# Chapter 3: Methodology

## 3.1 Tools used to Launch the Attacks

In every scenario, the tools used by the user in this research, are tools which vary from one another. An essential tool used in this research was *Wireshark*; this type of tool was used to gather the data in transit between the instances. This type of tool is a sniffer and gathers packets that are both sent or received. Another essential factor about this tool is, that every kind of information is presented with an IP address, together with the relevant information on the packet itself, such as its size, for example. Another tool that was utilized during the methodology stage was *Kali Linux*, which is a Virtual Machine. This virtual machine was another essential tool because attacks were launched by using the tool available in the Virtual Machine. One of the available tools was *Ettercap*; this type of tool was used to launch a Man-in-the-Middle attack between the instance and the virtual machine. Another critical tool presented by the Kali Linux VM was the *Metasploit*. This tool was applied to launch a vulnerability attack against the Linux VM where OpenStack was launched.

## 3.2 Type of Implementation of the Environment

As referred to in the Literature Review section, the only way to see whether something is working correctly and as expected - such as the case the features offered by OpenStack, is to keep our infrastructure and instances safe, in order to be utilized by the user, since it is the user who is willing to use the cloud environment instead of the physical environment. Keystone is going be used to carry out four tests to verify whether it is protecting the instance's data, while it is being transferred to another instance.

Referring to Figure 3, the plan of implementing the infrastructure was to create the instances, and then by putting the instances into a group using Nova. Afterwards, the tool *Swift* is used, to store the data of the instances and finally, the Keystone tool is implemented, as this tool is utilized for the instances to make sure that only the authorised user can access his data accordingly. This topology is done to test the safety of OpenStack, and to see whether OpenStack tools are secure enough for the user to feel

safe in a cloud environment.

## 3.3 Knowledge on the Implementation

Since implementing a cloud environment and making use of the IaaS was a novelty and new experience for the researcher, this research was rather challenging, despite the fact the researcher had limited knowledge and experience on how to implement cloud infrastructure. A slight understanding of how to use the commands of Linux came as an advantage to the researcher, as such a general understanding of some of the commands were later used and implemented in the cloud environment. The research gave the author a better insight into the subject matter. The knowledge of how to use the management of Linux, is an exciting area which the researcher has found an interest in, as this assisted the methodology stage of this research. The implementation of the topology took place by following an excellent source Ubuntu (2020) leading the researcher to the methodology stage.[21]

## 3.4 Queries and Limitations

The implementation of the topology of OpenStack is quite straightforward to implement, but in the researcher’s case, there were limitations on the IaaS as a license was not purchased. Some of the restrictions were:

* Only two instances can be in use i.e. the OpenStack Server and the instance created. Another instance was created but was not accessible either by SSH connection, by a desktop experience or by a terminal.
* The instance cannot be accessed neither by terminal nor through a desktop experience. The instance was accessible only by an SSH connection.
* While utilizing the instance by an SSH connection, the user did not have full control of the instance.

## 3.5 Specifications used for Virtual Machines

Not every device has the same set of specifications, and every specification needs to be set correctly for the computer to work correctly. In this study, the set of specifications set on a VM, were set correctly from a reference that was in use to complete the infrastructure of the topology used in this research. Many VMs were installed and used for testing, to be direct and these are the specifications to test and to launch the methodology stage.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Name** | **Type of system** | **Type** | **RAM** | **CPU** | **Hard-Disk** | **Network Adapter 1** | **Network Adapter 2** |
| Openstack vm (24-2-2020) | GUI | To present work | 16 GB | 2 | 50 GB | NAT | VMnet10 (192.168.100.0) |
| Windows 10 Machine | GUI | To present work | 4 GB | 2 | 60 GB | NAT | VMnet10 (192.168.100.0) |
| Kali Linux | GUI | To present work | 4 GB | 2 | 20 GB | NAT | VMnet10 (192.168.100.0) |
| Openstack (TestVM) | GUI | For testing | 16 GB | 2 | 60 GB | NAT | N/A |
| Linux 18.04 (Openstack) TEST\_VM | GUI | For testing | 16 GB | 2 | 100 GB | NAT | N/A |
| OpenStack (Server) | Command Prompt | For testing | 16 GB | 2 | 50 GB | NAT | N/A |

### Figure 5: Type of Virtual Machines used and their respective specifications.

Every VM was tested while having different types of specifications. Every test took place to try out whether the recommended specifications can achieve a better performance, however it is clear that, with the recommended specifications, the VMs operated smoothly.

## 3.6 Specifications used for the Instances

In addition to the subject of specifications of the devices, the right specifications were needed to be set on the instance to be tested. Although there were many limitations, the situation was manageable. An instance can be implementable in two ways. Firstly, by GUI, where every specification is manually chosen, for instance, its name, the type of specification, and the type of OS. The user can build an instance using one single code “microstack. launch cirros –name test”. When making use of this command, automatically, the instance will be created while having the minimum amount of specifications.



### Figure 6: The implemented instance and the respective specifications.

Specifications of the Instance created on OpenStack:

* Name: instance 1
* Type of OS: cirros
* IP address assigned: 10.20.20.73
* Type of instance: m1.tiny (Size of HDD: 1MB)
* Key Pair (Group): microstack
* Status: Active
* Availability zone: nova

## 3.7 Utilization of Virtual Machines

In the tests conducted during the research, the plan was to use multiple instances and to attack their communication packets while trying to gain access to the data sent by every instance. However, without the license, this is not possible because only two instances can be used. Virtual Machines that were in use in this project where the Linux 18.04, which was utilized for the OpenStack server to be online. The Windows 10 Machine, which was utilized for when the instance sends a folder to the Windows 10 Machine, and the last VM was the Kali Linux, was initiated to launch attacks against the Windows 10 VM and the Linux 18.04 VM.

## 3.8 Code used and Explanation of the Code

As mentioned before, there are two ways for the implementation to be done, either using GUI or by CLI. The GUI technique was used in the testing phase, but the implementation was so much easier while using the code, in fact, CLI was utilized in the implementation phase.

|  |  |
| --- | --- |
| **Code Used** | **Explanation** |
| sudo snap install microstack --classic --edge | The first step was to install OpenStack on Linux Virtual Machine to be able to access the server. |
| sudo microstack.init --auto | The second command was in use to initialize the OpenStack server before making use of the server. This command was very crucial because the databases and the networks should be configured instantly. |
| microstack.openstack catalog list | The third command was used to be able to view the list of all the tools that OpenStack provides to the user. |
| microstack.launch cirros --name instance1 | The fourth command was to create an instance that should be used by the user. |
| ssh -I $HOME/.ssh/id\_microstack cirros @10.20.20.179 | The command used to access the instance that was created before. In the session, the user has complete control of the instance. The user must have full rights to be able to read, write, and execute his data. |
| Uptime | Command used to verify that the user is up and active. To verify that the user is active by using a command is one way to verify that the instance is created. Another way is to enter into the server, click instances, and view the instance created. |

### Figure 7: Code used for the implementation of the OpenStack.

Figure 6 should be the result after creating an instance by CLI, an important note is that the specifications were set automatically.

## 3.9 Type of Tests Done

Since one of the limitations of this research was that only two instances could only be in use, the tests were a bit difficult because three instances were in need to test their communication and effectiveness while data is being in transit. The first basic test involved sending an empty folder to the host’s desktop, while Wireshark was in use to gather the data in transit. The connection between the instance and the host was visible, and therefore packets of this communication were caught. The second basic test was that of the instance, where the same packet was in transit, however this time, it was on another virtual machine on the same network, while Wireshark was active. This time, the communication between the instance and the VM was not visible. For the third and fourth tests, Kali was used. In the third test, Ettercap was utilized for a Man-in-the-Middle test to take place while Wireshark was active. In the fourth test, Metasploit was deployed to try to gather data by using an exploit. In both tests, Wireshark did not gather any of the data, no IP addresses of either the instance and the virtual machines were visible in the trace of Wireshark.

## 3.10 Types of Attacks

In the testing phase of this project, there were many ways to test whether the Keystone is functioning well or not. The type of attacks that took place during the methodology stage was the most basic. The first two attacks were made, the first attack was made by sending an empty file to the computer, which is the host, and Wireshark would be active to test whether it is possible to view the packets that were sent to the destination. The second attack was performed precisely like the first one, but the destination was a VM on the same network. The third can be considered as a MITM attack, and the other was to try and to compromise a Linux Virtual Machine by using a remote code vulnerability.

## 3.11 The Process of the Attacks and the Environment

The process of the attacks was not so difficult when testing to see whether Keystone was either working or not. The first process involved installing OpenStack on the Linux virtual machine, which was then tried to test the attacks with two instances, however, this was not possible since the instances were used. Then, in order to test whether Keystone was working or not, an empty folder was sent to the researcher’s personal computer while Wireshark was active. The other attack was rather similar to the first one. However, in the first attack, Linux VM OpenStack was deployed, and the Windows VM were put into the created network, after an empty folder was sent. During all of this, Wireshark was active. For the other attacks, the Kali Linux was installed. In the first attack, Ettercap was used to launch a MitM between the Openstack VM and the Windows VM, while in the fourth and final attack, Metasploit was deployed to launch a vulnerability scan against the Linux Virtual Machine.