

CLOUD COMPUTING

(PCA20D08J)

Name : Jatin Sharma

Registration No : RA2132241030008

Subject Code : PCA20D08J

Subject Title : Cloud Computing

Semester : 3rd Sem

Academic Year 2022 – 2023



DEPARTMENT OF COMPUTER APPLICATION
SRM INSTITUTE OF SCIENCE AND TECHNOLOGY
DELHI NCR CAMPUS
Modinagar, Ghaziabad – 201204

SRM INSITITUE OF SCIENCE AND TECHNOLOGY
DELHI NCR CAMPUS, MODINAGAR DEPARTMENT OF
COMPUTER APPLICATION

Registration No: _____

BONAFIDE CERTIFICATE

Certified to be the bonafide record of the work done by **JATIN SHARMA** of MCA, 2nd year, 3rd Semester for the award of Master Degree course in **COMPUTER APPLICATION** in **CLOUD COMPUTING (PCA20D08J)** during the academic year 2022-23.

LAB IN-CHARGE

HEAD OF DEPARTMANT

Submitted for the university examination held on _____

INTERNAL EXAMINER 1

INTERNAL EXAMINER 2

INDEX

S.No.	Title of the Experiment	Date	Signature/ Remarks
1.	Implement RPC and Bankers algorithm (using Python /Java)	13/7/22	
2.	Create and distribute a Torrent file to share a file in LAN Environment.	20/7/22	
3.	Use Google collaboration tools: Create Google Docs, Sheets and Slides and share it with other users.	04/8/22	
4.	Use Google collaboration tools: Create Google Docs, Sheets and Slides and share it with other users.	12/8/22	
5.	Quizzes on different service models and deployment models. Report submission - Comparison of various services provided by different Cloud Service Providers (configuration of VM, cost, Network bandwidth etc.).	17/8/22	
6.	Create a simple web service using Python Flask/ Java/ any language: Client - Server Model should be implemented using socket/http	24/8/22	
7.	Install Oracle Virtual Box / VMware Workstation	01/9/22	

8.	Review web services implementation - Proper Connection should be established between the client and server to make use of the service offered by the Server. Review the working of application in virtual environment.	07/9/22	
9.	Use Security tools like ACUNETIX, ETTERCAP to scan Web Applications on the Cloud.	14/9/22	
10.	Cloud Networks for finding vulnerabilities, verifying leakage of information to an unauthorized third party	21/9/22	
11.	Report Submission: Generate a detailed report describing vulnerabilities along with the suitable action that can be taken to remedy the loopholes	28/9/22	
12.	Install and configure OpenStack all-in-one using DevStack/Packstack	6/10/22	
13.	Launch VMs in OpenStack through Dashboard	12/10/22	
14.	OpenStack Dashboard should be accessed through Web Browser. Verify the working of the instance by logging or pinging the instance.	18/10/22	

Experiment 1

1. Implement RPC and Bankers algorithm (using Python /Java)

Remote Procedure Call (RPC)

RPC is a powerful technique for constructing distributed, client-server-based applications. It is based on extending the conventional local procedure calling so that the called procedure need not exist in the same address space as the calling procedure. The two processes may be on the same system, or they may be on different systems with a network connecting them.

Implementation in Python

Server Side : Defining a simple function that calculates the factorial of a number.

```
from xmlrpc.server import SimpleXMLRPCServer

def factorial(n):
    fact = 1
    for i in range(1,n+1):
        fact=fact*i
    return fact

server = SimpleXMLRPCServer(("localhost", 8000), logRequests=True)
server.register_function(factorial, "factorial_rpc")

try:
    print("Starting and listening on port 8000...")
    print("Press Ctrl + C to exit.")
    server.serve_forever()

except:
    print("Exit.")
```

Client Side

```
import xmlrpc.client
proxy = xmlrpc.client.ServerProxy("http://localhost:8000/")
print("factorial of 3 is : %s" % str(proxy.factorial_rpc(3)))
print("factorial of 5 is : %s" % str(proxy.factorial_rpc(5)))
```

Output

Server Side

```
Starting and listening on port 8000...  
Press Ctrl + C to exit.  
█
```

Client Side

```
C:\Users\wwwri\Downloads\python_rpc>python clie.py  
factorial of 3 is : 6  
factorial of 5 is : 120  
  
C:\Users\wwwri\Downloads\python_rpc>█
```

Banker's Algorithm

The banker's algorithm is a resource allocation and deadlock avoidance algorithm that tests for safety by simulating the allocation for predetermined maximum possible amounts of all resources, then makes an "s-state" check to test for possible activities, before deciding whether allocation should be allowed to continue.

Driver code:

```
if __name__ == "__main__":
```

```
    # P0, P1, P2, P3, P4 are the Process names here
```

```
    n = 5 # Number of processes
```

```
    m = 3 # Number of resources
```

```
    # Allocation Matrix
```

```
    alloc = [[0, 1, 0 ],[ 2, 0, 0 ],
```

```
             [3, 0, 2 ],[2, 1, 1 ],[ 0, 0, 2]]
```

```
    # MAX Matrix
```

```
    max = [[7, 5, 3 ],[3, 2, 2 ],
```

```
           [ 9, 0, 2 ],[2, 2, 2],[4, 3, 3]]
```

```
    avail = [3, 3, 2] # Available Resources
```

```
    f = [0]*n
```

```
    ans = [0]*n
```

```
    ind = 0
```

```
    for k in range(n):
```

```
        f[k] = 0
```

```
    need = [[ 0 for i in range(m)]for i in range(n)]
```

```
    for i in range(n):
```

```
        for j in range(m):
```

```
            need[i][j] = max[i][j] - alloc[i][j]
```

```
    y = 0
```



```

for k in range(5):
    for i in range(n):
        if (f[i] == 0):
            flag = 0
            for j in range(m):
                if (need[i][j] > avail[j]):
                    flag = 1
                    break

            if (flag == 0):
                ans[ind] = i
                ind += 1
                for y in range(m):
                    avail[y] += alloc[i][y]
                f[i] = 1

print("Following is the SAFE Sequence")

for i in range(n - 1):
    print(" P", ans[i], " ->", sep="", end="")
print(" P", ans[n - 1], sep="")

```

Output

```
1 # Driver code:
2 ▼ if __name__=="__main__":
3
4     # P0, P1, P2, P3, P4 are the Process names here
5     n = 5 # Number of processes
6     m = 3 # Number of resources
7
8     # Allocation Matrix
9 ▼ alloc = [[0, 1, 0 ],[ 2, 0, 0 ],
10           [3, 0, 2 ],[2, 1, 1 ],[ 0, 0, 2]]
11
12     # MAX Matrix
12 ▼ max = [[7, 5, 3 ],[3, 2, 2 ],
13          [ 9, 0, 2 ],[2, 2, 2],[4, 3, 3]]
14     avail = [3, 3, 2] # Available Resources
15     f = [0]*n
16     ans = [0]*n
17     ind = 0
18 ▼ for k in range(n):
19     f[k] = 0
20     need = [[ 0 for i in range(m)]for i in range(n)]
21 ▼ for i in range(n):
22 ▼     for j in range(m):
23         need[i][j] = max[i][j] - alloc[i][j]
24     y = 0
25 ▼ for k in range(5):
26 ▼     for i in range(n):
27 ▼         if (f[i] == 0):
28             flag = 0
29 ▼             for j in range(m):
30 ▼                 if (need[i][j] > avail[j]):
31                     flag = 1
32                     break
33
34 ▼             if (flag == 0):
35                 ans[ind] = i
```

Following is the SAFE Sequence
P1 -> P3 -> P4 -> P0 -> P2

Experiment 2

2. Create and distribute a Torrent file to share a file in LAN Environment.

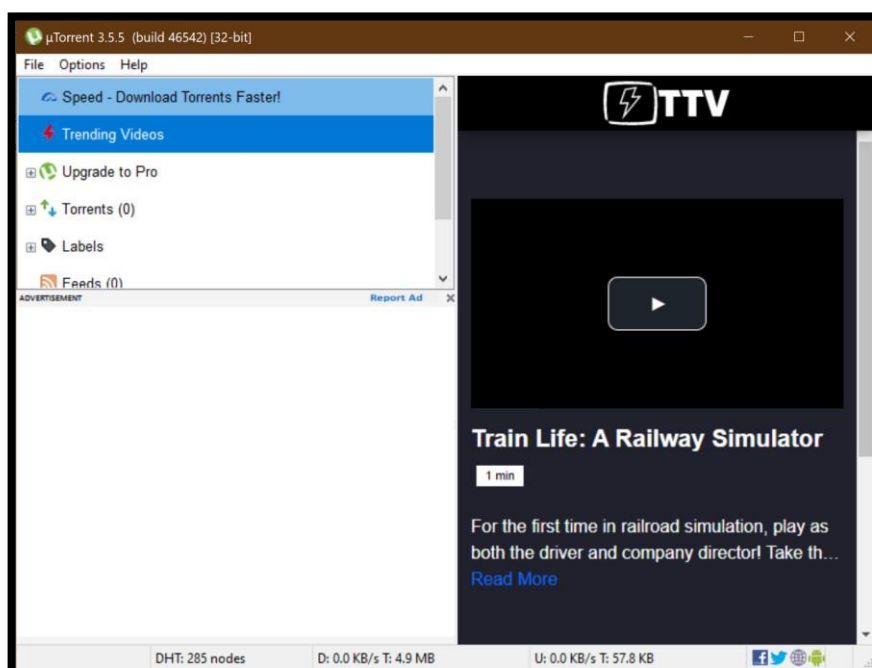
A Torrent File contains metadata of file system and directories that you want to distribute using the BitTorrent File Distribution System. Along with the meta data it also contains list of trackers. The Network location of trackers are noted in the tracker list.

The main principle behind the working of these torrents is the use of a peer-to-peer protocol, which implies that a group of computers is used for downloading and uploading the same torrent. Torrents are used to transfer data between each other without the need for a central server. In other words, they use a decentralized server in which every torrent participant is actively involved in downloading and uploading files.

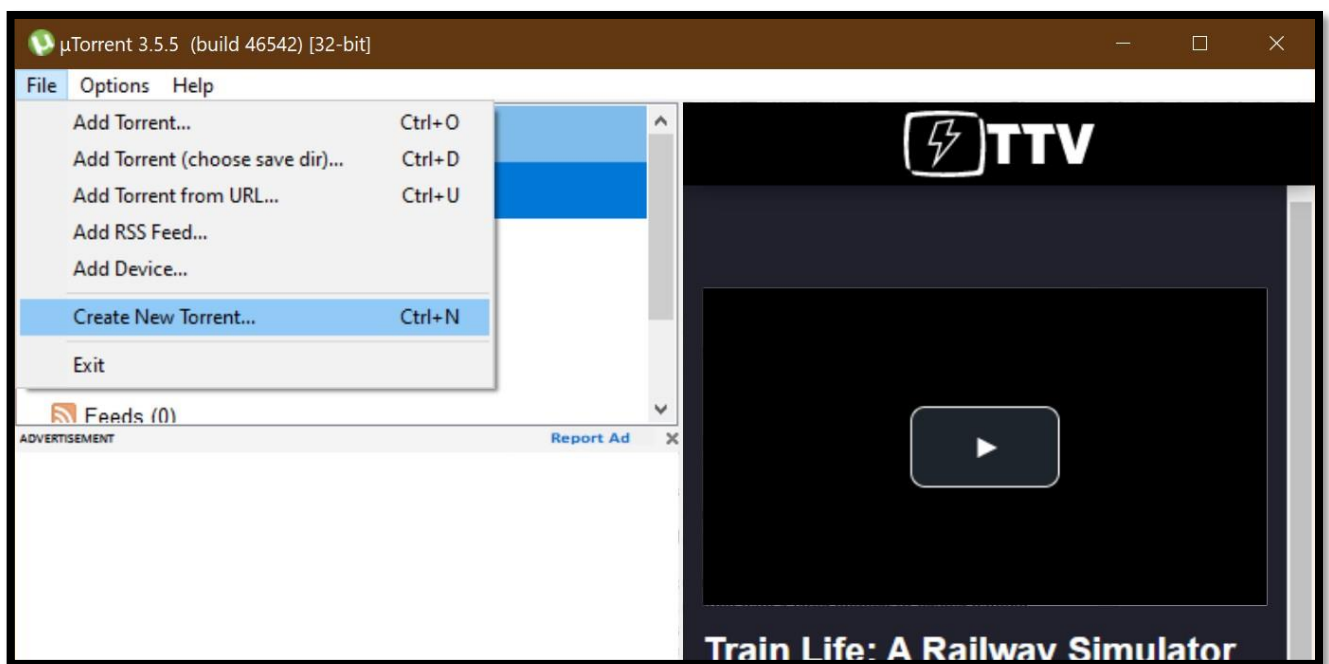
Although torrenting may mainly be associated with piracy in the present day. It surely provides very effective ways of communicating with a large number of people without current bandwidth requirements. For piracy, we cannot blame torrents as the technology can be used both constructively and destructively. In addition to that, the advantages of torrents surely outweigh its drawbacks. However, the use of unsanctioned copyright materials must be discouraged completely.

Following are the steps to create a torrent file and share it on LAN

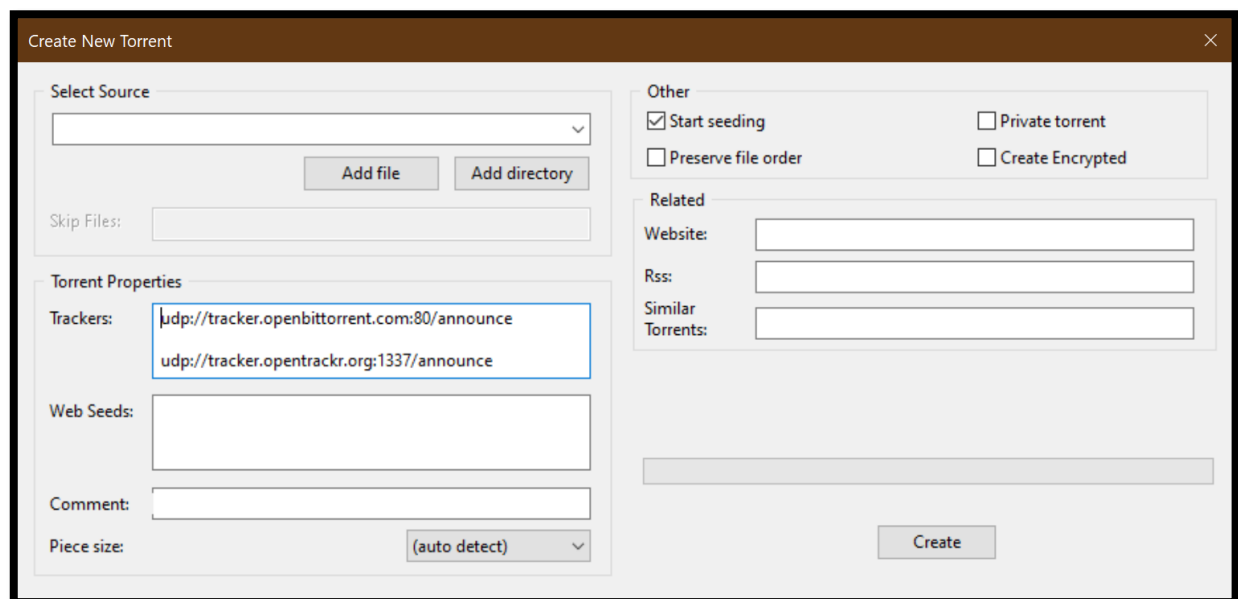
1) Download a torrent client (Bit torrent or uTorrent)



2) Open uTorrent → File → Create new Torrent (or CTRL + N)



3) Select the files and or directories

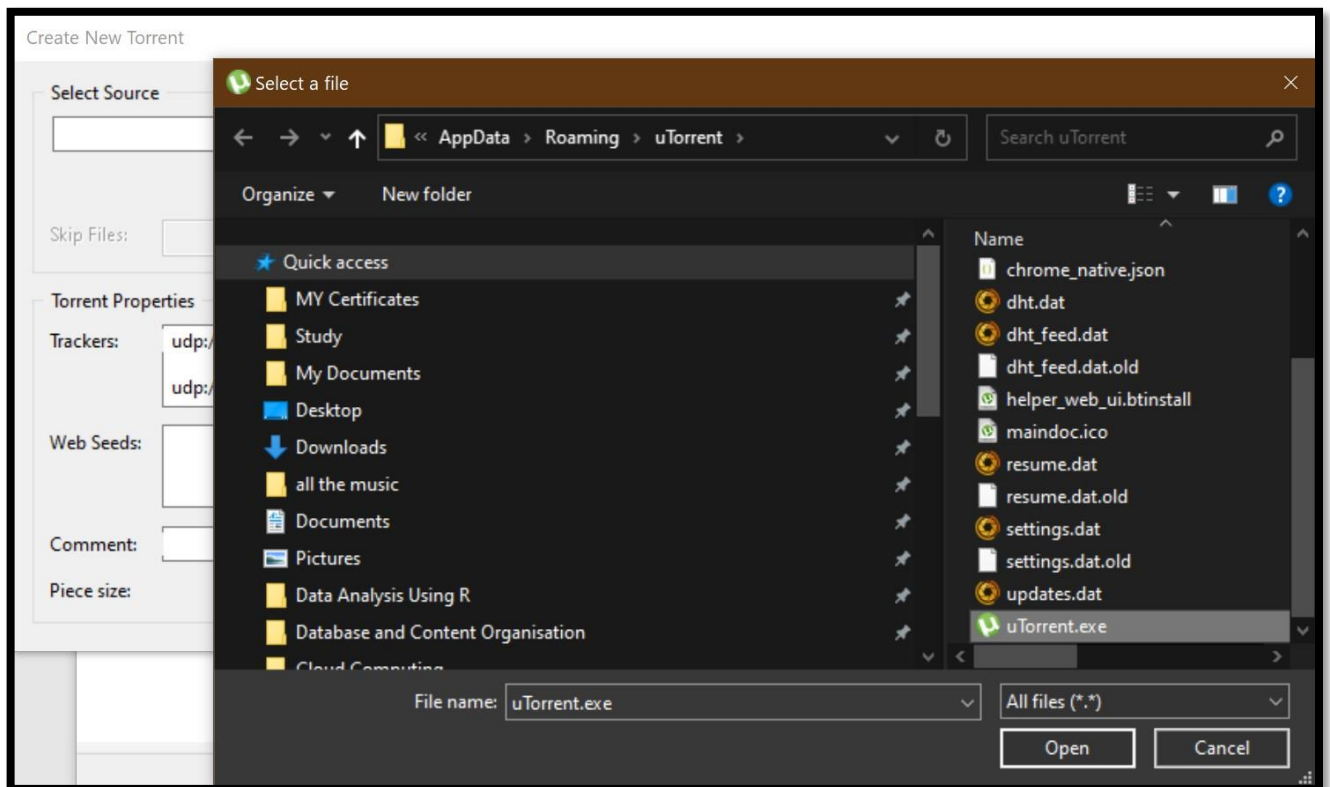


4) Trackers : A BitTorrent tracker is a special type of server that assists in the communication between peers using the BitTorrent protocol. torrent trackers are servers that keep track of the peers who are available at the moment to offer you the requested files.

Include the following trackers for faster communication

- 1) `udp://tracker.openbittorrent.com:80/announce`
- 2) `udp://tracker.publicbt.com:80/announce`
- 3) `udp://tracker.istole.it:80/announce`
- 4) Specifying wheather our torrent file is a private torrent or not (if we are using a private tracker, we'll need to. If you aren't, you can most likely leave this part alone.

6) Adding the source file and clicking on create








Experiment 3

3. Use Google collaboration tools: Create Google Docs, Sheets and Slides and share it with other users.

Creating new files

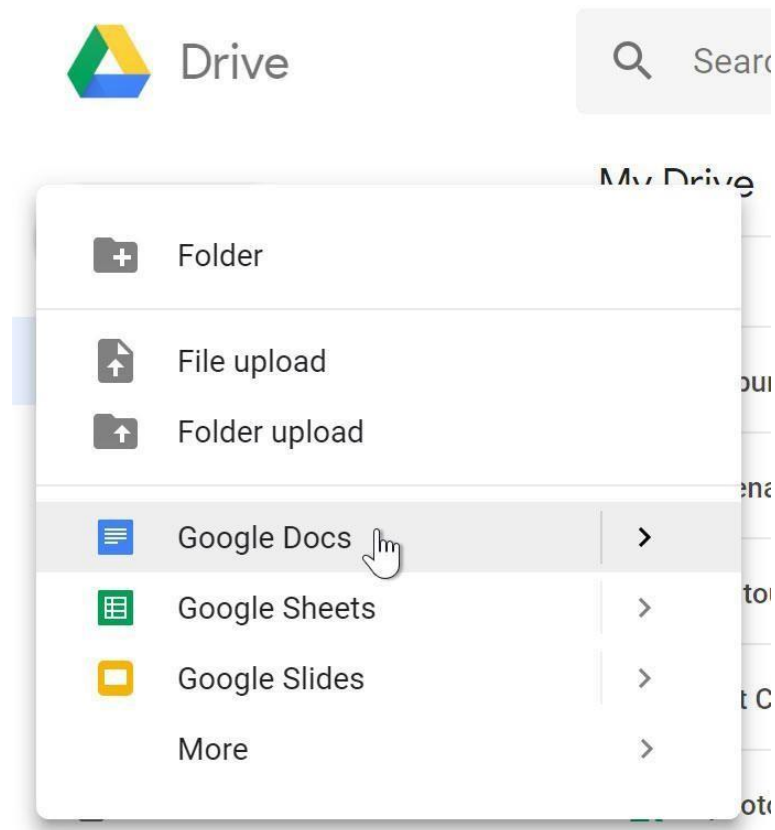
Google Drive gives you access to a suite of tools that allows you to create and edit a variety of files, including documents, spreadsheets, and presentations. There are five types of files you can create on Google Drive:

-  **Documents:** For composing letters, flyers, essays, and other text-based files (similar to Microsoft Word documents)
-  **Spreadsheets:** For storing and organizing information (similar to Microsoft Excel workbooks)
-  **Presentations:** For creating slideshows (similar to Microsoft PowerPoint presentations)
-  **Forms:** For collecting and organizing data
-  **Drawings:** For creating simple vector graphics or diagrams

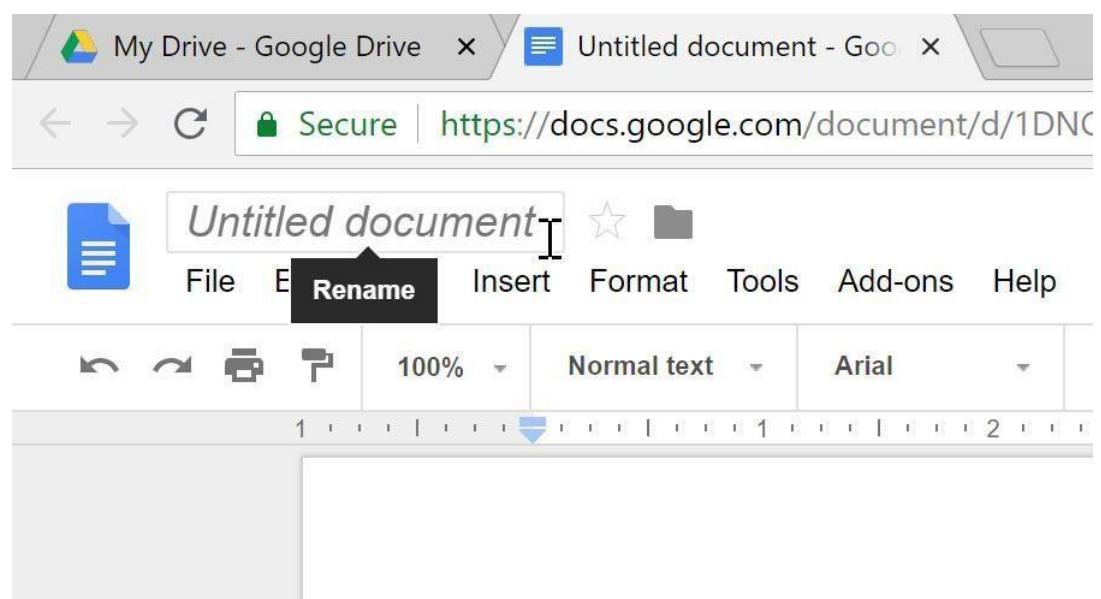
The process for creating new files is the same for all file types. Watch the video below to learn more.

To create a new file:

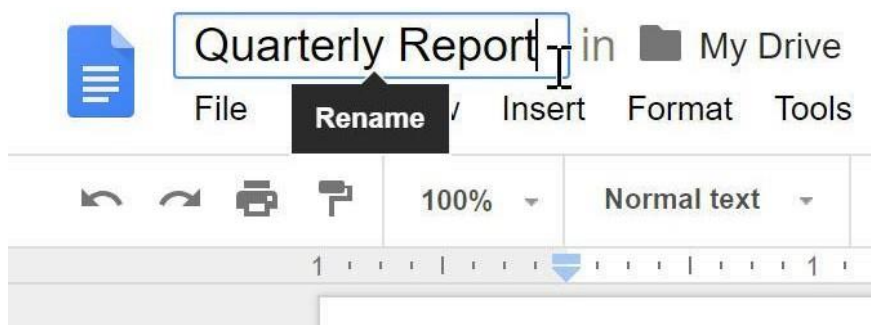
1. From Google Drive, locate and select the New button, then choose the type of file you want to create. In our example, we'll select Google Docs to create a new document.



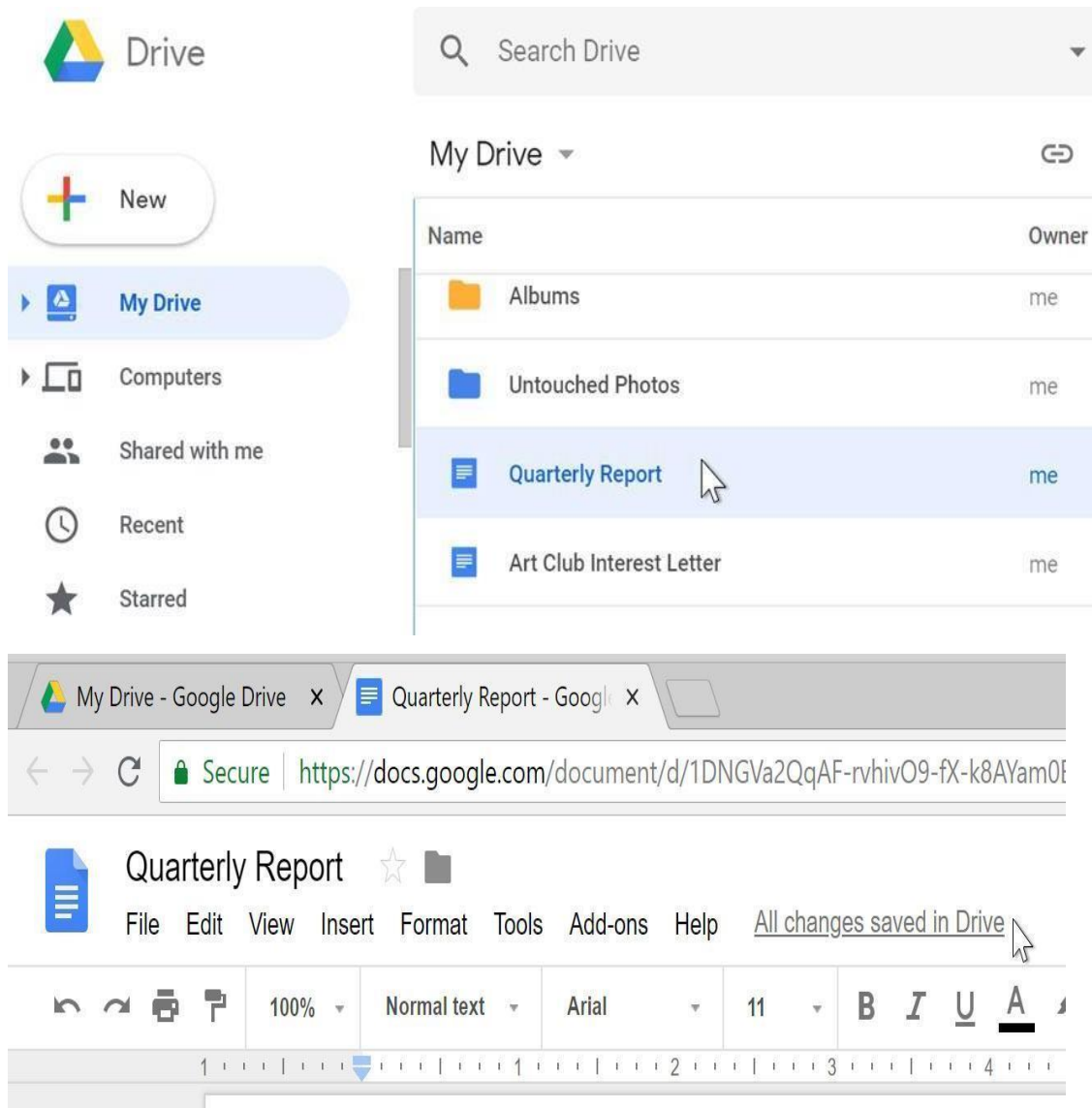
2. The new file will appear in a new tab on your browser. Locate and select 'Untitled document' in the upper-left corner.



3. The Rename dialog box will appear. Type a name for your file, then click OK.



4. Your file will be renamed. You can access the file at any time from your Google Drive, where it will be saved automatically. Simply double-click to open the file again.



We can notice that there is no Save button for your files. This is because Google Drive uses autosave, which automatically and immediately saves your files as you edit them.

Steps to run python scripts in the cloud

1. Launch your cloud computer
2. Below is a quick step-by-step guide to starting a cloud computer also called a cloud instance.

- 1) Create an account at AWS
- 2) Log in to AWS
- 3) Choose EC2 from the Services menu
- 4) (Optional) In the top-right corner select the region that is closest to you.
- 5) Click Launch Instance
- 6) Choose Amazon Linux
- 7) Click Review and Launch and on the next page click Launch
- 8) You can select Create a new key pair in the pop-up window and give it any name you want.
- 9) Click Launch Instance
- 10) Save the .pem file somewhere you'll remember. This is your AWS key and we will need it later
- 11) Click View Instances

After a short while you should see that a new instance is up and running. You can see this on the AWS browser page under Instances.

2. Connect to the cloud computer

- 1) Open the Terminal on Mac or Linux
- 2) You first need to change the access permissions of your AWS key file:

```
$ chmod 400 my-key-pair.pem
```

- 3) We now ssh into the cloud computer. This will create a secure connection between our laptop and the cloud

```
$ ssh -i /path/my-key-pair.pem ec2-user@public_dns_name
```

If all went well, we are now remotely logged in to your cloud computer.

3. Setup Python on the cloud

Your instance should come with Python 2.7 pre-installed. I will install Python 3 below. To see the Python-related packages available for install on your cloud computer type:

```
$ yum list | grep python
```

yum is a package manager for Linux and grep is a search command applied to the output of yum (this is what the | sign achieves). grep looks for anything with “python” in the output of yum.

To install Python 3:

```
$ sudo yum install python35
```

The sudo prefix ensures that the above command is run with administrator privileges, which is necessary to install stuff.

To install Python packages with pip (Python package manager), I also needed to run:

```
$ sudo yum install python35-devel$ sudo yum install gcc
```

4. Set up a Python virtual environment

It is best practice to set up a Python virtual environment and install any new packages there instead of installing packages in the Python root.

To create a virtual environment for Python 3

```
$ virtualenv -p python3 venv
```

We can choose a name other than “venv” for your virtual environment.

To activate the virtual environment

```
$ source venv/bin/activate
```

We can later deactivate the virtual environment by typing

```
$ deactivate
```

As an example, we can now install the Python package scrapy, a webscraping framework, with pip install.

```
$ pip install scrapy
```

5. Run a Python script

Create a blank Python script:

```
$ vi test.py
```

Press i to enter insert mode and type the following code.

```
import time
for i in range(1000, -1, -1):
    print("Time remaining: {}".format(i))
    time.sleep(1)
```

To exit the vi editor, first press Esc to exit insert mode and then press :x. This will also save the file.

Run the script:

```
$ python test.py
```

6. Upload Python code from your own machine

If you would like to upload existing code from your machine, use the secure copy command below.

```
$ scp -ri ~/documents/path/yourkey.pem ~/documents/path/example/ ec2-user@ec2-23-21-11-38.eu-central-1.compute.amazonaws.com:/home/ec2-user
```

Replace the above two files paths with the path to your AWS key and the folder containing the Python code. You can find out the last parameter that is needed above by right-clicking on the instance you have just started and selecting Connect.

The option -r is for recursive copy as we are copying a complete folder and option -i stands for identity_file to provide the path to your AWS key.

If you are only uploading a single file, the -r option is not necessary.

7. Keep your Python script alive and running

To keep Python running even after you disconnect from the cloud instance we install tmux.

```
$ sudo yum install tmux
```

Start tmux

```
$ tmux
```

Run the Python script inside tmux

```
$ python test.py
```

Press CTRL + B and type :detach. This will exit tmux, but the Python script will still be running in the background. Disconnect from the cloud by typing.

```
$ ~.
```

Reconnect to the cloud.

```
$ ssh -i "~/documents/path/mykey.pem" ec2-user@ec2-23-21-59-38.eu-central-1.compute.amazonaws.com:/home/ec2-user
```

Remember to change the above parameters to the ones you are using.

Enter tmux and see the Python code at work:

```
$ tmux attach
```

At this point, you should see our Python script running.

To stop the script, press CTRL + B and type:

```
:kill-session
```

8. Stop the cloud computer

On the AWS browser page under Instances, right-click the instance you want to terminate, select Instance State and select Terminate. This is also when billing stops.

Experiment 4

4. Explore public cloud services like Amazon, Google, Salesforce, Digital Ocean etc

With the public cloud raging the market, virtual hardware is shared by many companies. The multi-employer environment makes it easy to differentiate infrastructure costs for multiple users. Due to cost benefits and payment model, the public cloud is suitable for small and medium-sized businesses. In general, sensitive, community-oriented web applications that can receive unexpected traffic can increase the cloud mass very well.





While defining the cloud strategy for their business, enterprises can choose between a public cloud, a private cloud or a hybrid cloud for efficient scaling. The choice depends on several factors such as the type of business application, the costs involved, the technical specifications, and other business requirements. In this blog, we will take a closer look at the social cloud and its benefits for businesses.

The public cloud is the most popular computer model. In this case, cloud service providers use the Internet and make services such as infrastructure, storage, servers etc. available for businesses. Third-party providers own and use shared physical hardware and provide that to companies according to their needs. Amazon Elastic Compute Cloud (EC2), IBM Blue Cloud, Google App Engine, Sun Cloud, Microsoft Azure are some of the most popular cloud providers in the community.

Following are the top 3 cloud vendors that are high in demand with the IT sector.

1. Amazon Web Services (AWS)
2. Microsoft Azure
3. Google Cloud Platform
4. Salesforce

Other cloud service providers preceding the top 3 are DigitalOcean, IBM Cloud, Dell Technologies/ VMware, CISCO Systems, Salesforce, Oracle etc.

			
Free trial	12-month free trial	12-month free trial	12-month free trial
SLA	Amazon EC2- 99.5% annual uptime Amazon S3 - A monthly uptime of at least 99.9% for a billing cycle	99.9% uptime	99.95% monthly uptime
Max processors in Virtual Machines	128 nos	128 nos	96 nos
Maximum Memory in Virtual Machines	3904GiB	3800GiB	1433GiB
Supported OS	Core OS, Windows, SLES, Cloud Linux, Ubuntu, etc	SLES, Windows, CentOS, Oracle Linux, etc	Windows, SLES, CoreOS, FreeBSD, etc
Availability Zones	66 availability zones with 12 on the anvil	54 regions worldwide and is available in 140 countries.	in 20 regions around the world with 3 more on the way
Pros	Reliability, Top Professional Support, Quality	Flexible Infrastructure, Configurations, Ideal for large projects	Reliability, Costeffective, Easily affordable
Cons	Expensive despite low prices	Customer Support is longer process and tedious	Limited features, Less services

Amazon Web Services:

Popularly known as AWS, Amazon Web Services is the leading cloud service provider with a 33% market share.

AWS assists firms by providing quality services and supporting their businesses. One can run their business on a mobile phone or desktop. In addition, the user should focus only on creating the code and ignoring other features.

Microsoft Azure:

Back in 2017, Gartner called Azure a top leader in the Cloud Infrastructure as a service space.

Globally, 90% of Fortune 500 companies use Microsoft Azure to run their business.

Using the deeply integrated Azure cloud services, businesses can quickly build, deploy and manage simple and complex systems with ease.

Azure supports multiple programming languages, frameworks, operating systems, information, and devices, allowing businesses to use the tools and technologies they rely on.

Google Cloud Platform:

With precise looks, low cost, attractive features and flexible computer options, GCP is an attractive option for both AWS and Azure. It uses complete encryption for all data and communication channels including traffic between data centres.

Some of the areas where Google Cloud competes fiercely with AWS include model setting and payment, privacy and traffic security, cost-effectiveness, and machine learning.

While all three cloud providers offer up to 75% discounts on a one- to three-year commitment, Google Cloud additionally offers up to 30% continuous user discount on each model that works for more than 25% per month.

Google Cloud offers several in-house APIs related to computer viewing, native language processing and translation. Machine learning engineers can create models based on the Cloud Machine Learning Engine open-source TensorFlow open-source learning.

Salesforce

Salesforce is a cloud-based software company that provides businesses with tools that help them find more prospects, close more deals, and provide a higher level of service to their customers.

Salesforce, Inc. is a famous American cloud-based software company that provides CRM services. Salesforce is a popular CRM tool for support, sales, and marketing teams worldwide. Salesforce services allow businesses to use cloud technology to better connect with partners, customers, and potential customers. Using the Salesforce CRM, companies can track customer activity, market to customers, and many more services.

Conclusion:

Focusing your IT team on projects that can bring in more revenue rather than working furiously to manage on-premises systems is a predominant priority to most IT companies. With finite resources, companies are looking to adopt cloud models that can cater to their multiple IT requirements. Cloud-native technologies empower organizations to build and run scalable SRP-based micro servers-based applications in modern, dynamic environments.

Experiment 5

5. Quizzes on different service models and deployment models. Report submission - Comparison of various services provided by different Cloud Service Providers (configuration of VM, cost, Network bandwidth etc.).

System Property	<u>Amazon</u> Elastic Compute Cloud (EC2)	<u>Google</u> App Engine	<u>Microsoft</u> Live Mesh	<u>Sun</u> Network.com (Sun Grid)	<u>GRIDS Lab</u> Aneka
Focus	Infrastructure	Platform	Infrastructure	Infrastructure	Software Platform for enterprise Clouds
Service Type	Compute, Storage (Amazon S3)	Web application	Storage	Compute	Compute
Virtualisation	OS Level running on a Xen hypervisor	Application container	OS level	Job management system (Sun Grid Engine)	Resource Manager and Scheduler
Dynamic Negotiation of QoS Parameters	None	None	None	None	SLA-based Resource Reservation on Aneka side.
User Access Interface	Amazon EC2 Command-line Tools	Web-based Administration Console	Web-based Live Desktop and any devices with Live Mesh installed	Job submission scripts, Sun Grid Web portal	Workbench, Web-based portal
Web APIs	Yes	Yes	Unknown	Yes	Yes
Value-added Service Providers	Yes	No	No	Yes	No
Programming Framework	Customizable Linux-based Amazon Machine Image (AMI)	Python	Not applicable	Solaris OS, Java, C, C++, FORTRAN	APIs supporting different programming models in C# and other .Net supported languages

Experiment 6

6. Create a simple web service using Python Flask/ Java/ any language: Client - Server Model should be implemented using socket/http

Following are steps for the creation of a simple Python Flask App that provides a RESTful web service. This service will provide an endpoint to.

- 1) Ingest a JSON formatted payload (webhook) from Threat Stack
- 2) Parse the payload for Threat Stack Alert IDs
- 3) Retrieve detailed alert data from Threat Stack
- 4) Archive the webhook and alert data to AWS S3

Following key points for the Python flask creation

- 1) We don't need to worry about frontend functionality about HTML or CSS.
- 2) Following Flask is using Packages and Blueprints module pattern, we can also use single module pattern.

Project structure

The structure of the project that I'm going to build, which comes from Explore Flask, is shown below:

Threatstack-to-s3

```
├── app
|   ├── __init__.py
|   ├── models
|   |   ├── __init__.py
|   |   ├── s3.py
|   |   └── threatstack.py
|   └── views
```

```
|   |—__init__.py
|   |—s3.py
|—gunicorn.conf.py
|—requirements.osx.txt
|—requirements.txt
|—threatstack-to-s3.py
```

Top-level files

Following are the top-level files that are used to build the service

Gunicorn.conf.py: This is a configuration file for the Gunicorn WSGI HTTP server that will serve up this app. While the application can run and accept connections on its own, Gunicorn is more efficient at handling multiple connections and allowing the app to scale with load.

Requirements.txt/requirements.osx.txt: The app's dependencies are listed in this file. It is used by the pip utility to install the needed Python packages. For information on installing dependencies, see the Setup section of this README.md.

Threatstack-to-s3.py: This is the application launcher. It can be run directly using "python" if you are doing local debugging, or it can be passed as an argument to "gunicorn" as the application entry point. For information on how to launch a service, see README.md.

App package (app/ directory)

The app package is my application package. The logic for the application is underneath this directory. As I mentioned earlier, I have chosen to break the app into a collection of smaller modules rather than use a single, monolithic module file

The following four usable modules defined in this package are:

- 1) app
- 2) app.views.s3
- 3) app.models.threatstack
- 4) app.models.s3

Note: app.views and app.models do not provide anything and their __init__.py files are empty.

App module

The app module has the job of creating the Flask application. It exports a single function, create_app(), that will create a Flask application object and configure it. Currently it initializes application blueprints that correspond to my application views. Eventually, create_app() will do other things such as initialize logging, but I'm skipping that now for clarity and simplicity.

App/__init__.py

```
from flask import Flask
```

```
def _initialize_blueprints(application):
```

```
    """
```

```
    Register Flask blueprints
```

```
    """
```

```
    from app.views.s3 import s3
```

```
    application.register_blueprint(s3, url_prefix='/api/v1/s3')
```

```
def create_app():
```

```
    """
```

```
    Create an app by initializing components.
```

```
    """
```

```
    application = Flask(__name__)
```

```
    _initialize_blueprints(application)
```

```
    # Do it!
```

```
    return application
```

Copy

This module is used by threatstack-to-s3.py to start the application. It imports create_app() and then uses it to create a Flask application instance.

```
Threatstack-to-s3.py
#!/usr/bin/env python
from app import create_app

# Gunicorn entry point.
application = create_app()

if __name__ == '__main__':
    # Entry point when run via Python interpreter.
    print("== Running in debug mode ==")
    application.run(host='localhost', port=8080, debug=True)
Copy
```

App.views.s3 module

The threatstack-to-s3 service takes Threat Stack webhook HTTP requests in and stores a copy of the alert data in S3. This is where I store the set of API endpoints that allow someone to do this. If you look back at app/__init__.py, you will see that I have rooted the set of endpoints at /api/v1/s3.

From app/__init__.py:

```
from views.s3 import s3

app.register_blueprint(s3, url_prefix='/api/v1/s3')
```

Copy

In app.views.s3, Providing a single endpoint for now, /alert, which represents the object I'm manipulating, and that responds only to the HTTP POST request method.

Remember: When building APIs, URL paths should represent nouns and HTTP request methods should represent verbs.

App/views/s3.py

'''

API to archive alerts from Threat Stack to S3

'''

```
from flask import Blueprint, jsonify, request
import app.models.s3 as s3_model
import app.models.threatstack as threatstack_model
```

```
s3 = Blueprint('s3', __name__)
```

```
@s3.route('/alert', methods=['POST'])
```

```
def put_alert():
```

```
    '''
```

```
    Archive Threat Stack alerts to S3.
```

```
    '''
```

```
    webhook_data = request.get_json()
```

```
    for alert in webhook_data.get('alerts'):
```

```
        alert_full = threatstack_model.get_alert_by_id(alert.get('id'))
```

```
        s3_model.put_webhook_data(alert)
```

```
        s3_model.put_alert_data(alert_full)
```

```
    status_code = 200
```

```
    success = True
```

```
    response = {'success': success}
```

```
    return jsonify(response), status_code
```

Copy

Just a quick run through of a few spots of note:

```
THREATSTACK_BASE_URL = os.environ.get('THREATSTACK_BASE_URL',
'https://app.threatstack.com/api/v1')
```

```
THREATSTACK_API_KEY = os.environ.get('THREATSTACK_API_KEY')
```

Copy

Not using Stack API in the code, keeping code clean/security living.

Getting the api now as it's a quick and simple solution. At some point centralizing all the configuration in a single file instead of hiding it here, so the code and setup are a little cleaner.

```
def get_alert_by_id(alert_id):
    """
    Retrieve an alert from Threat Stack by alert ID.
    """
    alerts_url = '{ }/alerts/{ }'.format(THREATSTACK_BASE_URL, alert_id)

    resp = requests.get(
        alerts_url,
        headers={'Authorization': THREATSTACK_API_KEY}
    )

    return resp.json()
Copy
```

The `get_alert_by_id()` function takes an alert ID, queries the Threat Stack platform for the alert data, and returns that data. I'm using the Python requests module to make an HTTP GET request to the Threat Stack API endpoint that returns alert info for the given alert.

App.models.s3 Module

The `app.models.s3` module handles connectivity to AWS S3.

```
"""
Manipulate objects in AWS S3.
"""
import boto3
import json
import os
import time

TS_AWS_S3_BUCKET = os.environ.get('TS_AWS_S3_BUCKET')
TS_AWS_S3_PREFIX = os.environ.get('TS_AWS_S3_PREFIX', None)

def put_webhook_data(alert):
    """
    Put alert webhook data in S3 bucket.
    """
```

```

alert_time = time.gmtime(alert.get('created_at')/1000)
alert_time_path = time.strftime('%Y/%m/%d/%H/%M', alert_time)
alert_key = '/'.join([alert_time_path, alert.get('id')])
if TS_AWS_S3_PREFIX:
    alert_key = '/'.join([TS_AWS_S3_PREFIX, alert_key])

s3_client = boto3.client('s3')
s3_client.put_object(
    Body=json.dumps(alert),
    Bucket=TS_AWS_S3_BUCKET,
    Key=alert_key
)

return None

def put_alert_data(alert):
    """
    Put alert data in S3.
    """
    alert_id = alert.get('id')
    alert_key = '/'.join(['alerts',
                          alert_id[0:2],
                          alert_id[2:4],
                          alert_id
                          ])

    if TS_AWS_S3_PREFIX:
        alert_key = '/'.join([TS_AWS_S3_PREFIX, alert_key])

    s3_client = boto3.client('s3')
    s3_client.put_object(
        Body=json.dumps(alert),
        Bucket=TS_AWS_S3_BUCKET,
        Key=alert_key
    )

    return None
Copy

```

The functions `put_webhook_data()` and `put_alert_data()` have a lot of duplicate code. I haven't refactored them because it's easier to see the logic before refactoring. If you

look closely, you'll realize that the only difference between them is how the `alert_key` is defined. I'll focus on `put_webhook_data()`:

```
def put_webhook_data(alert):
    """
    Put alert webhook data in S3 bucket.
    """
    alert_time = time.gmtime(alert.get('created_at')/1000)
    alert_time_path = time.strftime('%Y/%m/%d/%H/%M', alert_time)
    alert_key = '/'.join(['webhooks', alert_time_path, alert.get('id')])
    if TS_AWS_S3_PREFIX:
        alert_key = '/'.join([TS_AWS_S3_PREFIX, alert_key])

    s3_client = boto3.client('s3')
    s3_client.put_object(
        Body=json.dumps(alert),
        Bucket=TS_AWS_S3_BUCKET,
        Key=alert_key
    )
    return None
```

Copy

This function takes in a single argument named `alert`. Looking back at `app/views/s3.py`, `alert` is just the JSON data that was sent to the endpoint. Webhook data is stored in S3 by date and time. The alert `587c0159a907346eccb84004` occurring at `2017-01-17 13:51` is stored in S3 as `webhooks/2017/01/17/13/51/587c0159a907346eccb84004`.

I start by getting the alert time. Threat Stack has sent the alert time in milliseconds since the Unix epoch, and that needs to be converted into seconds, which is how Python handles time. I take that time and parse it into a string that will be the directory path. I then join the top-level directory where I store webhook data, the time-based path, and finally the alert ID to form the path to the webhook data in S3.

Boto 3 is the primary module in Python for working with AWS resources. Initializing a `boto3` client object so I can talk to S3 and put the object there. The `s3_client.put_object()` is fairly straightforward with its `Bucket` and `Key` arguments, which are the name of the S3 bucket and the path to the S3 object I want to store. The `Body` argument is `alert` converted back to a string.

Experiment 7

7. Install Oracle Virtual Box / VMware Workstation

Below are the very easy steps with screenshots containing the installation procedure of Virtual Box

Step 1)

The Oracle VM VirtualBox installation can be started in either of the following ways:

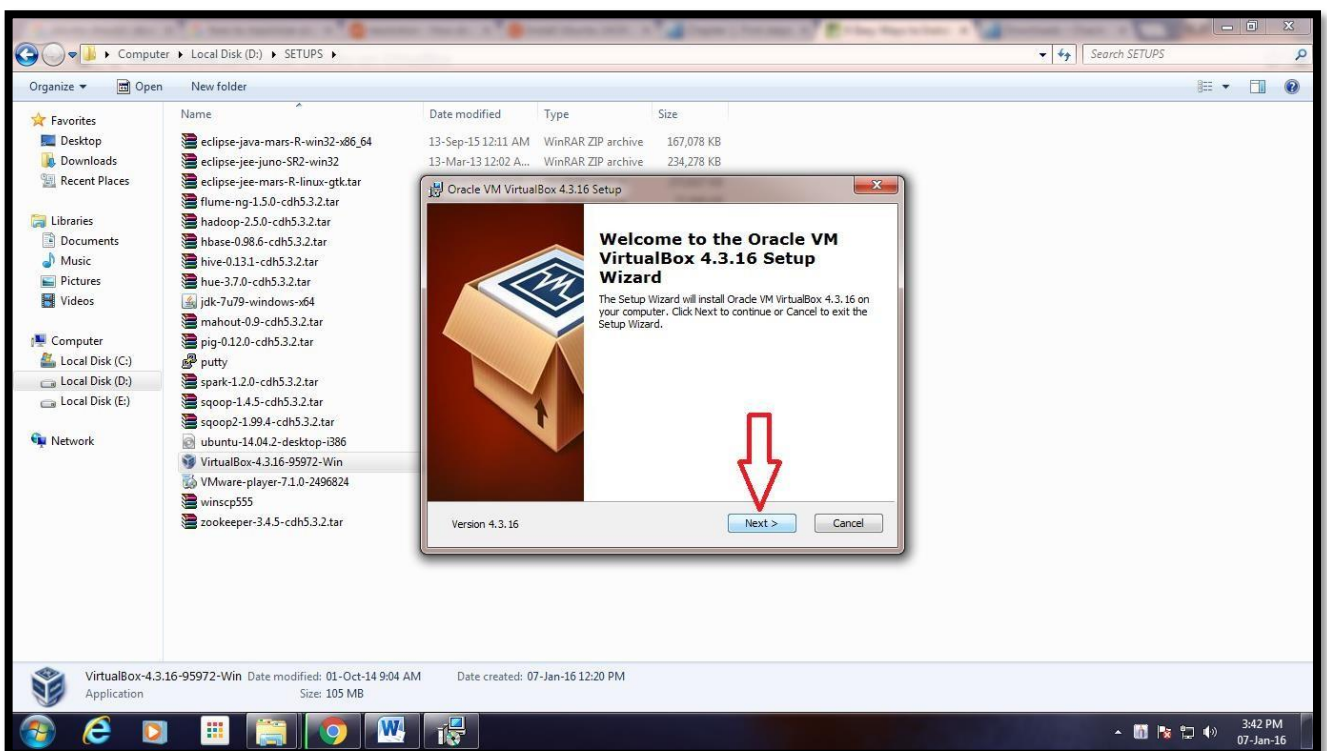
- 1) By double-clicking on the executable file.
- 2) By entering the following command:

```
VirtualBox-<version>-<revision>-Win.exe -extract
```

This will extract the installer into a temporary directory, along with the .MSI file. Run the following command to perform the installation:

```
msiexec /i VirtualBox-<version>-<revision>-Win.msi
```

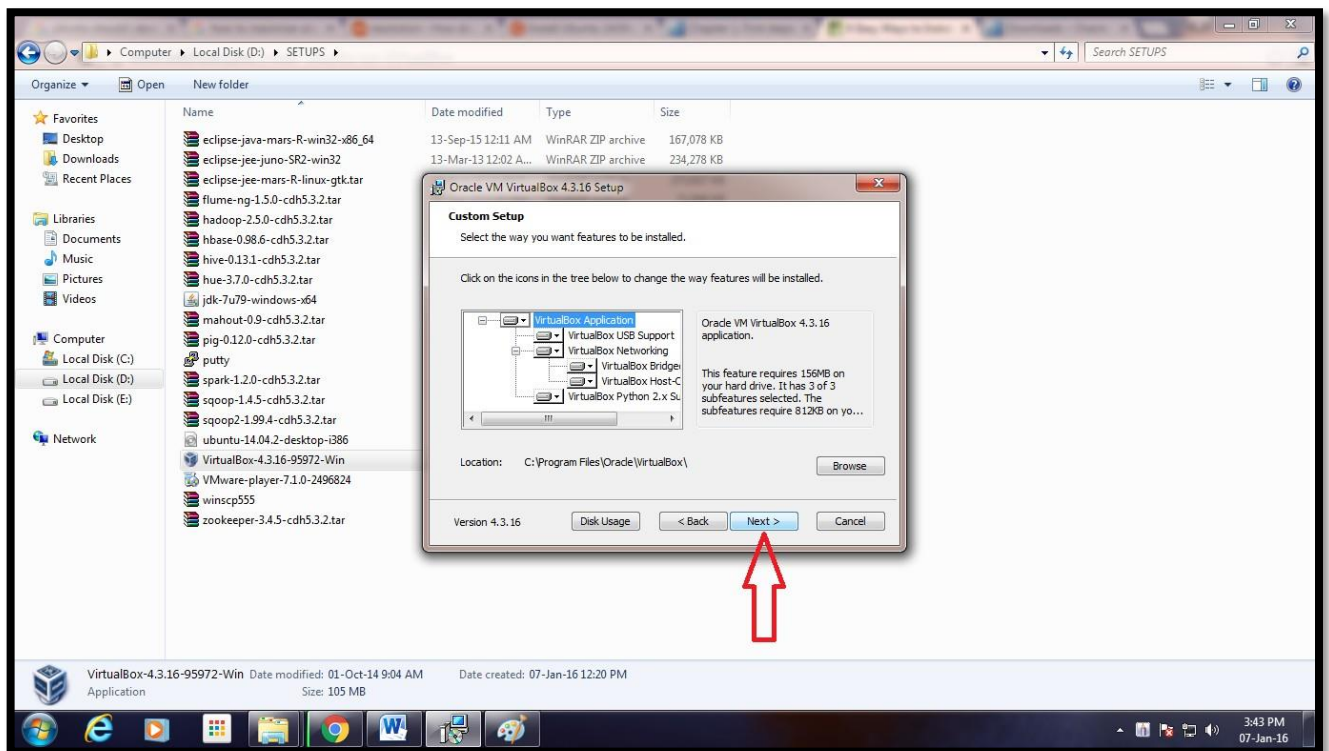
Run the windows executables files to displays the installation Welcome dialog and enables you to choose where to install Oracle VM VirtualBox, and which components to install.



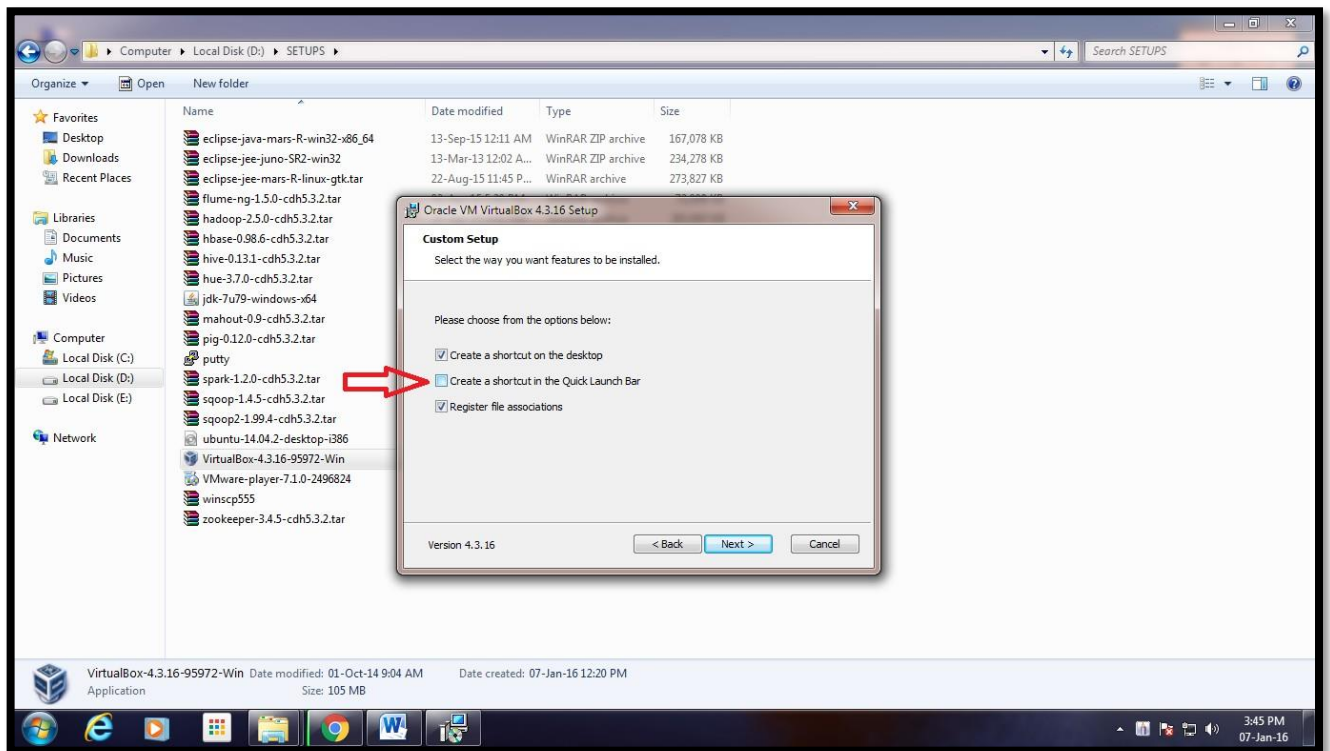
In addition to the Oracle VM VirtualBox application, the following components are available:

- 1) USB support. This package contains special drivers for your Windows host that Oracle VM VirtualBox requires to fully support USB devices inside your virtual machines.
- 2) Networking. This package contains extra networking drivers for your Windows host that Oracle VM VirtualBox needs to support Bridged Networking. This enables your VM's virtual network cards to be accessed from other machines on your physical network.
- 3) Python support. This package contains Python scripting support for the Oracle VM VirtualBox API

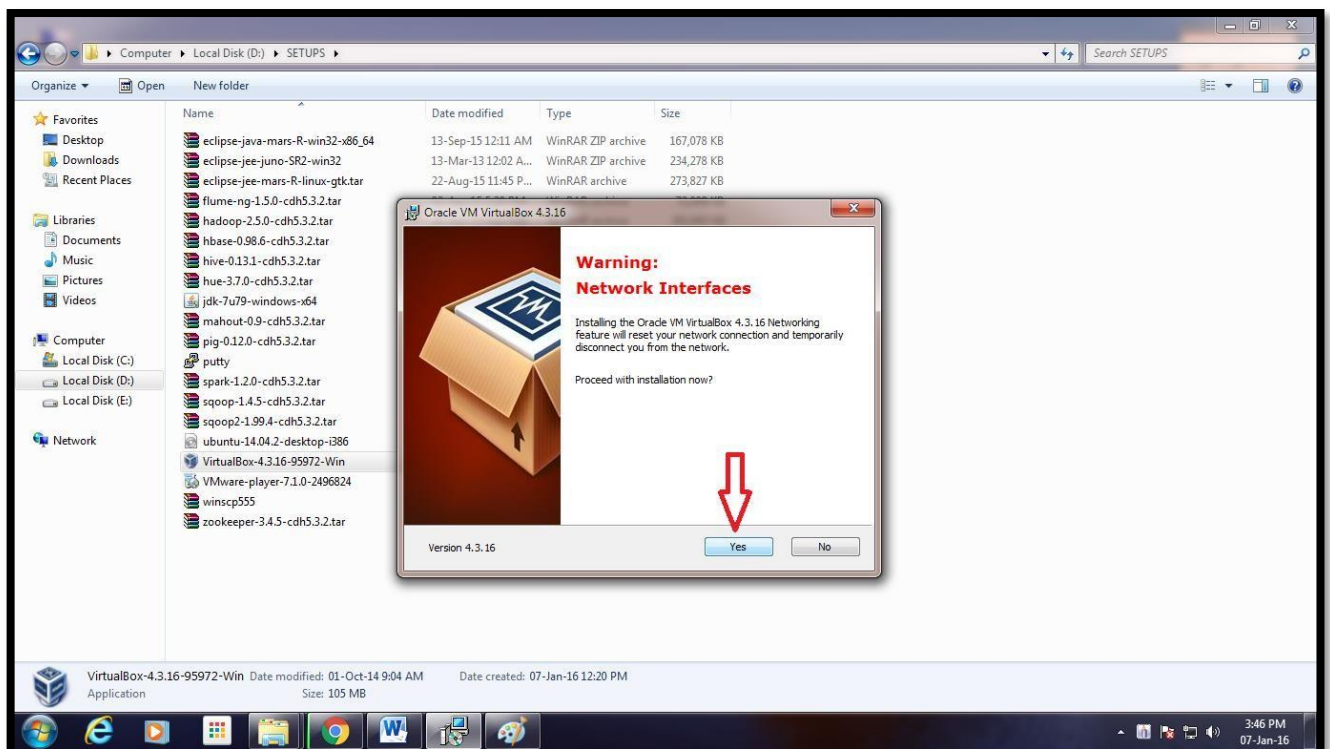
Click on Next



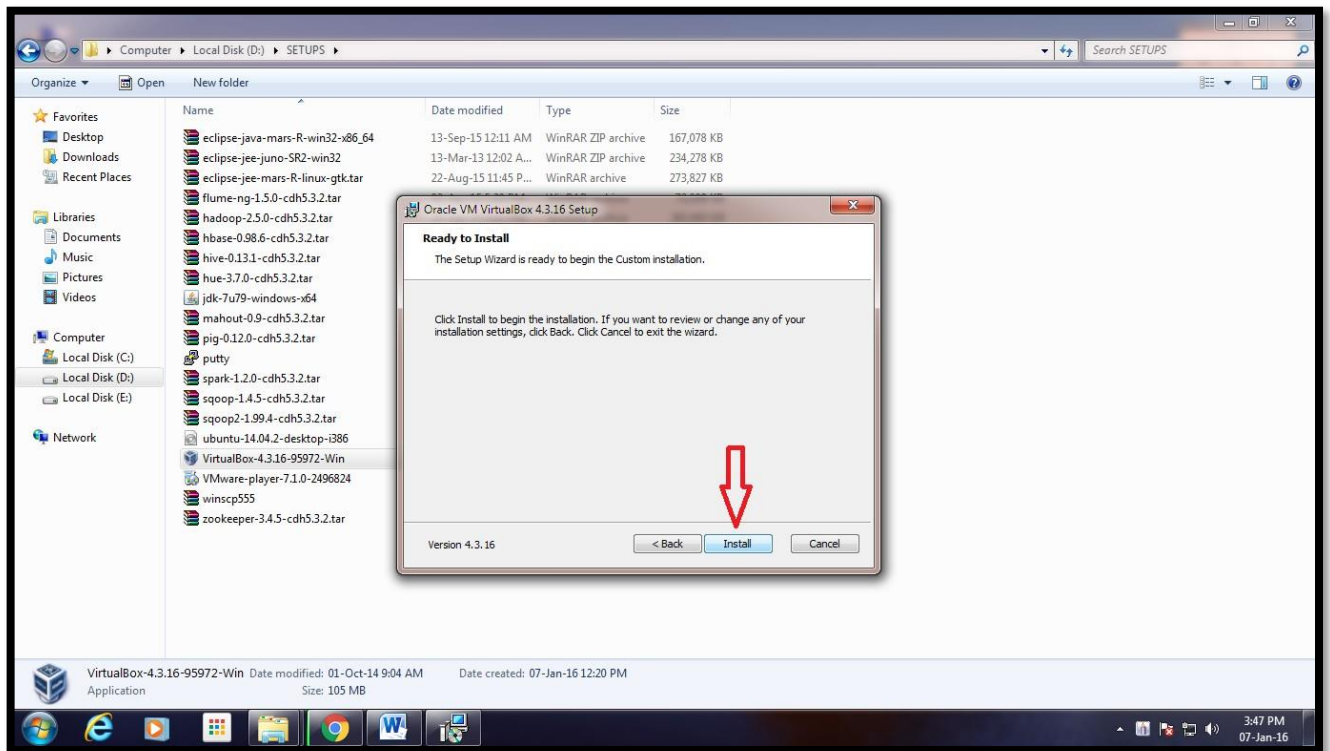
Step 2) Uncheck “Create a shortcut in the Quick Launch Bar” and click “Next”



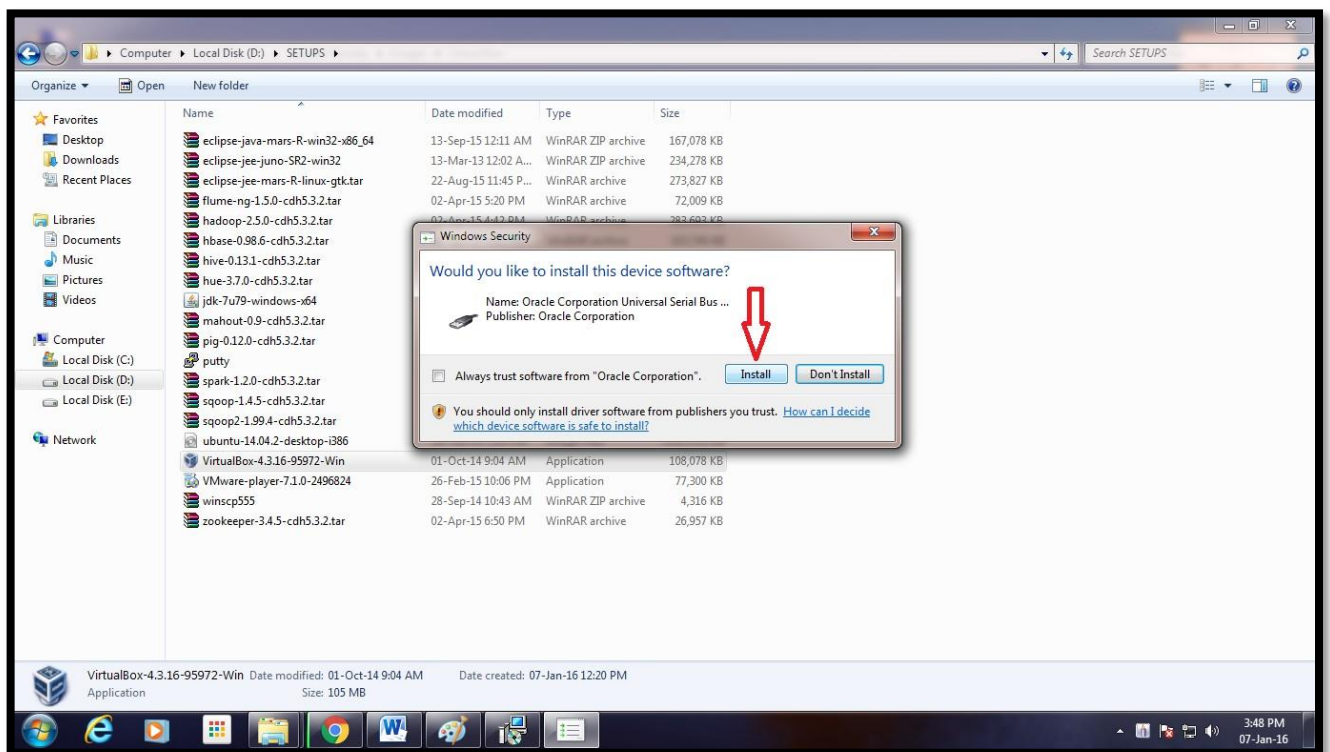
Step 3) Click “Yes”



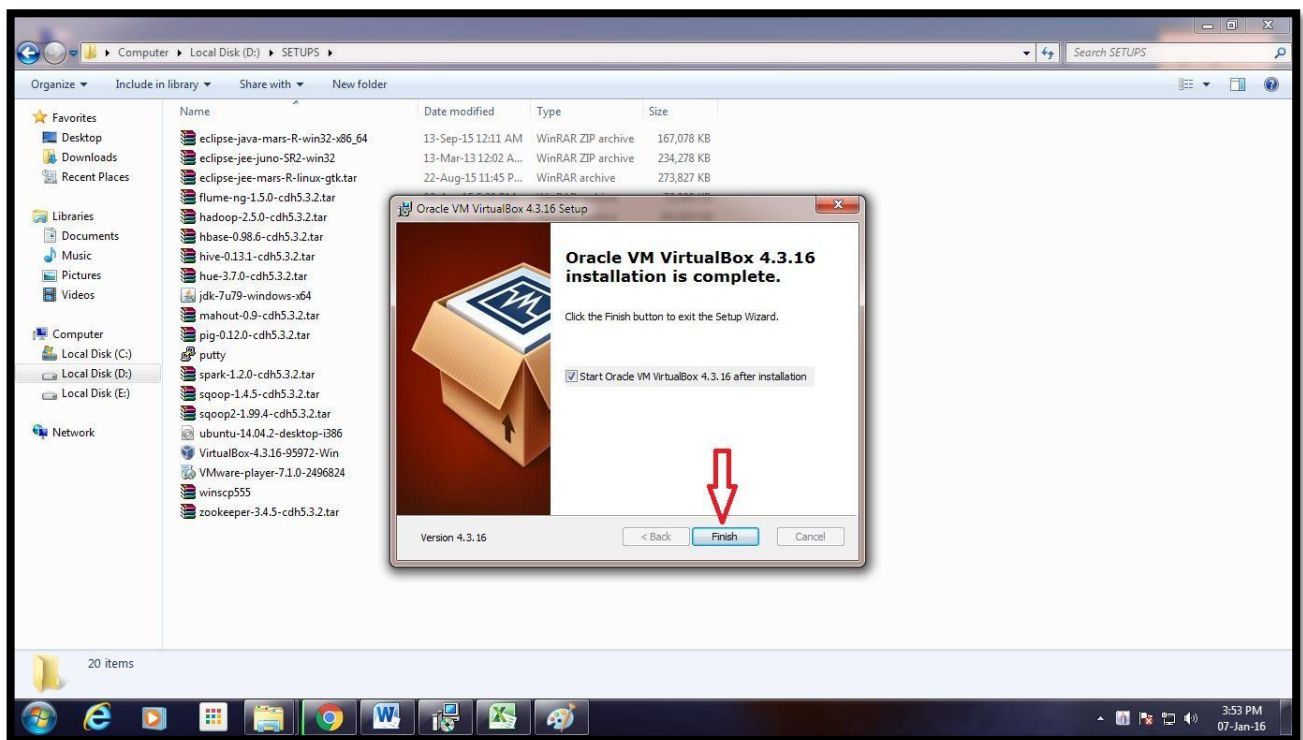
Step 4) Click on “Install”



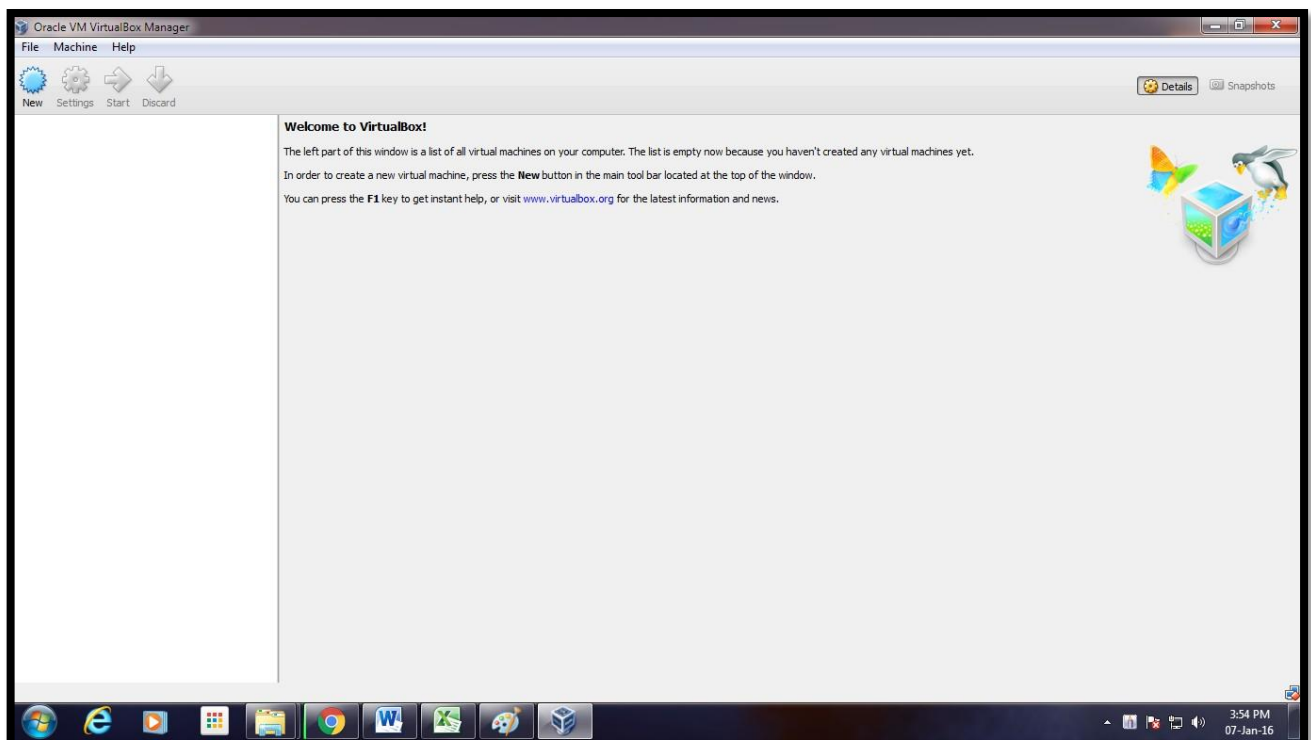
Click on install again for final confirmation



Step 4) Click on “Finish”



Step 5) After completion of Installation process, the Virtual Box Window gets opened



Experiment 8

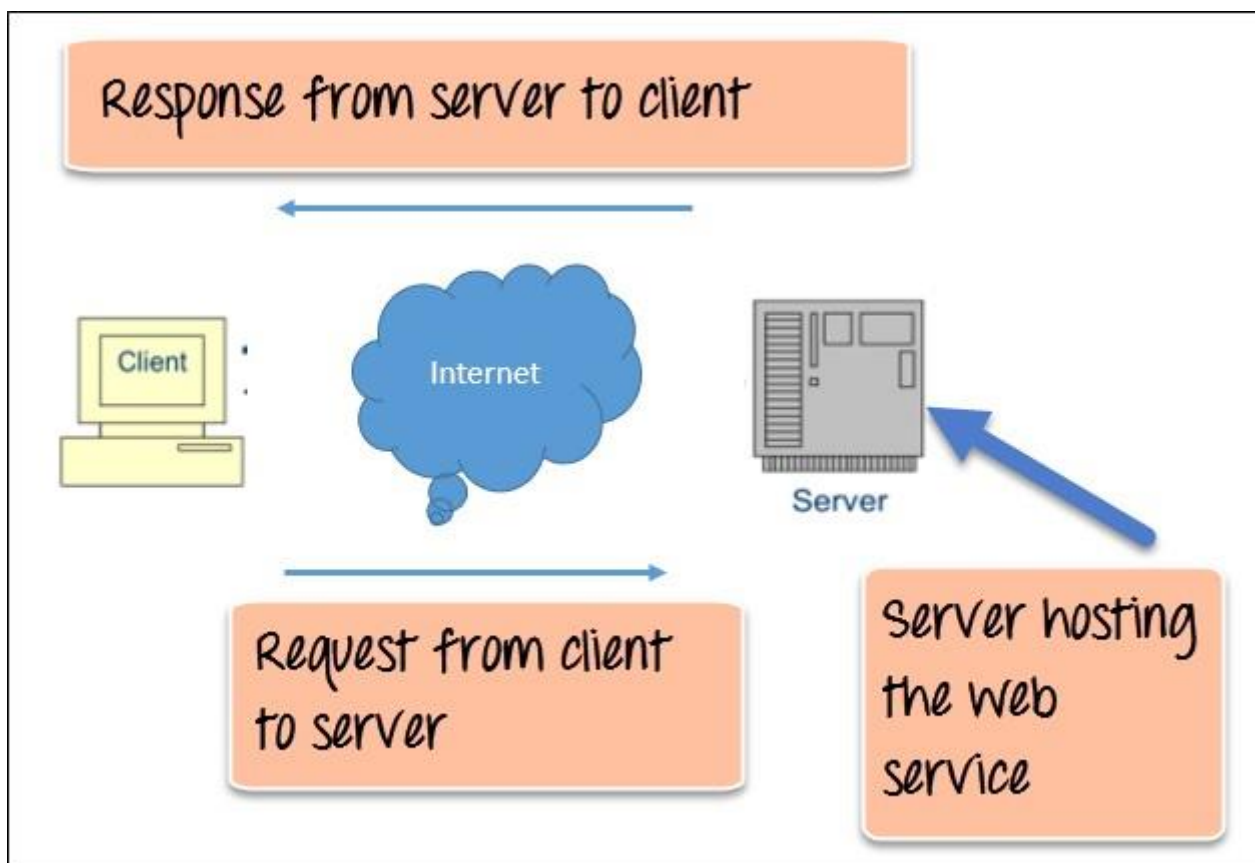
8. Review web services implementation - Proper Connection should be established between the client and server to make use of the service offered by the Server.
Review the working of application in virtual environment.

Web service is a standardized medium to propagate communication between the client and server applications on the WWW (World Wide Web). A web service is a software module that is designed to perform a certain set of tasks.

Web services in cloud computing can be searched for over the network and can also be invoked accordingly.

When invoked, the web service would be able to provide the functionality to the client, which invokes that web service.

Web Service implementation working



The above diagram shows a very simplistic view of how a web service would actually work. The client would invoke a series of web service calls via requests to a server which would host the actual web service.

These requests are made through what is known as remote procedure calls. Remote Procedure Calls(RPC) are calls made to methods which are hosted by the relevant web service.

As an example, Amazon provides a web service that provides prices for products sold online via amazon.com. The front end or presentation layer can be in .Net or Java but either programming language would have the ability to communicate with the web service.

The main component of a web service design is the data which is transferred between the client and the server, and that is XML. XML (Extensible markup language) is a counterpart to HTML and easy to understand the intermediate language that is understood by many programming languages.

So when applications talk to each other, they actually talk in XML. This provides a common platform for application developed in various programming languages to talk to each other.

Web services use something known as SOAP (Simple Object Access Protocol) for sending the XML data between applications. The data is sent over normal HTTP. The data which is sent from the web service to the application is called a SOAP message. The SOAP message is nothing but an XML document. Since the document is written in XML, the client application calling the web service can be written in any programming language.

The need of Web Services

Modern day business applications use variety of programming platforms to develop web-based applications. Some applications may be developed in Java, others in .Net, while some other in Angular JS, Node.js, etc.

Most often than not, these heterogeneous applications need some sort of communication to happen between them. Since they are built using different development languages, it becomes really difficult to ensure accurate communication between applications.

Here is where web services come in. Web services provide a common platform that allows multiple applications built on various programming languages to have the ability to communicate with each other.

Type of Web Service

There are mainly two types of web services.

- 1) SOAP web services.
- 2) RESTful web services.

In order for a web service to be fully functional, there are certain components that need to be in place. These components need to be present irrespective of whatever development language is used for programming the web service.

SOAP (Simple Object Access Protocol)

SOAP is known as a transport-independent messaging protocol. SOAP is based on transferring XML data as SOAP Messages. Each message has something which is known as an XML document. Only the structure of the XML document follows a specific pattern, but not the content. The best part of Web services and SOAP is that its all sent via HTTP, which is the standard web protocol.

Here is what a SOAP message consists of

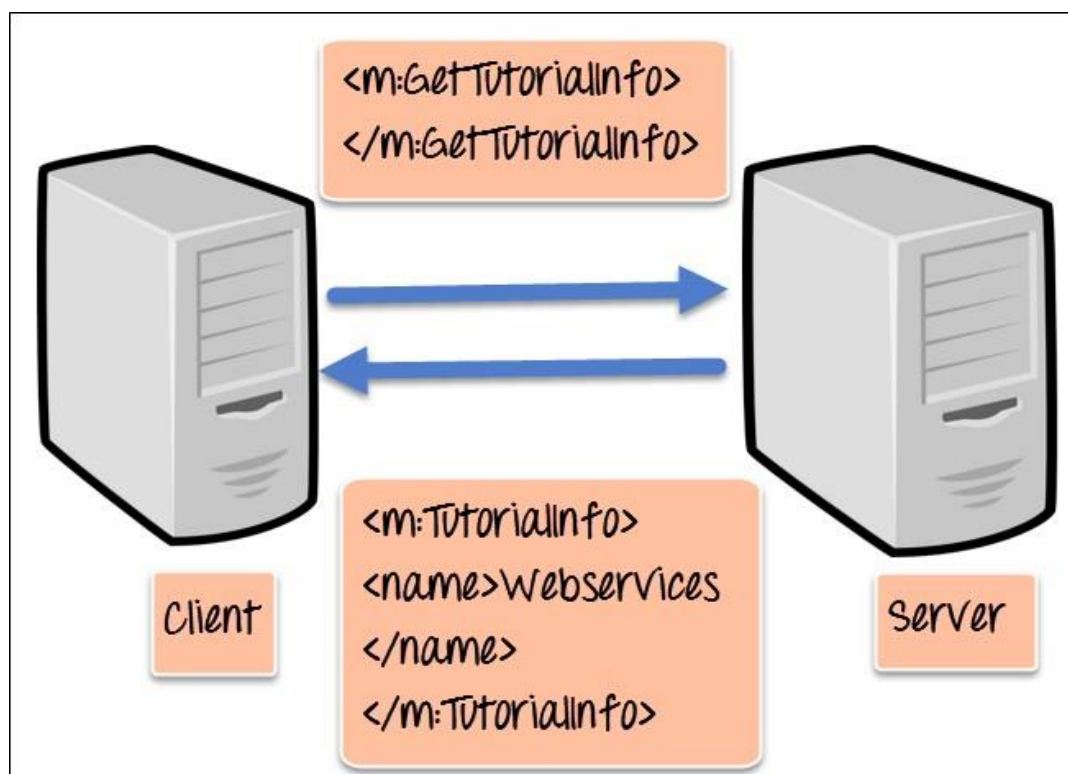
Each SOAP document needs to have a root element known as the `<Envelope>` element. The root element is the first element in an XML document.

The "envelope" is in turn divided into 2 parts. The first is the header, and the next is the body.

The header contains the routing data which is basically the information which tells the XML document to which client it needs to be sent to.

The body will contain the actual message.

The diagram below shows a simple example of the communication via SOAP.



Restful Web Services

Restful Web Services is a stateless client-server architecture where web services are resources and can be identified by their URIs. REST Client applications can use HTTP GET/POST methods to invoke Restful web services. REST doesn't specify any specific protocol to use, but in almost all cases it's used over HTTP/HTTPS. When compared to SOAP web services, these are lightweight and doesn't follow any standard. We can use XML, JSON, text or any other type of data for request and response.

REST API Implementations

There are two major implementations of JAX-RS API.

- 1) **Jersey**: Jersey is the reference implementation provided by Sun. For using Jersey as our JAX-RS implementation, all we need to configure its servlet in web.xml and add required dependencies. Note that JAX-RS API is part of JDK not Jersey, so we have to add its dependency jars in our application.
- 2) **RESTEasy**: RESTEasy is the JBoss project that provides JAX-RS implementation.

WSDL (Web services description language)

A web service cannot be used if it cannot be found. The client invoking the web service should know where the web service actually resides.

Secondly, the client application needs to know what the web service actually does, so that it can invoke the right web service. This is done with the help of the WSDL, known as the Web services description language. The WSDL file is again an XML-based file which basically tells the client application what the web service does. By using the WSDL document, the client application would be able to understand where the web service is located and how it can be utilized.

Universal Description, Discovery, and Integration (UDDI)

UDDI is a standard for describing, publishing, and discovering the web services that are provided by a particular service provider. It provides a specification which helps in hosting the information on web services.

Just as a telephone directory has the name, address and telephone number of a particular person, the same way the UDDI registry will have the relevant information for the web service. So that a client application knows, where it can be found.

The Need of Implementation of Web Services

Web Services provide following benefits:

- 1) Interoperability: Heterogeneous environments from various development environments (J2EE, CORBA, .Net) can be integrated into business applications.
- 2) Reuse: Developers can locate an existing web service provided by third parties and incorporate these into their solution.
- 3) Ubiquitous: Web Services make applications developed in various platforms visible. In future Web service tools will be available on most development platforms to make it easier to extend applications to Web Services.
- 4) Business flexibility: Developers can customize the existing services by wrapping them as per customer needs.
- 5) Easy application integration: As web services proliferate, independent software vendors can develop software packages that expose the services based on the Web services standard. These services can be easily integrated as compared to conventional enterprise application integration problems with differing information interfaces.

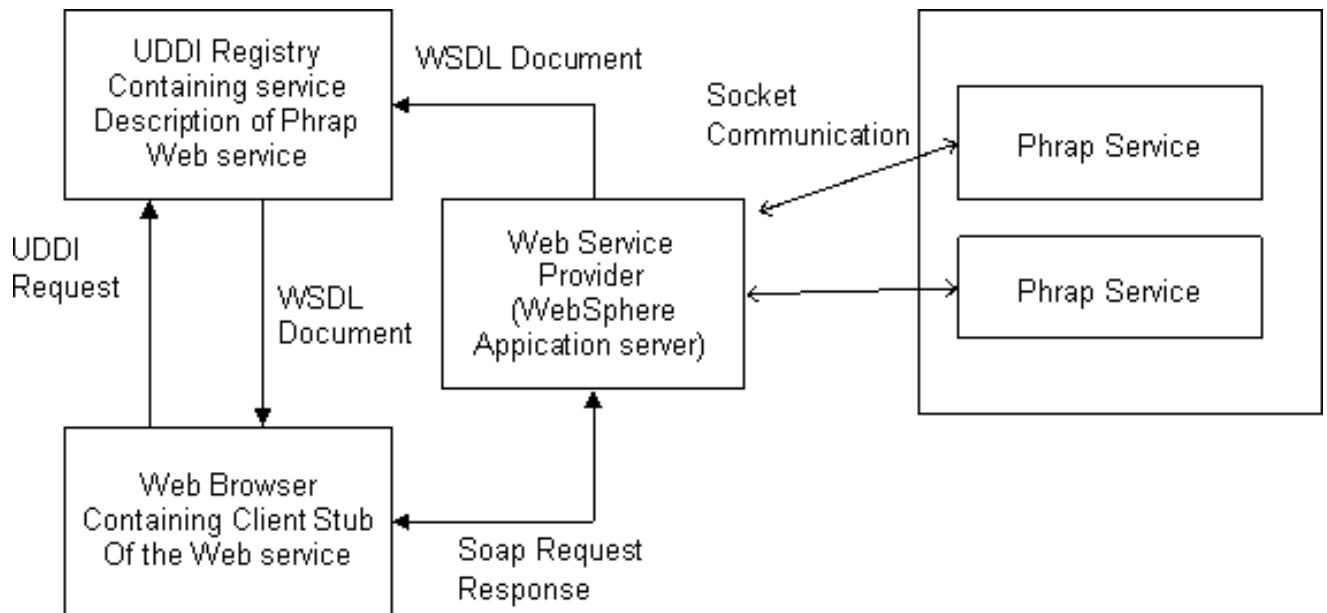
Web services are deployed on the web by services providers. The functions provided by the Web service are described using WSDL. The service providers publish deployed services on the Web through brokers.

A service broker helps the service providers and service requestors locate each other. A service requestor uses the UDDI API to ask the service broker about the services it needs. When the service broker returns the search results, the service requestor can use those results to bind to a particular service.

Implemented Architecture:

We are using Windows based WebSphere Studio and WebSphere Application Server to develop the Phrap Web Service. The WebSphere application server resides on a Windows machine, generates a WSDL document and registers to a private UDDI registry. The client can access this service through a web browser after a UDDI discovery process.

The service provider receives the input file and coordinates the proposed strategy in a reduced magnitude among two phrap services running on Strauss. The implementation is scalable to any number of phrap service instances and hence can be deployed on multiple processors as proposed.



Experiment 9

9. Use Security tools like ACUNETIX, ETTERCAP to scan Web Applications on the Cloud.

ACUNETIX

This information gathering tool scans web applications on the cloud and lists possible vulnerabilities that might be present in the given web application. Most of the scanning is focused on finding SQL injection and cross site scripting Vulnerabilities. It has both free and paid versions, with paid versions including added functionalities. After scanning, it generates a detailed report describing vulnerabilities along with the suitable action that can be taken to remedy the loophole.

This tool can be used for scanning cloud applications. Beware: there is always a chance of false positives. Any security flaw, if discovered through scanning, should be verified. The latest version of this software, Acunetix WVS version 8, has a report template for checking compliance with ISO 27001, and can also scan for HTTP denial of service attacks.

AIRCRAK-NG – A TOOL FOR WI-FI PEN TESTERS

This is a comprehensive suite of tools designed specifically for network pen testing and security. This tool is useful for scanning Infrastructure as a Service (IaaS) models. Having no firewall, or a weak firewall, makes it very easy for malicious users to exploit your network on the cloud through virtual machines. This suite consists of many tools with different functionalities, which can be used for monitoring the network for any kind of malicious activity over the cloud.

Its main functions include:

- 1) Aircrack-ng – Cracks WEP or WPA encryption keys with dictionary attacks
- 2) Airdcap-ng – Decrypts captured packet files of WEP and WPA keys
- 3) Airmmon-ng – Puts your network interface card, like Alfa card, into monitoring mode
- 4) Aircrack-ng – This is packet injector tool
- 5) Airodump-ng – Acts as a packet sniffer on networks
- 6) Airtun-ng – Can be used for virtual tunnel interfaces
- 7) Airolib-ng – Acts as a library for storing captured passwords and ESSID
- 8) Packetforge-ng – Creates forged packets, which are used for packet injection
- 9) Airbase-ng – Used for attacking clients through various techniques.
- 10) Airdecloak-ng – Capable of removing WEP clocking.

Several other tools are also available in this suite, including esside-ng, wesside-ng and tkiptun-ng. Aircrack-ng can be used on both command line interfaces and on graphical interfaces. In GUI, it is named Gerix Wi-Fi Cracker, which is a freely available network security tool licensed to GNU.

CAIN & ABEL

This is a password recovery tool. Cain is used by penetration testers for recovering passwords by sniffing networks, brute forcing and decrypting passwords. This also allows pen testers to intercept VoIP conversations that might be occurring through cloud. This multi functionality tool can decode Wi-Fi network keys, unscramble passwords, discover cached passwords, etc. An expert pen tester can analyze routing protocols as well, thereby detecting any flaws in protocols governing cloud security. The feature that separates Cain from similar tools is that it identifies security flaws in protocol standards rather than exploiting software vulnerabilities. This tool is very helpful for recovering lost passwords.

In the latest version of Cain, the ‘sniffer’ feature allows for analyzing encrypted protocols such as SSH-1 and HTTPS. This tool can be utilized for ARP cache poisoning, enabling sniffing of switched LAN devices, thereby performing Man in the Middle (MITM) attacks. Further functionalities have been added in the latest version, including authentication monitors for routing protocols, brute-force for most of the popular algorithms and cryptanalysis attacks.

ETTERCAP

Ettercap is a free and open-source tool for network security, designed for analyzing computer network protocols and detecting MITM attacks. It is usually accompanied with Cain. This tool can be used for pen testing cloud networks and verifying leakage of information to an unauthorized third party. It has four methods of functionality:

- 1) IP-based Scanning – Network security is scanned by filtering IP based packets.
- 2) Mac-based Scanning – Here packets are filtered based on MAC addresses. This is used for sniffing connections through channels.
- 3) ARP-based functionality – ARP poisoning is used for sniffing into switched LAN through an MITM attack operating between two hosts (full duplex).
- 4) Public-ARP based functionality – In this functionality mode, ettercap uses one victim host to sniff all other hosts on a switched LAN network (half duplex).

Experiment 10

10.Cloud Networks for finding vulnerabilities, verifying leakage of information to an unauthorized third party.

Using Ettercap for analysing computer network protocols and detecting MITM attacks on Cloud networks

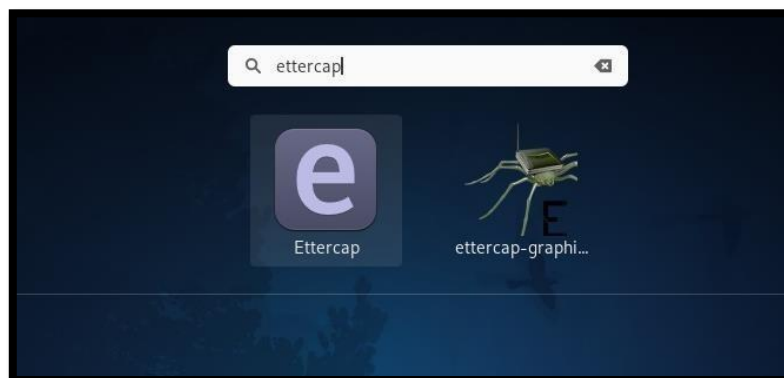
One of the most intriguing programs installed by default in Kali Linux is Ettercap. Unlike many of the programs that are command-line only, Ettercap features a graphical interface that's very beginner-friendly.

This tool can be used for pen testing cloud networks and verifying leakage of information to an unauthorized third party. However, Ettercap proves useful enough to feature for our demonstration. The general workflow of an Ettercap ARP spoofing attack is to join a network you want to attack, locate hosts on the network, assign targets to a "targets" file, and then execute the attack on the targets.

Step 1 : Connect to the Network

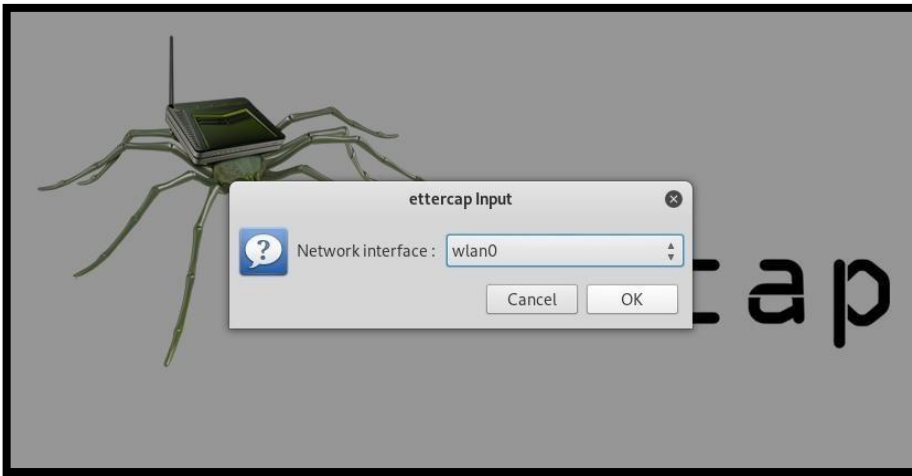
- The first step of ARP spoofing is to connect to the network you want to attack. If you're attacking an encrypted WEP, WPA, or WPA2 network, you'll need to know the password. This is because we're attacking the network internally, so we need to be able to see some information about the other hosts on the network and the data passing within it.
- We can connect to a network for ARP spoofing in two ways, ethernet and or by using wireless network adaptor.

Step 2: Start Ettercap

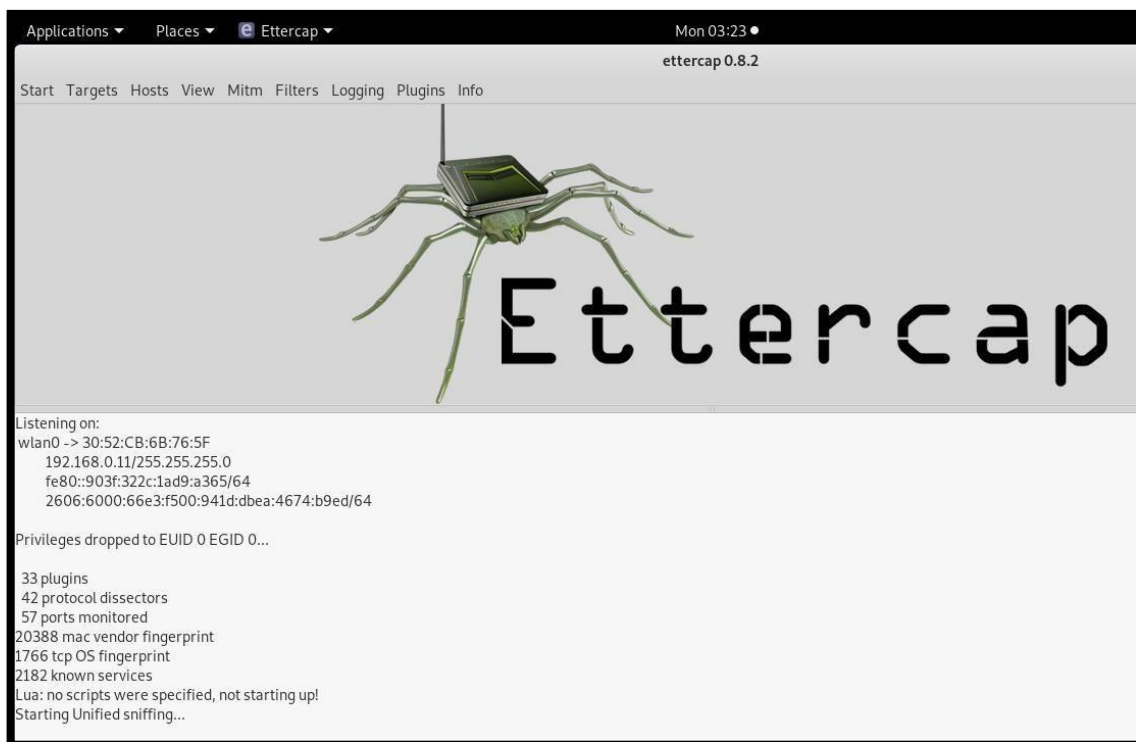


Step 2: Select Network Interface to Sniff On

Click on the "Sniff" menu item, and then select "Unified sniffing." A new window will open asking you to select which network interface you want to sniff on. You should select the network interface that is currently connected to the network you're attacking.

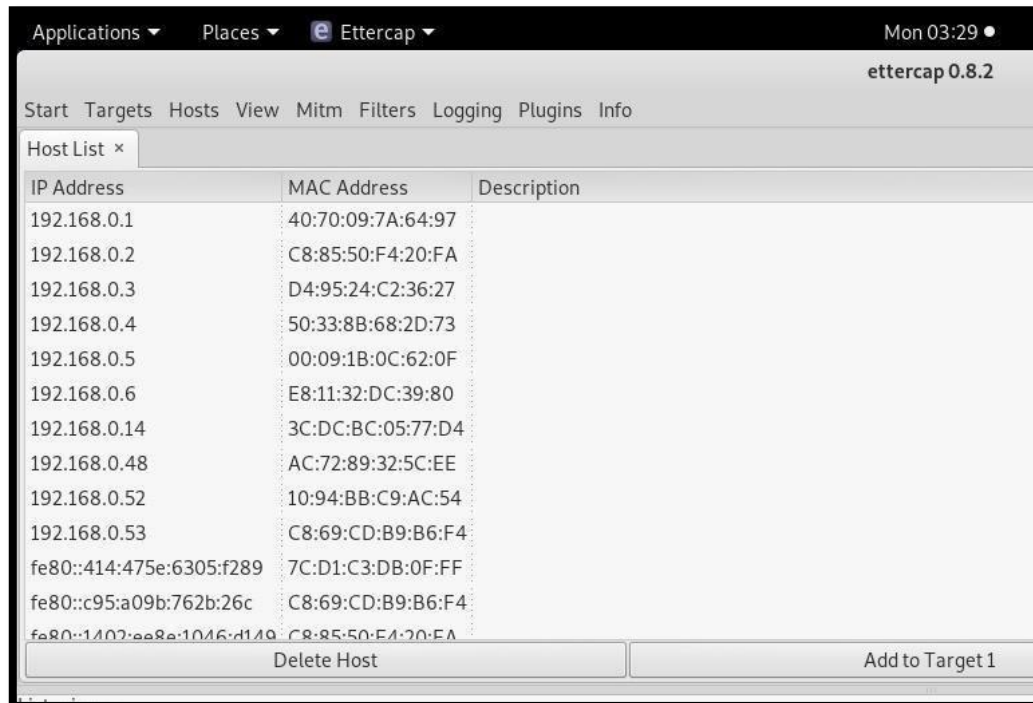


Now, we will be able to see some text confirming that sniffing has started, and we'll be able to access more advanced menu options such as Targets, Hosts, Mitm, Plugins, etc. Before we get started using any of them, we'll need to identify our target on the network.



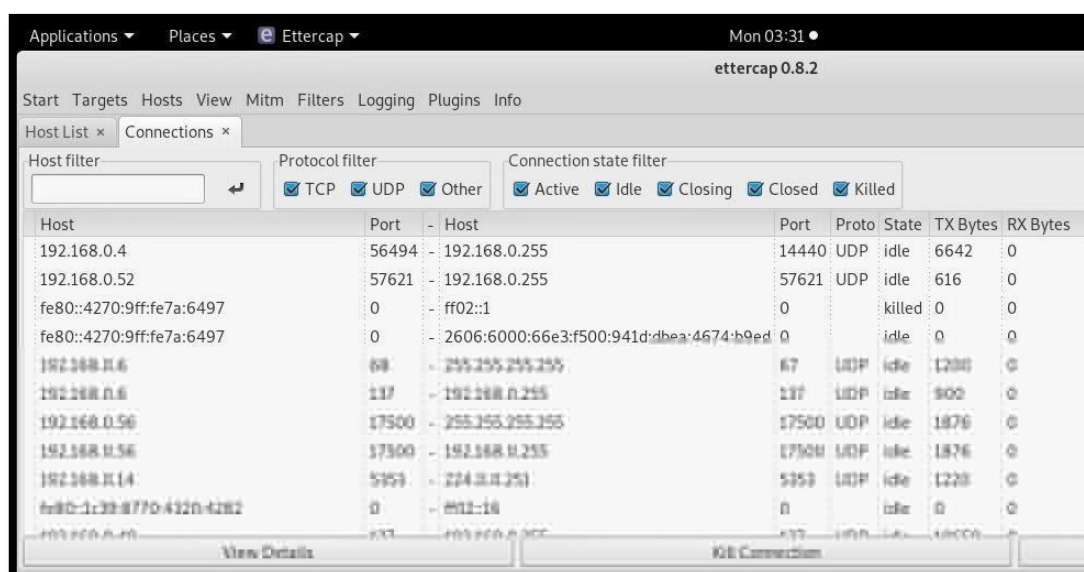
Step 3: Identify Hosts on a Network

To find the device we want to attack on the network, Ettercap has a few tricks up its sleeve. First, we can do a simple scan for hosts by clicking "Hosts," then "Scan for hosts." A scan will execute, and after it finishes, you can see the resulting hosts Ettercap has identified on the network by clicking "Hosts," then "Hosts list."



We can now see a list of targets we've discovered on the network.

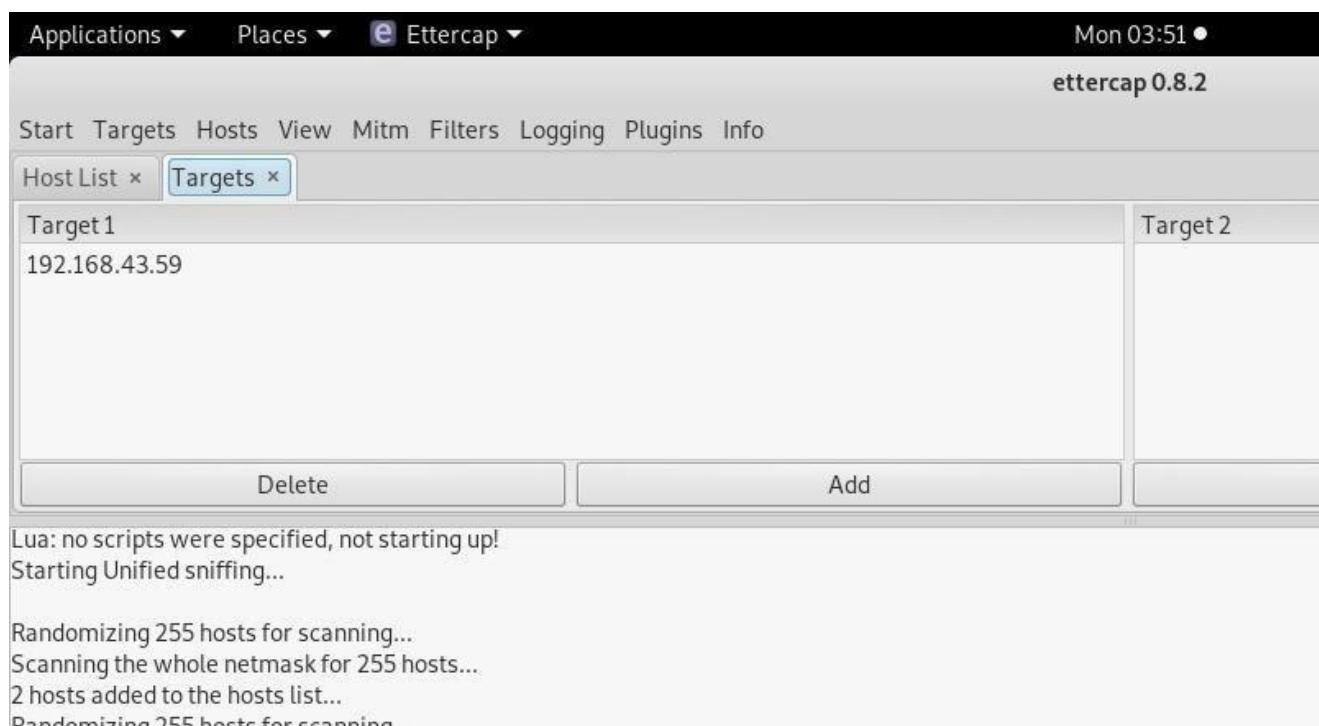
Once in the Connections view, you can filter the connections by IP address, type of connection, and whether the connection is open, closed, active, or killed. This gives you a lot of snooping power, which can be augmented by clicking the "View," then "Resolve IP addresses." This means Ettercap will try to resolve the IP addresses it sees other devices on the network connecting to.



Step 4: Select Hosts to Target with ARP Spoofing

- 1) we've identified our target's IP address
- 2) Now we'll add them to a target list.
- 3) Once we do this, we'll be telling Ettercap that we want to designate that IP address as one we want to pretend to be, so that we're receiving messages from the router that were meant to be sent to the target.

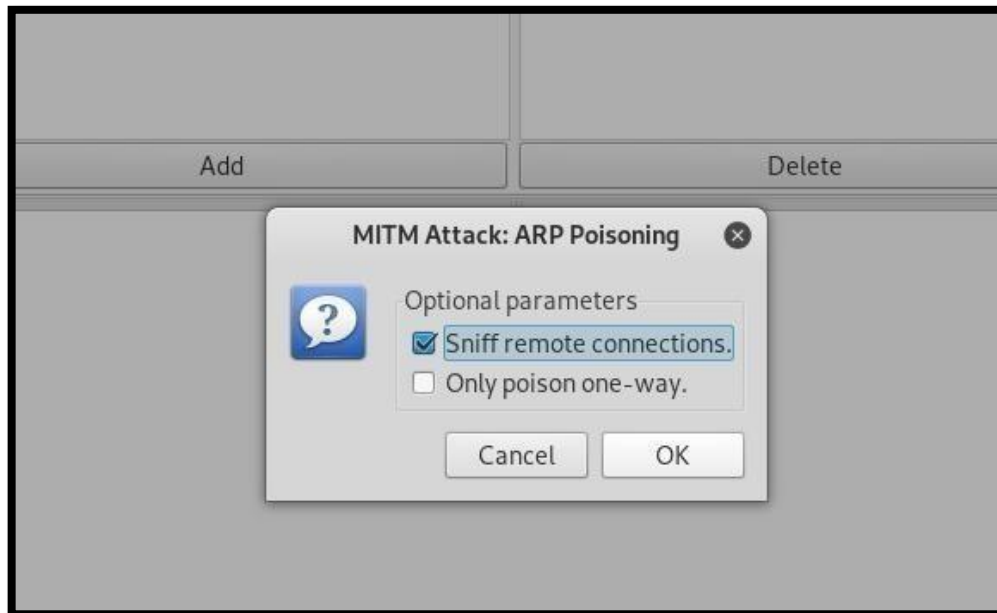
Go back to the "Hosts" screen and select the IP address of the target you want to target. Click the IP address to highlight it, then click on "Targets," followed by "Target list," to see a list of devices that have been targeted for ARP spoofing.



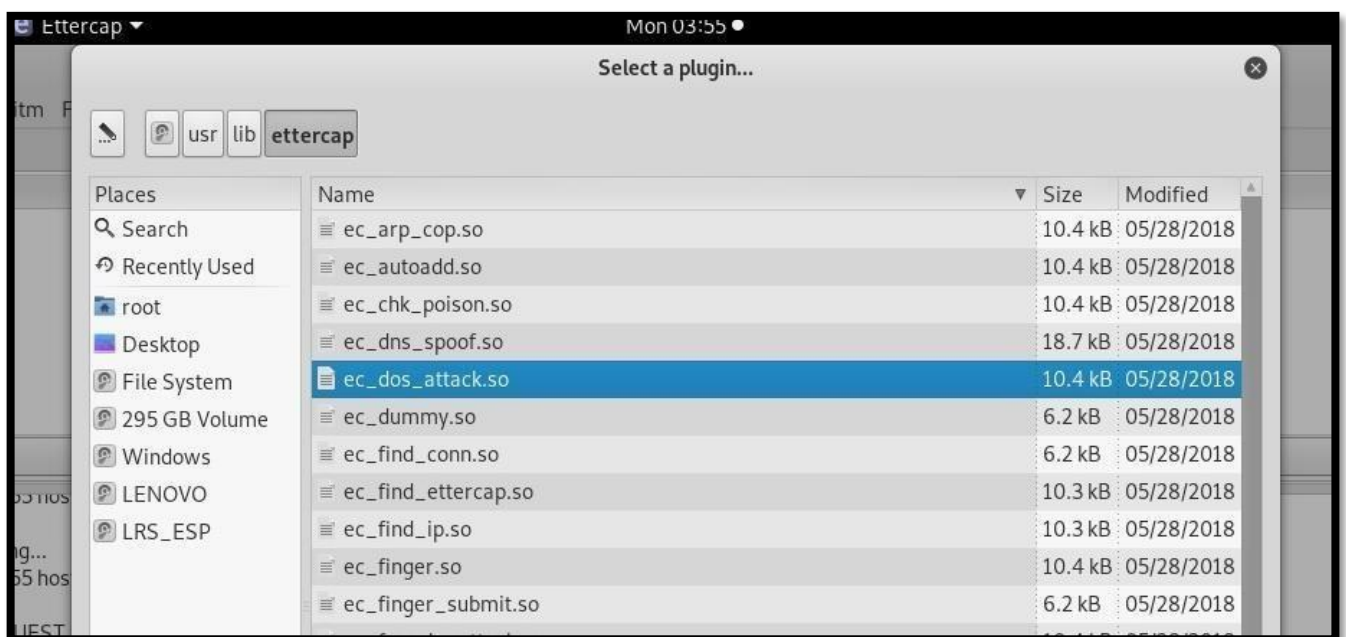
Now, we can go to the "Mitm" menu to start our attack on this target.

Step 5: Launch Attack on Targets

Click on the "Mitm" menu, and select "ARP poisoning." A popup will open, and you'll select "Sniff remote connections" to begin the sniffing attack.



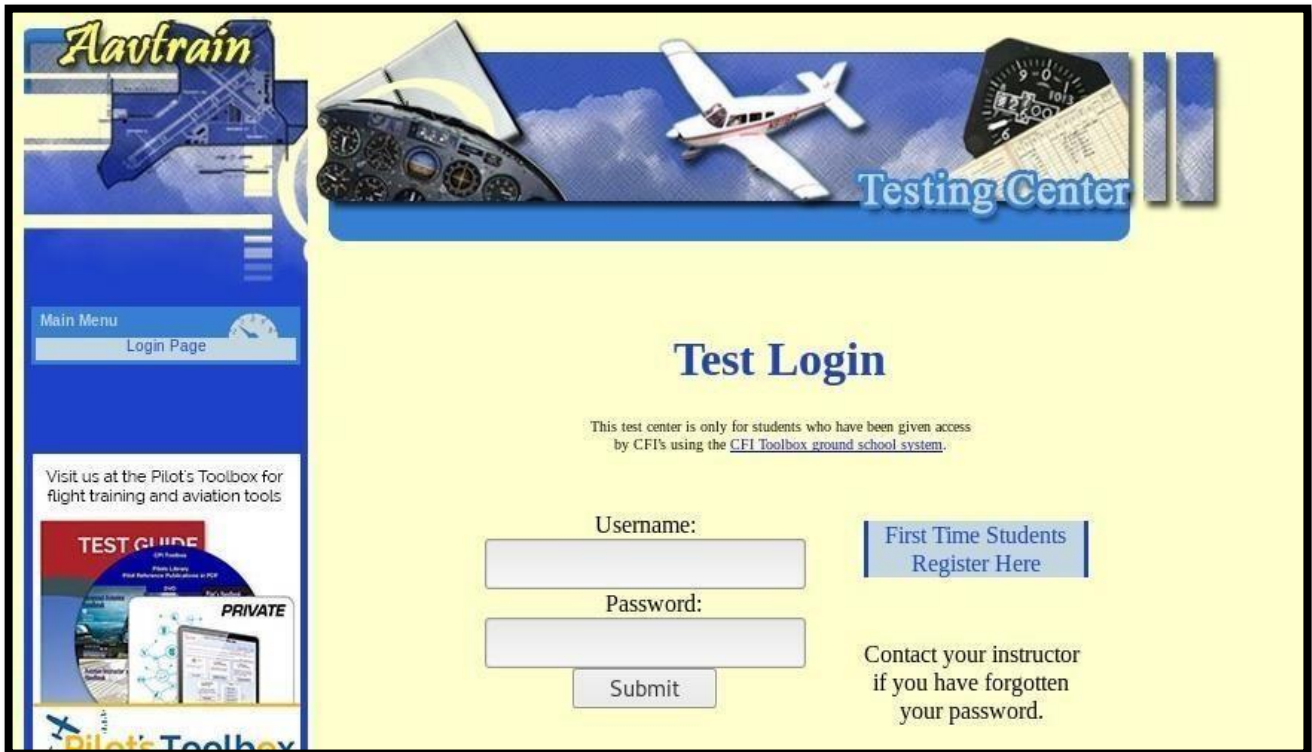
Once this attack has begun, we'll be able to intercept login credentials if the user you're targeting enters them into a website that doesn't use HTTPS. This could be a router or a device on the network or even a website that uses poor security.



Step 7: Try Intercepting a Password

Now, trying intercepting a password. A website that's great for testing is aavtain.com, which deliberately uses bad security so that you can intercept credentials. On the target

device, navigate to aavtrain.com. Once it loads, you'll see a login screen you can enter a fake login and password into.



Enter a username and password, then hit "Submit." If Ettercap is successful, we should see the login and password we have typed appear on the attacker's screen!

```
DHCP: [6C:7B:C8:A6:2A:42] DISCOVER
DHCP: [6C:7B:C8:A6:2A:42] DISCOVER
DHCP: [6C:7B:C8:A6:2A:42] REQUEST 192.168.43.59
Randomizing 255 hosts for scanning...
Scanning the whole netmask for 255 hosts...
3 hosts added to the hosts list...
Host 192.168.43.59 added to TARGET1

ARP poisoning victims:

GROUP 1 : 192.168.43.59 6C:7B:C8:A6:2A:42

GROUP 2 : ANY (all the hosts in the list)
HTTP : 192.185.11.183:80 -> USER: nullbyte PASS: averysecretpass INFO: http://aavtrain.com/
CONTENT: user_name=nullbyte&password=averysecretpass&Submit=Submit&login=true
```

- ❖ This way we can verify the various vulnerabilities using a ARP spoofing attacks implemented using Ettercap tool.

Experiment 11

11. Report Submission: Generate a detailed report describing vulnerabilities along with the suitable action that can be taken to remedy the loopholes

Vulnerability is a prominent factor of risk. ISO 27005 defines risk as “the potential that a given threat will exploit vulnerabilities of an asset or group of assets and thereby cause harm to the organization,” measuring it in terms of both the likelihood of an event and its consequence.

Cloud-Specific Vulnerabilities

Based on the abstract view of cloud computing we presented earlier, we can now move toward a definition of what constitutes a cloud-specific vulnerability. A vulnerability is cloud specific if it is intrinsic to or prevalent in a core cloud computing technology, has its root cause in one of NIST’s essential cloud characteristics, is caused when cloud innovations make tried-and-tested security controls difficult or impossible to implement or is prevalent in established state-of-the-art cloud offerings.

Following are the major Cloud Computing Security Vulnerabilities and Ways to Mitigate Them

1) Misconfigured Cloud Storage

- Cloud storage is a rich source of stolen data for cybercriminals. Despite the high stakes, organizations continue to make the mistake of misconfiguration of cloud storage which has cost many companies greatly.
- Cloud storage misconfiguration can quickly escalate into a major cloud security breach for an organization and its customers. There are several types of cloud misconfigurations that enterprises encounter. Some types of misconfigurations include
 - 1) AWS security group misconfiguration
 - 2) Lack of access restrictions
- When it comes to cloud computing, it’s always a good idea to double-check cloud storage security configurations upon setting up a cloud server. While this may seem obvious, it can easily get overlooked by other activities such as moving data into the cloud without paying attention to its safety.
- You can also use specialized tools to check cloud storage security configurations. These cloud security tools can help you check the state of security configurations on a schedule and identify vulnerabilities before it’s too late.

- Control who can create and configure cloud resources. Many cloud computing issues have come from people who want to move into the cloud without understanding how to secure their data.

2) Insecure APIs

- Application user interfaces (APIs) are intended to streamline cloud computing processes. However, if left insecure, APIs can open lines of communications for attackers to exploit cloud resources.
- A recent study also revealed that two-thirds of enterprises expose their APIs to the public so that external developers and business partners can access software platforms.

Prevent Insecure APIs

- Encourage developers to design APIs with strong authentication, encryption, activity monitoring, and access control. APIs must be secured.
- Conduct penetration tests that replicate an external attack targeting your API endpoints and get a secure code review as well. It is best to ensure you have a secure software development lifecycle (SDLC) to ensure you continually develop secure applications and APIs.
- Also, consider using SSL/TLS encryption for data-in-transit. Implement multi-factor authentication with schemas such as one-time passwords, digital identities, etc. to ensure strong authentication controls.

3) Loss or Theft of Intellectual Property

- Intellectual property (IP) is undeniably one of the most valuable assets of an organization, and it is also vulnerable to security threats, especially if the data is stored online.
- An analysis found that almost 21% of files uploaded to cloud-based file-sharing services contain sensitive information including IP. When these cloud services are breached, attackers can gain access to sensitive information stored in them.

Types of Common loss of data

- a) Data alteration
- b) Data deletion
- c) Loss of access

Prevent Loss or Theft of Intellectual Property

- Frequent backups are one of the most effective ways to prevent loss or theft of intellectual property. Set a schedule for regular backups and clear delineation of

what data is eligible for backups and what is not. Consider using data loss prevention (DLP) software to detect and prevent unauthorized movement of sensitive data.

- Another solution to prevent loss or theft of data is to encrypt your data and geo-diversify your backups. Having offline backups is also very important, especially with ransomware.

4) Poor Access Management

- Improper access management is perhaps the most common cloud computing security risk. In breaches involving web applications, stolen or lost credentials have been the most widely used tool by attackers for several years.
- Access management ensures that individuals can perform only the tasks they need to perform. The process of verifying what an individual has access to is known as authorization.

There are several other cloud-specific challenges that organizations face, including the following:

- a) Inactive assigned users
- b) Multiple administrator accounts
- c) Improper user and service provisioning and deprovisioning — for instance, companies not revoking access permissions of former employees
- d) Users bypassing enterprise access management controls

Prevent Poor Access Management

- To combat poor access management in cloud services, enterprises need to develop a data governance framework for user accounts. For all human users, accounts should be linked directly to the central directory services, such as Active Directory, which is responsible for provisioning, monitoring, and revoking access privileges from a centralized store.
- Additionally, enterprises should use cloud-native or third-party tools to regularly pull lists of roles, privileges, users, and groups from cloud service environments. AWS Command Line Interface and PowerShell for Azure can collect this type of data, and then the security team can sort, store, and analyze it.
- Organizations should also ensure logging and event monitoring mechanisms are in place in cloud environments to detect unusual activity or unauthorized

changes. Access keys should be tightly controlled and managed to avoid poor data handling or leakage.

5) Contractual Breaches with Customers or Business Partners

- Contracts in cloud computing are somewhat tricky. It often restricts who is authorized to access the data, how it can be used, and where and how it can be stored. When employees move restricted data into the cloud without authorization, the business contracts may be violated, and legal action could ensue.
- For instance, if your cloud service provider maintains the right to share all data uploaded to the cloud with third parties under their terms and conditions, they are breaching a confidentiality agreement with your company.
- This could lead to leakage of data from your customers, employees, and other stakeholders that may have been uploaded to the cloud.

Prevent Contractual Breaches with Customers or Business Partners

- The cloud service contract should include the rights to review, monitor, and audit reports. This way, any security risk can be identified at an early stage before it becomes an issue. Companies should also ensure that they are not locked into a service contract and switching vendors can be a smooth exercise.
- This means that the service contract should include service termination rights for the business (for example, change of control, service deterioration, regulatory requirements, security/confidentiality breach, etc.)
- The service contract should also highlight the intellectual property risk, as cloud services may include the use of IP or other software rights under a license agreement. The organization could then be dragged into a legal dispute if a third party claims infringement against the cloud service provider.

Experiment 12

12. Install and configure OpenStack all-in-one using DevStack/Packstack

Openstack Installation and Configuration

Key point

- The installation and configuration process requires all the prerequisites to be in place or installation will fail. It is also possible to setup Openstack on laptop using Openstack minimal setup.

What is RDO Openstack

RDO Openstack is an openstack deployment launched by RedHat in 2013. RDO Openstack is also known as RPM Distribution of RedHat. RDO Openstack was launched for Red Hat Enterprise Linux(RHEL) and other linux distributions such as CentOS, Fedora.

Packstack vs Devstack

a) Packstack is mostly suitable for Red Hat Distribution Linux like CentOS and Fedora. It basically uses puppet modules to deploy various part of Openstack Components through ssh.

b) Devstack is a script written to create an environment with Openstack minimal setup which can be used to setup Openstack on laptop as well.

Visit Openstack Packstack CentOS for openstack installation through Packstack on CentOS 7.

Openstack Configuration Step by Step

Step 1: Prerequisites

- a) Openstack minimal setup requires minimum memory of 4GB should be available in your system.
- b) Make sure latest version of python and pip is installed in your system.
- c) Install git using yum install git

```
root@localhost ~]# yum install git
Loaded plugins: fastestmirror
Loading mirror speeds from cached hostfile
epel/x86_64/metalink | 7.9 kB 00:00:00
```

```
* base: centos.excellmedia.net
* epel: mirrors.aliyun.com
* extras: centos.excellmedia.net
* updates: centos.excellmedia.net
base | 3.6 kB 00:00:00
epel | 5.3 kB 00:00:00
extras | 2.9 kB 00:00:00
kubernetes/signature | 454 B 00:00:00
kubernetes/signature | 1.4 kB 00:00:00 !!!
puppetlabs-pc1 | 2.5 kB 00:00:00
updates | 2.9 kB 00:00:00
(1/4): epel/x86_64/updateinfo | 1.0 MB 00:00:00
(2/4): kubernetes/primary | 60 kB 00:00:01
(3/4): puppetlabs-pc1/x86_64/primary_db | 234 kB 00:00:01
(4/4): epel/x86_64/primary_db | 6.9 MB 00:00:02
kubernetes 433/433
Resolving Dependencies
--> Running transaction check
.....
```

Step 2: Create an User

Devstack perform lot of changes in your system hence it does not perform installation through root user. Instead you need to create an user with sudo access to perform the installation. In our example, I have created stackuser.

```
[root@localhost ~]# useradd -d /home/stackuser -m stackuser
[root@localhost ~]# cat /etc/passwd | grep -i stackuser
stackuser:x:1000:1000::/home/stackuser:/bin/bash
```

Note:- You can also use create-stack-user.sh script to create an user.

Step 3: Provide sudo access to the User

You need to provide the sudo access to stackuser.

```
[root@localhost ~]# echo "stackuser ALL=(ALL) NOPASSWD: ALL" | tee
/etc/sudoers.d/stackuser
stackuser ALL=(ALL) NOPASSWD: ALL
```

Step 4: Openstack Download using Devstack

Switch user to stackuser and start the git cloning. Here we are using rocky version of Devstack. You can choose any version as per your requirement.

```
[root@localhost ~]# su - stackuser
[stackuser@localhost ~]$ git clone https://github.com/openstack-dev/devstack.git -b
stable/rocky devstack/
Cloning into 'devstack'...
remote: Enumerating objects: 54, done.
remote: Counting objects: 100% (54/54), done.
remote: Compressing objects: 100% (54/54), done.
remote: Total 44484 (delta 30), reused 17 (delta 0), pack-reused 44430
Receiving objects: 100% (44484/44484), 14.11 MiB | 3.84 MiB/s, done.
Resolving deltas: 100% (30950/30950), done.
```

Step 5: Configure local.conf for Openstack Deployment

Once Devstack is downloaded locally in your system. Go to devstack directory and configure local.conf for openstack deployment as mentioned below.

```
[stackuser@localhost devstack]$ cat local.conf
[[local|localrc]]
ADMIN_PASSWORD=test@123
DATABASE_PASSWORD=\$ADMIN_PASSWORD
RABBIT_PASSWORD=\$ADMIN_PASSWORD
SERVICE_PASSWORD=\$ADMIN_PASSWORD
HOST_IP=192.168.0.104
RECLONE=yes
```

Step 6: OpenStack Installation and Configuration

Now perform Openstack installation and configuration by running stack.sh Script as shown below.

```
[stackuser@localhost devstack]$ ./stack.sh
```

Openstack Installation and configuration usually takes around 20-25 mins depends on your network bandwidth.

Output:-

```
This is your host IP address: 192.168.0.104
This is your host IPv6 address: ::1
Horizon is now available at http://192.168.0.104/dashboard
Keystone is serving at http://192.168.0.104/identity/
The default users are: admin and demo
The password: test@123
DevStack Version: rocky
OS Version: CentOS Linux Release 7.7.1908(Core)
stack.sh completed in 3000 seconds.
```

Step 7: Test Openstack Installation

Congratulations!! Openstack Installation and Configuration is completed now. You can go to below URL Address and provide username/passwd.

URL: <http://192.168.0.104/dashboard>

User: admin

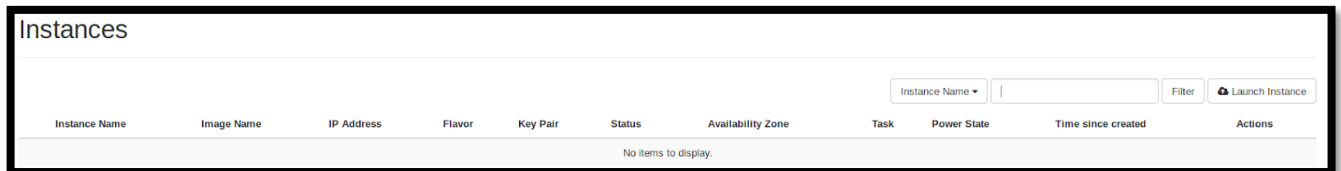
Pass: test@123

Experiment 13

13. Launch VMs in OpenStack through Dashboard

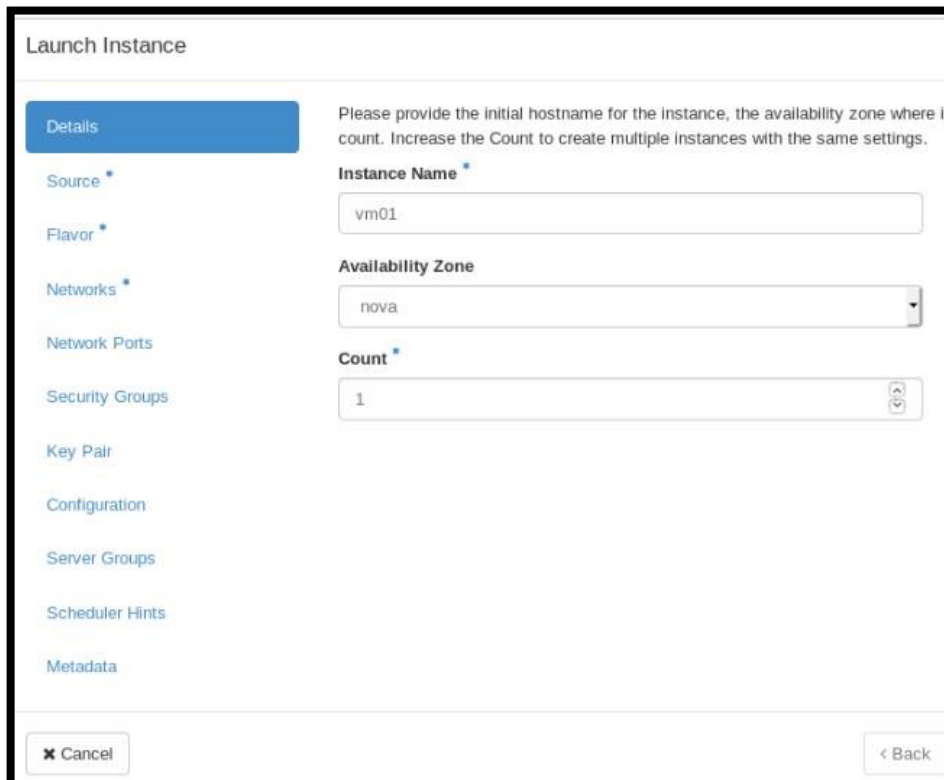
Following are the Steps to create a Linux VM in Openstack Dashboard (Horizon)

Step 1 : Go to Project → Compute → Instances



Click “Launch Instance”

Step 2 : Insert the name of the Instance (eg. "vm01") and click Next button.

The screenshot shows the 'Launch Instance' wizard. On the left is a sidebar with a 'Details' tab selected, and other tabs like Source, Flavor, Networks, Network Ports, Security Groups, Key Pair, Configuration, Server Groups, Scheduler Hints, and Metadata. The main area contains instructions: 'Please provide the initial hostname for the instance, the availability zone where it will be launched, and the count. Increase the Count to create multiple instances with the same settings.' Below this are three input fields: 'Instance Name' with the value 'vm01', 'Availability Zone' with a dropdown menu showing 'nova', and 'Count' with a value of '1'. At the bottom, there are 'Cancel' and '< Back' buttons.

Step 3 : Select Instance Boot Source (eg. "Image"), and choose desired image (eg. "Ubuntu 16.04 LTS") by clicking on arrow.

If we do not need to have the system disk bigger than the size defined in a chosen flavor, we recommend setting "Create New Volume" feature to "No" state.

Instance source is the template used to create an instance. You can use an image, a snapshot of an instance (image snapshot), a volume or a volume snapshot (if enabled). You can also choose to use persistent storage by creating a new volume. ?

Select Boot Source **Create New Volume**

Image

Allocated

Name	Updated	Size	Type	Visibility
Select an item from Available items below				

▼ Available 16 Select one

Q Click here for filters: x

Name	Updated	Size	Type	Visibility	
> Ubuntu 20.04 LTS	10/20/20 9:37 AM	2.94 GB	raw	Public	↑
> Ubuntu 18.04 LTS	10/16/20 6:21 AM	3.17 GB	raw	Public	↑
> Ubuntu 16.04 LTS	8/7/20 3:46 PM	2.69 GB	raw	Public	↑
> Ubuntu 18.04 + QGIS	8/7/20 12:36 AM	15.20 GB	raw	Public	↑

Step 4 : Choose Flavour (eg. eo1.xsmall)
xsmall).

Flavors manage the sizing for the compute, memory and storage capacity of the instance.

Allocated

Name	VCPUS	RAM	Total Disk	Root Disk	Ephemeral Disk	Public	
> eo1.xsmall	1	1 GB	8 GB	8 GB	0 GB	Yes	↓

▼ Available 22 Select one

Name	VCPUS	RAM	Total Disk	Root Disk	Ephemeral Disk	Public	
> ds.large.nvme	40	125 GB	64 GB	64 GB	0 GB	Yes	↑
> eo1.small	2	2 GB	16 GB	16 GB	0 GB	Yes	↑
> eo1.xmedium	1	2 GB	8 GB	8 GB	0 GB	Yes	↑
> eo1.medium	2	4 GB	16 GB	16 GB	0 GB	Yes	↑
> eo1.large	4	8 GB	32 GB	32 GB	0 GB	Yes	↑

Step 5 : Click “Networks” and then choose desired networks.

Networks provide the communication channels for instances in the cloud.

▼ Allocated 2 Select networks from those listed below.

Network	Subnets Associated	Shared	Admin State	Status	
1 > private_network_09064	private_subnet_09064	No	Up	Active	↓
2 > eodata	eodata	Yes	Up	Active	↓

▼ Available 0 Select at least one network

Network	Subnets Associated	Shared	Admin State	Status
No available items				

Step 6 : Open "Security Groups" After that, choose "allow_ping_ssh_rdp" and "default".

Select the security groups to launch the instance in. ?

▼ Allocated **2**

Name	Description	
> default	Default security group	↓
> allow_ping_ssh_rdp		↓

▼ Available **0** Select one or more

Name	Description
No available items	

Choose or generate SSH keypair for our VM. Next, launch your instance by clicking on blue button.

A key pair allows you to SSH into your newly created instance. You may select an existing key pair, import a key pair, or generate a new key pair. ?

Allocated

Displaying 1 item

Name	Fingerprint	
> ubuntu	7d:72:67:74:f9:ab:08:68:26:cd:73:e6:0c:8e:9b:ba	↓

Displaying 1 item

▼ Available 2 Select one

Displaying 2 items

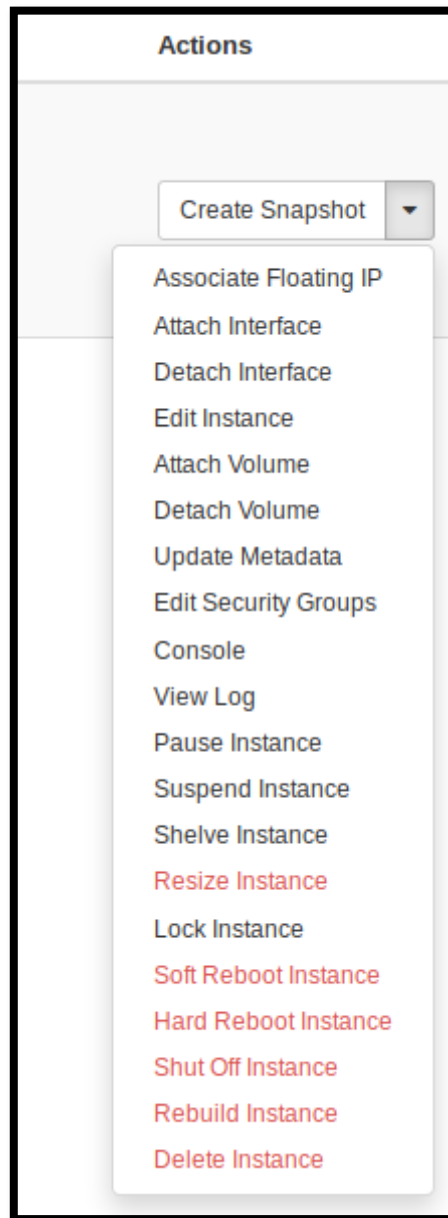
Name	Fingerprint	
> geolive	c7:9c:12:05:9f:af:3d:40:73:18:9d:11:c8:02:96:47	↑
> rh	2e:6d:48:46:d9:22:3a:b1:a4:2b:88:b0:06:41:f1:e7	↑

Displaying 2 items

You will see "Instances" menu with your newly created VM.

<input type="checkbox"/>	Instance Name	Image Name	IP Address	Flavor
<input type="checkbox"/>	vm01	Ubuntu 16.04 LTS	eodata 10.111.0.142 private_network_09064 192.168.0.8	eo1.xsmall

Open the drop-down menu and choose "Console".



Click on the black terminal area (to activate access to the console). Type: eoconsole and hit Enter.

```
Ubuntu 16.04.6 LTS vm01 tty1
vm01 login: eoconsole_
```

Insert and retype new password.

```
Ubuntu 16.04.6 LTS vm01 tty1

vm01 login: eoconsole
You are required to change your password immediately (root enforced)
Enter new UNIX password:
Retype new UNIX password:
```

Now you can type commands.

```
Ubuntu 16.04.6 LTS vm01 tty1

vm01 login: eoconsole
You are required to change your password immediately (root enforced)
Enter new UNIX password:
Retype new UNIX password:
Welcome to Ubuntu 16.04.6 LTS (GNU/Linux 4.4.0-165-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:    https://landscape.canonical.com
 * Support:       https://ubuntu.com/advantage

0 packages can be updated.
0 updates are security updates.


The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

eoconsole@vm01:~$ _
```

After finishing, type “exit”

```
eoconsole@vm01:~$ exit_
```

This will close the session.

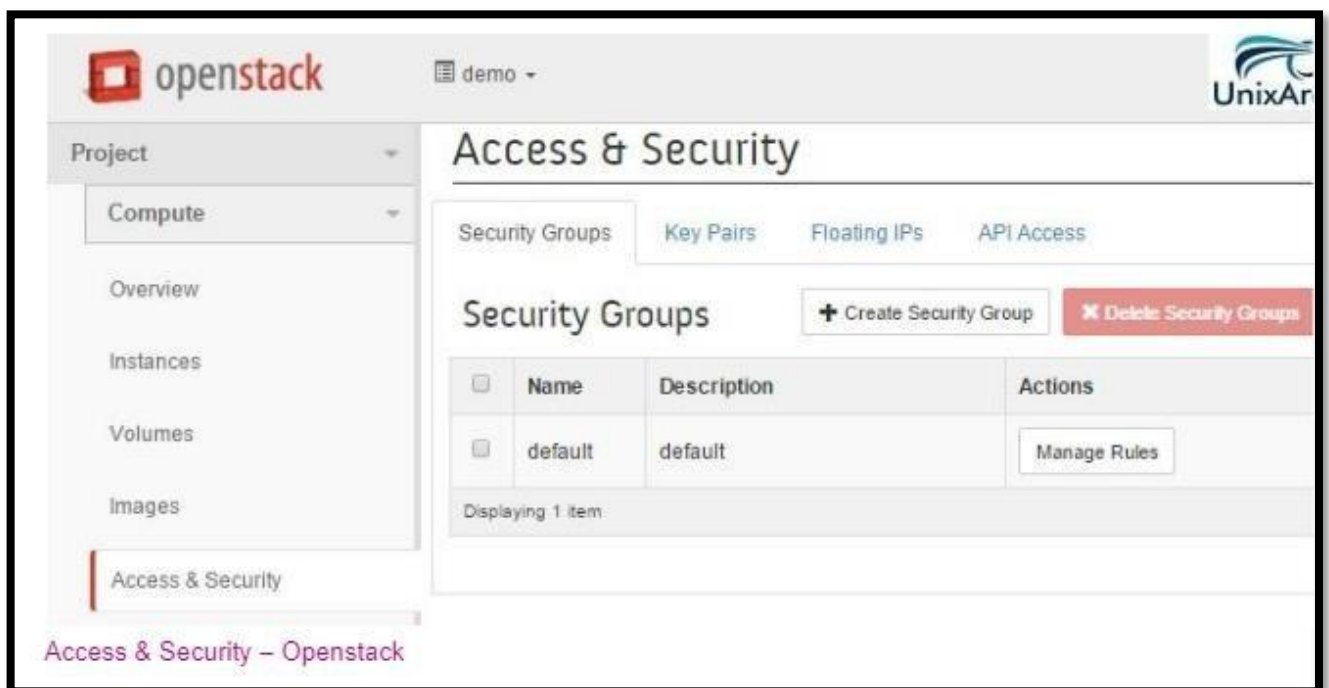
Experiment 14

14. OpenStack Dashboard should be accessed through Web Browser. Verify the working of the instance by logging or pinging the instance.

Launching the first Openstack Instance


Create the Network security group & Configure the Rules:

1. Login to Openstack Dashboard as normal user. (demo)
2. Navigate to Access & Security . Select the tab called “Security Groups”.



3. Click on “Create Security group”. Enter the name and description for the security group.

Create Security Group



Name *

UAssec1

Description *

UASECURITY

Description:

Security groups are sets of IP filter rules that are applied to the network settings for the VM. After the security group is created, you can add rules to the security group.

Cancel

Create Security Group

Create Security Group

4. Once the group has been created successfully, click on “Manage Rules”.

demo

Project

Compute

Overview

Instances

Volumes

Images

Access & Security

Orchestration

Identity

Access & Security

Security Groups

Key Pairs

Floating IPs

API Access

Security Groups

+ Create Security Group

Delete Security Groups

<input type="checkbox"/>	Name	Description	Actions
<input type="checkbox"/>	UAssec1	UASECURITY	Manage Rules
<input type="checkbox"/>	default	default	Manage Rules

Displaying 2 items



Success: Successfully created security group: UAssec1

Manage the Network Group Rules

5. Click on “Add Rule”

openstack demo

Project

Compute

Overview

Instances

Volumes

Images

Access & Security

Orchestration

Identity

Manage Security Group Rules: UAsec1

Security Group Rules

+ Add Rule

Direction	Ether Type	IP Protocol	Port Range	Remote	Actions
No items to display.					
Displaying 0 items					

UnixArena

Add Rule – Openstack

6. Allow ssh from anywhere to the instances.

Add Rule

Rule *

SSH

Remote * ?

CIDR

CIDR ?

0.0.0.0/0

Description:

Rules define which traffic is allowed to instances assigned to the security group. A security group rule consists of three main parts:

Rule: You can specify the desired rule template or use custom rules, the options are Custom TCP Rule, Custom UDP Rule, or Custom ICMP Rule.

Open Port/Port Range: For TCP and UDP rules you may choose to open either a single port or a range of ports. Selecting the "Port Range" option will provide you with space to provide both the starting and ending ports for the range. For ICMP rules you instead specify an ICMP type and code in the spaces provided.

Remote: You must specify the source of the traffic to be allowed via this rule. You may do so either in the form of an IP address block (CIDR) or via a source group (Security Group). Selecting a security group as the source will allow any other instance in that security group access to any other instance via this rule.

UnixArena

Cancel Add

Allow SSH – Openstack

7. Similarly , allow “ping” as well to this host from anywhere.

Add Rule

Rule *

All ICMP ▼

Remote * ?

CIDR ▼

CIDR ?

0.0.0.0/0

Description:

Rules define which traffic is allowed to instances assigned to the security group. A security group rule consists of three main parts:

Rule: You can specify the desired rule template or use custom rules, the options are Custom TCP Rule, Custom UDP Rule, or Custom ICMP Rule.

Open Port/Port Range: For TCP and UDP rules you may choose to open either a single port or a range of ports. Selecting the "Port Range" option will provide you with space to provide both the starting and ending ports for the range. For ICMP rules you instead specify an ICMP type and code in the spaces provided.

Remote: You must specify the source of the traffic to be allowed via this rule. You may do so either in the form of an IP address block (CIDR) or via a source group (Security Group). Selecting a security group as the source will allow any other instance in that security group access to any other instance via this rule.

Cancel

Add

Allow ICMP -Ping

8. Once you have added those rules to the security group, it will look like below.

openstack

demo ▼

Project ▼

Compute ▼

Overview

Instances

Volumes

Images

Access & Security

Manage Security Group Rules: UAssec1

Security Group Rules

+ Add Rule

✕ Delete Rules

<input type="checkbox"/>	Direction	Ether Type	IP Protocol	Port Range	Remote	Actions
<input type="checkbox"/>	Ingress	-	ICMP	-1 (All ICMP)	0.0.0.0/0 (CIDR)	<div>Delete Rule</div>
<input type="checkbox"/>	Ingress	-	TCP	22 (SSH)	0.0.0.0/0 (CIDR)	<div>Delete Rule</div>

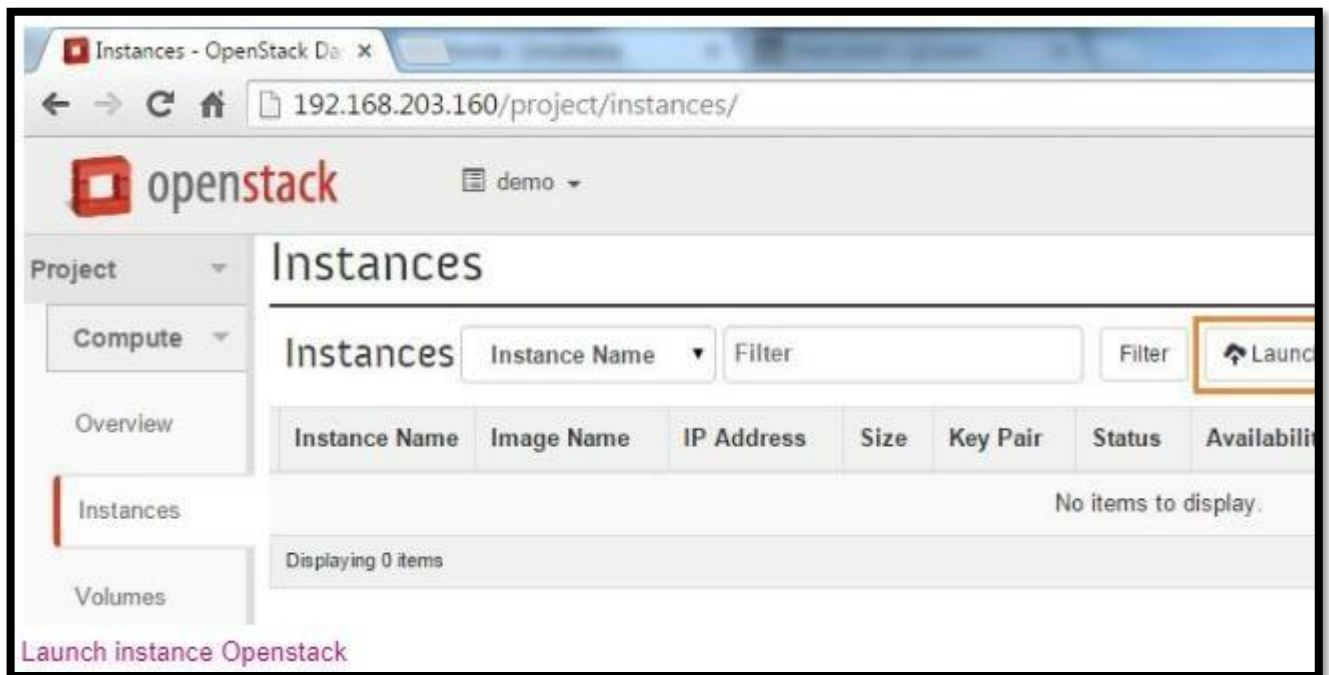
Displaying 2 items

UnixArena

Security Rules – Openstack

Launch the New Openstack Instance :

1. Login to Openstack Dashboard.
2. Click on “Launch Instance ” tab.



3. Select the instance details like below

Launch Instance

Details *
Access & Security *
Post-Creation
Advanced Options

Availability Zone
nova

Instance Name *
uainst1

Flavour * ?
m1.micro

Instance Count * ?
1

Instance Boot Source * ?
Boot from image

Image Name
cirros-0.3.2-x86_64-uec (24.0 MB)
Select Image
Fedora-x86_64-20-20140618-sda (199.9 MB)
cirros-0.3.2-x86_64-uec (24.0 MB)

Specify the details for launching an instance.
The chart below shows the resources used by this project in relation to the project's quotas.
Flavour Details

Name	m1.micro
VCPUs	1
Root Disk	0 GB
Ephemeral Disk	0 GB
Total Disk	0 GB
RAM	128 MB

Project Limits
Number of Instances 0 of 10 Used
Number of VCPUs 0 of 20 Used
Total RAM 0 of 51,200 MB Used

Cancel Launch

Enter the Instance Details

Enter the Instance Details

- Availability Zone – nova . (Need to select your compute node). In our case control node & compute nodes are same.
- Instance Name – Enter the desired instance name
- Flavour – Select the available flavour according to your need. (See the details in right side)
- Instance Count – Enter the instance Count
- Boot Source – Select boot from pre-defined image.
- Image Name – select “cirros” since its very small Linux foot print for testing openstack.

4. Click on Access & security tab for the instance. From the drop down box , select the key pair “UAPAIR” which we have created earlier. Also select the security group which we have created. Click “Launch” to launch the new instance.

Launch Instance

Details * Access & Security * Post-Creation Advanced Options

Key Pair ⓘ

UAPAIR1 +

Security Groups * ⓘ

☒ UAsec1

☐ default

Control access to your instance via key pairs, security groups, and other mechanisms.

Cancel **Launch**

UnixArena

Select the security group & Key Pair

5. Here you can see that instance has been launched. It will take few minutes to boot the instance depends on the image size which we have selected.

Instances

Launch Instance Soft Reboot Instances Terminate Instances

	Instance Name	Image Name	IP Address	Size	Key Pair	Status	Availability Zone	Task	Power State	Time since created	Actions
<input type="checkbox"/>	uainst1	cirros-0.3.2-x86_64-uec		m1.micro	-	Build	nova	Spawning	No State	0 minutes	Associate Floating IP

Displaying 1 item

Success: Launched instance named "uainst1".

Openstack Instance Launched

6. Once the instance is completely up , you can see the screen like below.

Instances

Launch Instance Soft Reboot Instances Terminate Instances

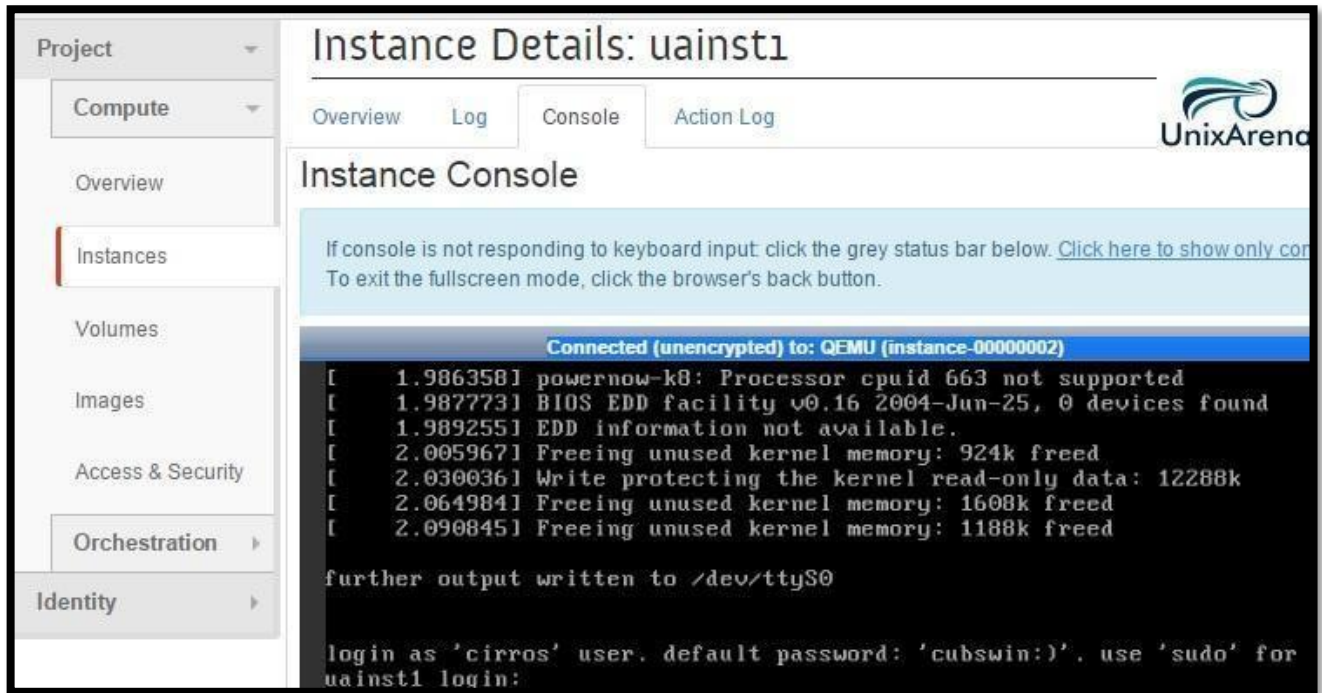
	Instance	Image Name	IP Address	Size	Key Pair	Status	Availab	Task	Power State	Time since	Actions
<input type="checkbox"/>	uainst1	cirros-0.3.2-x86_64-uec	192.168.204.2	m1.micro	UAPAIR1	Active	nova	None	Running	0 minutes	Create Snapshot

Displaying 1 item

Openstack Instance is up

In the IP address tab , you can get the private IP address for the instance. Using this IP , You should be able to access the instance.

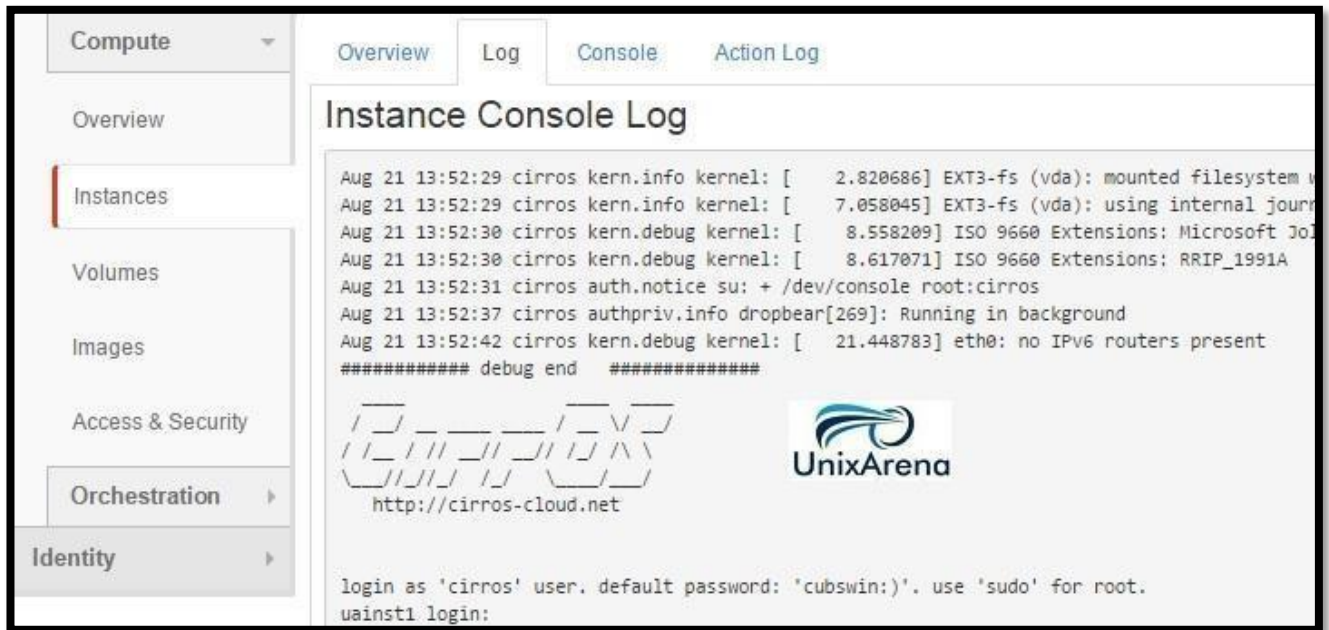
7. If you would like to see the instance console , click the instance name and select the console tab. You should be able to access the instance here as well by double clicking the console bar.



Instance Console

In Openstack's kilo branch, console may not load properly if you didn't add the below parameter in the local.conf file during the installation.
"enable_service n-cauth"

8. We can also check the log to know the instance is booted or not . (If console is not working due to above mentioned issue).



openstack instance log

We should be able to access the instance within the private IP range (if we didn't allocate the floating IP). We can access the instance from control node.

```
stack@uacloud:~$ ssh cirros@192.168.204.2
```

```
cirros@192.168.204.2's password:
```

```
$
```

```
$ sudo su -
```

```
# ifconfig -a
```

```
eth0    Link encap:Ethernet  HWaddr FA:16:3E:A6:81:BE
```

```
        inet addr:192.168.204.2  Bcast:192.168.204.255  Mask:255.255.255.0
```

```
        inet6 addr: fe80::f816:3eff:fea6:81be/64 Scope:Link
```

```
        UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
```

```
        RX packets:114 errors:0 dropped:0 overruns:0 frame:0
```

```
        TX packets:72 errors:0 dropped:0 overruns:0 carrier:0
```

```
        collisions:0 txqueuelen:1000
```

```
        RX bytes:14089 (13.7 KiB)  TX bytes:8776 (8.5 KiB)
```

```
lo      Link encap:Local Loopback
```

```
        inet addr:127.0.0.1  Mask:255.0.0.0
```

inet6 addr: ::1/128 Scope:Host

UP LOOPBACK RUNNING MTU:16436 Metric:1

RX packets:0 errors:0 dropped:0 overruns:0 frame:0

TX packets:0 errors:0 dropped:0 overruns:0 carrier:0

collisions:0 txqueuelen:0

RX bytes:0 (0.0 B) TX bytes:0 (0.0 B)

route

Kernel IP routing table

Destination	Gateway	Genmask	Flags	Metric	Ref	Use	Iface
default	192.168.204.3	0.0.0.0	UG	0	0	0	eth0
192.168.204.0	*	255.255.255.0	U	0	0	0	eth0