

## **16. Performance Dashboard Onboarding**

Power of Platform

Exported on 10/08/2024

## Table of Contents

1	Flowchart Representation of Automation: .....	4
1.1	JENKINS: .....	4
1.1.1	important parts of shell script.....	5
1.2	ELK STACK: .....	5
1.2.1	For detailed information on configuration please refer to this confluence page - ELK Automation .....	6
1.3	HOW TO GENERATE THE COMPARISON REPORT: .....	6
1.3.1	snapshot of our postman request .....	6
2	HTML REPORT: .....	7
3	Service Level Dashboards: .....	9
4	Grafana Screenshots of Dashboards: .....	10
5	TO DO:.....	11
6	ELK Automation .....	12
6.1	ELK(Elastic Search, Logstash, Kibana):.....	12
6.2	Installation steps of ELK and Filebeat in Linux Remote Machine:.....	13
6.3	Filebeat .....	13
6.4	Logstash.....	13
6.5	Elastic Search.....	14
6.6	Kibana .....	14
7	Golang.....	25
7.1	GoLang creating a build:.....	27
7.2	GOLANG code: .....	27
8	Jenkins Automation.....	28
8.1	Jenkin Setup : .....	28
9	New Automation Suite onboarding in IICS automation suite .....	35

Automation Story	 <a href="#">POP-24470</a> <sup>1</sup> - Phase I - IICS Platform Performance Autonomous Reporting Dashboard <span>CLOSED</span>  <a href="#">POP-24471</a> <sup>2</sup> - Phase II - IICS Platform Performance Autonomous Reporting Dashboard <span>CLOSED</span>  <a href="#">POP-24472</a> <sup>3</sup> - Phase III - IICS Platform Performance Autonomous Reporting Dashboard <span>CLOSED</span>  <a href="#">POP-24473</a> <sup>4</sup> - Phase IV - IICS Platform Performance Autonomous Reporting Dashboard <span>CLOSED</span>
------------------	--

Goal for this story is to create a centralized dashboard to collect all the performance regression KPI metrices for each build/release and publish a comparison report. Provide and store screenshots of Grafana Dashboard for further analysis.

#### Automation Requirement:

- Currently to provide the Regression statistics the results of the previous release and the current release are copied from jenkins and calculating the degradation from excel
- To streamline the process and enhance efficiency, this automation is introduced to collect the stats of regression and provide comparison report in email and dashboard level visualization of response time over the releases in Kibana.
- We are also automating the collection and storage of screenshots for further analysis. This will eliminate the need for manual labor and ensure that the process is more accurate and reliable.

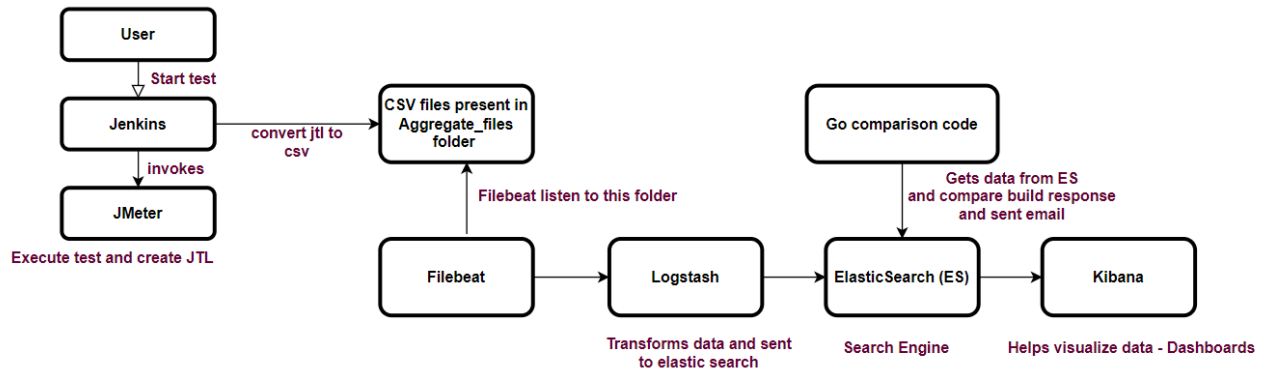
<sup>1</sup> <https://infajira.informatica.com/browse/POP-24470?src=confmacro>

<sup>2</sup> <https://infajira.informatica.com/browse/POP-24471?src=confmacro>

<sup>3</sup> <https://infajira.informatica.com/browse/POP-24472?src=confmacro>

<sup>4</sup> <https://infajira.informatica.com/browse/POP-24473?src=confmacro>

# 1 Flowchart Representation of Automation:



## 1.1 JENKINS:

- We trigger the test from Jmeter using Jenkins.
- Using cmdRunner we convert results from JTL format to CSV.

### 1.1.1 important parts of shell script

#### #Execute Testrun

```
sh ${JMETER_BIN_PATH}/jmeter.sh -n -t ${JMETER_SCRIPT_PATH}
-Jorg_name_prefix=$org_name_prefix -Jthread_num=$thread_num -Jramp_time=$ramp_time
-Jloop_num=$loop_num -Jma_ip=$ma_ip -Jserver_ip=$server_ip -Jpdm_ip=$pdm_ip
-Jusername=$username -Jpassword=$password -Jcon_username=$con_username
-Jcon_password=$con_password -JFRS_username=$FRS_username -JFRS_password=$FRS_password
-JMig_username=$Mig_username -JMig_password=$Mig_password
-JAdmin_password=$Admin_password -JAudit_username=$Audit_username
-JAudit_password=$Audit_password -JFilePath=$FilePath -JSch_username=$sch_username
-Jsch_password=$sch_password -JV3_username=$V3_username -JV3_password=$V3_password
-Jthread_num_2=$thread_num_2 -Jloop_num_2=$loop_num_2 -Jvcs_username=$vcs_username
-Jvcs_password=$vcs_password -JAC_username=$AC_username -JAC_password=$AC_password
-Jscim_username=$scim_username -Jscim_password=$scim_password
-JUpref_username=$Upref_username -JUpref_password=$Upref_password -l "${JTL_Results_File_Name}"
```

#### #Generate aggregate report from jtl

```
java -jar $JMETER_LIB_PATH/ext/cmdrunner-2.0.jar --tool Reporter --generate-csv "${
{Aggregate_Temp_Results_File_Name}}" --input-jtl "${JTL_Results_File_Name}" --plugin-type
AggregateReport
```

## 1.2 ELK STACK:

- **Filebeat** is a log shipper belonging to the Beats family – a group of lightweight shippers installed on hosts for shipping different kinds of data into the ELK Stack for analysis.
- **Logstash** is an open-source data ingestion tool that allows you to collect data from a variety of sources, transform it, and send it to your desired destination.
- **Elasticsearch** is a distributed search and analytics engine built on Apache Lucene. Support for various languages, high performance, and schema-free JSON documents makes Elasticsearch an ideal choice for various log analytics and search use cases.
- **Kibana** is a data visualization and exploration tool for reviewing logs and events. Kibana offers easy-to-use, interactive charts, pre-built aggregations and filters, and geospatial support and making it the preferred choice for visualizing data stored in Elasticsearch

### 1.2.1 For detailed information on configuration please refer to this confluence page - [ELK Automation](#)

## 1.3 HOW TO GENERATE THE COMPARISON REPORT:

- We use postman to get the comparison report.
- we compare using the release number and our test iteration.

### 1.3.1 snapshot of our postman request

POST http://localhost:6666/compareResults?oldReleaseNumber=2022.12&oldRelease\_Iteration=202212.1...

Params • Authorization Headers (7) Body Pre-request Script Tests Settings Cookies

Query Params

	KEY	VALUE	DESCRIPTION	...	Bulk Edit
<input checked="" type="checkbox"/>	oldReleaseNumber	2022.12			
<input checked="" type="checkbox"/>	oldRelease_Iteration	202212.1			
<input checked="" type="checkbox"/>	oldBuildNum	2806			
<input checked="" type="checkbox"/>	newReleaseNumber	2023.01			
<input checked="" type="checkbox"/>	newBuildNum	2906			
<input checked="" type="checkbox"/>	newRelease_Iteration	202301.2			
<input checked="" type="checkbox"/>	index	jmeter-aggregate-jtl			
<input checked="" type="checkbox"/>	email	svali@informatica.com			
<input checked="" type="checkbox"/>	metric	95th			
	Key	Value	Description		

Response

⌵ Cookies ⌵ Capture requests ⌵ Runner ⌵ Trash ⌵ ?

## 2 HTML REPORT:

Note: Below are the Demo Results and will use from upcoming Releases.

## 95th Response Time Comparison for 2022.12 (2806) &amp; 2023.01 (2821)

## IICS Platform Performance Regression

API Labeling : {Servicename}\_{API\_Name}\_{Concurrency}

Dashboard URL : <http://asviiicsperf03:5601/app/r/s/Gexca>

Performance Summary			
Label	Range	Use case Count	Color Code
Total API Count		240	-
% Improvement	> 0 %	108	
% Degradation	0 to 20 %	89	
% Degradation	> 20 %	32	

95th Response Time (ms) for 10 user Concurrency				
API	Release: 2022.12 (in ms)	Release: 2023.01 (in ms)	Time Difference	% Time Difference
AC_Login_10T	441	435	6	1.36 %
AC_Logout_10T	292	307	-15	-5.14 %
CA_ServiceSign_10T	352	718	-366	-103.98 %
FRS_BaseEntities_10T	21667	19431	2236	10.32 %
FRS_CreateFolders_10T	122	138	-16	-13.11 %
FRS_CreateProjects_10T	328	357	-29	-8.84 %
FRS_DeleteFolders_10T	159	135	24	15.09 %
FRS_DeleteProjects_10T	124	121	3	2.42 %
FRS_FetchDefaultLocation_10T	100	121	-21	-21.00 %
FRS_FetchProjectStatForRecentEntity_10T	109	123	-14	-12.84 %
FRS_GetDocumentTypes_10T	555	790	-235	-42.34 %
FRS_GetEffPrivilegeForDoctypeContainer_10T	84	91	-7	-8.33 %
FRS_GetFolders_10T	324	336	-12	-3.70 %
FRS_GetPermissions_10T	107	134	-27	-25.23 %
FRS_GetProjectsAll_10T	110	243	-133	-120.91 %
FRS_GetProjects_10T	211	208	3	1.42 %
FRS_Login_10T	433	477	-44	-10.16 %
FRS_LookupArtifactsDetailsByPath_10T	327	324	3	0.92 %
FRS_UpdateEntityAccess_10T	108	130	-22	-20.37 %
IDS_Login_10T	761	5737	-4976	-653.88 %
Identity_AssignLicense_10T	642	683	-41	-6.39 %
Identity_CreateOrg_10T	338	360	-22	-6.51 %
Identity_VerifyIdentity_10T	108	106	2	1.85 %
JLS_GetFlattenedJobLogEntries_10T	101	107	-6	-5.94 %
JLS_GetJobLogEntries_10T	326	325	1	0.31 %
JLS_GetJobStatusChart_10T	88	101	-13	-14.77 %
LDM_GetConnection_10T	1011	1153	-142	-14.05 %
LDM_getTasksModifiedAfter=0_10T	1138	1162	-24	-2.11 %
LDM_get_tasks_10T	1136	1161	-25	-2.20 %
Login_NotificationPublisher_10T	362	354	8	2.21 %
MA_V3User_10T	162	225	-63	-38.89 %
MA_CreateUser_10T	199	189	10	5.03 %
MA_DeleteUser_10T	222	231	-9	-4.05 %
MA_OrgLogin_10T	430	442	-12	-2.79 %
MA_V2Login_10T	765	672	93	12.16 %
MA_V2Logout_10T	78	82	-4	-5.13 %
MA_V3Login_10T	145	158	-13	-8.97 %
MA_V3Logout_10T	84	88	-4	-4.76 %
Notification-service-Publish-BANNER-User_10T	96	104	-8	-8.33 %
Notification-service-Publish-BELL_NOTIFICATION-ORG_10T	89	86	3	3.37 %
Notification-service_Fetch_notification_for_user_notificationIcon_10T	131	146	-15	-11.45 %
Notification-service_UI_Fetch_new_BANNERorMESSAGE_BOXorMESSAGE_BUBBLE_for_user_10T	96	96	0	0.00 %
Notification-service_UI_Fetch_new_BellNotificationCount_for_user_10T	95	93	2	2.11 %
Notification-service_UI_MarkAllNotificationsAsRead_BELL_NOTIFICATION_10T	127	127	0	0.00 %
Notification-service_count_of_new(unread)notifications_10T	315	326	-11	-3.49 %
SAAS_Login_10T	426	448	-22	-5.16 %
SCIM_GetUser_10T	164	147	17	10.37 %

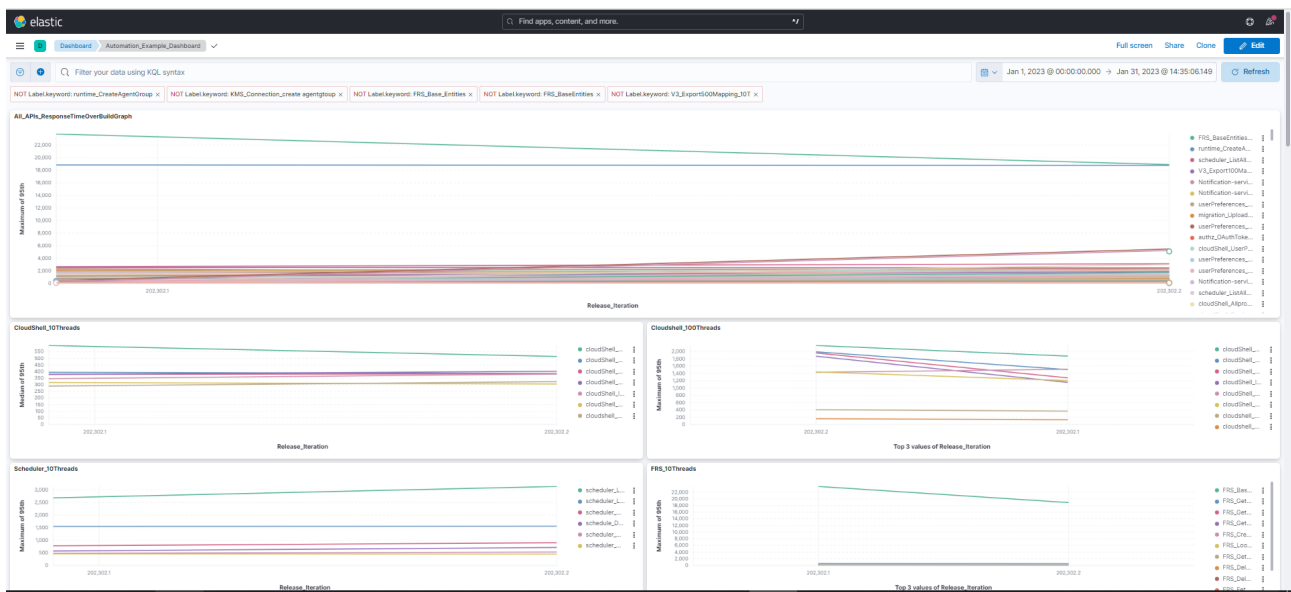


### 3 Service Level Dashboards:

To facilitate easy monitoring and comparison of service-level data across releases and test iterations, service-level dashboards have been created.

These dashboards provide a consolidated view of relevant data, enabling efficient analysis and informed decision-making.

[http://asviicsperf03:5601/app/dashboards#/view/add7e1c0-722b-11ed-b525-bf4b90ad2742?\\_g=\(filters:!\)&refreshInterval:\(pause:!t,value:0\),time:\(from:'2022-12-31T18:30:00.000Z',to:'2023-01-31T09:05:06.149Z'\)](http://asviicsperf03:5601/app/dashboards#/view/add7e1c0-722b-11ed-b525-bf4b90ad2742?_g=(filters:!)&refreshInterval:(pause:!t,value:0),time:(from:'2022-12-31T18:30:00.000Z',to:'2023-01-31T09:05:06.149Z'))

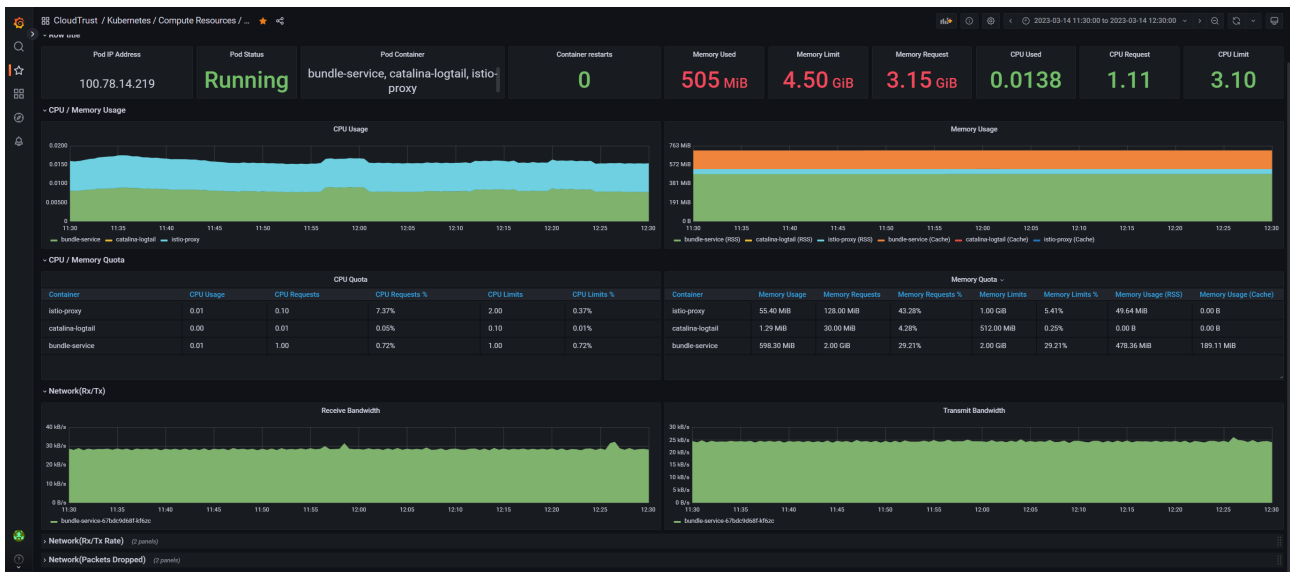


## 4 Grafana Screenshots of Dashboards:

we are building an selenium application which automates the manual work of taking screenshots of multiple dashboards for multiple services and pods.

Using this automation once the user provides the timestamp and pod-level details it takes all the screenshots and provides them on the local machine or on a wiki page.

This capturing and storing of screenshots helps us to later compare the results because the grafana data gets purged after some days.



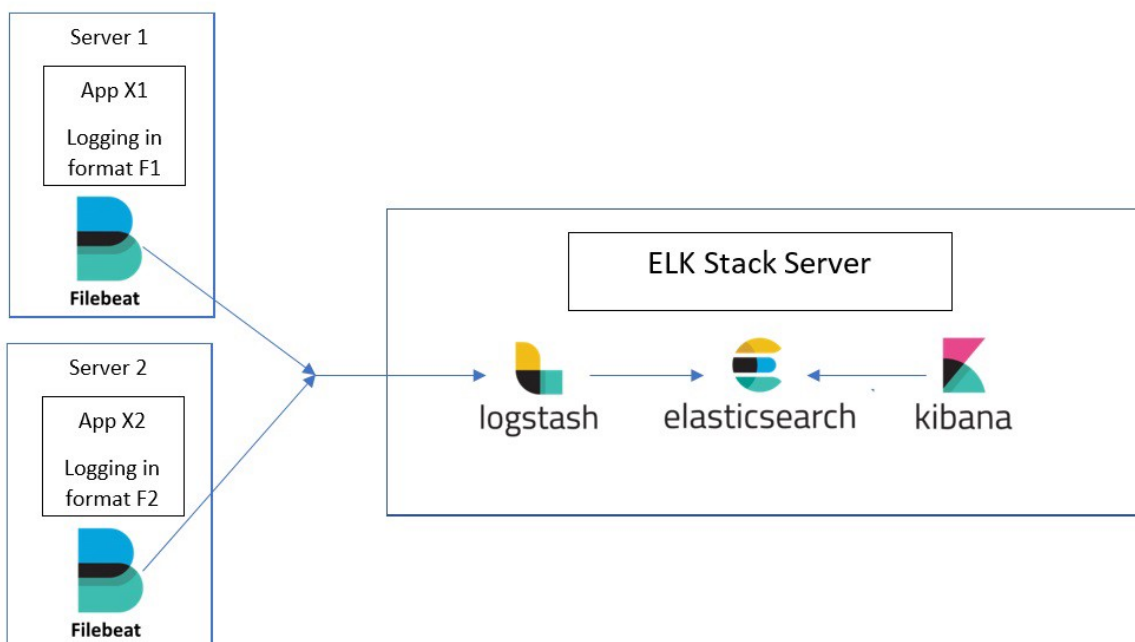
## 5 TO DO:

- Currently, we are using perf env for testing(shared environment) ,we are populating data in Dev-perf so that we can reduce the fluctuations in response time while doing regression testing.
- Grafana dashboards screenshots for one of the Dashboard are completed for all services. 2 more Dashboards need to be automated.

## 6 ELK Automation

### 6.1 ELK(Elastic Search, Logstash, Kibana):

The ELK stack is an acronym used to describe a stack that comprises of three popular projects: Elasticsearch, Logstash, and Kibana. Often referred to as Elasticsearch, the ELK stack gives you the ability to aggregate logs from all your systems and applications, analyze these logs, and create visualizations for application and infrastructure monitoring, faster troubleshooting, security analytics, and more



- **Filebeat** is a log shipper belonging to the Beats family – a group of lightweight shippers installed on hosts for shipping different kinds of data into the ELK Stack for analysis.
- **Logstash** is an open-source data ingestion tool that allows you to collect data from a variety of sources, transform it, and send it to your desired destination.
- **Elasticsearch** is a distributed search and analytics engine built on Apache Lucene. Support for various languages, high performance, and schema-free JSON documents makes Elasticsearch an ideal choice for various log analytics and search use cases.
- Kibana is a data visualization and exploration tool for reviewing logs and events. Kibana offers easy-to-use, interactive charts, pre-built aggregations and filters, and geospatial support and making it the preferred choice for visualizing data stored in Elasticsearch

#### Requirement:

- The csv which are generated from the jenkins are stored in a directory. These cvs logs are shipped to Logstash through filebeat. Initially the filebeat is used to ship these CSV files to Logstash

- The Logstash uses a config file to collect the data from filebeat and uses filter method to transform the data into required format from the CSV file received from filebeat and sends to elastic search in JSON Format.
- The Kibana is configured to get the data from the Elastic Search and an index is created in Kibana by Logstash and the CSV data is present in that particular Index.

## 6.2 Installation steps of ELK and Filebeat in Linux Remote Machine:

### 6.3 Filebeat

1. Give the below command in the linux directory where you want to install Filebeat.

**`sudo rpm --import https://packages.elastic.co/GPG-KEY-elasticsearch`**

2. Create file filebeat.repo

```
vi /etc/yum.repos.d/filebeat.repo
[elastic-8.x]
name=Elastic repository for 8.x packages
baseurl=https://artifacts.elastic.co/packages/8.x/yum
gpgcheck=1
gpgkey=https://artifacts.elastic.co/GPG-KEY-elasticsearch
enabled=1
autorefresh=1
type=rpm-md
```

3. Yum command to install filebeat

**`sudo yum install filebeat`**

4. Enable Filebeat

**`sudo systemctl enable Filebeat`**

**`sudo systemctl start Filebeat`**

5. To check the status and to stop the service use below commands

**`sudo systemctl status Filebeat`**

**`sudo systemctl stop Filebeat`**

### 6.4 Logstash

1. Give the below command in the linux directory where you want to install Logstash.

**`sudo rpm --import https://packages.elastic.co/GPG-KEY-elasticsearch`**

2. Yum command to install Logstash

**`sudo yum install logstash`**

4. Enable logstash

```
sudo systemctl enable logstash
```

```
sudo systemctl start logstash
```

5. To check the status and to stop the service use below commands

```
sudo systemctl status logstash
```

```
sudo systemctl stop logstash
```

## 6.5 Elastic Search

use the below commands in linux to install elastic search

```
wget https://artifacts.elastic.co/downloads/elasticsearch/elasticsearch-8.5.2-x86\_64.rpm
```

```
wget https://artifacts.elastic.co/downloads/elasticsearch/elasticsearch-8.5.2-x86\_64.rpm.sha512
```

```
shasum -a 512 -c elasticsearch-8.5.2-x86_64.rpm.sha512
```

```
sudo rpm --install elasticsearch-8.5.2-x86_64.rpm
```

1.Enable Elastic Search

```
sudo systemctl enable elasticsearch
```

```
sudo systemctl start elasticsearch
```

2. To check the status and to stop the service use below commands

```
sudo systemctl status elasticsearch
```

```
sudo systemctl stop elasticsearch
```

## 6.6 Kibana

use the below commands in linux to install Kibana

```
wget https://artifacts.elastic.co/downloads/kibana/kibana-8.5.2-x86\_64.rpm
```

```
shasum -a 512 kibana-8.5.2-x86_64.rpm
```

```
sudo rpm --install kibana-8.5.2-x86_64.rpm
```

1.Enable Kibana

```
sudo systemctl enable Kibana
```


```
sudo systemctl start Kibana
```

2. To check the status and to stop the service use below commands

```
sudo systemctl status Kibana
```

```
sudo systemctl stop Kibana
```

These are the files that require modification post installing in linux machine:

1	<div> elasticsearch.yml</div>	n e t w o r k .h o s t : "0 .0 .0 .0 " m a k e i t p u b l i c l y a c c e s s i b l e. [" <a href="#">http</a> :/ / 5 <a href="#">asviicsperf03</a> 6
---	--	---

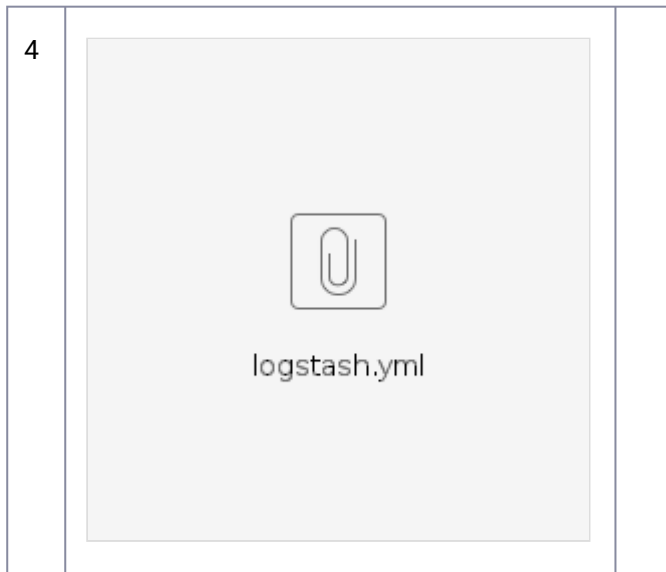
5 <http://asvlxrhperf33:9200>  
6 <http://asviicsperf03:5601/app/home#/>







2	<div><p>pipelines.yml</p></div>	<p>path.config: "/etc/logstash/conf.d/*.conf"</p>
---	--	---

		T e l w h e r e c o n f i l e c a n b e f o u n d
3	<div> logstash.conf</div>	m e n t i o n c o l u m n n a m e s



5	<div data-bbox="231 293 740 797">  <p>filebeat.yml</p> </div>	<p>paths:</p> <ul style="list-style-type: none"> <li>- /data/jenkins/jmeter-results/jtl/*.csv</li> </ul>
---	--	--

		te ll s to w hi c h lo c at io n th e fil e s n e e d s to b e re a d
--	--	---

6	 kibana.yml	server. host: "0. .0. .0. " Make it pub lic ly ac ce ssi ble.
---	---	---

		<a href="#">https://elasticsearch.ho.sts:[]https://</a>
--	--	---

	<div> <div>/</div> <div>8</div> <div>a</div> <div>s</div> <div>vi</div> <div>ic</div> <div>s</div> <div>p</div> <div>er</div> <div>f</div> <div>0</div> <div>3<sup>9</sup></div> <div>:9</div> <div>2</div> <div>0</div> <div>0<sup>10</sup></div> <div>"]</div> <div>G</div> <div>iv</div> <div>e</div> <div>c</div> <div>o</div> <div>nf</div> <div>ig</div> <div>ur</div> <div>e</div> <div>d</div> <div>el</div> <div>a</div> <div>st</div> <div>ic</div> <div>s</div> <div>e</div> <div>ar</div> <div>c</div> <div>h</div> <div>U</div> <div>R</div> <div>L</div> </div>
--	---

---

8 <http://asvlxrperf33:9200>

9 <http://asviicsperf03:5601/app/home#/>

10 <http://asvlxrperf33:9200>



## 7 Golang

```

type Response struct {
    Took      int    `json:"took"`
    TimedOut  bool   `json:"timed_out"`
    Shards    struct {
        Total      int    `json:"total"`
        Successful int    `json:"successful"`
        Skipped    int    `json:"skipped"`
        Failed     int    `json:"failed"`
    } `json:"_shards"`
    Hits struct {
        Total struct {
            Value      int    `json:"value"`
            Relation string `json:"relation"`
        } `json:"total"`
        Hits []struct {
            Index string `json:"_index"`
            ID    string `json:"_id"`
            Source struct {
                Message string `json:"message"`
                Log      struct {
                    File struct {
                        Path string `json:"path"`
                    } `json:"file"`
                    Offset int `json:"offset"`
                } `json:"log"`
            }
        } `json:"log"`
        Min      int    `json:"Min"`
        Throughput float64 `json:"Throughput"`
        Median    int    `json:"Median"`
        Tags      []string `json:"tags"`
        Error      string  `json:"Error"`
        ExecutionEndTime string `json:"Execution_End_Time"`
        ScriptName string  `json:"ScriptName"`
        TestDurationSeconds int `json:"Test_Duration_Seconds"`
        ExecutionStartTime string `json:"Execution_Start_Time"`
        Ecs struct {
            Version string `json:"version"`
        } `json:"ecs"`
        Version string `json:"@version"`
        Iteration int    `json:"Iteration"`
        Nine0Th int    `json:"90th"`
        ReceivedKBps float64 `json:"ReceivedKBps"`
        Timestamp time.Time `json:"@timestamp"`
        Event struct {
            Original string `json:"original"`
        } `json:"event"`
        Nine9Th int    `json:"99th"`
        Average int    `json:"Average"`
        Agent struct {

```

```

    Version      string `json:"version"`
    Type         string `json:"type"`
    EphemeralID  string `json:"ephemeral_id"`
    ID           string `json:"id"`
    Name         string `json:"name"`
} `json:"agent"`
Nine5Th        int `json:"95th"`
ReleaseNumber  int `json:"ReleaseNumber"`
Host           struct {
    IP []string `json:"ip"`
    Mac []string `json:"mac"`
    Os struct {
        Platform string `json:"platform"`
        Family    string `json:"family"`
        Name      string `json:"name"`
        Kernel    string `json:"kernel"`
        Codename  string `json:"codename"`
        Type      string `json:"type"`
        Version   string `json:"version"`
    } `json:"os"`
    Name      string `json:"name"`
    Architecture string `json:"architecture"`
    Containerized bool `json:"containerized"`
    ID        string `json:"id"`
    Hostname  string `json:"hostname"`
} `json:"host"`
Input struct {
    Type string `json:"type"`
} `json:"input"`
Samples      int `json:"Samples"`
BuildNumber  int `json:"BuildNumber"`
Max          int `json:"Max"`
UserLoadThreads int `json:"UserLoad_Threads"`
Label        string `json:"Label"`
} `json:"_source"`
} `json:"hits"`
} `json:"hits"`
}

```

```

var buffer bytes.Buffer
query := map[string]interface{}{
    "size": 2000,
    "query": map[string]interface{}{
        "bool": map[string]interface{}{
            "must": []map[string]interface{}{
                {
                    "match": map[string]interface{}{
                        "ReleaseNumber": release,
                    },
                },
                {
                    "match": map[string]interface{}{

```

```

    "BuildNumber": buildNum,
  },
},
},
},
}

```

## 7.1

### GoLang creating a build:

```

mkdir IICSRegression
git clone https://github.com/infa-mewilson/IICS\_Regression\_GO.git
go build -o IICSRegression .
nohup ./IICSRegression &
tail -f nohup.out

```

## 7.2

### GOLANG code:



IICS\_Regression\_GO-main.zip

## 8 Jenkins Automation

### 8.1 Jenkin Setup :

**Reference Page :** <https://www.linuxtechi.com/install-configure-jenkins-on-centos-7-rhel-7/>

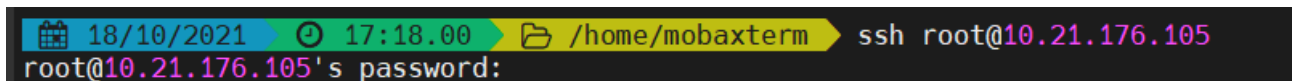
Steps to be followed to install Jenkins in the VM:

- 1) Login to the VM using the credentials and then type the command

**ssh username@ip\_address**

username: - username of your VM

ip\_address: - ip address of your VM



```
18/10/2021 17:18.00 /home/mobaxterm ssh root@10.21.176.105
root@10.21.176.105's password:
```

- 2) Then go to the desired folder where you want to install Jenkins.
- 3) Add Jenkins repository as they are not available in default in our VM. So run the below commands.

**# yum install -y wget** (only incase wget package is not present in your VM)

**# wget -O /etc/yum.repos.d/jenkins.repo** <http://pkg.jenkins.io/redhat-stable/jenkins.repo>

```
[root@asviicsperf01 Jenkins_Server]# wget -O /etc/yum.repos.d/jenkins.repo http://pkg.jenkins-ci.org/redhat-stable/jenkins.repo
--2021-10-18 17:29:05-- http://pkg.jenkins-ci.org/redhat-stable/jenkins.repo
Resolving pkg.jenkins-ci.org (pkg.jenkins-ci.org)... 52.202.51.185
Connecting to pkg.jenkins-ci.org (pkg.jenkins-ci.org)[52.202.51.185]:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 85
Saving to: '/etc/yum.repos.d/jenkins.repo'

100%[=====] 85 --.-K/s in 0s

2021-10-18 17:29:05 (23.0 MB/s) - '/etc/yum.repos.d/jenkins.repo' saved [85/85]

[root@asviicsperf01 Jenkins_Server]# sudo yum install jenkins
Loaded plugins: enabled_repos_upload, langpacks, package_upload, product-id, search-disabled-repos, subscription-manager
epel/x86_64/metalink | 10 kB 00:00:00
rhel-7-server-devtools-rpms | 2.4 kB 00:00:00
rhel-7-server-extras-rpms | 2.0 kB 00:00:00
rhel-7-server-optional-rpms | 2.0 kB 00:00:00
rhel-7-server-rh-common-rpms | 2.1 kB 00:00:00
rhel-7-server-rhn-tools-rpms | 2.1 kB 00:00:00
rhel-7-server-rpms | 2.0 kB 00:00:00
rhel-7-server-satellite-tools-6.7-rpms | 2.1 kB 00:00:00
rhel-7-server-satellite-tools-6.8-rpms | 2.1 kB 00:00:00
rhel-7-server-supplementary-rpms | 2.0 kB 00:00:00
rhel-server-rhsc-7-rpms | 2.0 kB 00:00:00
Resolving Dependencies
--> Running transaction check
--> Package jenkins.noarch 0:2.303.2-1.1 will be installed
--> Processing Dependency: daemonize for package: jenkins-2.303.2-1.1.noarch
--> Running transaction check
--> Package daemonize.x86_64 0:1.7.7-1.el7 will be installed
--> Finished Dependency Resolution

Dependencies Resolved

=====
Package Arch Version Repository Size
=====
Installing:
jenkins noarch 2.303.2-1.1 jenkins 69 M
Installing for dependencies:
=====

Dependencies Resolved

=====
Package Arch Version Repository Size
=====
Installing:
jenkins noarch 2.303.2-1.1 jenkins 69 M
Installing for dependencies:
daemonize x86_64 1.7.7-1.el7 epel 21 k
=====

Transaction Summary
-----
Install 1 Package (+1 Dependent package)

Total download size: 69 M
Installed size: 69 M
Is this ok [y/d/N]: y
Downloading packages:
(1/2): daemonize-1.7.7-1.el7.x86_64.rpm | 21 kB 00:00:00
(2/2): jenkins-2.303.2-1.1.noarch.rpm | 69 MB 00:00:25
-----
Total 2.7 MB/s | 69 MB 00:00:25

Running transaction check
Running transaction test
Transaction test succeeded
Running transaction
Warning: RPMDB altered outside of yum.
Installing : daemonize-1.7.7-1.el7.x86_64 1/2
Installing : jenkins-2.303.2-1.1.noarch 2/2
Uploading Package Profile
Verifying : daemonize-1.7.7-1.el7.x86_64 1/2
Verifying : jenkins-2.303.2-1.1.noarch 2/2

Installed:
jenkins.noarch 0:2.303.2-1.1

Dependency Installed:
daemonize.x86_64 0:1.7.7-1.el7

Complete!
Uploading Enabled Repositories Report
```

4) Install Jenkins and Java 11(Note: The latest jenkins supports java versions >11)

5) Run the following systemctl commands to start and enable the jenkins service

```
# systemctl start jenkins
```

```
# systemctl enable Jenkins
```

```
[root@asviicsperf01 Jenkins_Server]# sudo systemctl start jenkins
[root@asviicsperf01 Jenkins_Server]# sudo systemctl status jenkins
● jenkins.service - LSB: Jenkins Automation Server
   Loaded: loaded (/etc/rc.d/init.d/jenkins; bad; vendor preset: disabled)
   Active: active (running) since Mon 2021-10-18 17:32:12 IST; 21s ago
     Docs: man:systemd-sysv-generator(8)
   Process: 30564 ExecStart=/etc/rc.d/init.d/jenkins start (code=exited, status=0/SUCCESS)
   CGroup: /system.slice/jenkins.service
           └─30568 /etc/alternatives/java -Djava.awt.headless=true -DJENKINS_HOME=/var/lib/jenkins -jar /usr/lib/jenkins/jenkins.war --logfile=/var/l...

Oct 18 17:32:12 asviicsperf01.informatica.com systemd[1]: Starting LSB: Jenkins Automation Server...
Oct 18 17:32:12 asviicsperf01.informatica.com jenkins[30564]: Starting Jenkins [ OK ]
Oct 18 17:32:12 asviicsperf01.informatica.com systemd[1]: Started LSB: Jenkins Automation Server.
```

6) Open the ports (8080) in OS firewall.

In case firewall is enabled on your Linux server then run the following commands to open Jenkins related ports like 80 and 8080.

```
# firewall-cmd --zone=public --add-port=8080/tcp --permanent
```

```
# firewall-cmd --zone=public --add-service=http --permanent
```

# firewall-cmd --reload

```
[root@asviicsperf01 ~]# firewall-cmd --zone=public --add-port=8080/tcp --permanent
Warning: ALREADY_ENABLED: 8080:tcp
success
[root@asviicsperf01 ~]# firewall-cmd --zone=public --add-service=http --permanent
Warning: ALREADY_ENABLED: http
success
[root@asviicsperf01 ~]# firewall-cmd --reload
success
```

7) Then access the Jenkins web portal using the URL : `http://<Ip-Address-of-your-Server>:8080`

8) Admin password is created and stored in the log file `\secrets\initial\AdminPassword`

Copy the password and paste it in above windows and click on Continue.

9) In the next window Select the option : **Install suggested plugins**

10) Once the plugin installation was done then it will ask to create Admin User. Click on Save and Finish. Click on **"Start using Jenkins"**. Now the Jenkins is ready to use.

#### Access the Jenkins Web Portal :

Open the Browser and Access the URL : `http://<Ip-Address-of-your-Server>:8080`

Configure option has all setting required.

**Execute shell script:**

```

#Parameterize
#JMeter Bin PATH
JMETER_BIN_PATH=/data/Dashboard_Automation/Jmeter/apache-jmeter-5.5/bin

#JMeter JTL PATH
JMETER_JTL_PATH=/data/Dashboard_Automation/Jtl_files

#JMeter Aggregate PATH
JMETER_AGGREGATE_PATH=/data/Dashboard_Automation/Aggregate_files

#JMeter Lib PATH
JMETER_LIB_PATH=/data/Dashboard_Automation/Jmeter/apache-jmeter-5.5/lib

#JmeterScript PATH
JMETER_SCRIPT_PATH=/data/Dashboard_Automation/Scripts/newPlan_Linux.jmx

#Navigate to JMETER BIN PATH
cd $JMETER_BIN_PATH/ || exit

#rm -rf perf-username-password-tenantId-bearerToken.csv
#ScriptName_Trim=$(echo "${ScriptName}" | cut -f 1 -d '.')
ScriptName_Trim=newPlan_Linux
ScriptName=newPlan_Linux.jmx

#Configure-JTL-Results-FileName
# Desired JTL Path + ScriptName + Environment + ServiceName + Release + Build + Iteration + Threads +
Duration + Build_Number + CSV file suffix
JTL_Results_File_Name=$JMETER_JTL_PATH/${Result_Filename}.jtl
echo "${JTL_Results_File_Name}"

if [ -f "${JTL_Results_File_Name}" ];
then
echo "The Run is already Present with current Release and Build,Iteration"
echo "--Re-enter the Correct BuildNumber and ReleaseNumber"
else
#give permissions to write and read the file
#chmod 755 /data/Dashboard_Automation/Test_Data/Schedulerfile.csv
chmod 755 /data/Dashboard_Automation/Test_Data/UserIdfile.csv
chmod 755 /data/Dashboard_Automation/Test_Data/UserIdfile100.csv

```

```

#delete the previous csv files before execution
#rm -rf /data/Dashboard_Automation/Test_Data/Schedulerfile.csv
rm -rf /data/Dashboard_Automation/Test_Data/UserIdfile.csv
rm -rf /data/Dashboard_Automation/Test_Data/UserIdfile100.csv

#Execution-Start-Time
#Execution_Start_Time=$(date '+%Y/%m/%dT%H:%M:%S')
Execution_Start_Time=$(date --iso-8601=seconds)
StartTime=$(date +%s)

#Execute Testrun
sh ${JMETER_BIN_PATH}/jmeter.sh -Djavax.net11.ssl.keyStoreType=jks -Djavax.net12.ssl.keyStore=${
{JMETER_BIN_PATH}/metering-service-keystore.jks -Djavax.net13.ssl.keyStorePassword=changeit -n -t $
{JMETER_SCRIPT_PATH} -Jorg_name_prefix=$org_name_prefix -Jthread_numLoad=$thread_numLoad
-Jthread_num=$thread_num -Jramp_time=$ramp_time -Jloop_num=$loop_num -Jma_ip=$ma_ip
-Jserver_ip=$server_ip -Jpdm_ip=$pdm_ip -Jusername=$username -Jpassword=$password
-Jcon_username=$con_username -Jcon_password=$con_password -JFRS_username=$FRS_username
-JFRS_password=$FRS_password -JMig_username=$Mig_username -JMig_password=$Mig_password
-JAdmin_password=$Admin_password -JAudit_username=$Audit_username
-JAudit_password=$Audit_password -JFileBasePath=$FileBasePath -Jsch_username=$sch_username
-Jsch_password=$sch_password -JV3_username=$V3_username -JV3_password=$V3_password
-Jthread_num_notification=$thread_num_notification -Jloop_num_notification=$loop_num_notification
-Jvcs_username=$vcs_username -Jvcs_password=$vcs_password -JAC_username=$AC_username
-JAC_password=$AC_password -Jscim_username=$scim_username -Jscim_password=$scim_password
-JUpref_username=$Upref_username -JUpref_password=$Upref_password
-Jnotification_expiry=$notification_expiry -Jloop_numLoad=$loop_numLoad
-Jramp_timeLoad=$ramp_timeLoad -Jusername100=$username100 -Jpassword100=$password100
-Jnotification1500org_username=$notification1500org_username
-Jnotification1500org_password=$notification1500org_password -Jusernamepdm=$usernamepdm
-Jpasswordpdm=$passwordpdm -JtimeZoneId=$timeZoneId -JAgentGroupName=$AgentGroupName
-Jproject_prefix=$project_prefix -Jfolder_prefix=$folder_prefix -Jschedule_prefix=$schedule_prefix
-Jschedule_prefix100=$schedule_prefix100 -JSchedule_start_time=$Schedule_start_time -l "$
{JTL_Results_File_Name}"

#Execution-End-Time
#Execution_End_Time=$(date '+%Y/%m/%dT%H:%M:%S')
Execution_End_Time=$(date --iso-8601=seconds)
EndTime=$(date +%s)
TestDuration=$((EndTime - $StartTime))
echo $TestDuration

```

---

11 <http://Djavax.net>

12 <http://Djavax.net>

13 <http://Djavax.net>



```

#If user want to persist data to elastic search then he needs to check box the persist data variable while
building
if [ "${PersistData}" == "true" ];
then
#Configure-Temp-Aggregate-Results-FileName
Aggregate_Temp_Results_File_Name=${JMETER_AGGREGATE_PATH}/tmp/${ScriptName_Trim-
aggregate_Release_${Release_Num}_build_${Build}.csv
echo "${Aggregate_Temp_Results_File_Name}"

#temporary csv
Aggregate_Temp_Results_File_Name_actual=${JMETER_AGGREGATE_PATH}/tmp/${Result_Filename}
_actual.csv
echo "${Aggregate_Temp_Results_File_Name_actual}"

#Configure-Aggregate-Results-FileName
Aggregate_Results_File_Name=/data/Dashboard_Automation/CSVLogs/${Result_Filename}.csv
echo "${Aggregate_Results_File_Name}"

#Generate aggregate report from jtl
java -jar ${JMETER_LIB_PATH}/ext/cmdrunner-2.0.jar --tool Reporter --generate-csv "$
{Aggregate_Temp_Results_File_Name}" --input-jtl "${JTL_Results_File_Name}" --plugin-type
AggregateReport

#sudo cp ${JTL_Results_File_Name} ${WORKSPACE}
chmod 755 ${Aggregate_Temp_Results_File_Name}

#Add custom tags to aggregate report except header
sed "1 ! s/.*&${Release_Num},${Release_Iteration},${Build},${loop_num},${ScriptName_Trim},${
thread_num},${TestDuration},${Execution_Start_Time},${Execution_End_Time}/" "$
{Aggregate_Temp_Results_File_Name}" > "${Aggregate_Temp_Results_File_Name_actual}"

#Delete 1st line i.e old header file
sed -i '1d' "${Aggregate_Temp_Results_File_Name_actual}"

#Add Filebeat/Logstash/Kibana complaint Aggregate header
sed -i '1
i\Label,Samples,Average,Median,90th,95th,99th,Min,Max,Error,Throughput,ReceivedKBps,StdDev,Release
Number,Release_Iteration,BuildNumber,Loop_Count,ScriptName,UserLoad_Threads,Test_Duration_Secon
ds,Execution_Start_Time,Execution_End_Time' "${Aggregate_Temp_Results_File_Name_actual}"

#delete the last line of the csv
sed -i '$d' "${Aggregate_Temp_Results_File_Name_actual}"

#Copy the updated aggregate report to aggregate staging location
cp "${Aggregate_Temp_Results_File_Name_actual}" "${Aggregate_Results_File_Name}"

```

```
#Now delete the previous csv files to save space in MGMTNode
#sudo chmod 755 $LastbuildTemporaryActualFilename
#sudo chmod 755 $LastbuildTemporaryFilename
#sudo chmod 755 $LastbuildAggregateFilename

#sudo rm -rf $LastbuildTemporaryActualFilename
#sudo rm -rf $LastbuildTemporaryFilename
#sudo rm -rf $LastbuildAggregateFilename
fi
fi
```

## 9 New Automation Suite onboarding in IICS automation suite

### 1. Jenkins

- a. create new Folder under Jenkins Dashboard (<http://asviicsperf03:8080/>)
- b. Build project with required parameters and build configuration.
- c. Conversion of jtl to csv refer Build Steps of this configure file([http://asviicsperf03:8080/job/IICS\\_Regression/job/IICS\\_Suite/configure](http://asviicsperf03:8080/job/IICS_Regression/job/IICS_Suite/configure))
- d. Create separate JTL and Aggregate file locations in asviicsperf03 box, so that the files are read from this directory by ELK stack and can be visualized in kibana in a different index.

### 2. JMeter

- a. Add your script to path - /data/Dashboard\_Automation/Scripts

### 3. Filebeat

- a. create a new filestream type under filebeat input section in filebeat.yml file

```
filebeat.inputs:
- type: filestream
  # Change to true to enable this input configuration.
  enabled: true
  id: jtl
  #ignore_older: 15m
  #close_inactive: 5m
  # Paths that should be crawled and fetched. Glob based paths.
  paths:
    - /data/jenkins/jmeter-results/jtl/*.csv
  tags: ["complete_log"]
```

- b. Add custom tag to uniquely identify the filestream directory in logstash.conf file (to which index it needs to be pushed)

### 4. Logstash

- a. **Note: When creating a new index in the etc/logstash/conf.d/logstash.conf file the index name must be in *small letters*.**
- b. **parameter in the output section *index => "iics-regression"***

### 5. Kibana

- a. No changes required

### 6. Elastic Search

- a. No changes required

### 7. GoCode comparison

Note :

1. In order to check the newly created index "[http://asviicsperf03:5601/app/management/data/index\\_management/indices](http://asviicsperf03:5601/app/management/data/index_management/indices)", create a data view for this index in "<http://asviicsperf03:5601/app/management/kibana/dataViews>" create a new Data View .
2. If any yml/configuration changes are done then restart all the 4 services  
systemctl stop <service Name>  
systemctl start <service Name>
3. To check the status "systemctl status <service Name>"
4. To view the logs of particular service "systemctl status <service name> -l"
5. Also add data into the /data/Dashboard\_Automation/CSVLogs folder in order to view the newly created index in kibana Index management