**Configuration and setup for Docker environment**

**What is Docker and Container?**

-Docker is a computer program that performs operating-system-level virtualization, also known as "containerization".

-A Docker container image is a lightweight, standalone, executable package of software that includes everything needed to run an application: code, runtime, system tools, system libraries and settings.

**Features:**

* Standard: Docker created the industry standard for containers, so they could be portable anywhere
* Lightweight: Containers share the machine’s OS system kernel and therefore do not require an OS per application, driving higher server efficiencies and reducing server and licensing costs
* Secure: Applications are safer in containers and Docker provides the strongest default isolation capabilities in the industry.

**Configuration of Docker image and Containers on Ubuntu 16 environment:**

1.Before installing Docker package, we need to set up the docker path:

The default path of docker installation is /var/lib/docker/containers/containerid/. But by configuring the docker configuration path on /etc/docker/daemon.json we can change this path to any required location. By doing this change & removing the /home partition from the tool as a mount point we are able to store the data on /home partition only.

Below is the content need to be added in /etc/docker/daemon.json file:

**{**

**"graph":"/home/cavisson/docker"**

**}**

2.Installation of Docker package (**Version 18**):

Uninstall old versions:

apt-get remove docker docker-engine docker.io

a.Update the apt package index:

$ sudo apt-get update

b.Install packages to allow apt to use a repository over HTTPS:

$ sudo apt-get install \

apt-transport-https \

ca-certificates \

curl \

software-properties-common

c. Add Docker’s official GPG key:

$ curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo apt-key add -

Verify that you now have the key with the fingerprint:

$ sudo apt-key fingerprint 0EBFCD88

pub 4096R/0EBFCD88 2017-02-22

Key fingerprint = 9DC8 5822 9FC7 DD38 854A E2D8 8D81 803C 0EBF CD88

uid Docker Release (CE deb) <docker@docker.com>

sub 4096R/F273FCD8 2017-02-22

d. Use the following command to set up the **stable** repository.

$ sudo add-apt-repository \

"deb [arch=amd64] https://download.docker.com/linux/ubuntu \

$(lsb\_release -cs) \

stable"

#### **INSTALL DOCKER CE**

1. Update the apt package index.

$ sudo apt-get update

2. Install the latest version of Docker CE, or go to the next step to install a specific version:

$ sudo apt-get install docker-ce

For all the setup of docker version 18, you may also use the below link:

<https://docs.docker.com/install/linux/docker-ce/ubuntu/>

Now check the Docker version on machine using below command:

**docker --version**

Output should be like this:

**Docker version 18.03.1-ce, build 9ee9f40**

Up to now, you have installed the Docker on the machine which is considered as a host.

Now there is another requirement of port mapping in creation of controllers and that will be done for all the services like tomcat,cmon,lps,ndc on host machine.

**Need for port mapping on Host:**

Port mapping to the IP of container has to be done on host in order to conflicts running port while launching any container.

**For port mapping of tomcat:**

1.Specify the IP of the Host on path:

/home/cavisson/work/**apps**/apache-tomcat-7.0.59/conf/server.xml (Blade specific)

Content to be added:

Example: **address="66.220.31.142"**

**Output should be like (for HTTP):**

<Connector port="80" protocol="HTTP/1.1"

connectionTimeout="20000"

address="66.220.31.142"

maxPostSize="0"

Content to be added for HTTPS:

Example: **address="66.220.31.142"**

**Output should be like:**

<Connector SSLEnabled="true" port="443" address=”66.220.31.142” protocol="HTTP/1.1"

Example:

root@ubuntu:/home/cavisson/work# netstat -natp| grep -w 80

tcp 0 0 **66.220.31.147:80** 0.0.0.0:\* LISTEN 186849/docker-proxy

tcp 0 0 **66.220.31.145:80** 0.0.0.0:\* LISTEN 33004/docker-proxy

tcp 0 0 **66.220.31.146:80** 0.0.0.0:\* LISTEN 32164/docker-proxy

tcp6 0 0 **66.220.31.142:80** :::\* LISTEN 70795/java

In above output the host and 3 containers, all are mapped to port 80 with each IP.

**Below is the IP mapping content for cmon.env file:**

Output as an Example:

**CAV\_MON\_AGENT\_OPTS="-i 66.220.31.142 -p 7891"**

**Path - /home/cavisson/monitors/sys/cmon.env**

root@ubuntu:/home/cavisson/work# netstat -natp| grep 7891

tcp 0 0 **66.220.31.147:7891** 0.0.0.0:\* LISTEN 186583/docker-proxy

tcp 0 0 **66.220.31.145:7891** 0.0.0.0:\* LISTEN 32751/docker-proxy

tcp 0 0 **66.220.31.146:7891** 0.0.0.0:\* LISTEN 31873/docker-proxy

tcp6 0 0 **66.220.31.142:7891** :::\* LISTEN 33967/java

**Below is the IP mapping content for NDC:**

File to be edit: **ndc.conf**

Path- /home/cavisson/work/ndc/conf

Add the host IP in the file. Below is an output as an example:

PORT 66.220.31.142:7892 (7892 is the NDC Port)

Example:

root@ubuntu:/home/cavisson/work# netstat -natp| grep 7892

tcp 0 0 **66.220.31.142:7892** 0.0.0.0:\* LISTEN 66670/ndcollector

tcp 0 0 **66.220.31.147:7892** 0.0.0.0:\* LISTEN 186570/docker-proxy

tcp 0 0 **66.220.31.145:7892** 0.0.0.0:\* LISTEN 32737/docker-proxy

tcp 0 0 **66.220.31.146:7892** 0.0.0.0:\* LISTEN 31859/docker-proxy

**Below is the IP mapping content for LPS:**

File to be edit: **lps.conf**

Path- /home/cavisson/work/lps/conf

Add the host IP and output as an example should be like:

LPS\_PORT 66.220.31.142:7888

NDC\_LPS\_PORT 66.220.31.142:7889

**Port mapping for ssh service:**

If you want to enable the ssh port to be 22 on both containers and host, then you have to assign host IP in sshd\_config file.

Path- **/etc/ssh/sshd\_config**

Uncomment the field “ListenAddress” and assign the host ip instead of 0.0.0.0

Below is the example:

**ListenAddress 66.220.31.142**

Save this file and **run “service ssh restart”**

Proceeding further, on this host machine we have to launch containers which will be deployed using cavisson docker image and that image will be fetch from Docker Hub.

Below are the commands to deploy Cavisson Docker Image on Host machine:

1. **docker login --username=cavisson001container**
2. **password: (Enter the password as “cavisson”)**

After the successful login, we have to pull the Docker Image. Use below command:

**docker pull cavisson001container/cav\_nde\_4.1.12.30:4.1.12.30**

This will take approximately 7-8 minutes.

Now run the command "**docker images**" to check the present docker images on Host machine from only Root user.

Output will be like:

root@ubuntu:/home/cavisson/work# docker images

REPOSITORY TAG IMAGE ID CREATED SIZE

cavisson001container/cav\_nde\_4.1.12.30 4.1.12.30 ef46ff14910f 5 weeks ago 7.01GB

Moving on to the configuration and deployment of containers, we need some free IP’s that we will map to container IP for ssh. It means our containers will behave as separate machine. Therefore, free IP’s should be as equal to the number of containers.

For that you can also use “**nmap –PN**” command. This command will give you the output of the IP’s which are using in that subnet. You may target the missing IP’s in the output of that subnet, also try to ping them. If they are not able to ping, this means you can use those IP’s for containers.

For example, I have to launch a container a container with SSH IP “66.220.31.145”:

After this you have to assign the IP address in /etc/network/interfaces file as to make it accessible:

Below are the steps:

1.vi /etc/network/interfaces

2.Below content is an example you have to add in interfaces file:

**auto enp97s0f0:1**

**iface enp97s0f0:1 inet static**

**address 66.220.31.145**

**netmask 255.255.255.224**

(Here enp97s0f0 is an example of primary interface of host machine). Similarly for next container IP’s you may use :2, :3 with the interface name and can change the IP address respectively.

After this, you have to make the interface in “up” state. You may use the below command:

**ifup enp97s0f0:1**

After this, validate by pinging the container IP from host machine. It must be ping and proves that container interface is in up state successfully.

**Output of ifconfig as an example:**

enp97s0f0 Link encap:Ethernet HWaddr 0c:c4:7a:fc:03:08

inet addr:66.220.31.142 Bcast:66.220.31.159 Mask:255.255.255.224

inet6 addr: fe80::ec4:7aff:fefc:308/64 Scope:Link

UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1

RX packets:7635689 errors:0 dropped:0 overruns:0 frame:0

TX packets:18810022 errors:0 dropped:0 overruns:0 carrier:0

collisions:0 txqueuelen:1000

RX bytes:1819245252 (1.8 GB) TX bytes:21410994274 (21.4 GB)

enp97s0f0:1 Link encap:Ethernet HWaddr 0c:c4:7a:fc:03:08

inet addr:66.220.31.145 Bcast:66.220.31.159 Mask:255.255.255.224

UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1

enp97s0f0:2 Link encap:Ethernet HWaddr 0c:c4:7a:fc:03:08

inet addr:66.220.31.146 Bcast:66.220.31.159 Mask:255.255.255.224

UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1

enp97s0f0:3 Link encap:Ethernet HWaddr 0c:c4:7a:fc:03:08

inet addr:66.220.31.147 Bcast:66.220.31.159 Mask:255.255.255.224

UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1

**Deployment of container on host machine:**

For this you may use below shell to deploy container with that IP and required configuration:

Name- **cav\_docker\_config**

root@controller:/home/cavisson# ./cav\_docker\_config

Mandatory Arguments container is missing. Use container name with -n option.

cav\_docker\_config -i <virtual ip>[Mandatory for run operation] -n <Container name>[Mandatory]

-r <release image>[Mandatory for run operation] -R [Mark container for restart automatically]

-m <memory for container use k,m,g for units>{Optional} -c <no of core for cpu>

-C <explicit command if needed> -D[Optional for Debug log] -o <Docker operations start/stop/run/restart/remove>

After making the shell executable using chmod +x, you may refer to the below example to launch container:

**./cav\_docker\_config -i 66.220.31.145 -n NDPerformance1 -r cavisson001container/cav\_nde\_4.1.12.30:4.1.12.30 -m 360g -c 11**

(Here 66.220.31.145 is free IP that will be used as SSH IP for container)

You will find the shell cav\_docker\_config in an upcoming build **4.1.12 B 107**

For now, you can take this from local machine (**10.10.30.2**)

**Path**- /home/cavisson

After the container formation, you may run the command “**docker ps –a**” from Root user

This will give you an output wih the container ID including with the mapped IP address.

Below is the example:

root@ubuntu:/home/cavisson/work# **docker ps -a**

CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES

**1f5a5aa0e1df** cavisson001container/cav\_nde\_4.1.12.30:4.1.12.30 "/bin/sh -c '/root/s…" 5 days ago Exited (255) 5 days ago 66.220.31.148:22->22/tcp, 66.220.31.148:25->25/tcp, 66.220.31.148:80->80/tcp, 66.220.31.148:123->123/tcp, 66.220.31.148:443->443/tcp, 66.220.31.148:587->587/tcp, 66.220.31.148:623->623/tcp, 66.220.31.148:4430-4440->4430-4440/tcp, 66.220.31.148:5900->5900/tcp, 66.220.31.148:7890-7900->7890-7900/tcp, 66.220.31.148:8000-8010->8000-8010/tcp, 66.220.31.148:10001->21/tcp Test\_SSH

Here ‘**1f5a5aa0e1df**’ is the container ID. You may login into this container using below command:

**docker exec –it 1f5a5aa0e1df /bin/bash**

Now you have enter into the container. Here you have to set the root password of container using command “**passwd**”.

After this to enable SSH, you have to run “**service ssh restart**” of container.

At last, login/SSH the container into the new terminal. In this way you can launch and setup multiple containers from 1 Host machine.

**Server Side Configuration:**

A. Multiple CMON configuration on agent.

B. Multiple ND agent configuration agent.

In order to generate ND Data, we have to run a Continuous Monitoring Test (CM Test) on agent/server side that will generate the load on containers. To set up that environment we have to generate the load either from multiple CMON’s or multiple ND Agents at server side.

**A.Steps to run multiple CMON’s on server side:**

To configure multiple CMON’s, you may use the shell “**cav\_cmon\_config**”

Path- (/home/cavisson)

Machine- 10.10.30.2

Copy this shell to your agent.

**Usage**: cav\_cmon\_config -n <number of cmon to run> -m <NDC IP> -P <ndc ports> -c <Cav agent Ports> -p <jmx Ports> -t <Tier Name>

This will configure your multiple CMON’s by configuring cmon.env file.

Path of cmon.env file- /home/cavisson/monitors/sys

NDC IP will be the IP of container with ndc port.

Regarding cav agents port, you can use any random port e.g 17891.

Similarly specify the JMX port also and required Tier Name with the required number of cmon to be run.

For validation purpose, you can use “ **ps –ef| grep cmon**” command.

**B. Multiple ND agent configuration agent:**

To generate the ND load in CM Test, we have to configure multiple ND Agents on server side (On the same machine on which we ran multiple CMON’s)

For that we have to use a TAR file named as “**QATools.tar.gz**”

**Location**- 10.10.30.2

**Path**- /home/cavisson

Untar this file using command “**tar -xvzf QATools.tar.gz**”. Enter to the directory QATools.

Validate that there should be a TAR “thread-ip-callout-0.0.1-SNAPSHOT.jar” present on agent.

Run the shell “**init\_instances\_in\_servers.sh**” in the background using nohup command. Below is the command that you have to run on agent:

**nohup ./init\_instances\_in\_servers.sh 66.220.31.145 7898 30 1 > server.log 2>&1 &**

(Here 66.220.31.145 is the container IP on which ND Data will be generated)

**Validation:** You can validate the shell by checking **server.log** file. Data will be appending continuously in server.log file.

You may also check the thread-ip process. It should be running.

cavisson@CC-W-CA-Fremont-HE-131:~/work/QATools$ **ps -ef| grep thread-ip**

cavisson 1276 1 1 Sep07 ? 01:37:00 /home/cavisson/apps/jdk1.8.0\_161/bin/java -jar -javaagent:/home/cavisson/netdiagnostics/lib/ndmain.jar=time,ndAgentJar=/home/cavisson/netdiagnostics/lib/ndagent-with-dep.jar,ndHome=/home/cavisson/netdiagnostics,ndcHost=66.220.31.140,ndcPort=7898,ndlPort=181 -Dcom.sun.management.jmxremote -Dcom.sun.management.jmxremote.port=18181 -Dcom.sun.management.jmxremote.ssl=false -Dcom.sun.management.jmxremote.authenticate=false -Dserver.port=19181 /home/cavisson/work/QATools/thread-ip-callout-0.0.1-SNAPSHOT.jar

cavisson 6359 1 1 Sep07 ? 01:21:12 /home/cavisson/apps/jdk1.8.0\_161/bin/java -jar -javaagent:/home/cavisson/netdiagnostics/lib/ndmain.jar=time,ndAgentJar=/home/cavisson/netdiagnostics/lib/ndagent-with-dep.jar,ndHome=/home/cavisson/netdiagnostics,ndcHost=66.220.31.140,ndcPort=7898,ndlPort=201 -Dcom.sun.management.jmxremote -Dcom.sun.management.jmxremote.port=18201 -Dcom.sun.management.jmxremote.ssl=false -Dcom.sun.management.jmxremote.authenticate=false -Dserver.port=19201 /home/cavisson/work/QATools/thread-ip-callout-0.0.1-SNAPSHOT.jar

After this you can check the init instances process also:

cavisson@CC-W-CA-Fremont-HE-131:~/work/QATools$ **ps -ef| grep init\_instances**

cavisson 31817 1 1 Sep07 ? 01:36:38 /bin/bash ./init\_instances\_in\_servers.sh 66.220.31.140 7898 30 1

After configuring multiple CMON’s and ND agents, we have to configure a script and scenario to run a CM test on container.

**ND CM Test configuration on container:**

How to setup ND Config UI

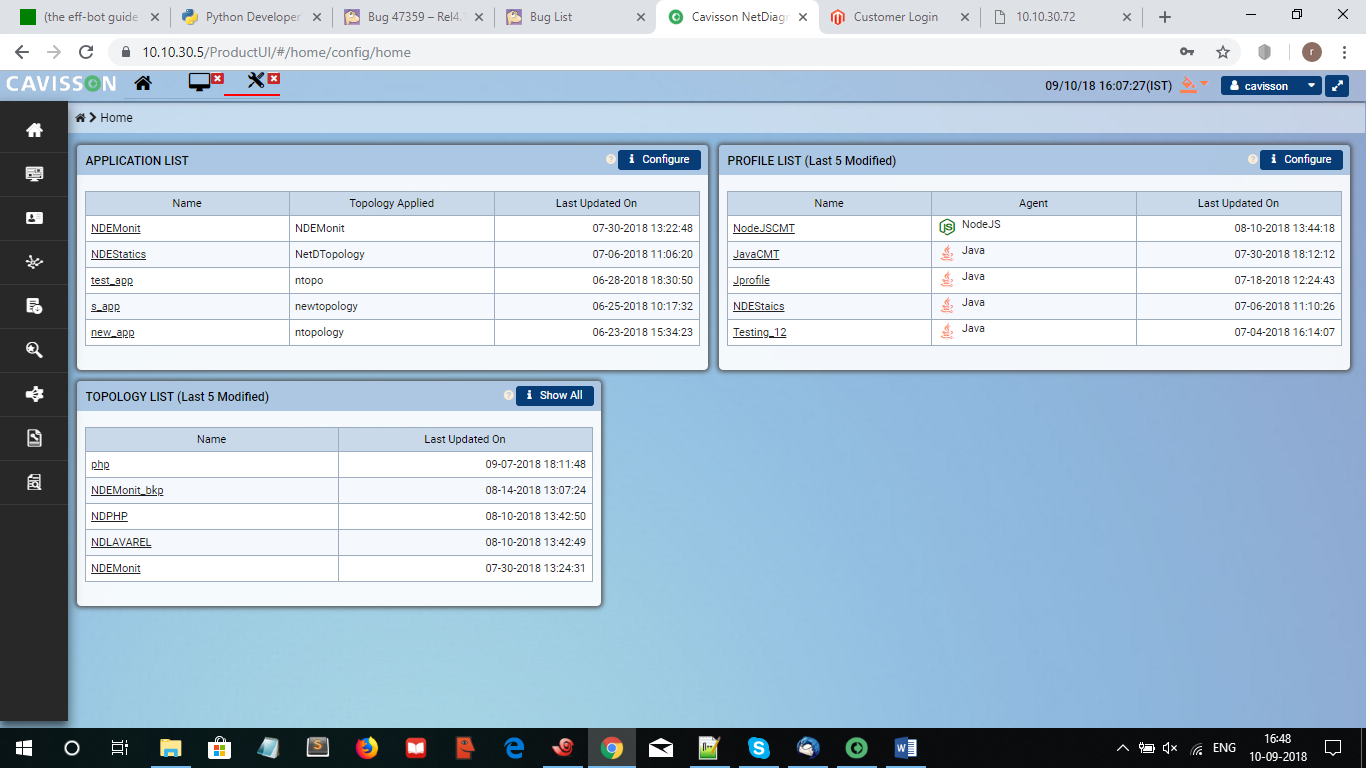
1st step->

Login to the machine from where you want to import the profile .

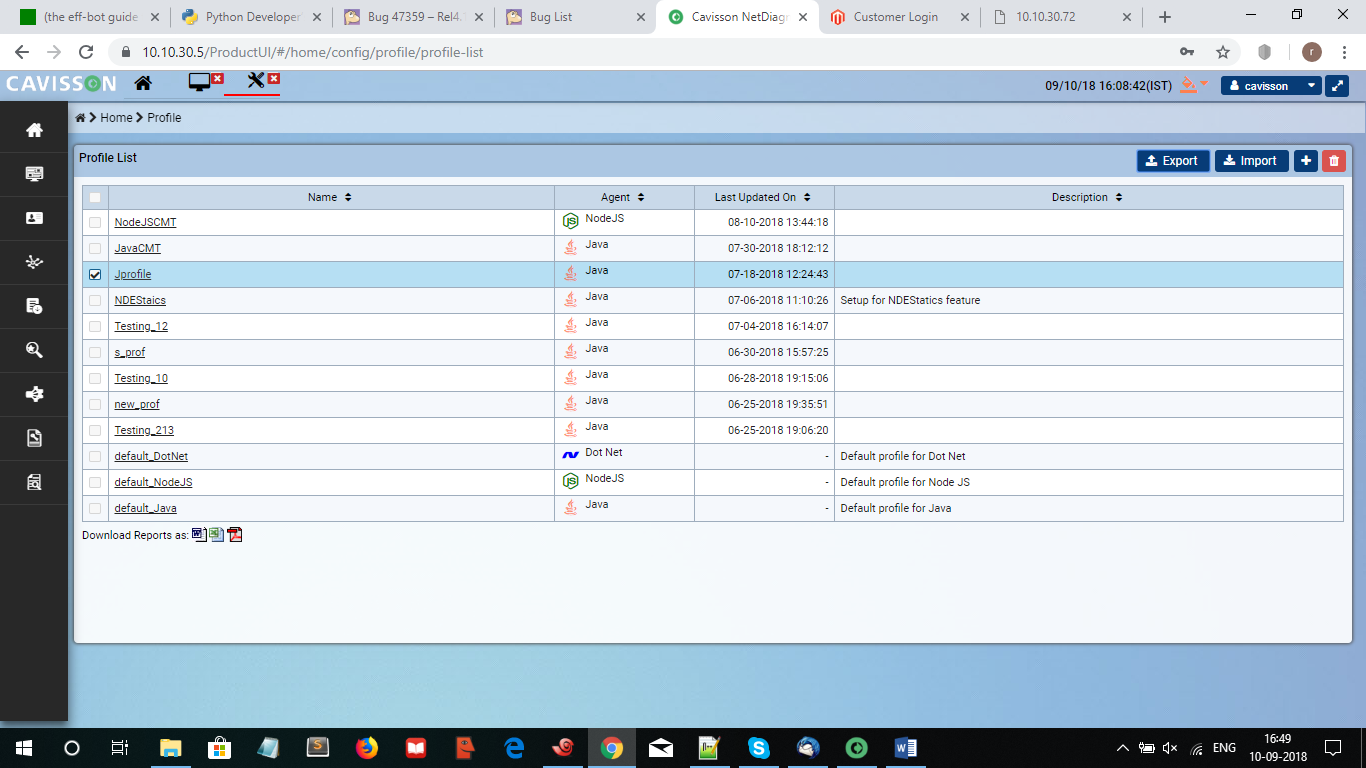
i.e from UI go to NDCONFIG -> profile ->select Profile ->Export

**Example:-**

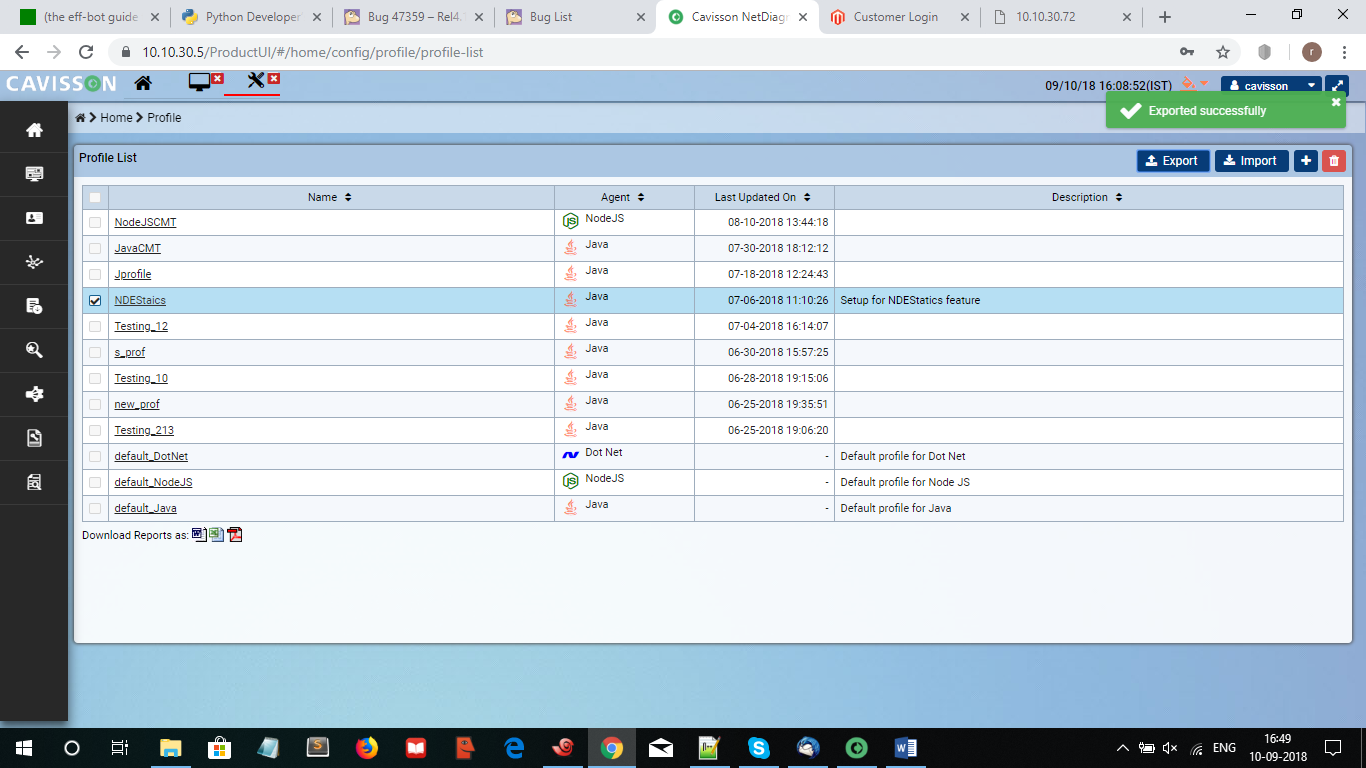
After clicking on config UI you will get this page.



2.Click on the configure in profile list and you will get all the profile and select among them which one you want to export.



3.After that click on the export and you will get a message that it will be done successfully like below.



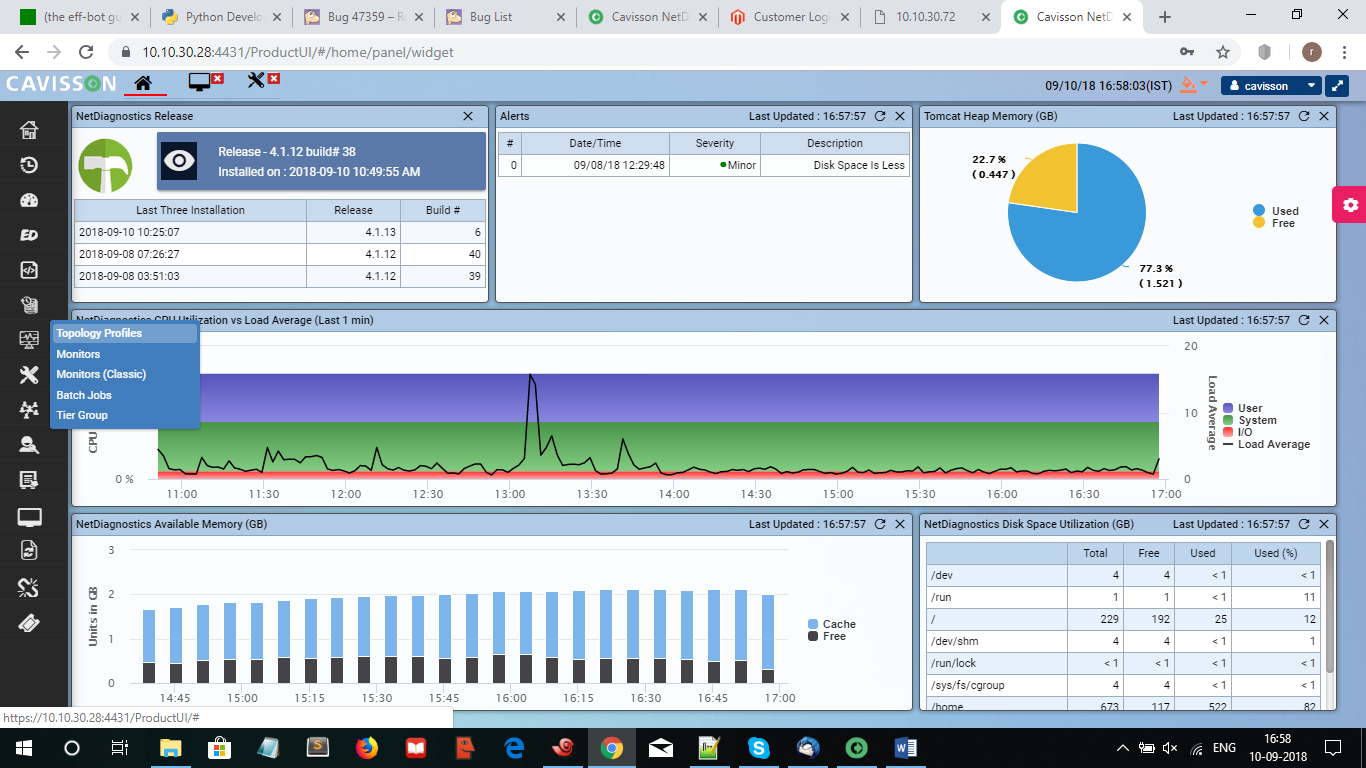
Next

Then open the machine UI where you want to import the same profile configuration.

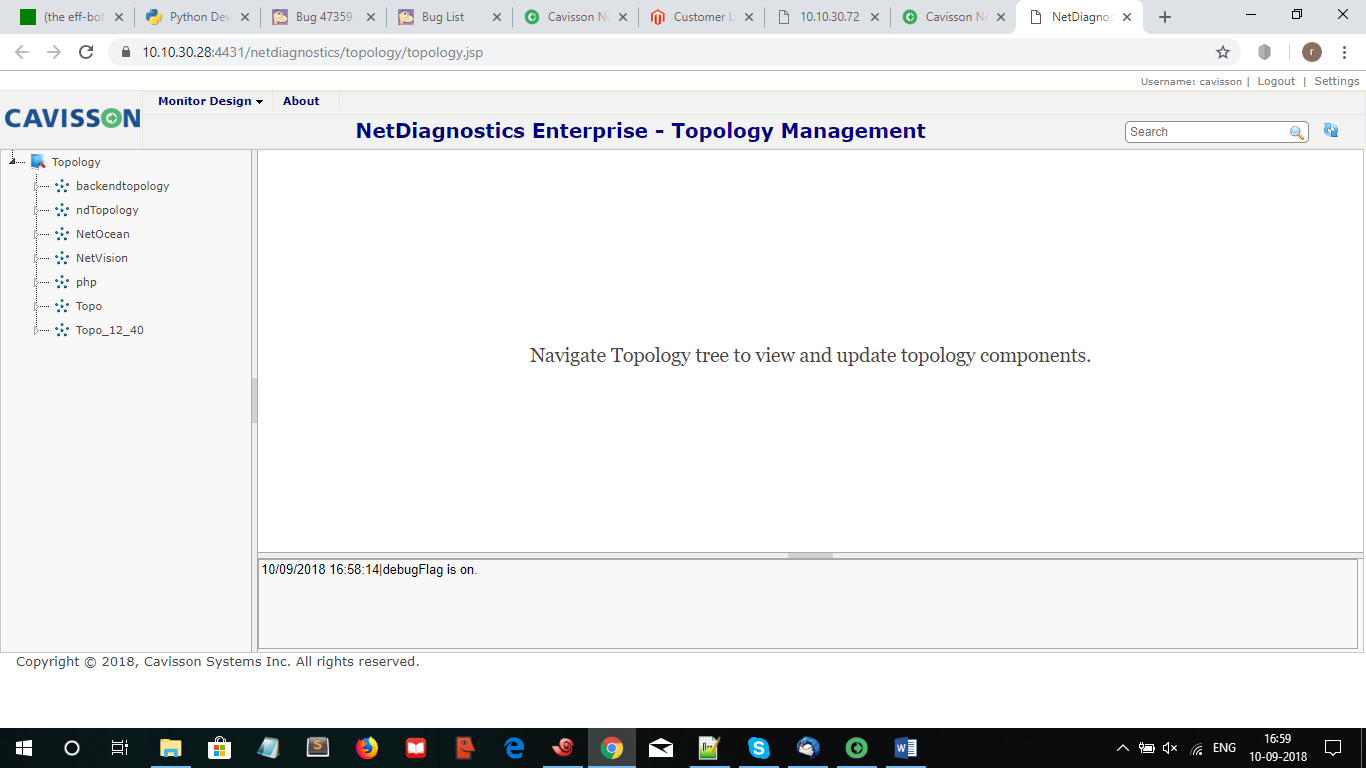
Step1:-

1-you need to create one topology which can be created by clicking on the Monitor Tab -> Topology Profile -> right click on topology -> add topology -> then give the topology name .

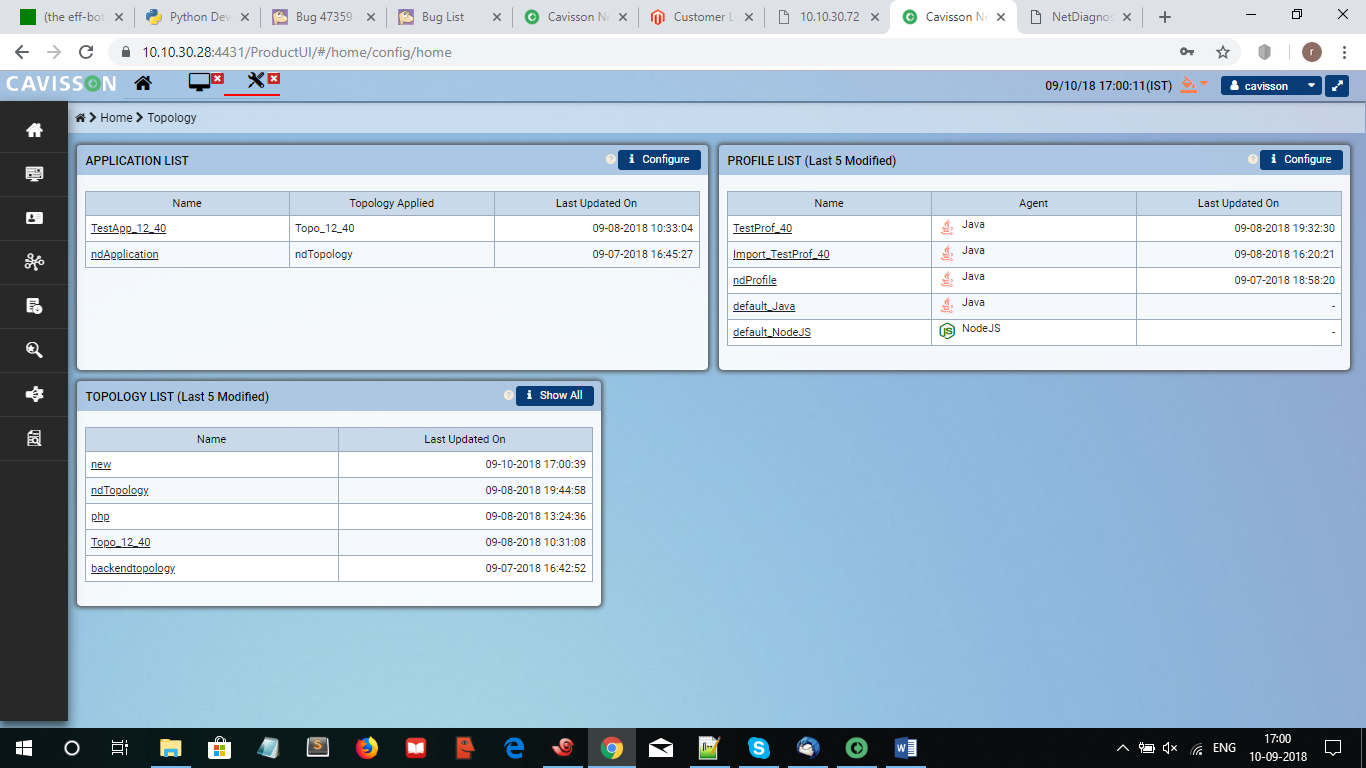
Example:



Next:-



Here you will able to see the topology name which you have created.

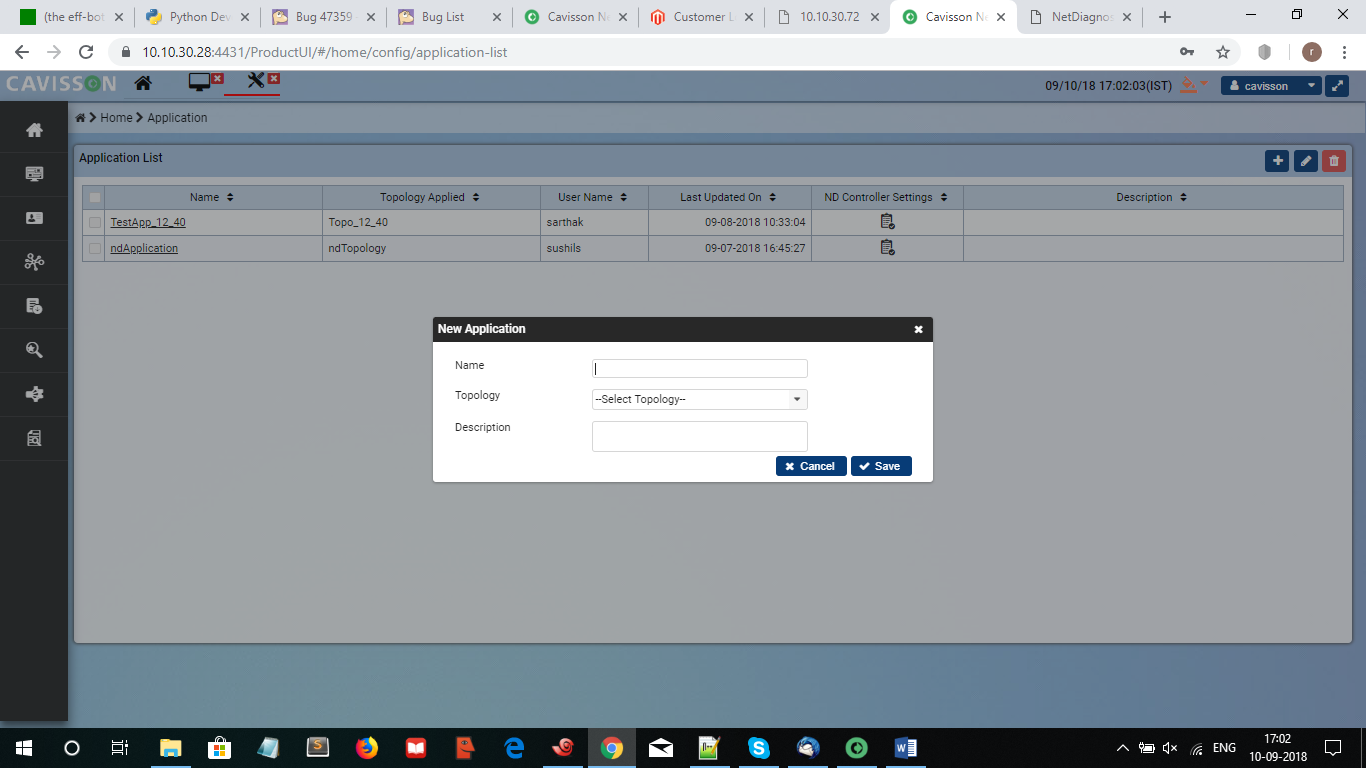


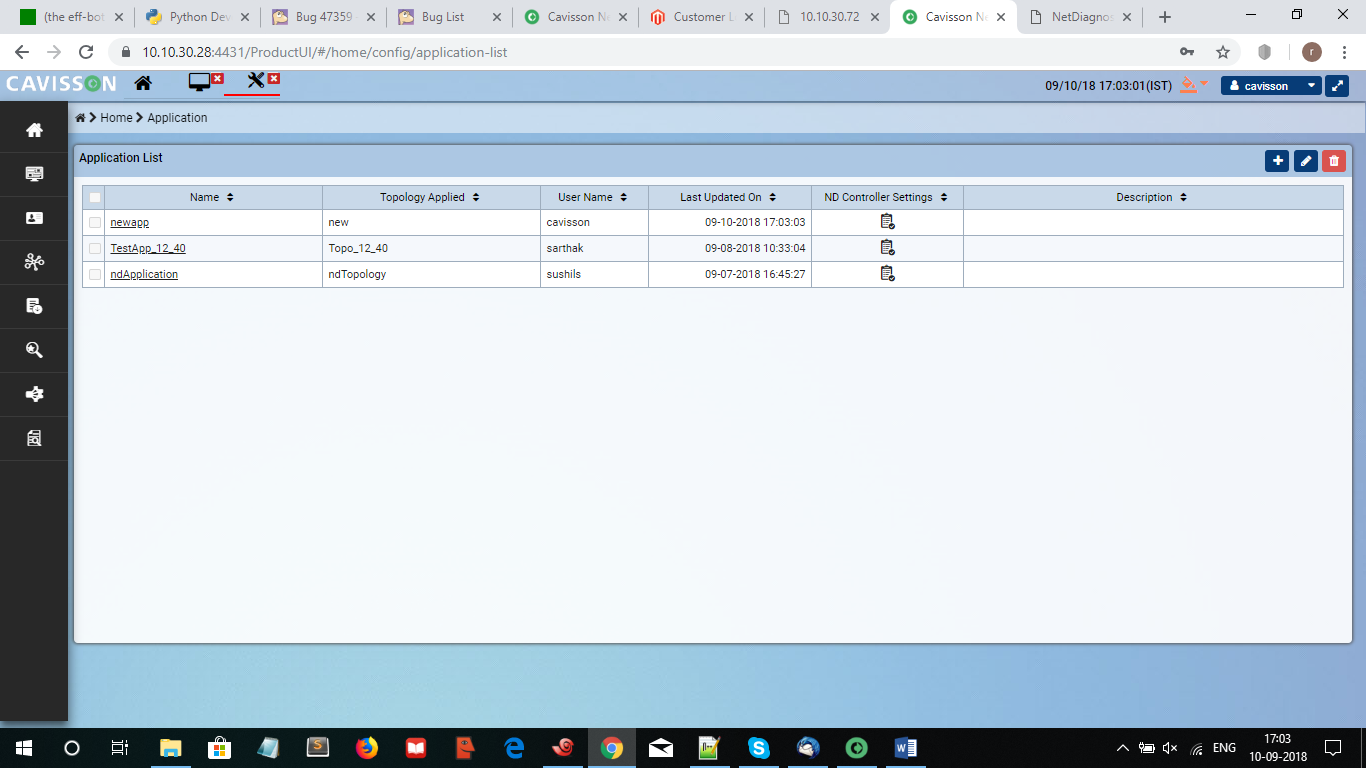
Step 2->

Now click on application and click on add to create an application.

Here you need to provide application name and the topology which you have created soo that it will attach the same in it.

**Example:**





NOTE:-

Before going to import the profile you need to copy the .zip file from the Shell from which ip you have export

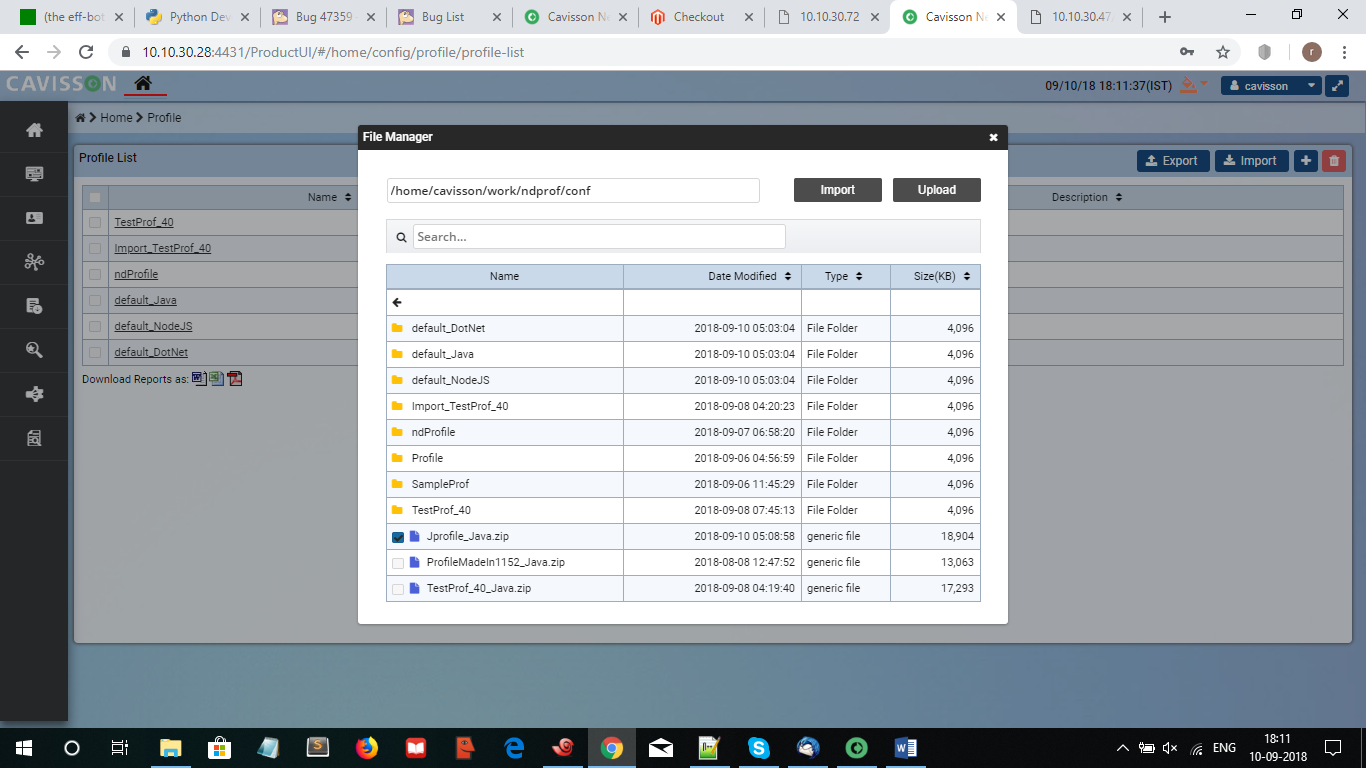
e.g:->

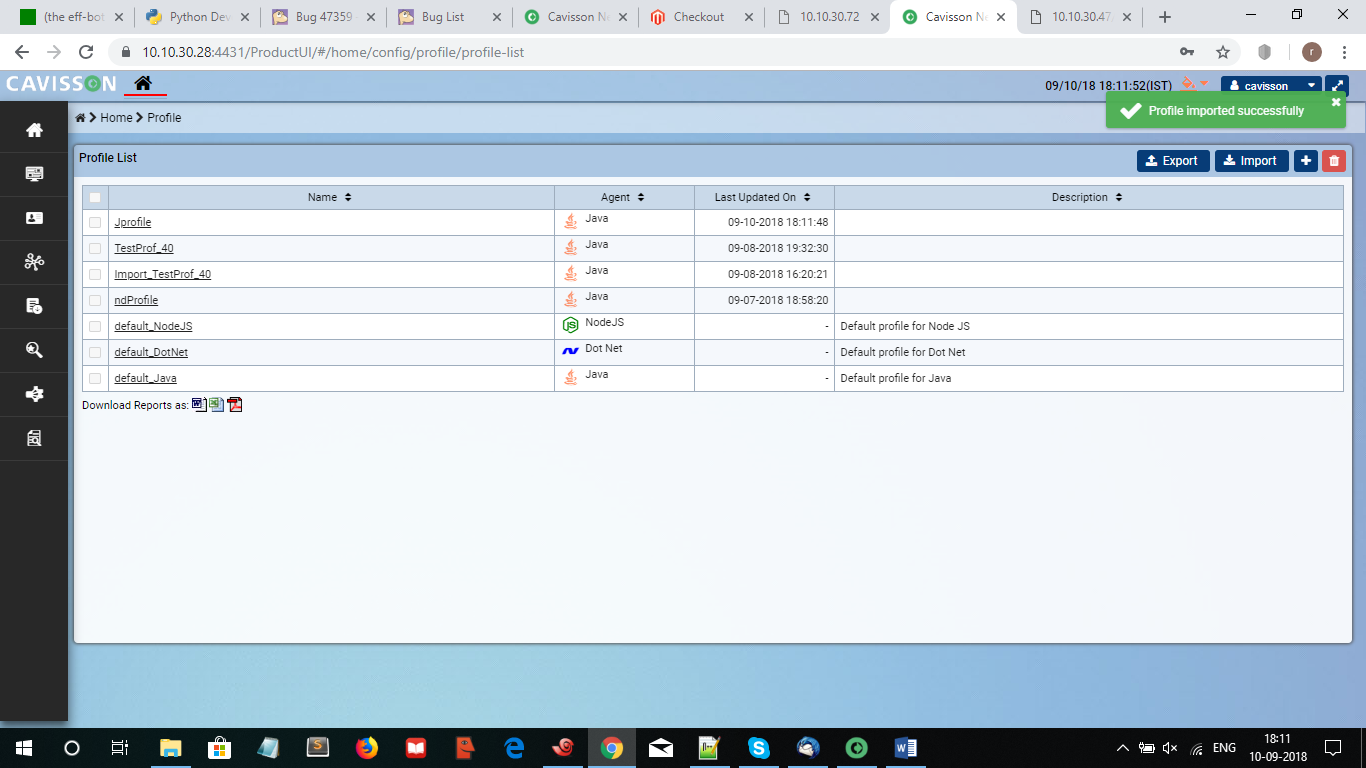
scp Jprofile\_Java.zip [cavisson@10.10.30.28:/home/cavisson/work/ndprof/conf/](mailto:cavisson@10.10.30.28:/home/cavisson/work/ndprof/conf/).

After this you have to import the same from your UI window.

Open profile configuration setting -> click on import ->provide Absolute path -> then select .zip file to import

**Example:-**





**Sample Scenario:**

SCHEDULE ALL Start START IMMEDIATELY

SCHEDULE ALL RampUp RAMP\_UP ALL RATE 120 M LINEARLY

SCHEDULE ALL Stabilize STABILIZATION TIME 0

SCHEDULE ALL RampDown RAMP\_DOWN ALL IMMEDIATELY

ENABLE\_TMPFS\_IN\_DBU 0 0

MAX\_DYNAMIC\_HOST 32

ENABLE\_AUTO\_JSON\_MONITOR 1 Docker.json

MONITOR\_PROFILE RunVectors

ND\_ENABLE\_MONITOR 1 cm\_java\_gc\_jmx\_sun8.gdf cm\_java\_gc\_jmx\_sun8

G\_SCRIPT\_MODE ALL 1 256

G\_PAGE\_THINK\_TIME ALL ALL 3 50 150

LPS\_SERVER 127.0.0.1 7879 1

G\_SERVER\_HOST ALL 127.0.0.1 66.220.31.138:81 -

ENABLE\_CMON\_AGENT 0

G\_CONTINUE\_ON\_PAGE\_ERROR ALL ALL 1

PROF\_PCT\_MODE NUM

NLR\_TRACE\_LEVEL 2

G\_OVERRIDE\_RECORDED\_THINK\_TIME ALL ALL 1

NUM\_NVM 10 MACHINE

ENABLE\_AUTO\_SERVER\_SIGNATURE 0

MULTIDISK\_PATH RAW\_DATA /rawdata1,/rawdata2

MULTIDISK\_PATH ns\_rawdata /rawdata2

MULTIDISK\_PATH processed\_data /processeddata

MULTIDISK\_PATH nscsv /processeddata

MULTIDISK\_PATH NSLOGS /cavlogs

MULTIDISK\_PATH NDLOGS /cavlogs

TABLESPACE\_INFO tablespace1 0 ndbciarg

TABLESPACE\_INFO tablespace2 0 ndmethodtiming,ndautosensorhotspotthreads

TABLESPACE\_INFO tablespace3 0 ALL

SGRP G1 NA NA Internet 0 default/default/Tour\_Script 30

HIERARCHICAL\_VIEW 1 Docker >

NET\_DIAGNOSTICS\_SERVER 2 NA Docker

GROUP\_HIERARCHY\_CONFIG 1

ENABLE\_OUTBOUND\_CONNECTION 1

PROGRESS\_MSECS 10000

For script, you can simply use a no-hit script or tours script.

Now you can run a CM test on container and validate the ND and monitor data in web dashboard.