# Practice M7: Elastic Stack

During this practice we will assume that we are working in Windows environment. It could be either a physical machine or a virtual one

Additionally, you must have **Vagrant** and **VirtualBox** (preferably) or another virtualization solution installed

All steps can be executed in most **Linux** distributions and/or **macOS** (Intel-based) environment as well

The lab infrastructure will vary during different parts of the module. For each part there is one **Vagrantfile** file

## Part 1: Elastic Stack

First, bring the just a part of the environment up with

**vagrant up server**

Then open a session to it

**vagrant ssh server**

Before version **8.x** we had to take care of ensuring that there is a supported version of **Java** installed

As of now, this is not our responsibility anymore as it comes integrated with the software

There are two valid ways to install **Elastic Stack** components – first is to add the repository and then install the packages, and the second is to download the package directly and then install it

#### Install Elasticsearch (on CentOS)

##### Repository

In order to follow this path, we must follow this set of steps

First, import the **Elasticsearch** GPG key

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

*You may see an error stating* ***Warning: Signature not supported. Hash algorithm SHA1 not available.***

*If this happens, then the only way to go is to allow using of legacy crypto policies:*

***sudo update-crypto-policies --set LEGACY***

Then, create a repository file **/etc/yum.repos.d/elasticsearch.repo** with the following content

**[elasticsearch]**

**name=Elasticsearch 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=0**

**autorefresh=1**

**type=rpm-md**

Please note that the repository is disabled by default. This is to prevent accidental update

And finally, install the package

**sudo dnf install --enablerepo=elasticsearch elasticsearch**

##### Direct

This is the easier and quicker installation method

First, download the package with

**wget https://artifacts.elastic.co/downloads/elasticsearch/elasticsearch-8.6.2-x86\_64.rpm**

Then, install it

**sudo rpm -Uvh elasticsearch-\*.rpm**

#### Install Elasticsearch (on Debian / Ubuntu) \*

*\* This section is provided just for completeness. For the purpose of this practice, we will skip it*

##### Repository

The steps are the following

Download and install the public key

**wget -qO - https://artifacts.elastic.co/GPG-KEY-elasticsearch | sudo apt-key add -**

Save the repository definition

**echo "deb https://artifacts.elastic.co/packages/8.x/apt stable main" | sudo tee /etc/apt/sources.list.d/elastic-8.x.list**

And then install the package

**sudo apt-get update && sudo apt-get install elasticsearch**

##### Direct

Direct installation is quicker

Download the package

**wget https://artifacts.elastic.co/downloads/elasticsearch/elasticsearch-8.6.2-amd64.deb**

And install it

**sudo dpkg -i elasticsearch\*.deb**

#### Post-installation Steps for Elasticsearch

*Note that on package installations, the* ***security mode*** *is* ***enabled*** *by default*

*This means that* ***HTTPS*** *is used instead of* ***HTTP***

*And there is a password set for the* ***elastic*** *(the only available) user. It is displayed during the installation process*

Once that we have the software installed, we can do the following

We can adjust the IP address that **Elasticsearch** will be bound to. Open the file:

**sudo vi /etc/elasticsearch/elasticsearch.yml**

And adjust the **network.host** parameter (line #56)

**network.host: ["localhost", "192.168.99.101"]**

And then the **http.port** parameter (line #61)

**http.port:9200**

We could adjust two other parameters as well

Change the **cluster.name** parameter (line #17) for example to

**cluster.name: mycluster**

And **node.name** parameter (line #23) to

**node.name: server**

Finally, don’t forget to change the **cluster.initial\_master\_nodes** parameter (line #74) to

**cluster.initial\_master\_nodes: ["server"]**

*Please keep in mind, that the above may already be* ***present at the end of the file*** *(line #115). So, make sure that you do not end up with* ***two entries*** *of the same setting as this will* ***prevent*** *the service from* ***starting***

Save and close the file

In addition, we can change the Java heap size either by editing **/etc/elasticsearch/jvm.options** file or creating a new **jvm.options** file in the **/etc/elasticsearch/jvm.options.d/** folder

For example, if our virtual machine has 4 GB of RAM, we should limit the heap to 50% of this

So, we can create **/etc/elasticsearch/jvm.options.d/jvm.options** file with the following content

**-Xms2g**

**-Xmx2g**

Save and close the file

Reload the services

**sudo systemctl daemon-reload**

Enable and start the service

**sudo systemctl enable elasticsearch**

**sudo systemctl start elasticsearch**

If we want to examine the events in the system log for the service either all, or after a specific point of time:

**sudo journalctl --unit elasticsearch**

**sudo journalctl --unit elasticsearch --since "2023-03-13 18:30:00"**

We can check if the service is responsive with:

**curl https://localhost:9200**

*Please note that depending on the configuration, we may use just* ***http*** *instead of* ***https***

*Furthermore, we should* ***specify*** *the* ***user*** *and either ignore the certificate check*

***curl -k -u elastic https://localhost:9200***

*Or point the command to the CA certificate*

***sudo curl --cacert /etc/elasticsearch/certs/http\_ca.crt -u elastic https://localhost:9200***

If the firewall is on, either disable it, or open the appropriate port(s) if you plan to access the software from outside

**sudo firewall-cmd --add-port 9200/tcp --permanent**

**sudo firewall-cmd --reload**

#### Explore Elasticsearch

Exploration is always a good way to become familiar with something

There is a **REST API** with which we can interact

Get cluster status

**curl -k -u elastic https://localhost:9200/\_cat/health?v**

Entering the password every time is a daunting task

Instead, we can export two variables

**export EUSR='elastic'**

**export EPWD='<password>'**

And use curl with the following arguments

**curl -k -XGET -u "${EUSR}:${EPWD}" https://localhost:9200/\_cat/health?v**

Get nodes information

**curl -k -XGET -u "${EUSR}:${EPWD}" https://localhost:9200/\_cat/nodes?v**

List all indices

**curl -k -XGET -u "${EUSR}:${EPWD}" https://localhost:9200/\_cat/indices?v**

There aren’t any

Create an index

**curl -k -XPUT -u "${EUSR}:${EPWD}" https://localhost:9200/customer?pretty**

List all indices again

**curl -k -XGET -u "${EUSR}:${EPWD}" https://localhost:9200/\_cat/indices?v**

Now, we can see one index

Create a simple document

**curl -k -XPUT -u "${EUSR}:${EPWD}" https://localhost:9200/customer/\_doc/1?pretty \**

**-H 'Content-Type: application/json' -d '{ "name": "John Doe" }'**

Retrieve the document

**curl -k -XGET -u "${EUSR}:${EPWD}" https://localhost:9200/customer/\_doc/1?**

Or with better formatted output

**curl -k -XGET -u "${EUSR}:${EPWD}" https://localhost:9200/customer/\_doc/1?pretty**

Create one more

**curl -k -XPUT -u "${EUSR}:${EPWD}" https://localhost:9200/customer/\_doc/2?pretty \**

**-H 'Content-Type: application/json' -d '{ "name": "Jane Smith" }'**

*Instead of the above, we can use this command*

***curl -k -XPOST -u "${EUSR}:${EPWD}" https://localhost:9200/customer/\_doc/?pretty \***

***-H 'Content-Type: application/json' -d '{ "name": "Jane Smith" }'***

*But this way, we leave the identity creation to* ***Elasticsearch***

Now, list the documents in the index

**curl -k -XGET -u "${EUSR}:${EPWD}" https://localhost:9200/customer/\_search?pretty**

And delete the first document

**curl -k -XDELETE -u "${EUSR}:${EPWD}" https://localhost:9200/customer/\_doc/1?pretty**

List them again

**curl -k -XGET -u "${EUSR}:${EPWD}" https://localhost:9200/customer/\_search?pretty**

Now delete the whole index

**curl -k -XDELETE -u "${EUSR}:${EPWD}" https://localhost:9200/customer?pretty**

Ask again for the list of indices

**curl -k -XGET -u "${EUSR}:${EPWD}" https://localhost:9200/\_cat/indices?v**

#### Turn Off the Security Mode of Elasticsearch

As of version **8.0**, by default the **security mode** is enabled

This will require special attention and extra stuff, and it will interfere with our learning process

So, we will turn this mode off

For this, we must open the main configuration file

**sudo vi /etc/elasticsearch/elasticsearch.yml**

Go to row #98 and change it from ***true*** to ***false***

**xpack.security.enabled: false**

Then save the file and close it

Do not forget to restart the service

**sudo systemctl restart elasticsearch**

#### Install Logstash (on CentOS)

We can apply the selected approach for the **Elasticsearch** installation here as well

If we have the repository added, then we need to execute just the following

**sudo dnf install --enablerepo=elasticsearch logstash**

Or if we decided to go with the package download, then again, we can repeat the procedure and download it with

**wget https://artifacts.elastic.co/downloads/logstash/logstash-8.6.2-x86\_64.rpm**

Finally, we can install the package with:

**sudo rpm -Uvh logstash-\*.rpm**

#### Install Logstash (on Debian / Ubuntu) \*

\* This section is provided just for completeness. For the purpose of this practice, we will skip it

We can apply the selected approach for the Elasticsearch installation here as well

If we have the repository added, then we need to execute just the following

**sudo apt-get install logstash**

If we decided to go with the package download, then again, we can repeat the procedure and download it with

**wget https://artifacts.elastic.co/downloads/logstash/logstash-8.6.2-amd64.deb**

Finally, we can install the package with:

**sudo dpkg -i logstash-\*.deb**

#### Explore Logstash

Go to **/usr/share/logstash/bin** and execute

**sudo ./logstash -e 'input { stdin { } } output { stdout {} }'**

Wait a while until you see some messages on your screen *(for example, something like* ***Pipelines running****)*

Type a few words and press the **Enter** key

You should see a message that what you typed is accepted

Okay, our service is working, press **Ctrl+C** to stop it (instead, we can press **Ctrl+D** to stop the service)

Now let’s send a few messages to the **Elasticsearch** service

Start the **logstash** service with

**sudo ./logstash -e 'input { stdin {} } output { elasticsearch { hosts => ["localhost:9200"] } }'**

Wait a while until you see some messages on your screen *(for example, something like* ***Pipelines running****)*

Again, type a few words and press the **Enter** key

Open a second terminal session and check the list of available indexes

**curl -X GET http://localhost:9200/\_cat/indices?v**

Ask the **Elasticsearch** for the entered data

**curl -X GET http://localhost:9200/.ds-logs-\*/\_search**

We should see some output but not in the most readable form

Execute this instead

**curl -X GET http://localhost:9200/.ds-logs-\*/\_search?pretty**

Now, we can easily see the messages we typed on the console

Return in the first session and hit **Ctrl+C** (or **Ctrl+D**) to stop the service

#### Post-installation Steps for Logstash

Now, that we have a basic idea how and if the **Logstash** is working, we can setup the service

We can alter the settings in this file **/etc/logstash/jvm.options**

For example, set the ***initial*** and ***maximum*** heap size to ***512m***

And then, reload the services

**sudo systemctl daemon-reload**

Enable and start the service

**sudo systemctl enable logstash**

**sudo systemctl start logstash**

#### Install Kibana (on CentOS)

We can apply the selected approach for the **Elasticsearch** installation here as well

If we have the repository added, then we need to execute just the following

**sudo dnf install --enablerepo=elasticsearch kibana**

If we decided to go with the package download, then again, we can repeat the procedure and download it with

**wget https://artifacts.elastic.co/downloads/kibana/kibana-8.6.2-x86\_64.rpm**

Finally, we can install the package with

**sudo rpm -Uvh kibana-\*.rpm**

#### Install Kibana (on Debian / Ubuntu) \*

\* This section is provided just for completeness. For the purpose of this practice, we will skip it

We can apply the selected approach for the **Elasticsearch** installation here as well

If we have the repository added, then we need to execute just the following

**sudo apt-get install kibana**

If we decided to go with the package download, then again, we can repeat the procedure and download it with

**wget https://artifacts.elastic.co/downloads/kibana/kibana-8.6.2-amd64.deb**

Finally, we can install the package with:

**sudo dpkg -i kibana-\*.deb**

#### Post-Installation Steps for Kibana

Before we start our interaction with **Kibana**, we must take care for a few additional steps

Open the main configuration file **/etc/kibana/kibana.yml** for editing

Adjust the values of the following settings if needed

(line #6) **server.port: 5601**

(line #11) **server.host: "192.168.99.101"**

(line #32) **server.name: "server"**

(line #43) **elasticsearch.hosts: ["http://localhost:9200"]**

*In our current setup – all services on one machine, we can leave them as they are – commented, or just adjust the* ***server.host*** *and set it to something like* ***server.host: 192.168.99.101***

Save and close the file

Reload the services

**sudo systemctl daemon-reload**

Enable and start the service

**sudo systemctl enable kibana**

**sudo systemctl start kibana**

If the firewall is on, either disable it, or open the appropriate port(s) if you plan to access the software from outside

**sudo firewall-cmd --add-port 5601/tcp --permanent**

**sudo firewall-cmd --reload**

#### Explore Kibana

The time has come. Let’s get to know **Kibana**

Open a browser tab and navigate to **http://192.168.99.101:5601**

Keep in mind that it may take some time for **Kibana** to start

*Even if after quite some time you are not able to access the UI, you should check the following*

* *Service status*

***systemctl status kibana***

* *Log file contents (in case the service output is not enough)*

***sudo cat /var/log/kibana/kibana.log***

* *The status of the network interface on your host (especially if Linux-based)*

***ip a***

*And if down, bring it up and assign an address to it (for example, on vboxnet1)*

***sudo ip link set vboxnet1 up***

***sudo ip address add 192.168.99.1/24 dev vboxnet1***

Once in, click the **Explore on my own** button

Let’s continue exploring **Kibana**, by adding a sample data and visualization set

Click on the **Try sample data** button

Click on the **Other sample data sets** link

Pick up the **Sample flight data** option and click on the **Add data** button

Once installed, click on **View data**, and then on **Dashboard** to explore a bit

*If you want to add another sample set, click on the menu button on the top left, then on* ***Home***

*Then again select* ***Try sample data*** *and then* ***Other sample data sets****, and choose another one*

We can also see our initial test message(s) that we stashed during the **Logstash** exploration

Open the menu and under the **Management** section click on **Stack Management**

Click on **Index Management** under **Data**

Here, we can see both indices - the one for the sample data, and ours created during experiments with **Logstash**

*If it is only one (the sample one), try to turn on the switch* ***Include hidden indices***

The one from **Logstash** should be named like ***.ds-logs-generic-default-\****

Now, again in the menu, choose **Data Views** under **Kibana** section

And then on **Create data view**

Click the **Show advanced settings** link

Then turn on the **Allow hidden and system indices** option

Now in the **Name** field, enter **Logstash Data**

Then, in the **Index pattern** field, enter **.ds-logs\***

This will select only the hidden index that comes from **Logstash**

*Alternatively, we can enter* ***logs-generic-\****

*This will select the whole stream*

Select whichever you want

Under the **Timestamp field** select **@timestamp**

Click the **Save data view to Kibana** button

Now, we can examine the structure of our new data view, and mark it as default (click on the star icon in the top-right corner)

Now, let’s explore the content of our view

Switch to the **Discover** section (in the menu click on **Analytics > Discover**)

From the drop-down menu select **Logstash Data** (if you marked it as default, it should be already selected)

Examine the data collected for our event(s)

If no data is shown, try to adjust the time range (top-right corner)

This is it for now 😊

## Part 2: Beats

Extend the environment by executing the following on the host

**vagrant up node1 node2**

#### Heartbeat (on the server)

Depending on the way we choose to install the software, we have two options for the beats as well – repository or individual packages

We will continue following the latter approach and will download and install packages

On a **CentOS** machine we must do

**wget https://artifacts.elastic.co/downloads/beats/heartbeat/heartbeat-8.6.2-x86\_64.rpm**

**sudo rpm -Uvh heartbeat-8.6.2-x86\_64.rpm**

On a **Debian** / **Ubuntu** machine we can install it with

**wget https://artifacts.elastic.co/downloads/beats/heartbeat/heartbeat-8.6.2-amd64.deb**

**sudo dpkg -i heartbeat-8.6.2-amd64.deb**

Now that we have it installed, we must configure it

Open the main configuration file

**sudo vi /etc/heartbeat/heartbeat.yml**

There is a default but disabled **HTTP** monitor

So, just below its definition (around line 39), we can add the following

**- type: icmp**

**id: icmp-monitor**

**name: ICMP Monnitor**

**schedule: '\*/30 \* \* \* \* \* \*'**

**hosts: ["192.168.99.101", "192.168.99.102", "192.168.99.103"]**

Then, we should disable the **Elasticsearch** output (around line #106) and enable the **Logstash** output (around line #119)

Don’t worry about the **5044** port, we will deal with this in a bit

Save and close the file

Test the configuration with

**sudo heartbeat test config**

Install the beat’s template in **Elasticsearch**

**sudo heartbeat setup --index-management -E output.logstash.enabled=false -E 'output.elasticsearch.hosts=["localhost:9200"]'**

Go to **/usr/share/logstash/bin** and list all installed plugins

**./logstash-plugin list**

Our plugin (**logstash-input-beats**) is listed, so we can create a configuration file

**sudo vi /etc/logstash/conf.d/beats.conf**

Enter the following and save

**input {**

**beats {**

**port => 5044**

**}**

**}**

**output {**

**elasticsearch {**

**hosts => ["http://localhost:9200"]**

**index => "%{[@metadata][beat]}-%{[@metadata][version]}-%{+YYYY.MM.dd}"**

**}**

**}**

Restart **Logstash** service

**sudo systemctl restart logstash**

Start and enable the **Heartbeat** service

**sudo systemctl daemon-reload**

**sudo systemctl enable heartbeat-elastic**

**sudo systemctl start heartbeat-elastic**

Go to **Kibana** UI, add the new data view (**heartbeat-\***), and explore the messages

For example, open a message and check its details

Or filter for the fields that starts with **monitor**

Then, move on to the next beat

#### Metricbeat (on nodes)

Depending on the way we choose to install the software, we have two options for the beats as well – repository or individual packages

We will continue following the latter approach and download and install individual packages

Establish a session to node 1 (**node1**)machine

On a **CentOS** machine we must do

**wget https://artifacts.elastic.co/downloads/beats/metricbeat/metricbeat-8.6.2-x86\_64.rpm**

**sudo rpm -Uvh metricbeat-8.6.2-x86\_64.rpm**

On a **Debian** / **Ubuntu** machine we can install it with

**wget https://artifacts.elastic.co/downloads/beats/metricbeat/metricbeat-8.6.2-amd64.deb**

**sudo dpkg -i metricbeat-8.6.2-amd64.deb**

Now that we have it installed, we must configure it

Open the main configuration file

**sudo vi /etc/metricbeat/metricbeat.yml**

Change the following

Disable the **Elasticsearch** output (line #92) and enable the **Logstash** output (line #105)

Don’t forget to substitute the localhost with the IP of the **Logstash** server (192.168.99.101)

Save and close the file

Test the configuration with

**sudo metricbeat test config**

To check what modules are available, you can either

**ls -al /etc/metricbeat/modules.d**

Or execute the following

**sudo metricbeat modules list**

*Now, if we want to enable a module, we can execute*

***sudo metricbeat modules enable system***

*This one may be enabled by default*

Install the beat’s template in **Elasticsearch** *(when we repeat the procedure for the second node, we should skip this)*

**sudo metricbeat setup --index-management -E output.logstash.enabled=false -E 'output.elasticsearch.hosts=["192.168.99.101:9200"]'**

Start and enable the **Metricbeat** service

**sudo systemctl daemon-reload**

**sudo systemctl enable metricbeat**

**sudo systemctl start metricbeat**

Go to **Kibana** UI, add the new data view (**metricbeat-\***), and explore the messages

If there aren’t any messages captured, go to the **node1** machine and check the logs (with **journalctl**)

**sudo journalctl -xe --unit metricbeat**

You will notice that a connection cannot be established with the **Logstash** machine (**server**)

To correct this, we must open port **5044/tcp** in the firewall on **server** machine

**sudo firewall-cmd --add-port 5044/tcp --permanent**

**sudo firewall-cmd --reload**

Return again in the **node1** and check the logs (with **journalctl**)

Now, go to the **Kibana** UI and check if the messages started to appear

**Repeat the same steps** (without the template installation) on node 2 (**node2**) and check the results in **Kibana** UI

Once, done with the above on node 2, let’s add additional service

*In our case it is Debian-based, so we must adjust the steps (at least the installation procedure)*

We can install **NGINX** for example on node 2 (**node2**), and enable the appropriate module

Go to **node2** and install **NGINX**

**sudo apt-get update**

**sudo apt-get install nginx**

Change the **Metricbeat** configuration

**sudo metricbeat modules enable nginx**

Restart the service

**sudo systemctl restart metricbeat**

Return to **Kibana** UI and explore the results

Filter the results for **node 2**

And then filter them by service type

The **KQL** should be **agent.name:node2 and service.type:"nginx"**

More details and samples can be found here:

**https://www.elastic.co/guide/en/kibana/current/kuery-query.html**

#### Auditbeat (on nodes)

Depending on the way we choose to install the software, we have two options for the beats as well – repository or individual packages

We will continue following the latter approach and download and install individual packages

Establish a session to node 1 (**node1**)machine

On a **CentOS** machine we must do:

**wget https://artifacts.elastic.co/downloads/beats/auditbeat/auditbeat-8.6.2-x86\_64.rpm**

**sudo rpm -Uvh auditbeat-8.6.2-x86\_64.rpm**

On a **Debian** / **Ubuntu** machine we can install it with:

**wget https://artifacts.elastic.co/downloads/beats/auditbeat/auditbeat-8.6.2-amd64.deb**

**sudo dpkg -i auditbeat-8.6.2-amd64.deb**

Now that we have it installed, we must configure it

Open the main configuration file

**sudo vi /etc/auditbeat/auditbeat.yml**

Change the following

Disable the **Elasticsearch** output (line #147) and enable the **Logstash** output (line #160)

Don’t forget to substitute the localhost with the IP of the **Logstash** server (192.168.99.101)

Save and close the file

Test the configuration with

**sudo auditbeat test config**

Install the beat’s template in **Elasticsearch**

**sudo auditbeat setup --index-management -E output.logstash.enabled=false -E 'output.elasticsearch.hosts=["192.168.99.101:9200"]'**

Start and enable the **Auditlbeat** service

**sudo systemctl daemon-reload**

**sudo systemctl enable auditbeat**

**sudo systemctl start auditbeat**

Go to **Kibana** UI, add the new data view (**auditbeat-\***), and explore the messages

**Repeat the same steps** (without the template installation) on node 2 and check the results in **Kibana** UI

#### Filebeat (on nodes)

Depending on the way we choose to install the software, we have two options for the beats as well – repository or individual packages

We will continue following the latter approach and download and install individual packages

Establish a session to node 1 (**node1**)machine

On a **CentOS** machine we must do

**wget https://artifacts.elastic.co/downloads/beats/filebeat/filebeat-8.6.2-x86\_64.rpm**

**sudo rpm -Uvh filebeat-8.6.2-x86\_64.rpm**

On a **Debian** / **Ubuntu** machine we can install it with

**wget https://artifacts.elastic.co/downloads/beats/filebeat/filebeat-8.6.2-amd64.deb**

**sudo dpkg -i filebeat-8.6.2-amd64.deb**

Now that we have it installed, we must configure it

Open the main configuration file

**sudo vi /etc/filebeat/filebeat.yml**

Change the following

Disable the **Elasticsearch** output (line #139) and enable the **Logstash** output (line #152)

Don’t forget to substitute the localhost with the IP of the **Logstash** server (192.168.99.101)

Save and close the file

Test the configuration with

**sudo filebeat test config**

To check what modules are available, you can either

**ls -al /etc/filebeat/modules.d**

Or execute the following

**sudo filebeat modules list**

*If no module is enabled, we can enable the* ***system*** *module with (****do not do it****)*

***sudo filebeat modules enable system***

*We will continue with the default module configured*

Reopen the main configuration file

**sudo vi /etc/filebeat/filebeat.yml**

and add an input (around line #56)

**- type: filestream**

**enabled: true**

**paths:**

**- /tmp/app.log**

Save and close the file

Test the configuration with

**sudo filebeat test config**

Execute the following to generate a few messages in the monitored log file

**for i in $(seq 1 10); do echo "$(date) -> test message #$i" >> /tmp/app.log; sleep 3; done**

Install the beat’s template in **Elasticsearch**

**sudo filebeat setup --index-management -E output.logstash.enabled=false -E 'output.elasticsearch.hosts=["192.168.99.101:9200"]'**

Start and enable the **Filebeat** service

**sudo systemctl daemon-reload**

**sudo systemctl enable filebeat**

**sudo systemctl start filebeat**

Go to **Kibana** UI, add the new data view (**filebeat-\***), and explore the messages

**Repeat the same steps** (without the template installation) on node 2 and check the results in **Kibana** UI

We can install **Apache** for example on node 1, and enable the appropriate module

Go to node 1

Install, enable, and start **Apache**

**sudo dnf install httpd**

**sudo systemctl enable httpd**

**sudo systemctl start httpd**

Create a simple **index.html** page

**echo 'Simple Page' | sudo tee /var/www/html/index.html**

Try a few **URLs** with

**curl http://localhost**

**curl http://localhost/err**

Change the **Filebeat** configuration

**sudo filebeat modules enable apache**

Open the module’s configuration file

**sudo vi /etc/filebeat/modules.d/apache.yml**

And change lines #7 and #15 to **true**

Save and close the file

Restart the service

**sudo systemctl restart filebeat**

Return to **Kibana** UI and explore the results

Check the fields **input.type**, **fileset.name**, **event.dataset**, **host.name**, and **log.file.path**

#### Create a Dashboard

Let’s create a dashboard

In order to do this, first we should create a few **Visualizations**

First, we will create a visualization of the average **RAM Usage**:

* In the menu, under the **Analytics** section, go to **Visualize Library**
* Click on **Create visualization**
* Select **Aggregation based**
* Choose the **Line** for visualization type
* Select the **Metricbeat Data** based on the **metricbeat-\*** index pattern
* For **Y-axis** set:
  + **Aggregation** to ***Average***
  + **Field** to ***system.memory.actual.used.pct***
  + **Custom label** to **RAM % Used**
* Click the **Add** button in the **Buckets** section and select **X-axis**
* For **X-axis** set:
  + **Aggregation** to ***Date Histogram***
  + **Field** to ***@timestamp***
* Click the **Add** button in the **Buckets** section and select **Split series**
* For **Split series**
  + Set the **Sub aggregation** to ***Terms***
  + Set the **Field** to **agent.name.keyword**
* Click the **Update** button to see the result
* Click on **Save** and save it under **RAM Usage**
* Change the **Add to dashboard** option to **None** and confirm

Create a second one for **CPU Usage**:

* Return to the **Analytics** section and go to **Visualize Library**
* Click on **Create visualization**
* Select **Aggregation based**
* Choose the **Area** chart type
* Select the **Metricbeat Data** based on the **metricbeat-\*** index pattern
* For **Y-axis** set:
  + **Aggregation** to ***Average***
  + **Field** to ***system.cpu.system.pct***
  + **Custom label** to ***CPU: System***
* Click the **Add** button in the **Metrics** section
* Add another **Y-axis** with:
  + **Aggregation** set to ***Average***
  + **Field** set to ***system.cpu.user.pct***
  + **Custom label** set to ***CPU: User***
* Click the **Add** button in the **Buckets** section and select **X-axis**
* For **X-axis** set:
  + **Aggregation** to ***Date Histogram***
  + **Field** to ***@timestamp***
* Click the **Add** button in the **Buckets** section and select **Split chart**
* In **Split chart** set:
  + **Sub aggregation** to ***Terms***
  + **Field** to ***agent.name.keyword***
* Click the **Update** button to see the result
* Click on **Save** and save it under **CPU Usage**
* Change the **Add to dashboard** option to **None** and confirm

Let’s create one more. This one will be showing the free disk space:

* Return to the **Analytics** section and go to **Visualize Library**
* Click on **Create visualization**
* Select **Aggregation based**
* Choose the **Vertical bar** chart type
* Select the **Metricbeat Data** based on the **metricbeat-\*** index pattern
* For **Y-axis** set:
  + **Aggregation** to ***Average***
  + **Field** to ***system.fsstat.total\_size.free***
  + **Custom label** to ***Disk Free***
* Click the **Add** button in the **Buckets** section and select **X-axis**
* For **X-axis** set:
  + **Aggregation** to ***Terms***
  + **Field** to ***agent.name.keyword***
  + **Custom label** to ***Host***
* Click the **Update** button to see the result
* Click on **Save** and save it under **Disk Free**
* Change the **Add to dashboard** option to **None** and confirm

We are ready to create our first dashboard:

* In the menu, click **Home**
* Then, click on **Dashboard** under **Analytics**
* Click on **Create dashboard**
* Click on **Add from library** to add an existing visualization
* Select, position, and resize the three visualizations we created earlier
* Click **Save** and for name enter **Resources**

## Part 3: Docker and Elastic Stack

Extend the environment by executing the following on the host

**vagrant up docker**

#### Docker Metrics and Logs

Let’s install the appropriate beats on the **Docker** host

On **Docker** node (**docker**), download and install the package:

**wget https://artifacts.elastic.co/downloads/beats/filebeat/filebeat-8.6.2-x86\_64.rpm**

**sudo rpm -Uvh filebeat-8.6.2-x86\_64.rpm**

Open the main configuration file

**sudo vi /etc/filebeat/filebeat.yml**

Change the following

Disable the **Elasticsearch** output (line #139) and enable the **Logstash** output (line #152)

Don’t forget to substitute the localhost with the IP of the **Logstash** server (192.168.99.101)

Add the following (in the **Filebeat inputs** section - around line #56)

**- type: container**

**paths:**

**- '/var/lib/docker/containers/\*/\*.log'**

Test the configuration with

**sudo filebeat test config**

Start and enable the **Filebeat** service

**sudo systemctl daemon-reload**

**sudo systemctl enable filebeat**

**sudo systemctl start filebeat**

Start a few containers. For example

**docker container run --name web1 -d -p 8081:80 nginx**

**docker container run --name web2 -d -p 8082:80 nginx**

**docker container run --name sleeper -d alpine sleep 1d**

Go to **Kibana** UI and switch to **Discovery** mode (**Analytics > Discover**)

Filter the **filebeat-\*** index pattern messages with **agent.name:docker**

Check the messages

Now let’s set to monitor the **Docker** metrics

On **Docker** node, download and install

**wget https://artifacts.elastic.co/downloads/beats/metricbeat/metricbeat-8.6.2-x86\_64.rpm**

**sudo rpm -Uvh metricbeat-8.6.2-x86\_64.rpm**

Open the main configuration file

**sudo vi /etc/metricbeat/metricbeat.yml**

Change the following

Disable the **Elasticsearch** output (line #92) and enable the **Logstash** output (line #105)

Don’t forget to substitute the localhost with the IP of the **Logstash** server (192.168.99.101)

Save and close the file

Test the configuration with

**sudo metricbeat test config**

Enable the **docker** module

**sudo metricbeat modules enable docker**

Open its file for editing

**sudo vi /etc/metricbeat/modules.d/docker.yml**

And make sure it looks like this

**- module: docker**

**metricsets:**

**- container**

**- cpu**

**- diskio**

**- event**

**- healthcheck**

**- info**

**- memory**

**- network**

**- network\_summary**

**period: 10s**

**hosts: ["unix:///var/run/docker.sock"]**

**enabled: true**

Save and close the file

Start and enable the **Metricbeat** service

**sudo systemctl daemon-reload**

**sudo systemctl enable metricbeat**

**sudo systemctl start metricbeat**

Go to **Kibana** UI, and navigate to the dashboard created earlier, we should see the third machine

Explore the **metricbeat-\*** index pattern in **Discover** mode (**Analytics > Discover**) and filter the messages with **agent.name:docker**

Run a few more containers, and check again the **Kibana** UI

#### Run Elastic Stack on Docker

The example is borrowed from: <https://github.com/aboullaite/docker-elk>

Either copy the **elk-on-docker** folder to the **Docker** host or execute it on your host machine (you should have **Docker** and **Docker Compose** installed)

Ensure that the host where **Docker** is running meets the requirements

Check the parameter for system limits on **mmap** counts

**sudo sysctl vm.max\_map\_count**

And set it to

**sudo sysctl -w vm.max\_map\_count=262144**

*More info here:* [*https://www.elastic.co/guide/en/elasticsearch/reference/current/vm-max-map-count.html*](https://www.elastic.co/guide/en/elasticsearch/reference/current/vm-max-map-count.html)

Create a folder for the stack’s data *(in the folder where the* ***docker-compose.yml*** *file is)*

**mkdir -p data/es**

Finally, start the stack with

**docker compose up -d**

*If you see a permission denied error, then correct the ownership with this command (should be executed in* ***/home/vagrant****):*

***sudo chown -R vagrant:vagrant .docker***

Please note, that it may take up to 10 minutes or even more for the stack to spin up

Meanwhile, we can monitor the progress with

**docker compose logs -f**

**docker container stats**

Once the process settles, we can try to connect

Open a browser tab and navigate to <http://192.168.99.104:5601> to explore the stack