, B. and Gkioulos, V., 2020. Cyber ranges and security testbeds: Scenarios, functions, tools and architecture. *Computers & Security*, *88*, p.101636.

[2] Lau, F., Rubin, S.H., Smith, M.H. and Trajkovic, L., 2000, October. Distributed denial of service attacks. In *Smc 2000 conference proceedings. 2000 ieee international conference on systems, man and cybernetics.'cybernetics evolving to systems, humans, organizations, and their complex interactions'(cat. no. 0* (Vol. 3, pp. 2275-2280). IEEE.

[3] Khalaf, B.A., Mostafa, S.A., Mustapha, A., Mohammed, M.A. and Abduallah, W.M., 2019. Comprehensive review of artificial intelligence and statistical approaches in distributed denial of service attack and defense methods. IEEE Access, 7, pp.51691-51713.

[4] Schuba, C.L., Krsul, I.V., Kuhn, M.G., Spafford, E.H., Sundaram, A. and Zamboni, D., 1997, May. Analysis of a denial of service attack on TCP. In *Proceedings. 1997 IEEE Symposium on Security and Privacy (Cat. No. 97CB36097)* (pp. 208-223). IEEE.

[5] MUVI. UDP (User Data Protocol). Available at: https://www.muvi.com/wiki/udpuser-datagram-protocol.html. Accessed 19/09/2022

[6] Ghazali, K.W. and Hassan, R., 2011. Flooding distributed denial of service attacks-a review. *Journal of Computer Science*, *7*(8), p.1218.

[7] IBM. 2021. EZY2393I UDP character generator. Available at: https://www.ibm.com/docs/en/zos/2.2.0?topic=messages-ezy2393i. Accessed: 20/07/2022

[8] Beitollahi, H. and Deconinck, G., 2012. Analyzing well-known countermeasures against distributed denial of service attacks. *Computer Communications*, *35*(11), pp.1312-1332.

[9] Chang, R.K., 2002. Defending against flooding-based distributed denial-of-service attacks: A tutorial. *IEEE communications magazine*, *40*(10), pp.42-51.

[10] Bhuyan, M.H., Kashyap, H.J., Bhattacharyya, D.K. and Kalita, J.K., 2014. Detecting distributed denial of service attacks: methods, tools and future directions. *The Computer Journal*, *57*(4), pp.537-556.

[11] Fekolkin, Roman. (2015). Botnet-originated DDoS attacks: Overview and Mitigation.

[12] Global, I.T., *security risks survey.(2014). Distributed denial of service (DDoS) attacks*. Technical report, Kaspersky. Retrieved from https://media. kaspersky. com/en/B2B-International-2014-Survey-DDoS-Summary-Report. pdf.

[13] Arora, K., Kumar, K. and Sachdeva, M., 2011. Impact analysis of recent DDoS attacks. *International Journal on Computer Science and Engineering*, *3*(2), pp.877-883.

[14] Solomon, B., Fox-Brewster, T. 2016. Hacked Cameras Were Behind Friday's Massive Web Outage. Available at: https://www.forbes.com/sites/briansolomon/2016/10/21/hacked-cameras-cyber-attack-hacking-ddos-dyn-twitter-netflix/?sh=2ec7d0404fb7 – Accessed 19/09/2022

[15] Radware. 2015. DDoS Case Study: Boston Children's Hospital DDoS Attack Mitigation. Available at: https://www.radware.com/security/ddos-experts-insider/ert-case-studies/boston-childrens-hospital-ddos-mitigation-case-study/. Accessed 19/09/2022

[16] Ali, I. and Meghanathan, N., 2011. Virtual machines and networks-installation, performance study, advantages and virtualization options. *arXiv preprint arXiv:1105.0061*.
[17] IBM. Virtualisation – A Complete Guide. Available at: https://www.ibm.com/cloud/learn/virtualization-a-complete-guide. Accessed 01/08/2022
[18] Che, J., Shi, C., Yu, Y. and Lin, W., 2010, December. A synthetical performance evaluation of openvz, xen and kvm. In *2010 IEEE Asia-Pacific Services Computing Conference* (pp. 587-594). IEEE.
[19] Majumdar, P. 2014. Virtualisation: Why it's a Crucial Technology for the Cloud. Available at: https://cloudacademy.com/blog/virtualization-why-its-a-crucial-technology-for-the-cloud-world/. Accessed 19/09/2022
[20] Vaughan-Nichols, S.J., 2008. Virtualization sparks security concerns. *Computer*, *41*(8), pp.13-15.
[21] Bazargan, F., Yeun, C.Y. and Zemerly, M.J., 2012. State-of-the-art of virtualization, its security threats and deployment models. *International Journal for Information Security Research (IJISR)*, *2*(3/4), pp.335-343.
[22] Carroll, M., Kotzé, P. and Merwe, A.V.D., 2011, May. Securing virtual and cloud environments. In *International Conference on Cloud Computing and Services Science* (pp. 73-90). Springer, New York, NY.
[23] IBM. Get Started With Cloud Computing. Available at: https://developer.ibm.com/learningpaths/get-started-cloud/ - Accessed 12/08/2022
[24] Amazon. AWS. Available at: https://aws.amazon.com/. Accessed 13/08/2022
[25] Microsoft. Azure. Available at: https://azure.microsoft.com/en-gb/. Accessed 13/08/2022
[26] Fernandes, D.A., Soares, L.F., Gomes, J.V., Freire, M.M. and Inácio, P.R., 2014. Security issues in cloud environments: a survey. *International Journal of Information Security*, *13*(2), pp.113-170.
[27] Qian, L., Luo, Z., Du, Y. and Guo, L., 2009, December. Cloud computing: An overview. In *IEEE international conference on cloud computing* (pp. 626-631). Springer, Berlin, Heidelberg.
[28] Sadiku, M.N., Musa, S.M. and Momoh, O.D., 2014. Cloud computing: opportunities and challenges. IEEE potentials, 33(1), pp.34-36.
[29] DIATEAM. What is a cyber range? Available at: https://www.diateam.net/what-is-a-cyber-range/. Accessed 13/08/2022
[30] Chouliaras, N., Kittes, G., Kantzavelou, I., Maglaras, L., Pantziou, G. and Ferrag, M.A., 2021. Cyber ranges and testbeds for education, training, and research. *Applied Sciences*, *11*(4), p.1809.
[31] DIATEAM. Hns_platform_presentation_en.pdf [Presentation] Accessed 19/09/2022
[32] Mierzwa, Stan. 2017. Next Generation Defenses for a Hyper Evolving Threat Landscape An Anthology of ICIT Fellow Essays Volume I.
[33] anil933. 2009. *Protocol Pt1* [Presentation]. Accessed 01/09/2022
[34] Niznan, J., Papousek, J. and Pelánek, R., 2015. Exploring the Role of Small Differences in Predictive Accuracy using Simulated Data. In AIED Workshops.
[35] Somani, G., Gaur, M.S. and Sanghi, D., 2015, September. DDoS/EDoS attack in cloud: affecting everyone out there!. In *Proceedings of the 8th International Conference on Security of Information and Networks* (pp. 169-176).
[36] Ficco, M. and Rak, M., 2014. Stealthy denial of service strategy in cloud computing. *IEEE transactions on cloud computing*, *3*(1), pp.80-94.

[37] Wireshark. About Wireshark. Available at: https://www.wireshark.org/. Accessed 19/09/2022
[38] Srivastava, K., 2020. Socket Programming: Implement it using Python (TCP/IP). Available at: https://kartiksrivastava238.medium.com/socket-programming-implementing-it-using-python-tcp-ip-899ac914d284#:~:text=Socket%20Package%20%3A-%20Python%20has%20an%20inbuilt%20package,we%20are%20establishing%2C%20is%20it%20TCP%2FIP%20or%20UDP. – Accessed 19/09/2022
[39] Agile Alliance. Agile 101. Available at: https://www.agilealliance.org/agile101/ - Accessed 20/08/2022
[40] Sharma, M. 2021. The average DDoS attack only lasts a few minutes. Available at: https://www.techradar.com/news/the-average-ddos-attack-only-lasts-a-few-minutes. Accessed 12/08/2022
[41] Microsoft. Install and configure xrdp to use Remote Desktop with Ubuntu. Available at: https://docs.microsoft.com/en-us/azure/virtual-machines/linux/use-remote-desktop?tabs=azure-cli – Accessed 10/09/2022
[42] Ohsita, Y., Ata, S. and Murata, M., 2004, November. Detecting distributed Denial-of-Service attacks by analyzing TCP SYN packets statistically. In IEEE Global Telecommunications Conference, 2004. GLOBECOM'04. (Vol. 4, pp. 2043-2049). IEEE.
[43] Haggerty, J., Shi, Q. and Merabti, M., 2002, December. Beyond the perimeter: the need for early detection of denial of service attacks. In 18th Annual Computer Security Applications Conference, 2002. Proceedings. (pp. 413-422). IEEE.
[44] Alekseev, I.V., 2020. Detection of distributed denial of service attacks in large-scale networks based on methods of mathematical statistics and artificial intelligence. Automatic Control and Computer Sciences, 54(8), pp.952-957.
[45] Aad, I., Hubaux, J.P. and Knightly, E.W., 2008. Impact of denial of service attacks on ad hoc networks. IEEE/ACM transactions on networking, 16(4), pp.791-802.