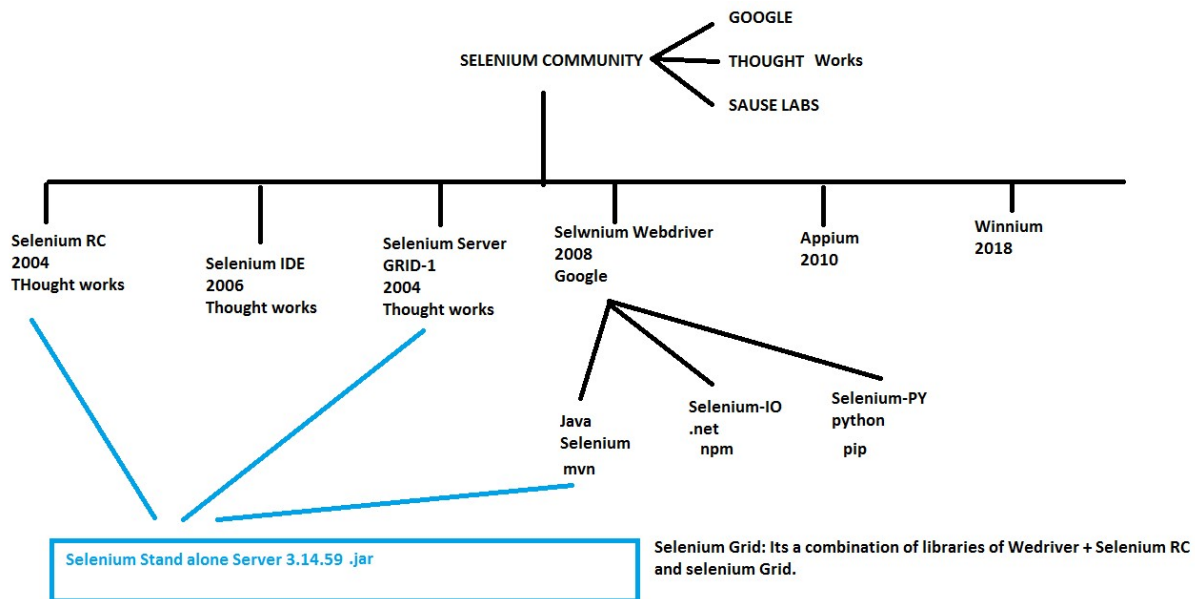


Selenium Grid

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 Compatibility testing.

Selenium grid is used to perform

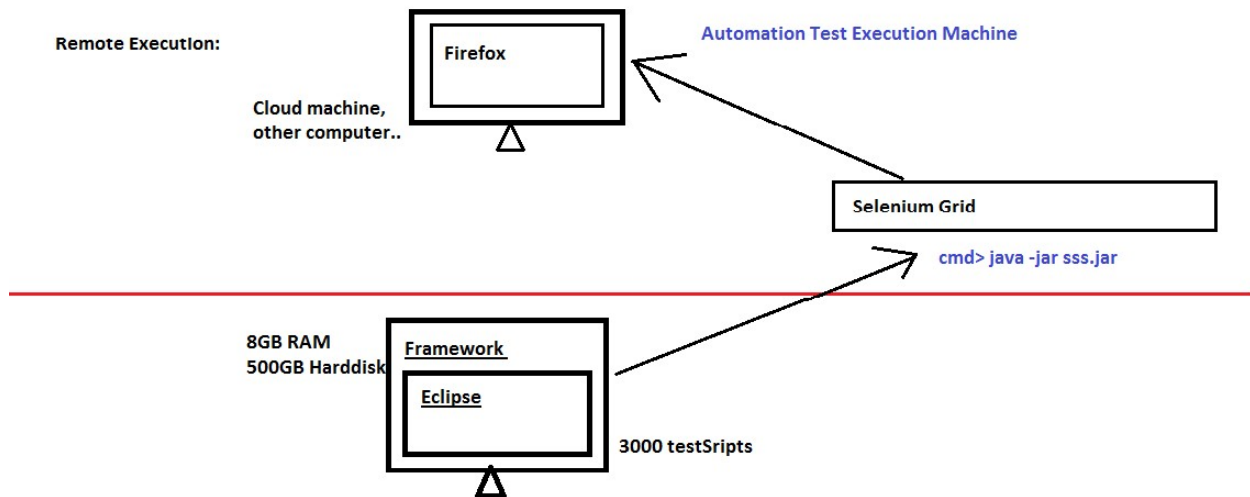
- Remote Execution
- Cross Browser testing
- Cross Platform Testing

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-standalone-3.141.59.jar
```

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

@Test

```
public void 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-standalone-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://192.168.43.103:5555/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 -Dwebdriver.chrome.driver=C:\Users\Sreenivas\Desktop\chromedriver.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
```

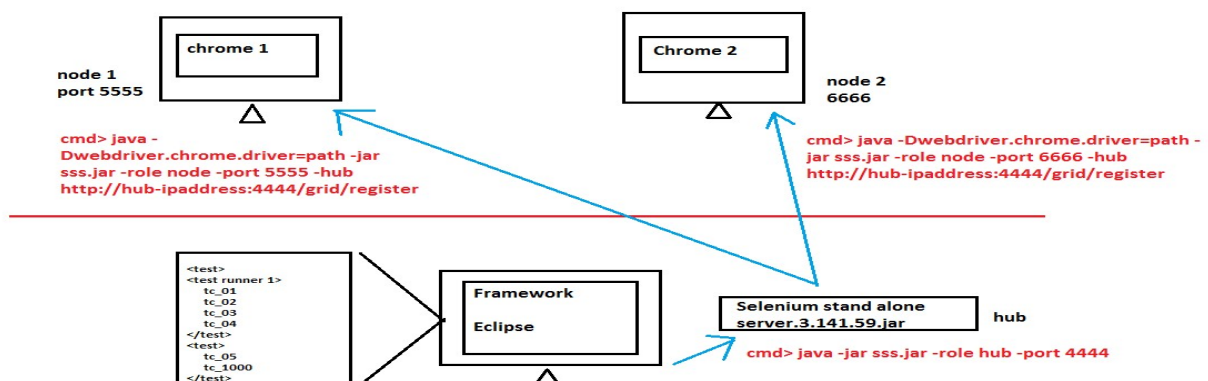
```
18:22:47.487 INFO [SelfRegisteringRemote.registerToHub] - Registering the node to the hub: http://192.168.43.103:5555/grid/register
18:22:48.027 INFO [SelfRegisteringRemote.registerToHub] - The node is registered to the hub and ready to use
```

Node registration shown in Hub:

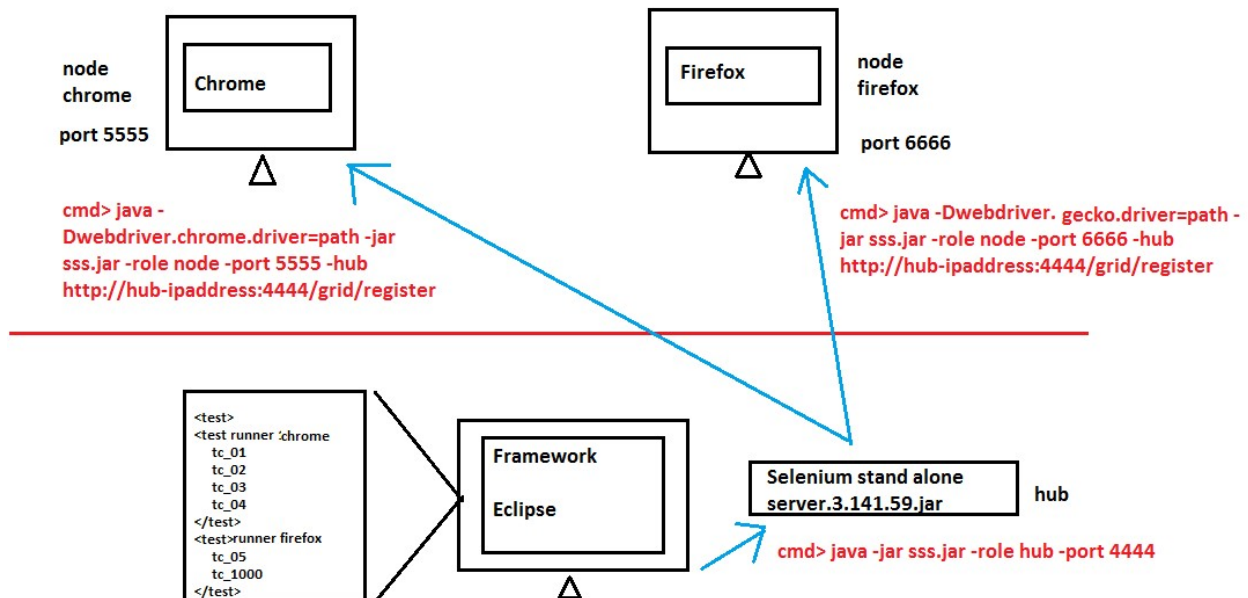
```
18:22:48.027 INFO [DefaultGridRegistry.add] - Registered a node http://192.168.43.103:6666
```

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



```
public class RemoteExecution {
```

```
    RemoteWebDriver driver;
```

```
    @Parameters("Browser")
```

```
    @BeforeClass
```

```
    public void 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
```

```
public void remotteExecution() 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

EC2 > Instances > i-0cb741c3c7f7e2daf

Instance summary for i-0cb741c3c7f7e2daf Info

Updated less than a minute ago

Instance ID: i-0cb741c3c7f7e2daf

Instance state: **Running**

Instance type: t2.micro

AWS Compute Optimizer finding: **User:** am:aws:iam::568476071820:user/Deepak is not authorized to perform: compute-optimizer:GetEnrollmentStatus on resource: *

Public IPv4 address: 3.143.250.72 | open address

Public IPv4 DNS: ec2-3-143-250-72.us-east-2.compute.amazonaws.com | open address

Elastic IP addresses: -

IAM Role: -

Private IPv4 addresses: 172.31.43.192

Private IPv4 DNS: ip-172-31-43-192.us-east-2.compute.internal

VPC ID: vpc-2cbe8244

Subnet ID: subnet-7f20e533

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

Details **Security** Networking Storage Status checks Monitoring Tags

▼ Security details

IAM Role: -

Owner ID: 568476071820

Launch time: Tue May 11 2021 10:36:32 GMT+0530 (India Standard Time)

Security groups: sg-0fb2b464ac0ffb110 (launch-wizard-11)

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

EC2 > Security Groups > sg-0fb2b464ac0ffb110 - launch-wizard-11

sg-0fb2b464ac0ffb110 - launch-wizard-11

Actions ▾

Details

Security group name launch-wizard-11	Security group ID sg-0fb2b464ac0ffb110	Description launch-wizard-11 created 2021-05-10T14:36:13.993+05:30	VPC ID vpc-2cbe8244
Owner 568476071820	Inbound rules count 5 Permission entries	Outbound rules count 1 Permission entry	

Inbound rules | Outbound rules | Tags

Click on Edit inbound rules

Inbound rules (5)

Edit inbound rules

3. Initially only All TCP rule will be present,

Inbound rules Info

Type Info	Protocol Info	Port range Info	Source Info	Description - optional Info	
All TCP ▾	TCP	0 - 65535	Custom ▾ Q 0.0.0.0/0 ✕		Delete
All traffic ▾	All	All	Custom ▾ Q 0.0.0.0/0 ✕		Delete
All traffic ▾	All	All	Custom ▾ Q ::/0 ✕		Delete
RDP ▾	TCP	3389	Custom ▾ Q 0.0.0.0/0 ✕		Delete
All ICMP - IPv4 ▾	ICMP	All	Custom ▾ Q 0.0.0.0/0 ✕		Delete

Click on Add rule

Add rule

4. Hit on add rule and add the following rules

Add these 3 rules with default parameters

All traffic ▾	All	All	Custom ▾ Q ::/0 ✕		Delete
RDP ▾	TCP	3389	Custom ▾ Q 0.0.0.0/0 ✕		Delete
All ICMP - IPv4 ▾	ICMP	All	Custom ▾ Q 0.0.0.0/0 ✕		Delete

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.

EC2 > Instances > i-0cb741c3c7f7e2daf

Instance summary for i-0cb741c3c7f7e2daf Info

Updated less than a minute ago

Click on connect to connect to VM

Instance ID i-0cb741c3c7f7e2daf	Public IPv4 address 3.134.108.100 open address	Private IPv4 addresses 172.31.43.192
Instance state Running	Public IPv4 DNS ec2-3-134-108-100.us-east-2.compute.amazonaws.com open address	Private IPv4 DNS ip-172-31-43-192.us-east-2.compute.internal
Instance type	Elastic IP addresses	VPC ID

6. Choose RDP client for connecting the windows VM

Connect to instance Info

Connect to your instance i-0cb741c3c7f7e2daf using any of these options

Choose RDP client to connect to windows VM

Session Manager | **RDP client** | EC2 Serial Console

Session Manager usage:

- Connect to your instance without SSH keys or a bastion host.
- Sessions are secured using an AWS Key Management Service key.
- You can log session commands and details in an Amazon S3 bucket or CloudWatch Logs log group.
- Configure sessions on the Session Manager [Preferences](#) page.

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

Connect to instance Info

Connect to your instance i-0cb741c3c7f7e2daf using any of these options

Session Manager | **RDP client** | EC2 Serial Console

You can connect to your Windows instance using a remote desktop client of your choice, and by downloading and running the RDP shortcut file below:

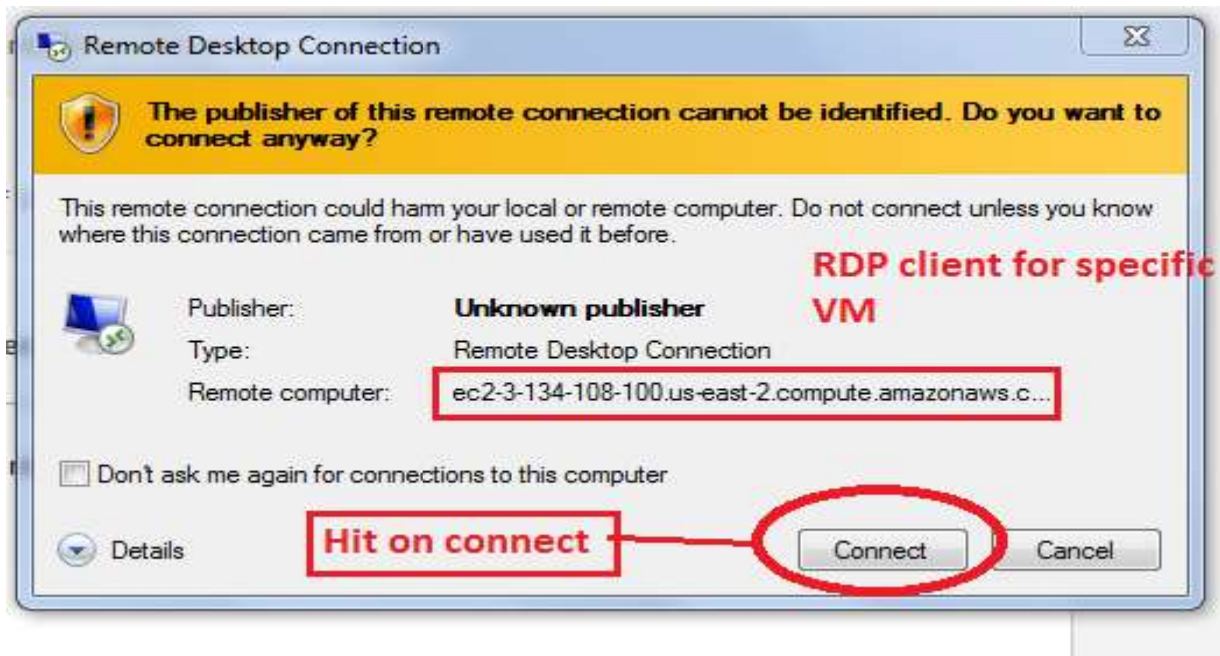
click on download RDP file

Download remote desktop file

When prompted, connect to your instance using the following details:

Public DNS ec2-3-134-108-100.us-east-2.compute.amazonaws.com	User name Administrator
---	----------------------------

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



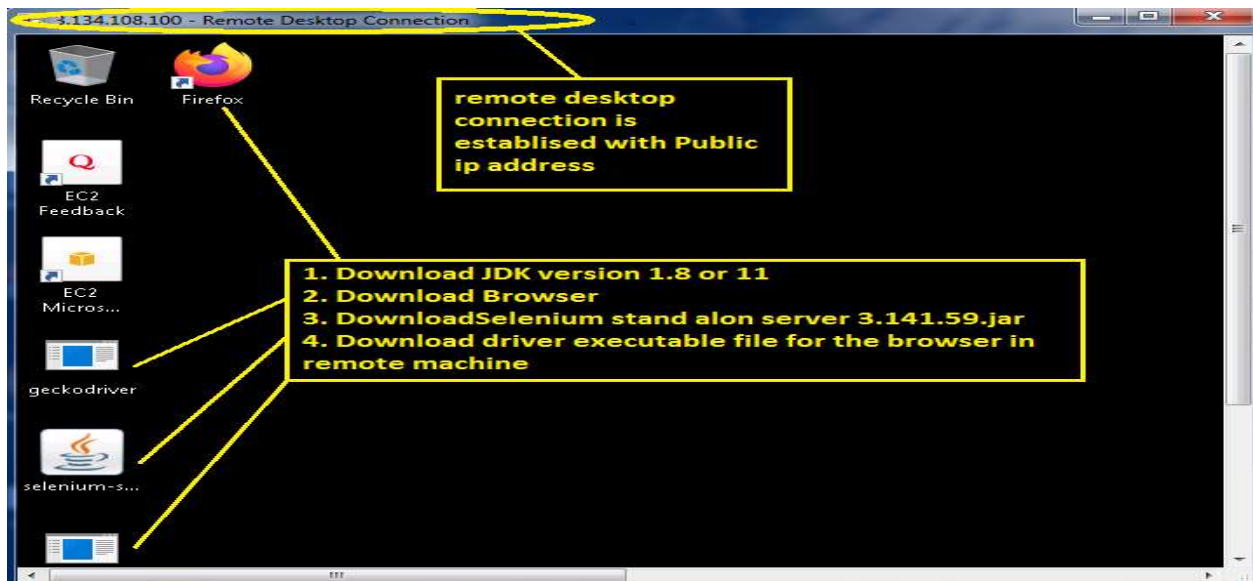
9. Enter the password decrypted during instance launch to login to the VM



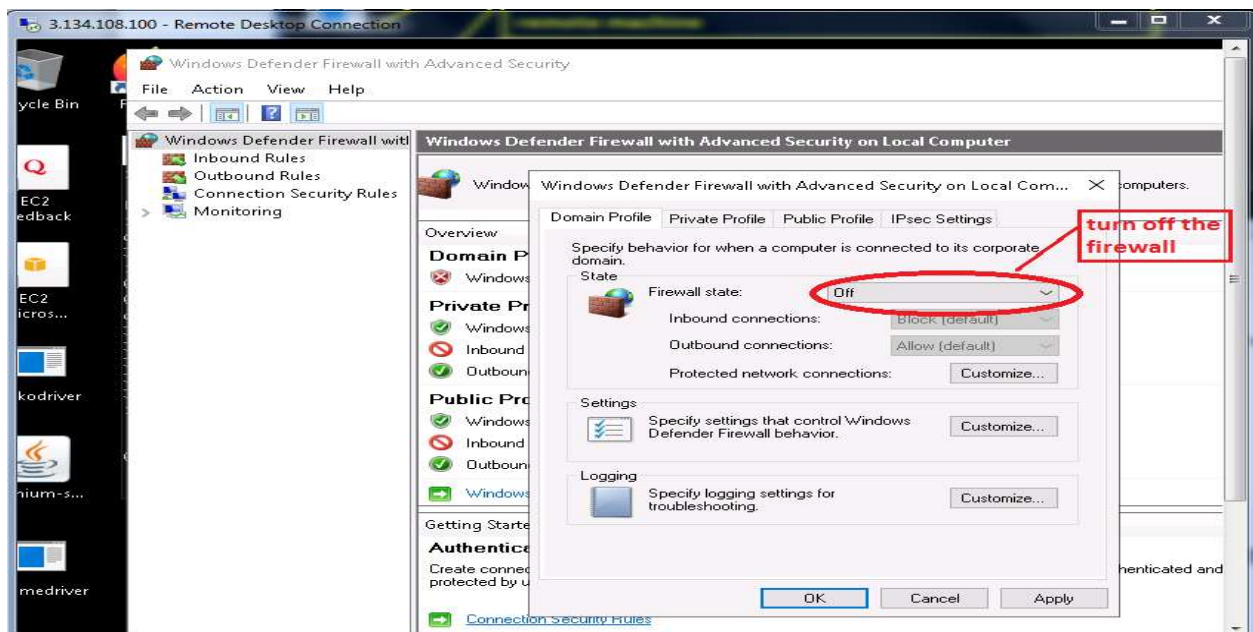
10. Remote desktop connection is established and a windows virtual computer is almost done, download all the pre-requisites to run the selenium server

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

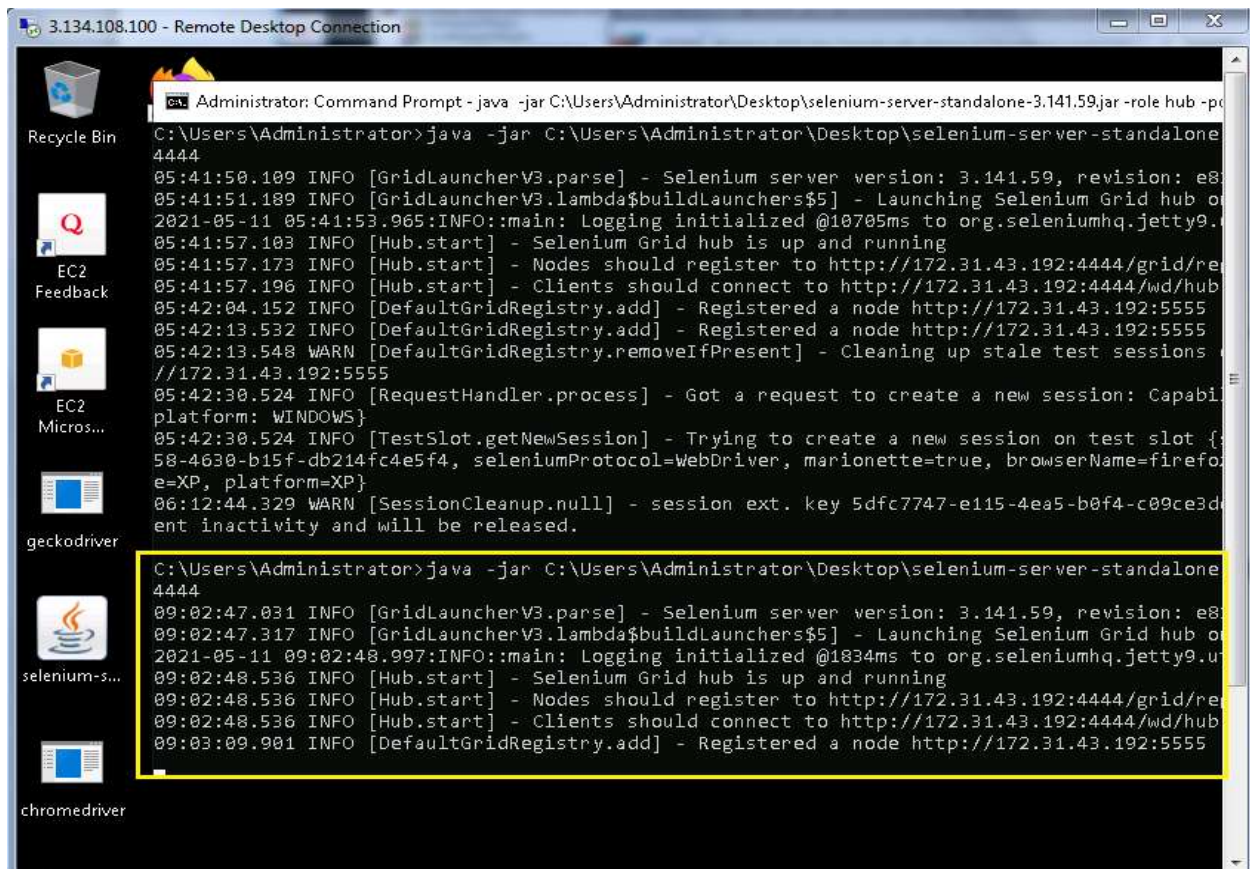
1. Selenium stand-alone-server.3.141.59.jar
2. Driver executable files for necessary browser
3. JDK
4. Browser



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



The screenshot shows a Remote Desktop Connection window titled "3.134.108.100 - Remote Desktop Connection". The desktop background is dark blue. On the left sidebar, there are icons for Recycle Bin, EC2 Feedback, EC2 Micros..., geckodriver, selenium-s..., and chromedriver. The main window is a Command Prompt titled "Administrator: Command Prompt" showing the execution of the following command: `java -jar C:\Users\Administrator\Desktop\selenium-server-standalone-3.141.59.jar -role hub -p 4444`. The output shows the Selenium server starting, logging initialized, and registering nodes. A yellow box highlights the second command prompt window showing the same command and output.

```
Administrator: Command Prompt - java -jar C:\Users\Administrator\Desktop\selenium-server-standalone-3.141.59.jar -role hub -p 4444
C:\Users\Administrator>java -jar C:\Users\Administrator\Desktop\selenium-server-standalone-3.141.59.jar -role hub -p 4444
05:41:50.189 INFO [GridLauncherV3.parse] - Selenium server version: 3.141.59, revision: e88d7c1
05:41:51.189 INFO [GridLauncherV3.lambda$buildLaunchers$5] - Launching Selenium Grid hub on port 4444
2021-05-11 05:41:53.965:INFO::main: Logging initialized @10705ms to org.seleniumhq.jetty9.util.log.StdErrLog
05:41:57.103 INFO [Hub.start] - Selenium Grid hub is up and running
05:41:57.173 INFO [Hub.start] - Nodes should register to http://172.31.43.192:4444/grid/register/
05:41:57.196 INFO [Hub.start] - Clients should connect to http://172.31.43.192:4444/wd/hub/
05:42:04.152 INFO [DefaultGridRegistry.add] - Registered a node http://172.31.43.192:5555
05:42:13.532 INFO [DefaultGridRegistry.add] - Registered a node http://172.31.43.192:5555
05:42:13.548 WARN [DefaultGridRegistry.removeIfPresent] - Cleaning up stale test sessions from http://172.31.43.192:5555
05:42:30.524 INFO [RequestHandler.process] - Got a request to create a new session: Capabilities {browserName: firefox, platform: WINDOWS}
05:42:30.524 INFO [TestSlot.getNewSession] - Trying to create a new session on test slot {id: 58-4630-b15f-db214fc4e5f4, seleniumProtocol=WebDriver, marionette=true, browserName=firefox, platform=XP}
06:12:44.329 WARN [SessionCleanup.null] - session ext. key 5dfc7747-e115-4ea5-b0f4-c09ce3d0b15f is inactive and will be released.

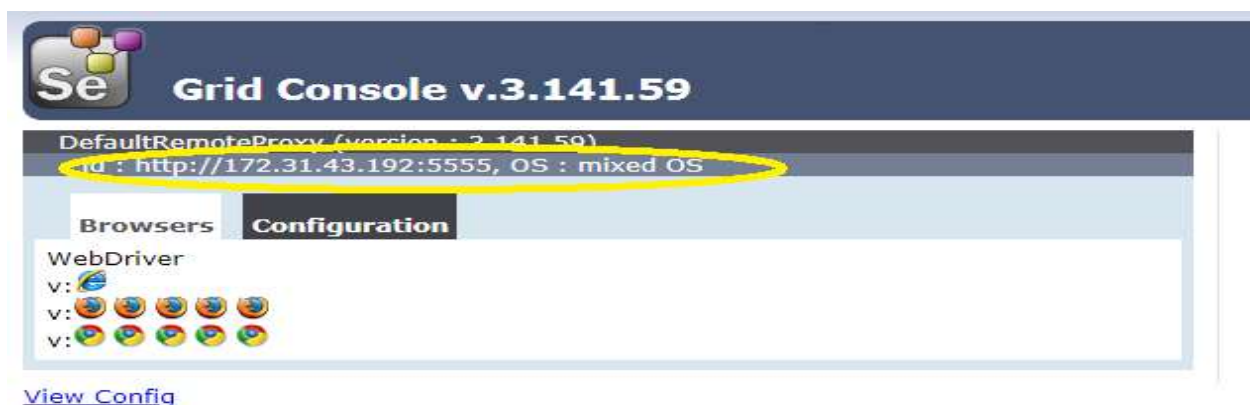
C:\Users\Administrator>java -jar C:\Users\Administrator\Desktop\selenium-server-standalone-3.141.59.jar -role hub -p 4444
09:02:47.031 INFO [GridLauncherV3.parse] - Selenium server version: 3.141.59, revision: e88d7c1
09:02:47.317 INFO [GridLauncherV3.lambda$buildLaunchers$5] - Launching Selenium Grid hub on port 4444
2021-05-11 09:02:48.997:INFO::main: Logging initialized @1834ms to org.seleniumhq.jetty9.util.log.StdErrLog
09:02:48.536 INFO [Hub.start] - Selenium Grid hub is up and running
09:02:48.536 INFO [Hub.start] - Nodes should register to http://172.31.43.192:4444/grid/register/
09:02:48.536 INFO [Hub.start] - Clients should connect to http://172.31.43.192:4444/wd/hub/
09:03:09.901 INFO [DefaultGridRegistry.add] - Registered a node http://172.31.43.192:5555
```

Navigate back to local machine, open any browser and enter

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

⚠ Not secure **3.134.108.100:4444/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
```

The screenshot displays the Selenium Grid Console interface. At the top, it says 'Grid Console v.3.141.59'. Below this, there are three panels, each representing a node. Each panel has a 'Browsers' tab and a 'Configuration' tab. The 'Browsers' tab shows a list of browsers (Chrome, Firefox, Internet Explorer, Safari) and their versions. The 'Configuration' tab shows the node's configuration details. The top bar indicates the version is v.3.141.59. A 'View Config' link is visible at the bottom left.

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