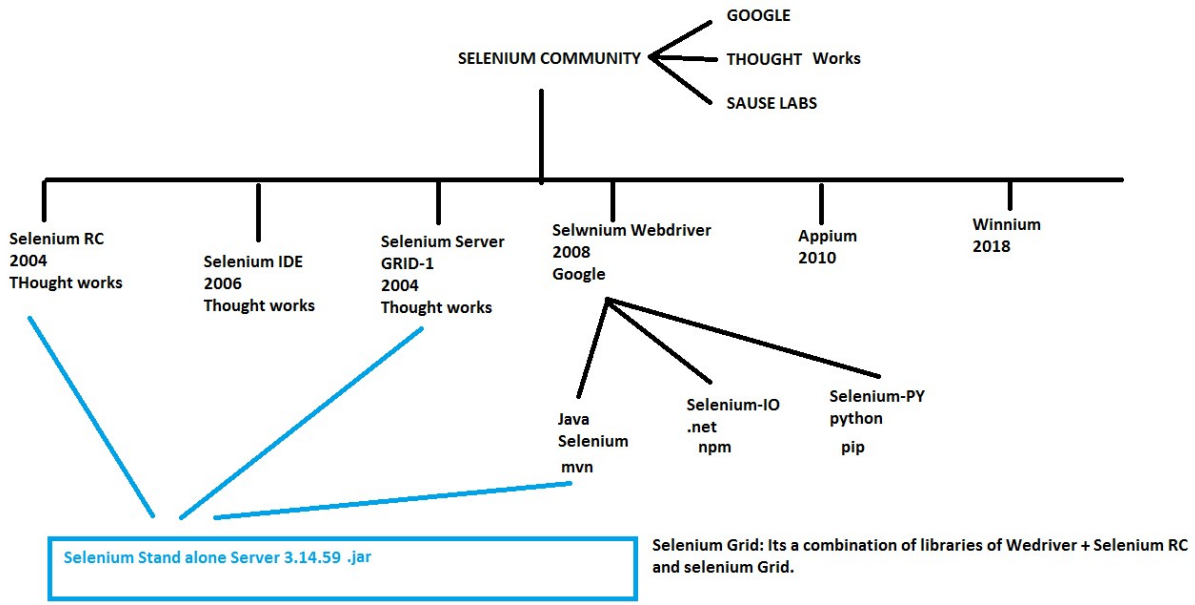
Selenium Grid

Selenium grid is a collection of libraries from Selenium RC + Selenium Webdriver + Selenium Grid Server



Selenium Grid: It's a open source available in Selenium community and it acts like a server for remote execution and Compatability testing.

Selenium grid is used to perform

- Remote Execution
- Cross Browser testing
- Cross Platform Testing

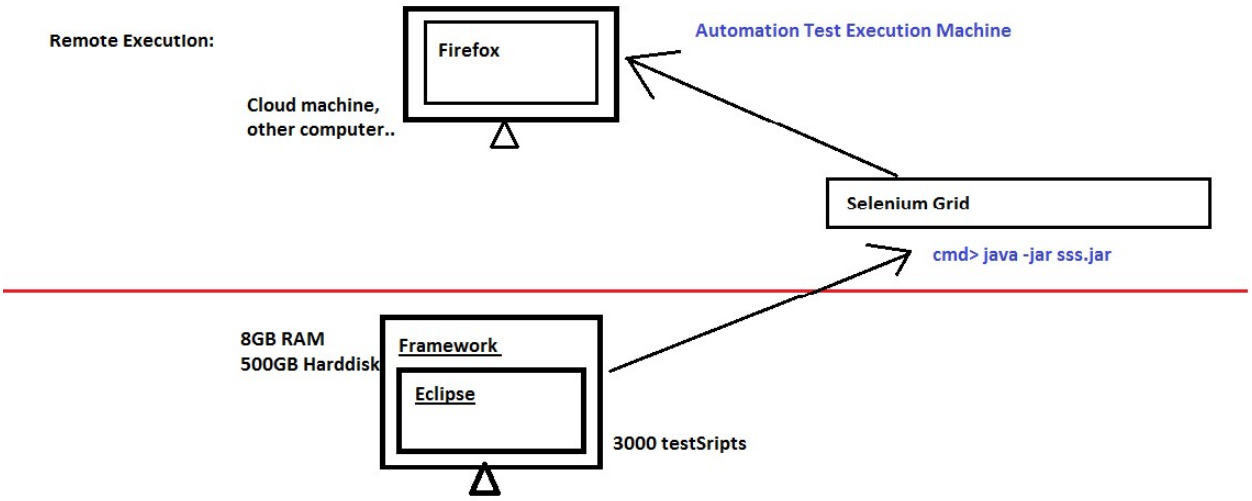
REMOTE EXECUTION:executing the test script in any remote devices like cloud machines, other computers in the same network, mobile devices like android or IOS with the help of selenium grid is called as Remote Execution.

In order to perform Remote Execution we make use of:

1. Remote WebDriver:- It's a subclass of webdriver which implements webdriver interface, its helps to achieve remote execution using selenium grid. It is present in `org.openqa.selenium.remote.RemoteWebDriver`

2. DesiredCapabilities:- It's a class used to set or change the capabilities of webdriver. In remote execution we have to set the capabilities for browser name, browser version and platform where the test script has to be executed. It is present in `org.openqa.selenium.remote.DesiredCapabilities`

3. URL : It's a class in java which helps to store the remote address for remote execution, present in `java.net.URL`



How to Run Selenium grid?

1. Download Selenium stand-alone-server-3.141.59.jar from the below link https://www.selenium.dev/downloads
2. Go to command prompt and execute the below command
   Java –jar selenium-stand-alone-server.jar

```
C:\Users\Sreenivas>java -jar C:\Users\Sreenivas\Desktop\selenium-server-standalo
ne-3.141.59.jar
```

Default Port number of Selenium grid is **4444** and can be changed if required.

```
@Test
publicvoid gridPractice() throws MalformedURLException
{

      URL url = new URL("http://localhost:4444");
      DesiredCapabilities cap = new DesiredCapabilities();
      cap.setBrowserName("chrome");
      cap.setPlatform(Platform.WINDOWS);
      RemoteWebDriver driver = new RemoteWebDriver(url, cap);
      driver.get("http://gmail.com");

}
```

HUB and NODE:

In selenium grid, concept of hub and node helps to configure more than one node for a hub.

HUB acts as the master which will receive the command from selenium and bypass that to nodes. Default port number is 4444, if its busy, port number can be changed like below

```
C:\Users\Sreenivas>java -jar C:\Users\Sreenivas\Desktop\selenium-server-standalo
ne-3.141.59.jar -port 5555 -role hub
```

```
18:18:25.934 INFO [Hub.start] - Selenium Grid hub is up and running
18:18:25.934 INFO [Hub.start] - Nodes should register to http://192.168.43.103:5
555/grid/register/
18:18:25.944 INFO [Hub.start] - Clients should connect to http://192.168.43.103:
5555/wd/hub
```

NODE acts like slave which will receive the command from master/Hub and act execute the request. Maximum 5 nodes can be connected to 1 hub.

```
C:\Users\Sreenivas>java -Dewebdriver.chrome.driver=C:\Users\Sreenivas\Desktop\ch
romedriver.exe -jar C:\Users\Sreenivas\Desktop\selenium-server-standalone-3.141.
59.jar -role node -port 6666 -hub http://192.168.43.103:5555/grid/register
```

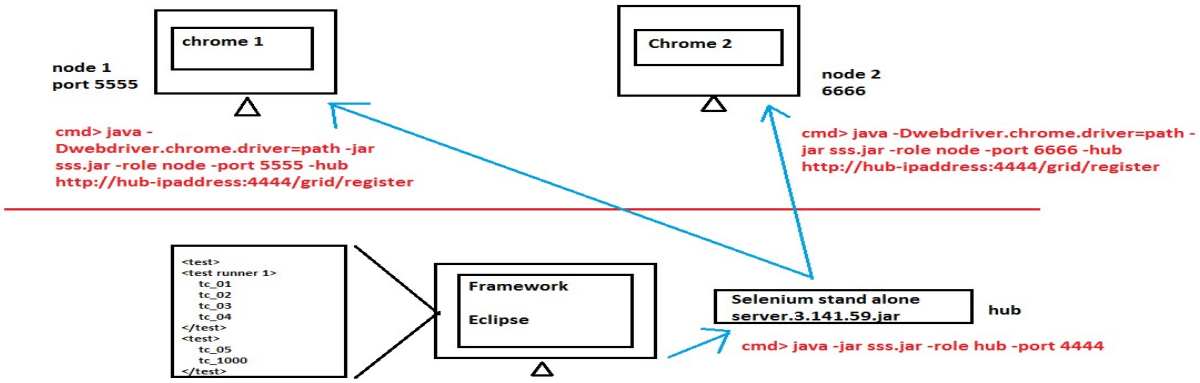
```
18:22:47.487 INFO [SelfRegisteringRemote.registerToHub] - Registering the node t
o the hub: http://192.168.43.103:5555/grid/register
18:22:48.027 INFO [SelfRegisteringRemote.registerToHub] - The node is registered
 to the hub and ready to use
```

Node registration shown in Hub:

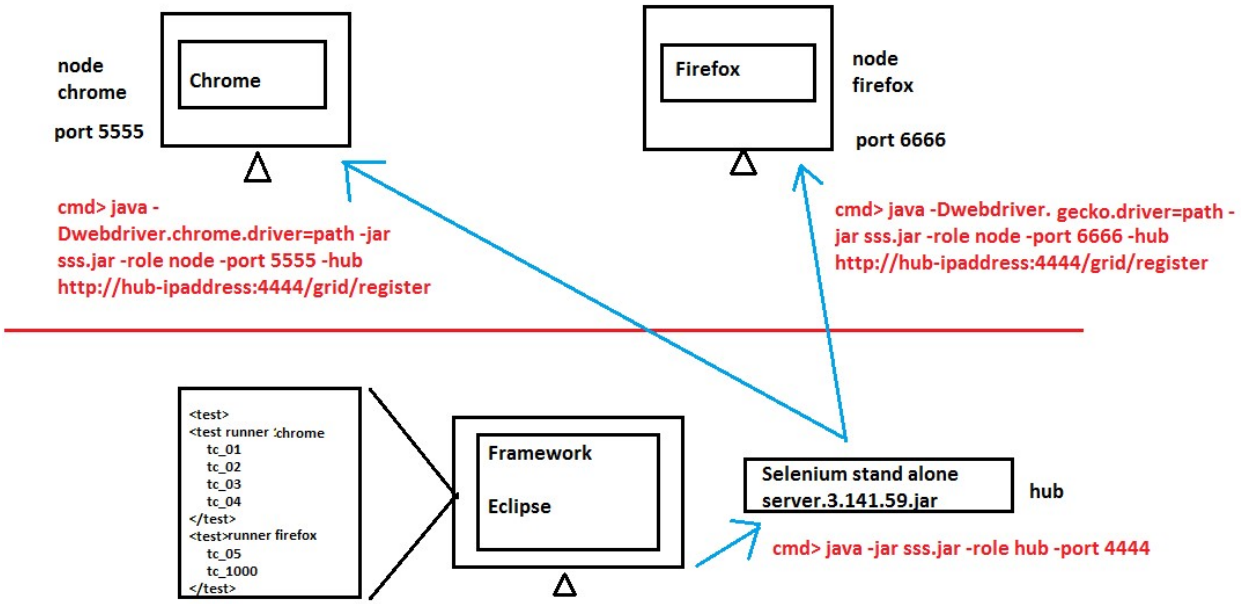
```
18:22:48.027 INFO [DefaultGridRegistry.add] - Registered a node http://192.168.4
3.103:6666
```

This configuration helps in parallel execution and cross browser parallel execution where the hub is configured in local system and nodes for various capabilities is configured in remote machines and the execution happens in nodes via hub.

PARALLEL EXECUTION:



CROSS BROWSER TESTING/COMPATABILITY TESTING

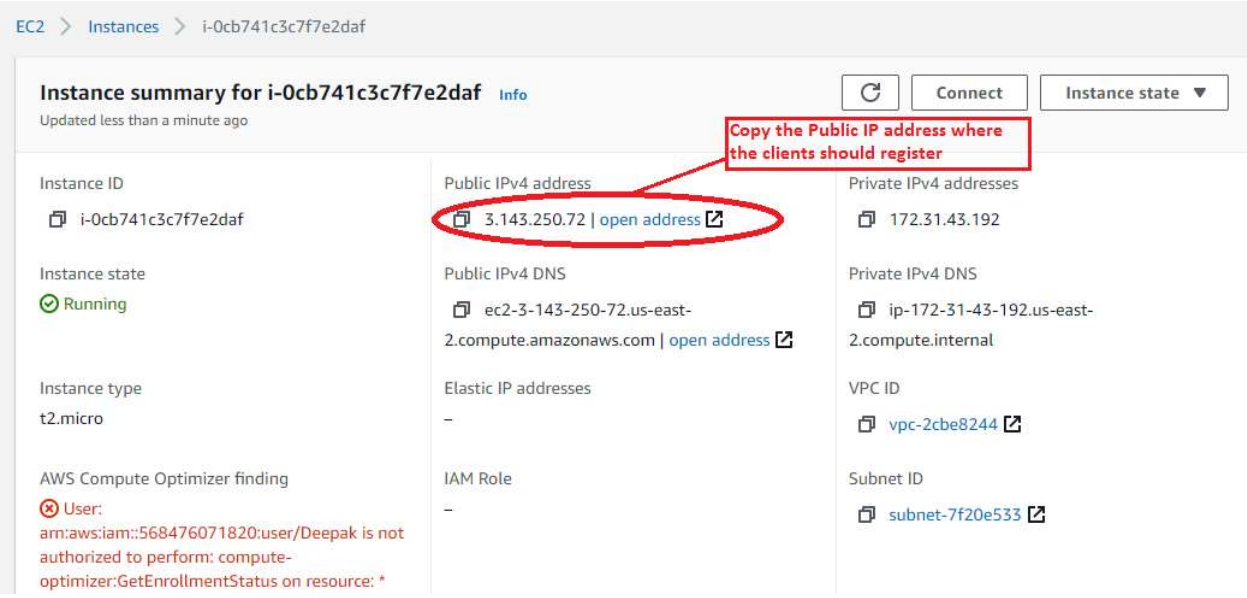



```
publicclass RemoteExcecution {

RemoteWebDriver driver;
@Parameters("Browser")
@BeforeClass
publicvoid configBeforeClass(String BROWSER) throws
MalformedURLException {
            URL url = new URL("http://192.168.1.100:4444/wd/hub");
             DesiredCapabilities cap= new DesiredCapabilities();

           if(BROWSER.equals("chrome")) {
                 cap.setPlatform(Platform.WINDOWS);
                 cap.setBrowserName("chrome");
           }elseif(BROWSER.equals("firefox")){
                 cap.setPlatform(Platform.WINDOWS);
                 cap.setBrowserName("firefox");
           }
           driver = new RemoteWebDriver(url, cap);
      }

@Test
publicvoid remotteExcecution() throws MalformedURLException
{
           driver.get("http://gmail.com");
}
```

Remote Execution using AWS cloud EC2 machine

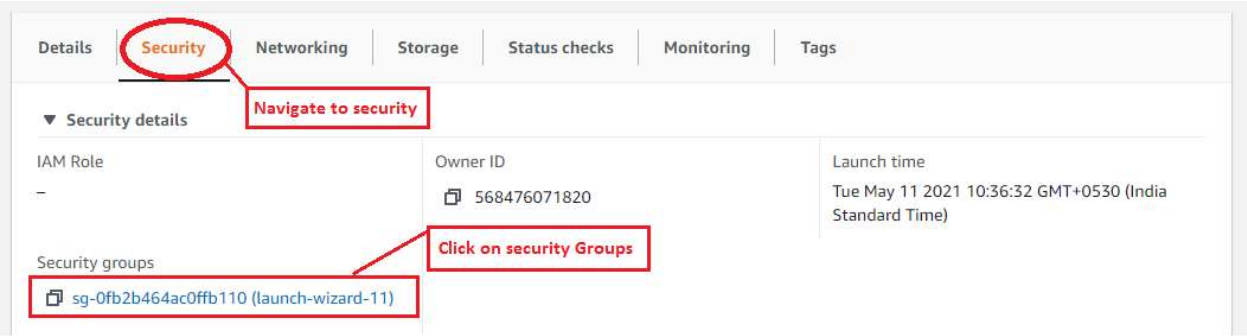
Create EC2 instance for windows using launch instance option

Launch the instance using all default status

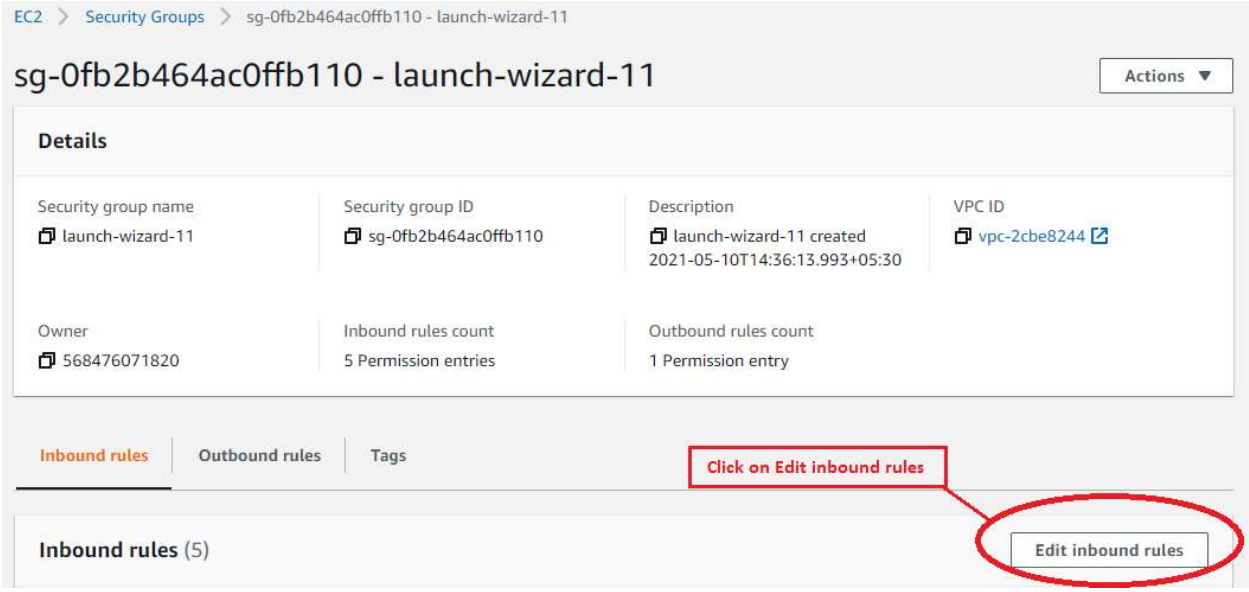


Since the VM has to be accessed through external system, we have to set the inbound rules to enable RDP client to access the Virtual Machine. Follow the procedure to set inbound rules:

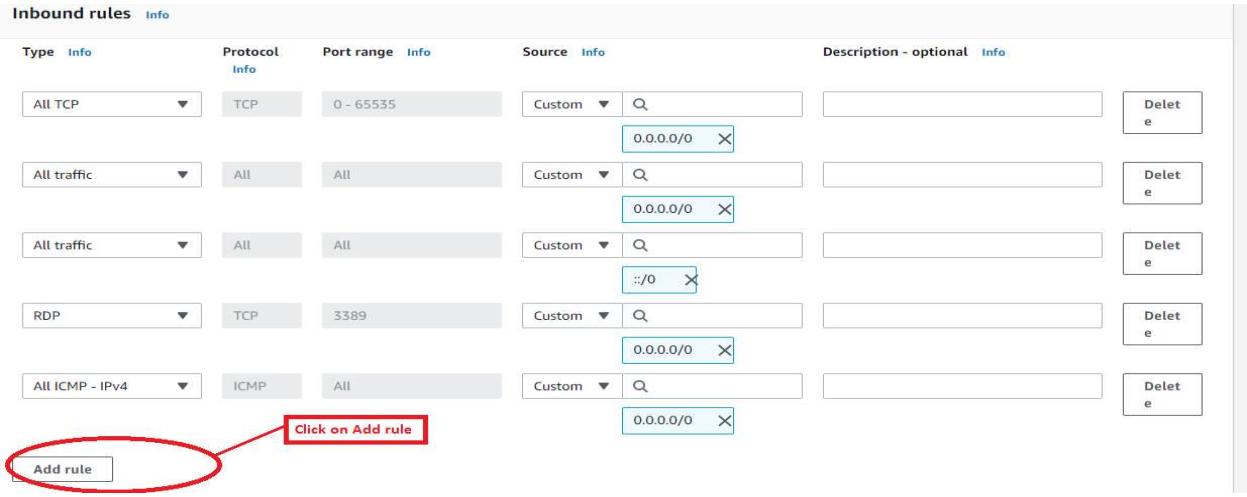
1. Scroll down to Security



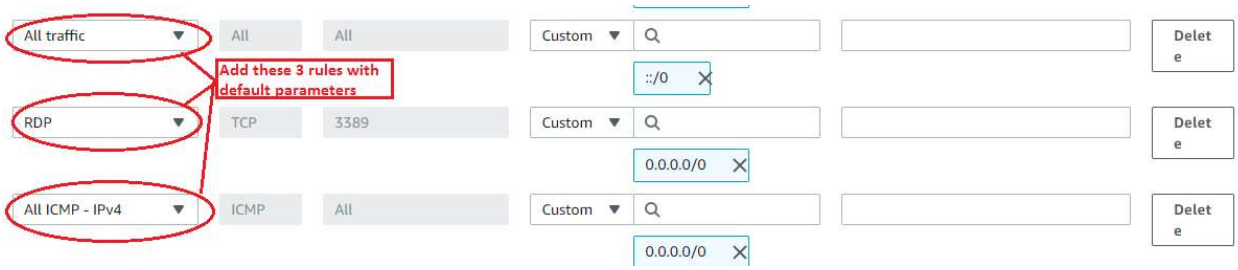
2. Click on the security groups and click on Edit inbound rules



3. Initially only All TCP rule will be present,

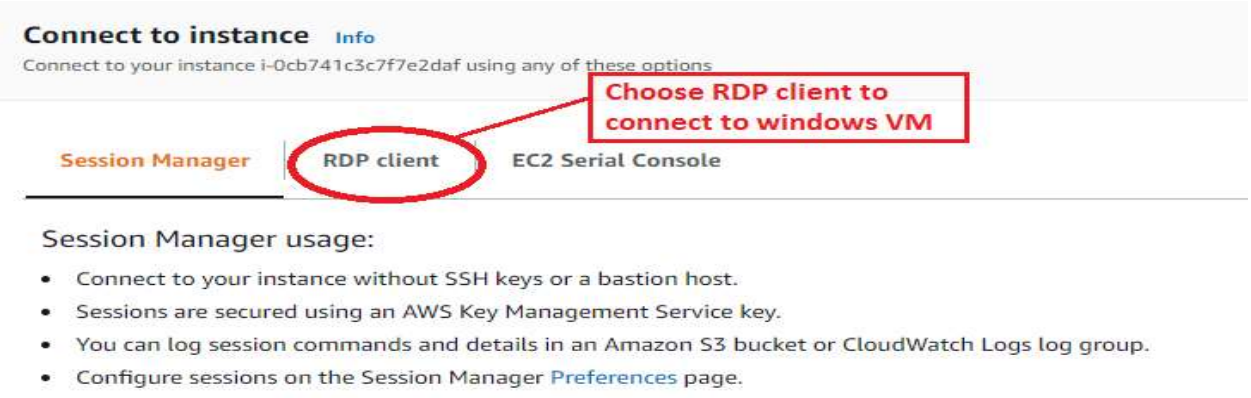


4. Hit on add rule and add the following rules



5. Once the inbound rules are set, we will have to connect to the virtual machine using Remote desktop Connection. Navigate back to the instance and hit on Connect.

6. Choose RDP client for connecting the windows VM



7. Download the Remote desktop file, which will be named after the public IP

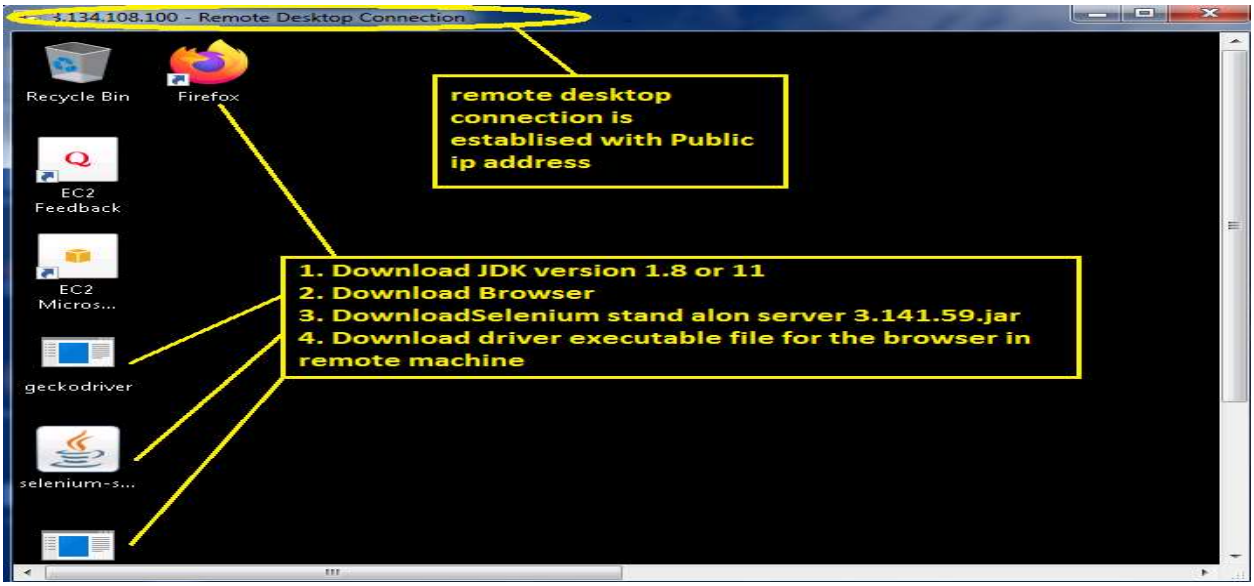
8. Hit on connect to establish the connection with Windows VM

9. Enter the password decrypted during instance launch to login to the VM

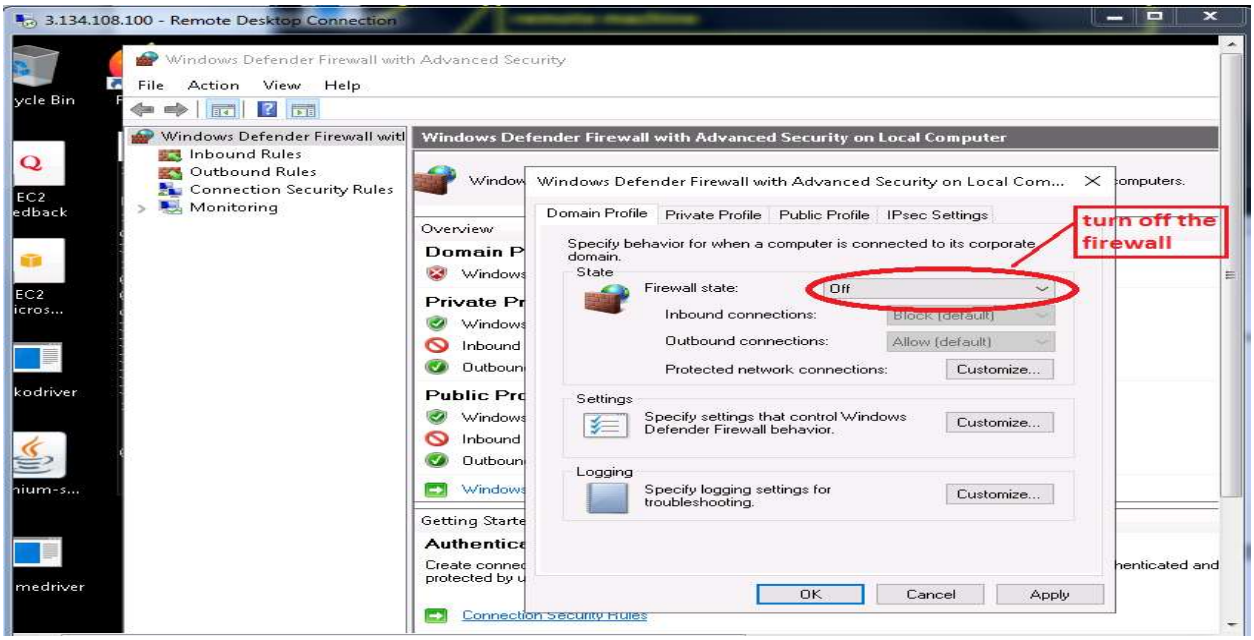
10. Remote desktop connection is established and a windows virtual computer is almost done, download all the pre-requisites to run the selenium server

Pre-Requisites for remote execution: remote machine should have:

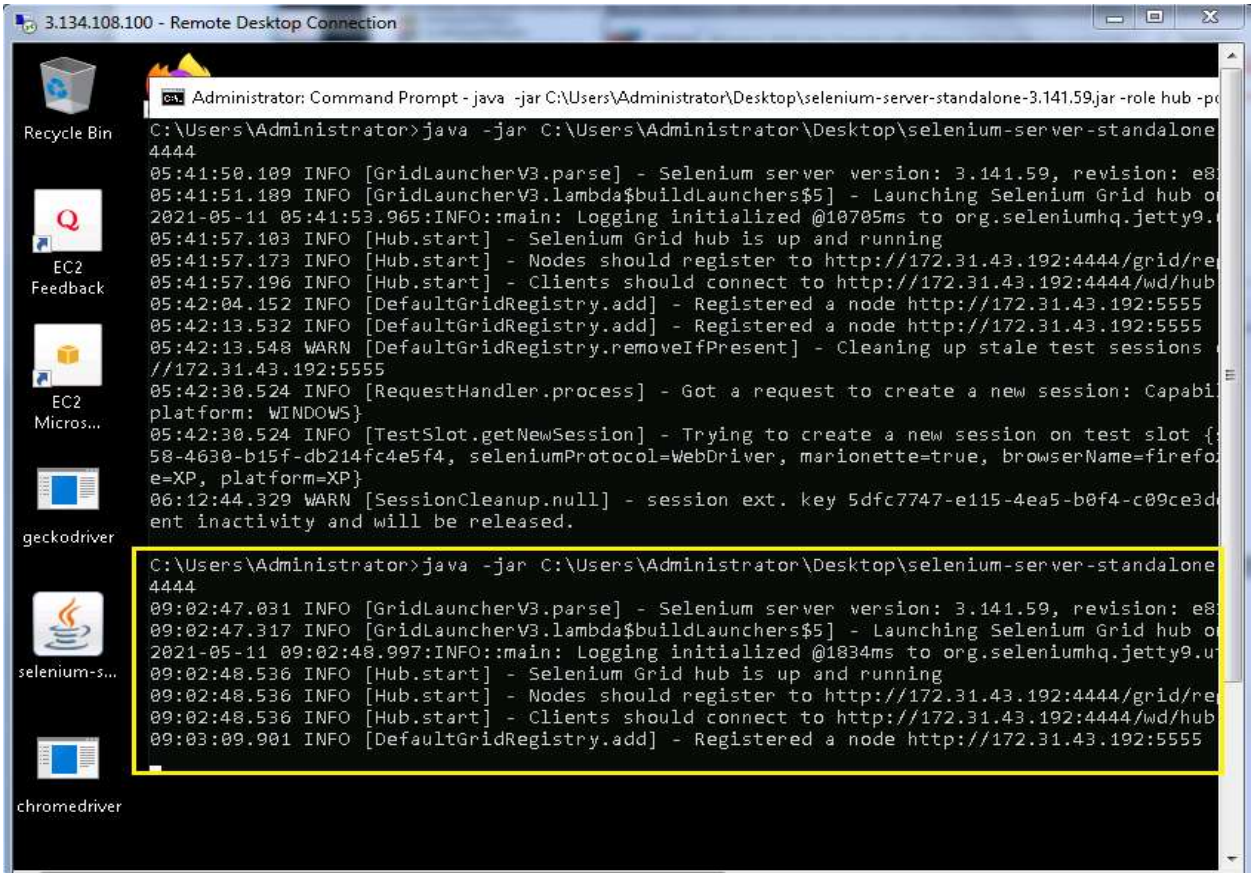
1. Selenium stand-alone-server.3.141.59.jar
2. Driver executable files for necessary browser
3. JDK
4. Browser



11. Navigate to windows defender and disable the firewall in order to allow access to Remote desktop connection

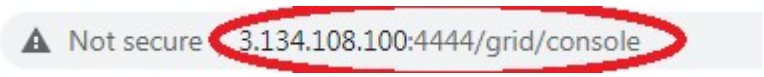


Run the Selenium server in hub mode in remote machine

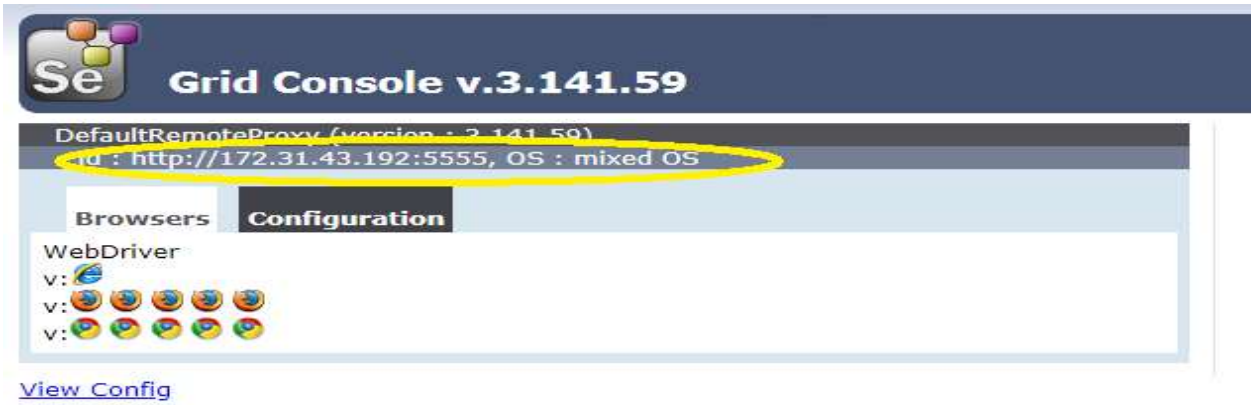


Navigate back to local machine, open any browser and enter

public-ip-address:port-number-of-hub/grid/console



If the selenium Grid console is displayed then the configuration of remote machine is successful.

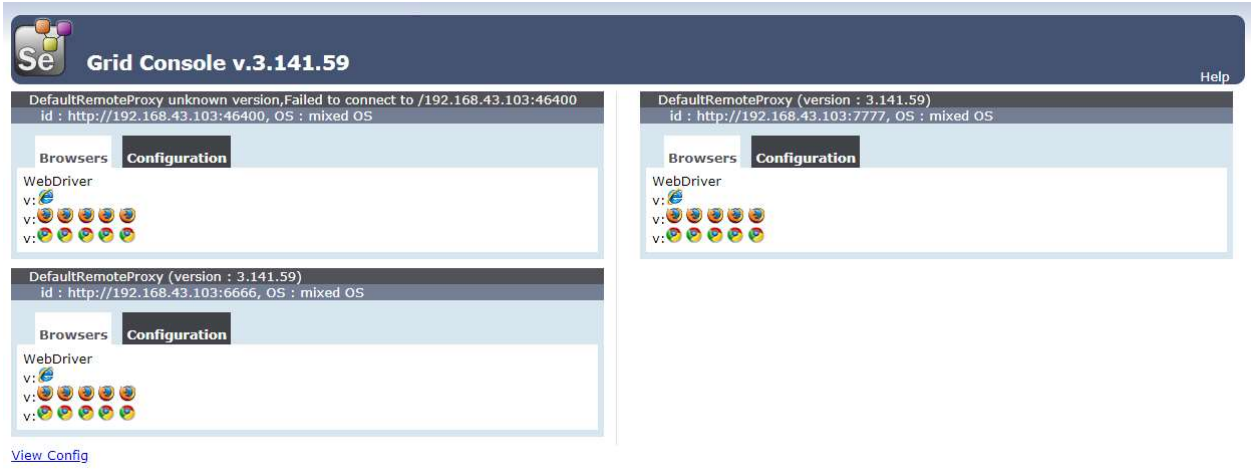


SELENIUM GRID CONSOLE:

Once the selenium grid is configured, go to any browser and ping ipaddress/grid/console to verify the number of nodes connected, browsers launched and the port numbers of nodes connected.

http://localhost:4444/grid/console

Config for the hub :
browserTimeout : 0
debug : false
host : 192.168.43.103
port : 5555
role : hub
timeout : 1800
cleanUpCycle : 5000
capabilityMatcher : org.openqa.grid.internal.utils.DefaultCapabilityMatcher
newSessionWaitTimeout : -1
throwOnCapabilityNotPresent : true
registry : org.openqa.grid.internal.DefaultGridRegistry



Resume Points on Selenium Grid

1. Involved in Remote Execution using Selenium Grid
2. Experienced in configuring Hub and node for Compatibility Testing
3. Experienced in performing cross browser execution using multiple machines
4. Experienced in configuring AWS machines for remote Execution
5. Involved AWS tool configuration