ELK Stack Documentation

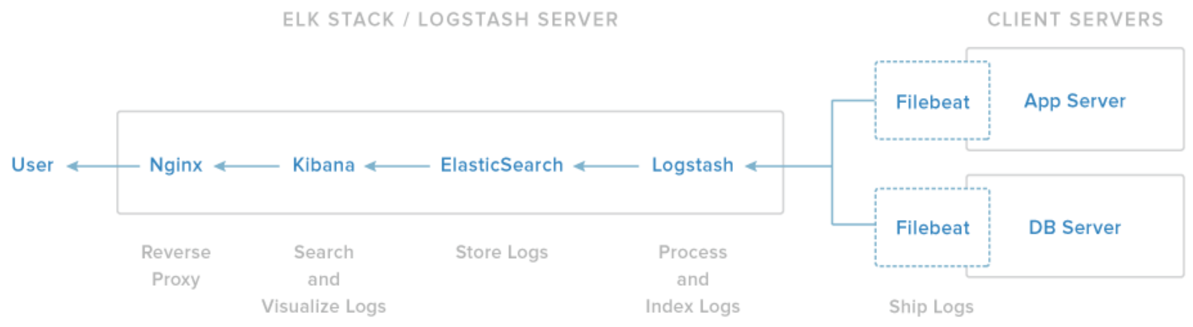
Abhishek Singh, Data Science Intern @ International Networks

Introduction

This is the setup of ELK stack on RedHat Linux OS version – Red Hat Enterprise Linux Server release 7.2 (Maipo).

In order to achieve full functionality of the stack, we need to install the following components –

- **Logstash**: The server component of Logstash that processes incoming logs
- **Elasticsearch**: Stores all of the logs
- **Kibana**: Web interface for searching and visualizing logs, which will be proxied through Nginx
- **Filebeat**: Installed on client servers that will send their logs to Logstash, Filebeat serves as a log shipping agent that utilizes the lumberjack networking protocol to communicate with Logstash



Setting up the Stack

1- Install Java

- Ensure you have root access on the system/server you are installing ELK stack on.
  - `sudo su`
- Navigate to the home directory
  - `cd ~`
- Download latest java sdk supported by ES (elasticsearch)

  -
    ```
    wget --no-cookies --no-check-certificate --header "Cookie: gpw_e24=http%3A%2F%2Fwww.oracle.com%2F; oraclelicense=accept-securebackup-cookie" http://download.oracle.com/otn-pub/java/jdk/8u91-b14/jdk-8u91-linux-x64.rpm
    ```
- Now Java should be installed at /usr/java/jdk1.8.0_91/jre/bin/java, and linked from /usr/bin/java.
- Remove the archive that we downloaded earlier –
  - `rm ~/jdk-8u91-linux-x64.rpm`

Now that Java 8 is installed, let's install ElasticSearch.

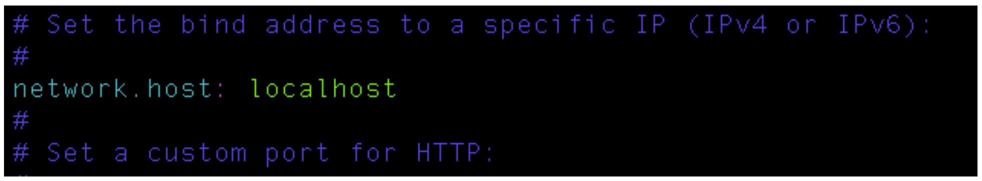
2- Install Elasticsearch

- Elasticsearch can be installed with a package manager by adding Elastic's package repository.
- Run the following command to import the Elasticsearch public GPG key into rpm:
  - `sudo rpm --import http://packages.elastic.co/GPG-KEY-elasticsearch`
- Create a new yum repository file for Elasticsearch. Note that this is a single command:

  -
    ```
    echo '[elasticsearch-2.x] name=Elasticsearch repository for 2.x packages
    baseurl=http://packages.elastic.co/elasticsearch/2.x/centos gpgcheck=1
    gpgkey=http://packages.elastic.co/GPG-KEY-elasticsearch enabled=1 ' |
    sudo tee /etc/yum.repos.d/elasticsearch.repo
    ```
- Install Elasticsearch with this command:
  - `sudo yum -y install elasticsearch`
- Elasticsearch is now installed. Let's edit the configuration:
  - `sudo vim /etc/elasticsearch/elasticsearch.yml`
- Set permission
  - `chown -R elasticsearch:elasticsearch /mnt/ssd/elkdata/`

3- Protect elasticsearch from being accessed via HTTP API

- We will want to restrict outside access to your Elasticsearch instance (port 9200), so outsiders can't read your data or shutdown your Elasticsearch cluster through the HTTP API. Find the line that specifies network.host, uncomment it, and replace its value with "localhost" so it looks like this:

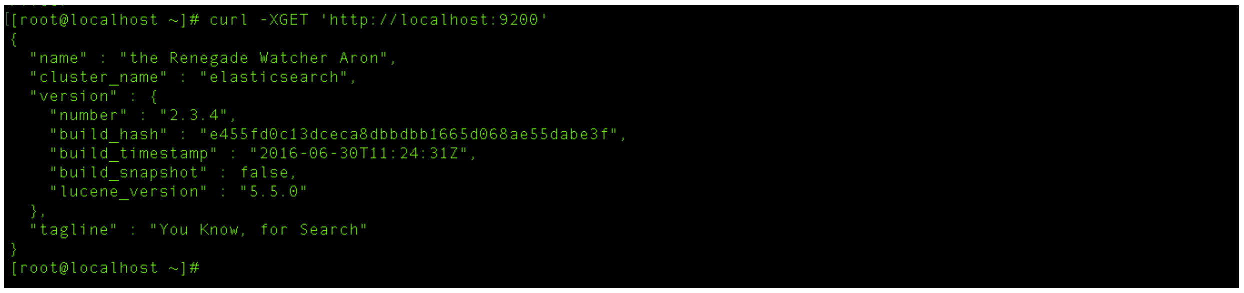


```
# Set the bind address to a specific IP (IPv4 or IPv6):
#
network.host: localhost
#
# Set a custom port for HTTP:
```

4- Run Elasticsearch:

- `sudo systemctl start elasticsearch`

- Then run the following command to start Elasticsearch automatically on boot up:
    - `sudo systemctl enable elasticsearch`
- In order to make sure if elasticsearch started properly, we can try this command:
    - `curl -XGET “http://localhost:9200”`

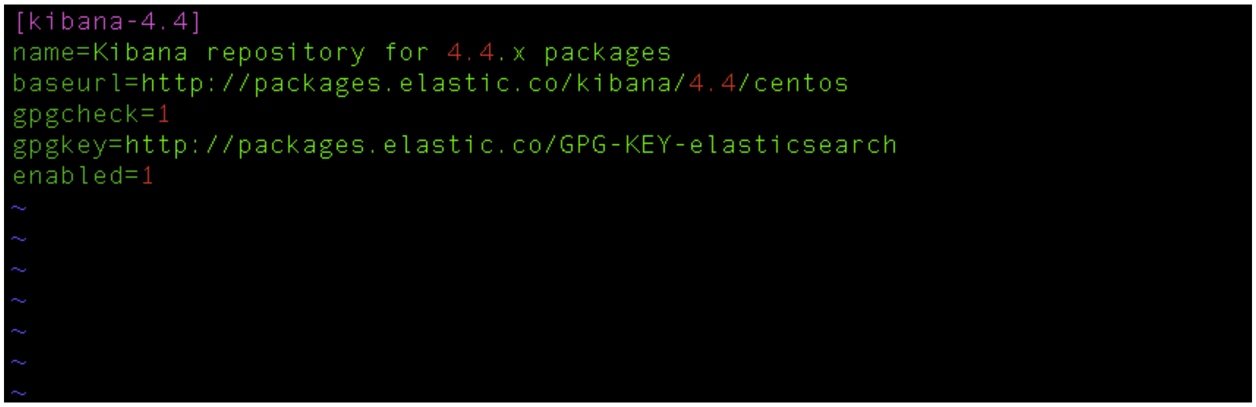


```
[root@localhost ~]# curl -XGET 'http://localhost:9200'
{
  "name" : "the Renegade Watcher Aron",
  "cluster_name" : "elasticsearch",
  "version" : {
    "number" : "2.3.4",
    "build_hash" : "e455fd0c13dceca8dbbdbb1665d068ae55dabe3f",
    "build_timestamp" : "2016-06-30T11:24:31Z",
    "build_snapshot" : false,
    "lucene_version" : "5.5.0"
  },
  "tagline" : "You Know, for Search"
}
[root@localhost ~]#
```

Now that Elasticsearch is up and running, let's install Kibana.

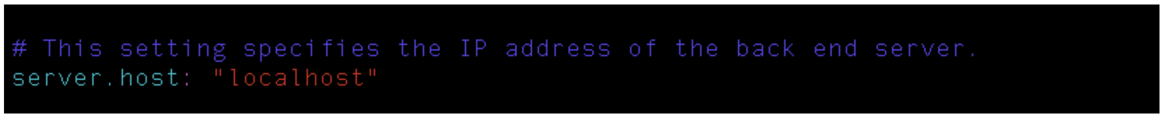
5- Install Kibana

- The Kibana package shares the same GPG Key as Elasticsearch, and we already installed that public key.
- Create and edit a new yum repository file for Kibana:
  - `sudo vim /etc/yum.repos.d/kibana.repo`
- Save and exit.



```
[kibana-4.4]
name=Kibana repository for 4.4.x packages
baseurl=http://packages.elastic.co/kibana/4.4/centos
gpgcheck=1
gpgkey=http://packages.elastic.co/GPG-KEY-elasticsearch
enabled=1
~
~
~
~
~
~
~
```

- Install Kibana with this command:
  - `sudo yum -y install kibana`
- Open the Kibana configuration file for editing:
  - In the Kibana configuration file, find the line that specifies server.host, and replace the IP address ("0.0.0.0" by default) with "localhost":

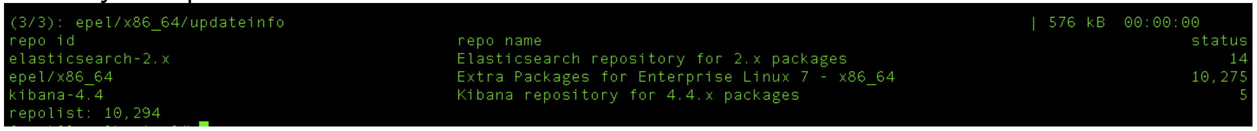


```
# This setting specifies the IP address of the back end server.
server.host: "localhost"
```

- Save and exit. This setting makes it so Kibana will only be accessible to the localhost. This is fine because we will install an Nginx reverse proxy, on the same server, to allow external access.
- Now start the Kibana service, and enable it:
  - `sudo systemctl start kibana`
  - `sudo chkconfig kibana on`

6- Install NGINX

- Because we configured Kibana to listen on localhost, we must set up a reverse proxy to allow external access to it. We will use Nginx for this purpose.
- Add the EPEL repository to yum.
- But before doing that, we need to enable EPEL for Red Hat. First, we need to download the file using Wget and then install it using RPM on your system to enable the EPEL repository.
    - `wget http://dl.fedoraproject.org/pub/epel/7/x86_64/e/epel-release-7-7.noarch.rpm`
    - `rpm -ivh epel-release-7-7.noarch.rpm`
- To verify -
    - `yum repolist`



```
(3/3): epel/x86_64/updateinfo                                                                  | 576 kB  00:00:00
repo id                                    repo name                                                        status
elasticsearch-2.x                          Elasticsearch repository for 2.x packages                            14
epel/x86_64                                Extra Packages for Enterprise Linux 7 - x86_64                   10,275
kibana-4.4                                 Kibana repository for 4.4.x packages                                  5
repolist: 10,294
```

- Now use yum to install Nginx and httpd-tools:
    - `sudo yum -y install nginx httpd-tools`
- Sometimes when epel release doesn’t work, we use other ways-
    - `sudo rpm -Uvh http://nginx.org/packages/centos/7/noarch/RPMS/nginx-release-centos-7-0.el7.ngx.noarch.rpm`
- Use htpasswd to create an admin user, called "kibanaadmin" (you should use another name), that can access the Kibana web interface:
    - `sudo htpasswd -c /etc/nginx/htpasswd.users kibanaroot`
- Enter a password at the prompt. Remember this login, as you will need it to access the Kibana web interface.
- Now open the Nginx configuration file in your favorite editor.
    - `sudo vim /etc/nginx/nginx.conf`

- Copy the nginx.conf file from our dev server under the location - /etc/nginx/nginx.conf
- Copy the kibana.conf file from dev too under location-
    - `cp kibana.conf /etc/nginx/conf.d/kibana.conf`

Now when I try to run nginx with this command –

- `sudo systemctl start nginx`

It gives me the following error –

```
Job for nginx.service failed because the control process exited with
error code. See "systemctl status nginx.service" and "journalctl -xe"
for details.
```

So I do -

```
systemctl status nginx.service
```

which gives me -

```
● nginx.service - nginx - high performance web server

   Loaded: loaded (/usr/lib/systemd/system/nginx.service; disabled;
vendor preset: disabled)

   Active: failed (Result: exit-code) since Fri 2016-07-08 13:24:33
EDT; 24s ago

     Docs: http://nginx.org/en/docs/

  Process: 48080 ExecStartPre=/usr/sbin/nginx -t -c
/etc/nginx/nginx.conf (code=exited, status=1/FAILURE)
```

This seems like a permissions issue, thus I need to give permission to nginx on the folder /etc/nginx

Note: This tutorial assumes that SELinux is disabled. If this is not the case, you may need to run the following command for Kibana to work properly: `sudo setsebool -P httpd_can_network_connect 1`

Note: The above issue was resolved by removing nginx fully and doing a fresh installation by following process.

- Now open the Nginx configuration file in your favorite editor. We will use vim:
    - `sudo vi /etc/nginx/nginx.conf`
- Find the default server block (starts with server {), the last configuration block in the file, and delete it. When you are done, the last two lines in the file should look like this:

nginx.conf excerpt

```
    include /etc/nginx/conf.d/*.conf;

}
```

- Save and exit.
- Now we will create an Nginx server block in a new file:
    - `sudo vi /etc/nginx/conf.d/kibana.conf`
- Paste the following code block into the file. Be sure to update the server_name to match your server's name:
    - `/etc/nginx/conf.d/kibana.conf`

```
server {

    listen 80;

    server_name 156.56.6.99;

    auth_basic "Restricted Access";

    auth_basic_user_file /etc/nginx/htpasswd.users;

#    allow all;

    location / {

        proxy_pass http://localhost:5601;

        proxy_http_version 1.1;

        proxy_set_header Upgrade $http_upgrade;

        proxy_set_header Connection 'upgrade';

        proxy_set_header Host $host;

        proxy_cache_bypass $http_upgrade;

   }

}
```

- Save and exit. This configures Nginx to direct your server's HTTP traffic to the Kibana application, which is listening on localhost:5601. Also, Nginx will use the htpasswd.users file, that we created earlier, and require basic authentication.
- Now start and enable Nginx to put our changes into effect:
  - `sudo systemctl start nginx`
  - `sudo systemctl enable nginx`

Kibana is now accessible via your FQDN or the public IP address of your ELK Server i.e. http://elk_server_public_ip/. If you go there in a web browser, after entering the "kibanaadmin" credentials, you should see a Kibana welcome page which will ask you to configure an index pattern. Let's get back to that later, after we install all of the other components.

7- Install Logstash

- Now install logstash on our server.
  - `sudo vim /etc/yum.repos.d/logstash.repo`

  paste this to the file

```
[logstash-2.3]

name=Logstash repository for 2.3.x packages

baseurl=https://packages.elastic.co/logstash/2.3/centos

gpgcheck=1

gpgkey=https://packages.elastic.co/GPG-KEY-elasticsearch

enabled=1
```

- Save and exit
- Install logstash with this command.
  - `sudo yum -y install logstash`

Logstash is now installed but not yet configured.

8- Generate SSL Certificate

- Since we are going to use Filebeat to ship logs from our Client Servers to our ELK Server, we need to create an SSL certificate and key pair. The certificate is used by Filebeat to verify the identity of ELK Server. Create the directories that will store the certificate and private key with the following commands:
    - `sudo vim /etc/pki/tls/openssl.cnf`
- Find the [v3_ca] section in the file, and add this line under it (substituting in the ELK Server's private IP address):

```
subjectAltName = IP: 127.0.0.1
```

- Save and exit.
- Now generate the SSL certificate and private key in the appropriate locations (/etc/pki/tls/), with the following commands:
    - `cd /etc/pki/tls`

    -
      ```
      sudo openssl req -config /etc/pki/tls/openssl.cnf -x509 -days 3650 -batch
        -nodes -newkey rsa:2048 -keyout private/logstash-forwarder.key -out
        certs/logstash-forwarder.crt
      ```
- The logstash-forwarder.crt file will be copied to all of the servers that will send logs to Logstash but we will do that a little later. Let's complete our Logstash configuration.

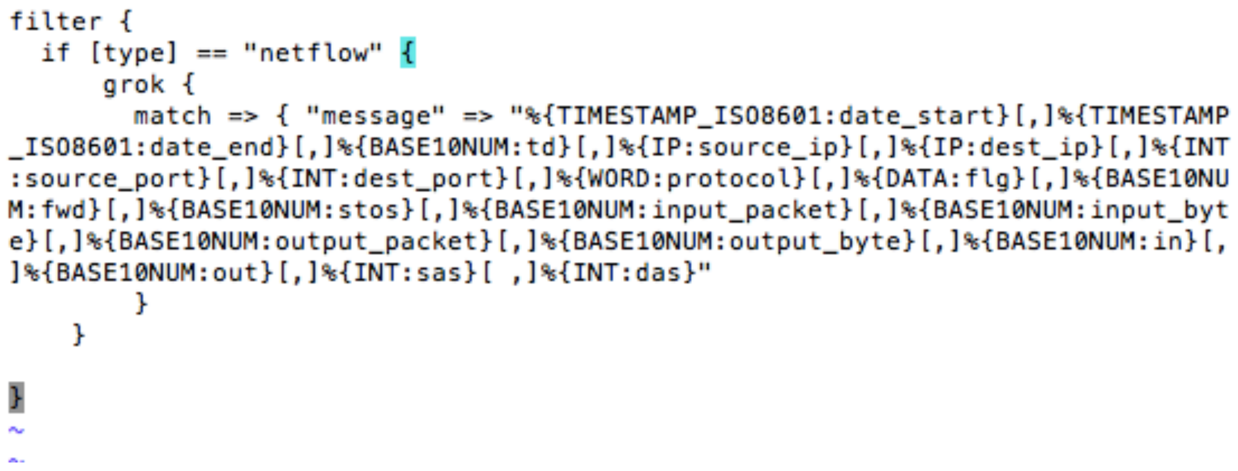
9- Generate SSL Certificate

- Logstash configuration files are in the JSON-format, and reside in /etc/logstash/conf.d. The configuration consists of three sections: inputs, filters, and outputs.
- Let's create a configuration file called 02-beats-input.conf (the 02 in the filename matters here, since logstash looks at numeric in file name to decide which one to implement first) and set up our "filebeat" input:
- Create a new configuration file 15-netlog-filter.conf with the following command-
    - `sudo vim /etc/logstash/conf.d/15-netlog-filter.conf`

- Paste the following code on the new file –

```
filter {

  if [type] == "netflow" {

      grok {

        match => { "message" =>
"%{TIMESTAMP_ISO8601:date_start}[,]%{TIMESTAMP_ISO8601:date_end}[,]%{BASE10NUM:td}[,]%{IP:source_ip}[,]%{IP:dest_ip}[,]%{INT:source_port}[,]%{INT:dest_port}[,]%{WORD:protocol}[,]%{DATA:flg}[,]%{BASE10NUM:fwd}[,]%{BASE10NUM:stos}[,]%{BASE10NUM:input_packet}[,]%{BASE10NUM:input_byte}[,]%{BASE10NUM:output_packet}[,]%{BASE10NUM:output_byte}[,]%{BASE10NUM:in}[,]%{BASE10NUM:out}[,]%{INT:sas}[ ,]%{INT:das}"

        }

    }

}
```

- The file will look as follows –



```
filter {
  if [type] == "netflow" {
      grok {
        match => { "message" => "%{TIMESTAMP_ISO8601:date_start}[,]%{TIMESTAMP_ISO8601:date_end}[,]%{BASE10NUM:td}[,]%{IP:source_ip}[,]%{IP:dest_ip}[,]%{INT:source_port}[,]%{INT:dest_port}[,]%{WORD:protocol}[,]%{DATA:flg}[,]%{BASE10NUM:fwd}[,]%{BASE10NUM:stos}[,]%{BASE10NUM:input_packet}[,]%{BASE10NUM:input_byte}[,]%{BASE10NUM:output_packet}[,]%{BASE10NUM:output_byte}[,]%{BASE10NUM:in}[,]%{BASE10NUM:out}[,]%{INT:sas}[ ,]%{INT:das}"
        }
    }

}
~
~
```

- Lastly, we will create a configuration file called 30-elasticsearch-output.conf:
  - `sudo vim /etc/logstash/conf.d/30-elasticsearch-output.conf`
- Edit the file so it looks as follows:

```
output {

 elasticsearch {
    hosts => ["localhost:9200"]

    sniffing => true

    manage_template => false

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

    document_type => "%{[@metadata][type]}"

  }

}
```

- Save and exit. This output basically configures Logstash to store the beats data in Elasticsearch which is running at localhost:9200, in an index named after the beat used (filebeat, in our case).
- If you want to add filters for other applications that use the Filebeat input, be sure to name the files so they sort between the input and the output configuration (i.e. between 02- and 30-).
- Test your Logstash configuration with this command:
  - `sudo service logstash configtest`
- It should display Configuration OK if there are no syntax errors. Otherwise, try and read the error output to see what's wrong with your Logstash configuration.
- Restart and enable Logstash to put our configuration changes into effect:
  - `sudo systemctl restart logstash`
  - `sudo chkconfig logstash on`

Next, we'll load the sample Kibana dashboards.

10- Load Kibana Dashboards

Elastic provides several sample Kibana dashboards and Beats index patterns that can help you get started with Kibana. Although we won't use the dashboards in this tutorial, we'll load them anyway so we can use the Filebeat index pattern that it includes.

- First, download the sample dashboards archive to your home directory:
  - `cd ~`
  - `curl -L -O https://download.elastic.co/beats/dashboards/beats-dashboards-1.1.0.zip`
- Install the unzip package with this command:
  - `sudo yum -y install unzip`
- Next, extract the contents of the archive:
  - `unzip beats-dashboards-*.zip`
- These are the index patterns that we just loaded:
  - [packetbeat-]YYYY.MM.DD
  - [topbeat-]YYYY.MM.DD
  - [filebeat-]YYYY.MM.DD
  - [winlogbeat-]YYYY.MM.DD
- When we start using Kibana, we will select the Filebeat index pattern as our default.

11- Load Filebeat Index Template in Elasticsearch

- Because we are planning on using Filebeat to ship logs to Elasticsearch, we should load a Filebeat index template. The index template will configure Elasticsearch to analyze incoming Filebeat fields in an intelligent way.
- First, make new Filebeat index template in your home directory:
  - `cd ~`
  - `curl ~`
  - `sudo vim filebeat.index.template.json`

- Paste this to the file

```
{
  "mappings": {
    "_default_": {
      "_all": {
        "enabled": true,
        "norms": {
          "enabled": false
        }
      },
      "dynamic_templates": [
        {
          "template1": {
            "mapping": {
              "doc_values": true,
              "ignore_above": 1024,
              "index": "not_analyzed",
              "type": "{dynamic_type}"
            },
            "match": "*"
          }
        }
      ],
      "properties": {
        "@timestamp": {
          "type": "date"
        },
        "message": {
          "type": "string",
          "index": "analyzed"
        },
      "offset": {
        "type": "long",
        "doc_values": "true"
      },
   "input_byte" : {
        "type" : "long",
        "doc_values": true,
        "index": "not_analyzed"
      },
      "output_byte" : {
        "type" : "long",
        "doc_values": true,
        "index": "not_analyzed"
      },
      "input_packet": {
        "type" : "long",
        "doc_values": true,
        "index": "not_analyzed"
      },

      "output_packet": {
        "type" : "long",
        "doc_values": true,
        "index": "not_analyzed"
      },
      "sas": {
        "type" : "integer",
        "doc_values": true,
        "index": "not_analyzed"
      },
      "das": {
        "type" : "integer",
      "doc_values": true,
      "index": "not_analyzed"
    },
    "in": {
      "type" : "integer",
      "doc_values": true,
      "index": "not_analyzed"
 },
    "out": {
      "type" : "integer",
      "doc_values": true,
      "index": "not_analyzed"
    },
    "date_start": {
      "type": "date",
      "format": "yyyy-MM-dd HH:mm:ss",
      "doc_values": true,
      "index": "not_analyzed"
    },
    "date_end": {
      "type": "date",
      "format": "yyyy-MM-dd HH:mm:ss",
      "doc_values": true,
      "index": "not_analyzed"
    },
    "source_ip": {
      "type" : "ip",
      "doc_values": true
    },
    "source_port": {
      "type" : "integer",
      "doc_values": true,
    "index": "not_analyzed"
  },
  "dest_ip": {
    "type" : "ip",
    "doc_values": true
  },
  "dest_port": {
    "type" : "integer",
    "doc_values": true,
    "index": "not_analyzed"
},
    "source_port": {
    "type" : "integer",
    "doc_values": true,
    "index": "not_analyzed"
  },
  "dest_ip": {
    "type" : "ip",
    "doc_values": true
  },
  "dest_port": {
    "type" : "integer",
    "doc_values": true,
    "index": "not_analyzed"
  },
  "stos": {
    "type" : "integer",
    "doc_values": true,
    "index": "not_analyzed"
  },
  "td": {
    "type" : "double",
          "doc_values": true,
          "index": "not_analyzed"
        },
        "protocol": {
          "type" : "string"
        }
      }
    }
  },
  "settings": {
    "index.refresh_interval": "5s"
  },
  "template": "filebeat-*"
}
```

- Then load the template with this command:
  - `curl -XPUT 'http://localhost:9200/_template/filebeat?pretty' -d@filebeat.template.json`
- If the template loaded properly, you should see a message like this:

```
Output:
{
  "acknowledged" : true
}
```

Now that our ELK Server is ready to receive Filebeat data, let's move onto setting up Filebeat on each client server.

12- SSL Certificate

- Make sure you have the logstash-forwarder.crt file in the correct location with this command –
  - `ls /etc/pki/tls/certs/`

13- Install Filebeat Package

- Create and edit a new yum repository file for Filebeat:
  - `sudo vi /etc/yum.repos.d/elastic-beats.repo`
- Add the following repository configuration:
  - `/etc/yum.repos.d/elastic-beats.repo`

```
•    [beats]
•    name=Elastic Beats Repository
•    baseurl=https://packages.elastic.co/beats/yum/el/$basearch
•    enabled=1
•    gpgkey=https://packages.elastic.co/GPG-KEY-elasticsearch
•    gpgcheck=1
```

- Save and exit.
- Install Filebeat with this command:
  - `sudo yum -y install filebeat`

Filebeat is installed but it is not configured yet.

14- Configure Filebeat

- rm the existing /etc/filebeat/filebeat.yml file and create a new file with the same name –
  - `sudo vim /etc/filebeat/filebeat.yml`
- Paste the following code in the new file.

```
filebeat:

  prospectors:

    -

      paths:

        - /home/elkdata/files/*.csv

      document_type: netflowOneDay



  registry_file: /var/lib/filebeat/registry



output:

  logstash:

    hosts: ["127.0.0.1:5044"]

    bulk_max_size: 5000
    tls:
      certificate_authorities: ["/etc/pki/tls/certs/logstash-forwarder.crt"]

shipper:

logging:
    to_files: true
    files:
      path: /var/log/filebeat/
      name: filebeat_eslogs.log
      rotateeverybytes: 10485760 # = 10MB
    level: info
```

- Save and quit.
- Now start and enable Filebeat to put our changes into place:
    - `sudo systemctl start filebeat`
    - `sudo systemctl enable filebeat`
- To ensure, your filebeat is working fine, check the logs –
    - `more /var/log/filebeat/filebeat_eslogs.log`
- They should look like this-

```
2016-07-12T12:31:10-04:00 ERR SSL client failed to connect with: dial tcp
127.0.0.1:5044

: getsockopt: connection refused

2016-07-12T12:31:10-04:00 INFO Activated logstash as output plugin.

2016-07-12T12:31:10-04:00 INFO Publisher name: localhost.localdomain

2016-07-12T12:31:10-04:00 INFO Flush Interval set to: 1s

2016-07-12T12:31:10-04:00 INFO Max Bulk Size set to: 5000

2016-07-12T12:31:10-04:00 INFO Init Beat: filebeat; Version: 1.2.3

2016-07-12T12:31:10-04:00 INFO filebeat sucessfully setup. Start running.

2016-07-12T12:31:10-04:00 INFO Registry file set to:
/var/lib/filebeat/registry
```

```
2016-07-12T12:31:10-04:00 INFO Set ignore_older duration to 0

2016-07-12T12:31:10-04:00 INFO Set close_older duration to 1h0m0s

2016-07-12T12:31:10-04:00 INFO Set scan_frequency duration to 10s

2016-07-12T12:31:10-04:00 INFO Invalid input type set:

2016-07-12T12:31:10-04:00 INFO Input type set to: log

2016-07-12T12:31:10-04:00 INFO Set backoff duration to 1s

2016-07-12T12:31:10-04:00 INFO Set max_backoff duration to 10s

2016-07-12T12:31:10-04:00 INFO force_close_file is disabled

2016-07-12T12:31:10-04:00 INFO Starting prospector of type: log

2016-07-12T12:31:10-04:00 INFO Starting spooler: spool_size: 2048;
idle_timeout: 5s

2016-07-12T12:31:10-04:00 INFO All prospectors initialised with 0 states to
persist

2016-07-12T12:31:10-04:00 INFO Starting Registrar

2016-07-12T12:31:10-04:00 INFO Start sending events to output
```

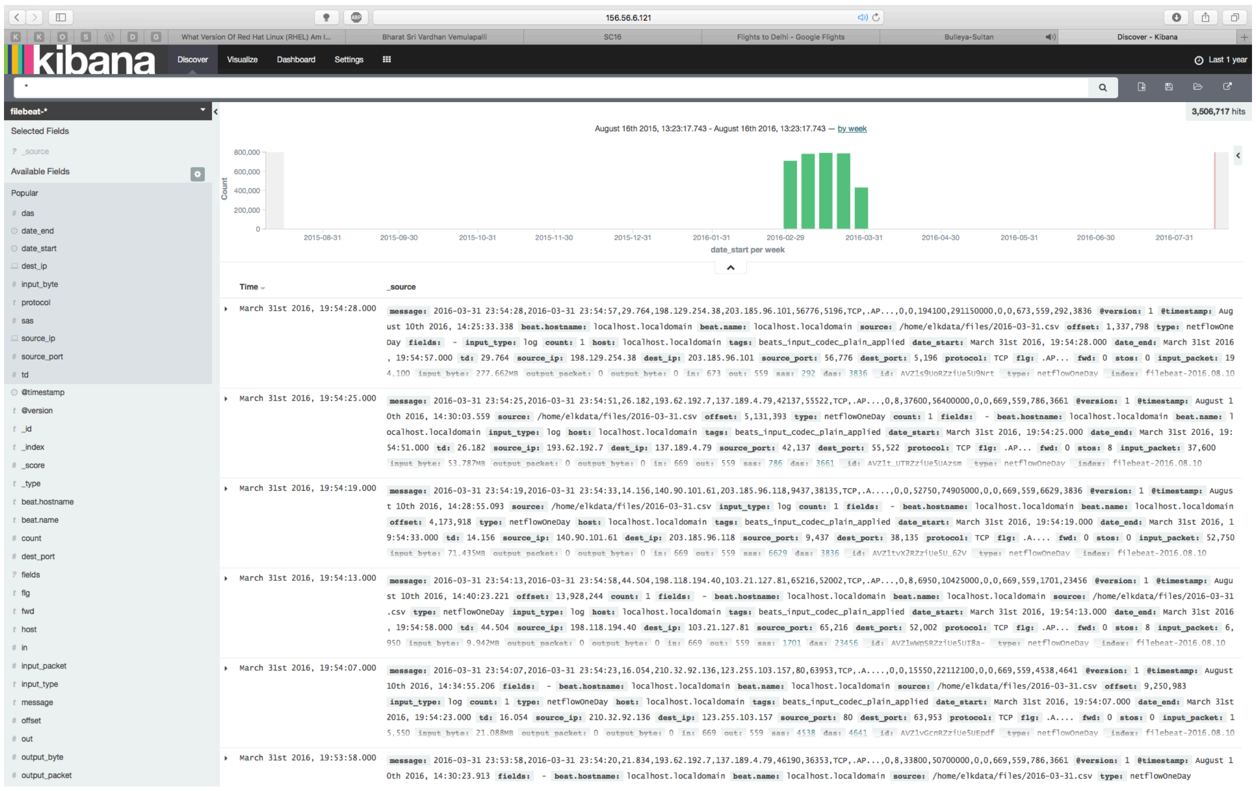
This means, filebeat is working fine and we are good to input data.

15- Install Filebeat Package

- It's about time now that we get data getting in ES through filebeat.
- Create a new directory in root home directory
  - `cd ~`
  - `mkdir /home/elkdata`
  - `mkdir /home/elkdata/files`
- Copy the csv file to import -
  - `cp /home/aasingh/01.csv /home/elkdata/files/01.csv`

16- Install Sense

- Go to the directory –
  - `cd /opt/kibana`
  - `./bin/kibana plugin --install elastic/sense`
  - `systemctl restart kibana`

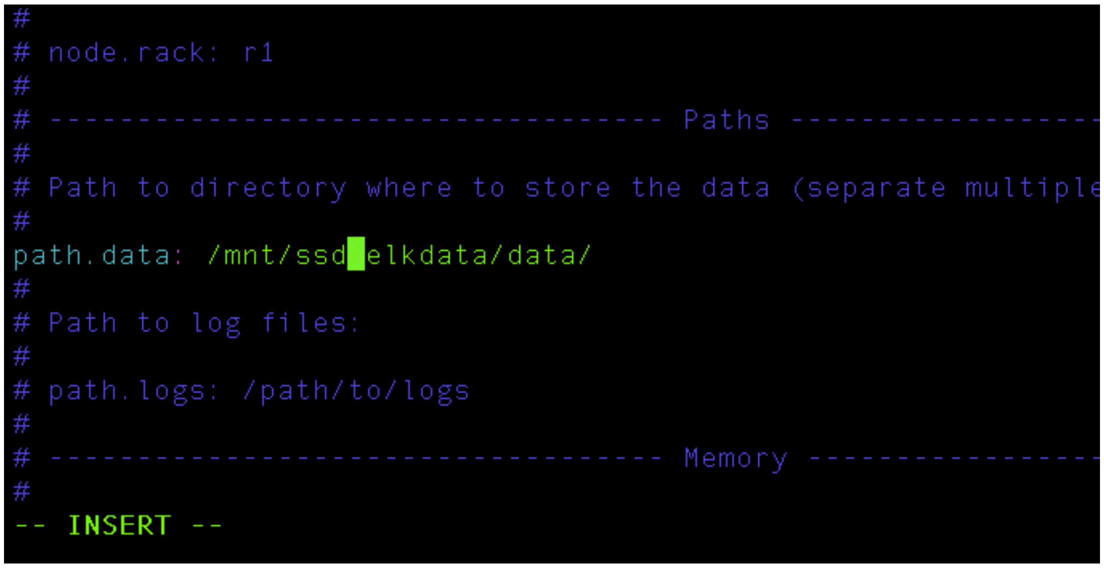


```
March 31st 2016, 19:54:28.000  message: 2016-03-31 23:54:28,2016-03-31 23:54:57,29.764,198.129.254.38,203.185.96.101,56776,5196,TCP,.AP...,0,0,194100,291150000,0,0,673,559,292,3836 @version: 1 @timestamp: August 10th 2016, 14:25:33.338 beat.hostname: localhost.localdomain beat.name: localhost.localdomain source: /home/elkdata/files/2016-03-31.csv offset: 1,337,798 type: netflowOneDay fields: - input_type: log count: 1 host: localhost.localdomain tags: beats_input_codec_plain_applied date_start: March 31st 2016, 19:54:28.000 date_end: March 31st 2016, 19:54:57.000 td: 29.764 source_ip: 198.129.254.38 dest_ip: 203.185.96.101 source_port: 56,776 dest_port: 5,196 protocol: TCP flg: .AP... fwd: 0 stos: 0 input_packet: 194,100 input_byte: 277.662MB output_packet: 0 output_byte: 0 in: 673 out: 559 sas: 292 das: 3836 _id: AVZ1s9UoRZziUe5U9Nrt _type: netflowOneDay _index: filebeat-2016.08.10

March 31st 2016, 19:54:25.000  message: 2016-03-31 23:54:25,2016-03-31 23:54:51,26.182,193.62.192.7,137.189.4.79,42137,55522,TCP,.AP...,0,8,37600,56400000,0,0,669,559,786,3661 @version: 1 @timestamp: August 10th 2016, 14:30:03.559 source: /home/elkdata/files/2016-03-31.csv offset: 5,131,393 type: netflowOneDay count: 1 fields: - beat.hostname: localhost.localdomain beat.name: localhost.localdomain input_type: log host: localhost.localdomain tags: beats_input_codec_plain_applied date_start: March 31st 2016, 19:54:25.000 date_end: March 31st 2016, 19:54:51.000 td: 26.182 source_ip: 193.62.192.7 dest_ip: 137.189.4.79 source_port: 42,137 dest_port: 55,522 protocol: TCP flg: .AP... fwd: 0 stos: 8 input_packet: 37,600 input_byte: 53.787MB output_packet: 0 output_byte: 0 in: 669 out: 559 sas: 786 das: 3661 _id: AVZ1t_UTRZziUe5UAzsm _type: netflowOneDay _index: filebeat-2016.08.10

March 31st 2016, 19:54:19.000  message: 2016-03-31 23:54:19,2016-03-31 23:54:33,14.156,140.90.101.61,203.185.96.118,9437,38135,TCP,.A....,0,0,52750,74905000,0,0,669,559,6629,3836 @version: 1 @timestamp: August 10th 2016, 14:28:55.093 source: /home/elkdata/files/2016-03-31.csv input_type: log count: 1 fields: - beat.hostname: localhost.localdomain beat.name: localhost.localdomain offset: 4,173,918 type: netflowOneDay host: localhost.localdomain tags: beats_input_codec_plain_applied date_start: March 31st 2016, 19:54:19.000 date_end: March 31st 2016, 19:54:33.000 td: 14.156 source_ip: 140.90.101.61 dest_ip: 203.185.96.118 source_port: 9,437 dest_port: 38,135 protocol: TCP flg: .A.... fwd: 0 stos: 0 input_packet: 52,750 input_byte: 71.435MB output_packet: 0 output_byte: 0 in: 669 out: 559 sas: 6629 das: 3836 _id: AVZ1tvX2RZziUe5U_62V _type: netflowOneDay _index: filebeat-2016.08.10

March 31st 2016, 19:54:13.000  message: 2016-03-31 23:54:13,2016-03-31 23:54:58,44.504,198.118.194.40,103.21.127.81,65216,52002,TCP,.AP...,0,8,6950,10425000,0,0,669,559,1701,23456 @version: 1 @timestamp: August 10th 2016, 14:40:23.221 offset: 13,928,244 count: 1 fields: - beat.hostname: localhost.localdomain beat.name: localhost.localdomain source: /home/elkdata/files/2016-03-31.csv type: netflowOneDay input_type: log host: localhost.localdomain tags: beats_input_codec_plain_applied date_start: March 31st 2016, 19:54:13.000 date_end: March 31st 2016, 19:54:58.000 td: 44.504 source_ip: 198.118.194.40 dest_ip: 103.21.127.81 source_port: 65,216 dest_port: 52,002 protocol: TCP flg: .AP... fwd: 0 stos: 8 input_packet: 6,950 input_byte: 9.942MB output_packet: 0 output_byte: 0 in: 669 out: 559 sas: 1701 das: 23456 _id: AVZ1wWpSRZziUe5UI8a- _type: netflowOneDay _index: filebeat-2016.08.10

March 31st 2016, 19:54:07.000  message: 2016-03-31 23:54:07,2016-03-31 23:54:23,16.054,210.32.92.136,123.255.103.157,80,63953,TCP,.A....,0,0,15550,22112100,0,0,669,559,4538,4641 @version: 1 @timestamp: August 10th 2016, 14:34:55.206 fields: - beat.hostname: localhost.localdomain beat.name: localhost.localdomain source: /home/elkdata/files/2016-03-31.csv offset: 9,250,983 input_type: log count: 1 type: netflowOneDay host: localhost.localdomain tags: beats_input_codec_plain_applied date_start: March 31st 2016, 19:54:07.000 date_end: March 31st 2016, 19:54:23.000 td: 16.054 source_ip: 210.32.92.136 dest_ip: 123.255.103.157 source_port: 80 dest_port: 63,953 protocol: TCP flg: .A.... fwd: 0 stos: 0 input_packet: 15,550 input_byte: 21.088MB output_packet: 0 output_byte: 0 in: 669 out: 559 sas: 4538 das: 4641 _id: AVZ1vGcnRZziUe5UEpdf _type: netflowOneDay _index: filebeat-2016.08.10

March 31st 2016, 19:53:58.000  message: 2016-03-31 23:53:58,2016-03-31 23:54:20,21.834,193.62.192.7,137.189.4.79,46190,36353,TCP,.AP...,0,8,33800,50700000,0,0,669,559,786,3661 @version: 1 @timestamp: August 10th 2016, 14:30:23.913 fields: - beat.hostname: localhost.localdomain beat.name: localhost.localdomain source: /home/elkdata/files/2016-03-31.csv type: netflowOneDay
```

Troubleshooting Guide

1- Change the path of the elk data

- Stop elasticsearch with this command:
  - `sudo systemctl stop elasticsearch`
- Go to the elasticsearch configuration file –
  - `sudo vim /etc/elasticsearch/elasticsearch.yml`
- Change the path.location to the required location as follows:



```
#
# node.rack: r1
#
# ---------------------------------- Paths -----------------
#
# Path to directory where to store the data (separate multiple
#
path.data: /mnt/ssd/elkdata/data/
#
# Path to log files:
#
# path.logs: /path/to/logs
#
# ---------------------------------- Memory ----------------
#
-- INSERT --
```

- Assign proper permission to elasticsearch user
  - `sudo chown -R elasticsearch: /mnt/ssd/elkdata/`
  - `sudo chmod -R ug+rw /mnt/ssd/elkdata/`
  - `vim /usr/lib/systemd/system/elasticsearch.service`

  LimitMEMLOCK=infinity

  - `systemctl daemon-reload`

- Restart elasticsearch
  - `sudo systemctl start elasticsearch`

- Useful link for elasticsearch.yml/mlockall not reflecting.
- https://github.com/elastic/elasticsearch/issues/9357

2- **Filebeat logs say this**

```
2016-07-12T12:48:14-04:00 INFO Harvester started for
file: /home/elkdata/files/01.csv

2016-07-12T12:48:14-04:00 ERR SSL client failed to
connect with: dial tcp 127.0.0.1:5044

: getsockopt: connection refused

2016-07-12T12:48:14-04:00 INFO Connecting error
publishing events (retrying): dial tcp 1

27.0.0.1:5044: getsockopt: connection refused

2016-07-12T12:48:14-04:00 INFO send fail

2016-07-12T12:48:14-04:00 INFO backoff retry: 1s

2016-07-
```

Although the problem seems from filebeat, its actually due to logstash filter not being correct.

2015-12-01 20:41:43,2015-12-01 23:59:10,11847.190,155.69.203.103,67.58.52.143,51200,22,T

CP,.AP...,0,0,10251900,14999132600,0,0,559,669,9419,26397

Got solved by fixing the file.

3- **ANOTHER ERROR**

Open port 80 if the page is not accessible –

- `iptables -I INPUT -p tcp --dport 80 --syn -j ACCEPT`

Delete old indexes

- `curl -XDELETE 'http://localhost:9200/filebeat-*'`

ANOTHER PROBLEM: Adding new filters:

- Go to the following directory to add a new logstash filter-
  - `cd /etc/logstash/conf.d`
- Create a new filter
  - `vim`

  and paste the following in the newly created file

```
filter {

  if [type] == "flowprofile" {

      grok {         match => { "message" =>
"%{WORD:id}[,]%{TIME:time_start}[,]%{BASE10NUM:duration}[,]%{BASE1
0NUM:bytes}[,]%{BASE10NUM:rates}"

        }

    }

}

}
```

  - `vim /etc/filebeat/filebeat.yml`

- Edit the file as follows

```
  -

        paths:

          - /home/elkdata/files/flowprofile/*.csv

        document_type: flowprofile
```

- Save the file and exit
  - `sudo service logstash restart`

In case things are breaking, check logstash config with

   - `sudo service logstash configtest restart`

- Run Logstash manually

   - `/opt/logstash/bin/logstash –quiet -f /etc/logstash/conf.d &`

Adding a new filter for tuple:

- Create a grok pattern

```
%{WORD:id}[,]%{IP:source_ip_flowpro}[,]%{POSINT:source_port_flowpro}
[,]%{IP:dest_ip_flowpro}[,]%{POSINT:dest_port_flowpro}[,]%{WORD:prot
ocol_pro}
```

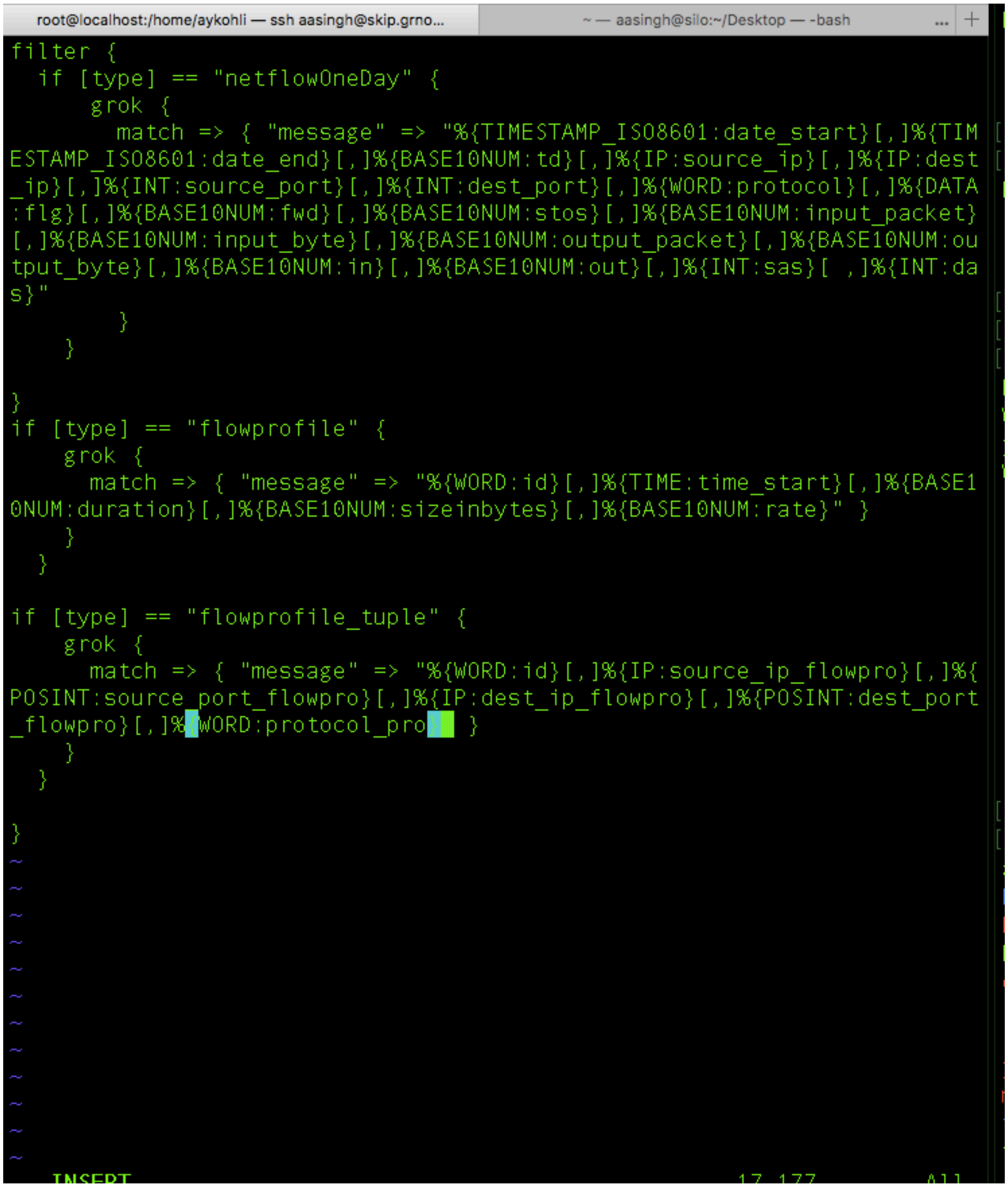
- Stop logstash and filebeat
   - `sudo service logstash stop`
   - `systemctl stop filebeat`

- Go over to the logstash configuration file and change the filter(basically add a new filter.
   - `vim /etc/logstash/conf.d/15-netlog-filter.conf`

- Add the filter:

```
if [type] == "flowprofile_tuple" {

    grok {      match => { "message" =>
"%{WORD:id}[,]%{IP:source_ip_flowpro}[,]%{POSINT:sou
rce_port_flowpro}[,]%{IP:dest_ip_flowpro}[,]%{POSINT
:dest_port_flowpro}[,]%{WORD:protocol_pro}" }

    }

  }
```

- So that the updated file looks like this –



root@localhost:/home/aykohli — ssh aasingh@skip.grno...   ~ — aasingh@silo:~/Desktop — -bash

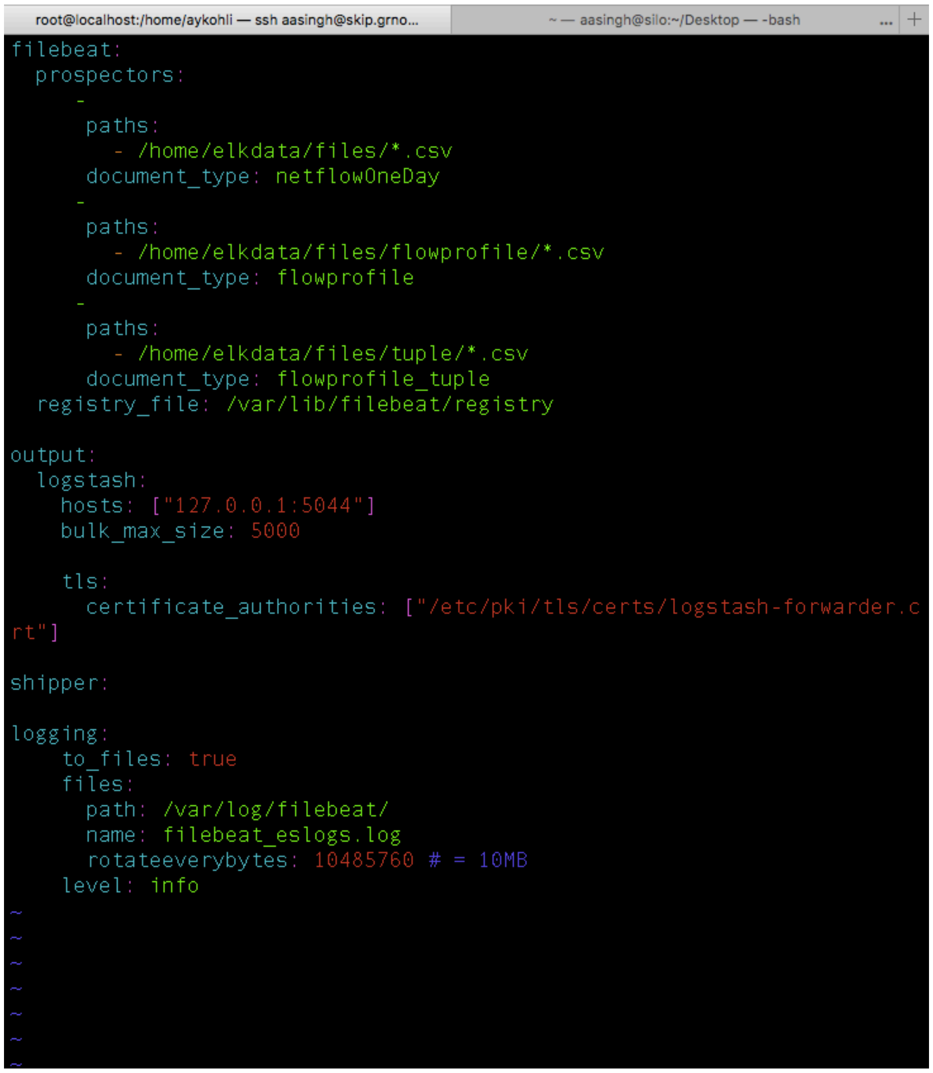
```
filter {
  if [type] == "netflowOneDay" {
      grok {
        match => { "message" => "%{TIMESTAMP_ISO8601:date_start}[,]%{TIMESTAMP_ISO8601:date_end}[,]%{BASE10NUM:td}[,]%{IP:source_ip}[,]%{IP:dest_ip}[,]%{INT:source_port}[,]%{INT:dest_port}[,]%{WORD:protocol}[,]%{DATA:flg}[,]%{BASE10NUM:fwd}[,]%{BASE10NUM:stos}[,]%{BASE10NUM:input_packet}[,]%{BASE10NUM:input_byte}[,]%{BASE10NUM:output_packet}[,]%{BASE10NUM:output_byte}[,]%{BASE10NUM:in}[,]%{BASE10NUM:out}[,]%{INT:sas}[ ,]%{INT:das}"
        }
    }

}
if [type] == "flowprofile" {
    grok {
      match => { "message" => "%{WORD:id}[,]%{TIME:time_start}[,]%{BASE10NUM:duration}[,]%{BASE10NUM:sizeinbytes}[,]%{BASE10NUM:rate}" }
    }
  }

if [type] == "flowprofile_tuple" {
    grok {
      match => { "message" => "%{WORD:id}[,]%{IP:source_ip_flowpro}[,]%{POSINT:source_port_flowpro}[,]%{IP:dest_ip_flowpro}[,]%{POSINT:dest_port_flowpro}[,]%{WORD:protocol_pro}" }
    }
  }

}
~
~
~
~
~
~
~
~
~
~
~
~
-- INSERT --                                             17,177         All
```

- Save and close the file.

- Now go to the filebeat config and add the new path for tuple csv.
    - `vim /etc/filebeat/filebeat.yml`
- Add the new path in the yml file.

```
-
          paths:
            - /home/elkdata/files/tuple/*.csv
          document_type: flowprofile_tuple
```

- The new file looks as follows:

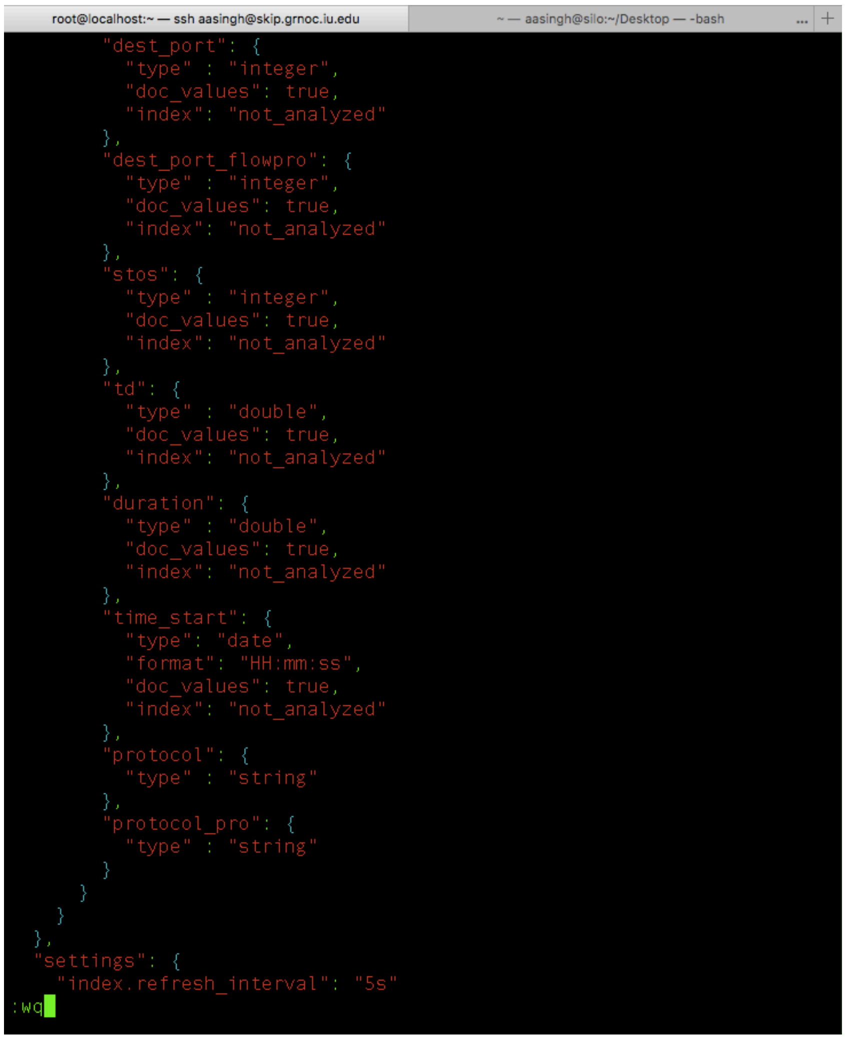


root@localhost:/home/aykohli — ssh aasingh@skip.grno... ~ — aasingh@silo:~/Desktop — -bash

```
filebeat:
  prospectors:
    -
      paths:
        - /home/elkdata/files/*.csv
      document_type: netflowOneDay
    -
      paths:
        - /home/elkdata/files/flowprofile/*.csv
      document_type: flowprofile
    -
      paths:
        - /home/elkdata/files/tuple/*.csv
      document_type: flowprofile_tuple
  registry_file: /var/lib/filebeat/registry

output:
  logstash:
    hosts: ["127.0.0.1:5044"]
    bulk_max_size: 5000

    tls:
      certificate_authorities: ["/etc/pki/tls/certs/logstash-forwarder.c
rt"]

shipper:

logging:
    to_files: true
    files:
      path: /var/log/filebeat/
      name: filebeat_eslogs.log
      rotateeverybytes: 10485760 # = 10MB
    level: info
~
~
~
~
~
~
~
```

- 7- Now we need to add the new variables to Kibana by changing kibana template file :
    - `cd ~`
    - `vim filebeat.template.json`
- Edit the file by adding all new variable. Eg-

```
"dest_port_flowpro": {
          "type" : "integer",
          "doc_values": true,
          "index": "not_analyzed"
        },
```



root@localhost:~ — ssh aasingh@skip.grnoc.iu.edu
~ — aasingh@silo:~/Desktop — -bash

```
        "dest_port": {
          "type" : "integer",
          "doc_values": true,
          "index": "not_analyzed"
        },
        "dest_port_flowpro": {
          "type" : "integer",
          "doc_values": true,
          "index": "not_analyzed"
        },
        "stos": {
          "type" : "integer",
          "doc_values": true,
          "index": "not_analyzed"
        },
        "td": {
          "type" : "double",
          "doc_values": true,
          "index": "not_analyzed"
        },
        "duration": {
          "type" : "double",
          "doc_values": true,
          "index": "not_analyzed"
        },
        "time_start": {
          "type": "date",
          "format": "HH:mm:ss",
          "doc_values": true,
          "index": "not_analyzed"
        },
        "protocol": {
          "type" : "string"
        },
        "protocol_pro": {
          "type" : "string"
        }
      }
    }
  },
  "settings": {
    "index.refresh_interval": "5s"
:wq
```

- Save and exit.
- Stop Kibana
    - `systemctl stop kibana  / service kibana start`
- Reload edited template
    - `curl -XPUT 'http://localhost:9200/_template/filebeat?pretty' -d@filebeat.template.json`
- Start kibana
    - `systemctl start kibana`
- Refresh Kibana filebeat from kibana interface
- Start logstash
    - `sudo service logstash start`
- Copy the new files where needed.
- Start filebeat

ANOTHER PROBLEM :

Change password to Kibana Interface or add a user?

- Type in the following command:
  - `sudo su`
  - `sudo htpasswd -c /etc/nginx/htpasswd.users username`
- Now put in the new password and you are all set!

NOTE: This will replace all the other users. So this is the only user account to reach kibana.

ANOTHER PROBLEM :

Stitch flows?

- Get csv files to */home/sylyon/* folder.
- Navigate to */home/sylyon/Stitching/* and run *FolderWalker.java* with command –
  - `java Folderwalker`
- This will generate stitched files in the folder */home/sylyon/STITCHED/* which you can export to */home/elkdata/files/*

Problem - Increase heap size?

or anyone looking to do this on Centos 7 or with another system running SystemD, you change it in

/etc/sysconfig/elasticsearch

Uncomment the ES_HEAP_SIZE line, and set a value, eg:

# Heap Size (defaults to 256m min, 1g max)

ES_HEAP_SIZE=16g

NETFLOW DATA

1 - How to import raw data to ELK from FlowProc

On flowproc navigate to the directory –

```
cd /media/netflow/profiles-data/live/rtr_seat_transpac/2017/03/
```

Create the output directory in this location –

```
mkdir /home/aasingh/newTranspacData/2017/03
```

Then run the following the script with needed modifications. This will run nfdump to create new data in the output folder-

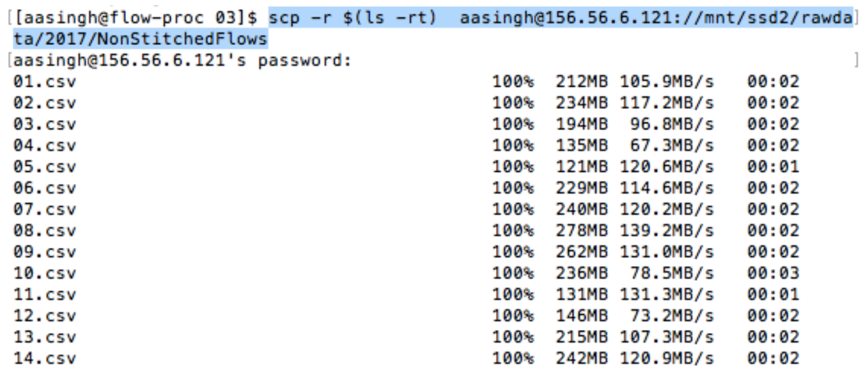
```
time for file in $(ls); do if [[ -d "$file" ]]; then nfdump -R
$file -o csv -s record/bytes -n 0 'bytes>100k'|cut --complement -
d, -f 20-48 | tail -n +2 |head -n -3 >>
/home/aasingh/newTranspacData/2017/03/$file.csv; elif [[ -f
"$file" ]]; then echo "$file is a regular file"; fi; done
```

Note- *This took me 22m37s for March 2017 files.*

Navigate to - `cd /home/aasingh/newTranspacData/2017/03/`

With the new data created from netflow, import it on 121 (elkprod) with following script which sends data in orderly fashion-

```
scp -r $(ls -rt)  aasingh@156.56.6.121://mnt/ssd2/rawdata/2017/NonStitchedFlows
```



```
[aasingh@flow-proc 03]$ scp -r $(ls -rt)  aasingh@156.56.6.121://mnt/ssd2/rawda
ta/2017/NonStitchedFlows
aasingh@156.56.6.121's password:
01.csv                                     100%  212MB 105.9MB/s   00:02
02.csv                                     100%  234MB 117.2MB/s   00:02
03.csv                                     100%  194MB  96.8MB/s   00:02
04.csv                                     100%  135MB  67.3MB/s   00:02
05.csv                                     100%  121MB 120.6MB/s   00:01
06.csv                                     100%  229MB 114.6MB/s   00:02
07.csv                                     100%  240MB 120.2MB/s   00:02
08.csv                                     100%  278MB 139.2MB/s   00:02
09.csv                                     100%  262MB 131.0MB/s   00:02
10.csv                                     100%  236MB  78.5MB/s   00:03
11.csv                                     100%  131MB 131.3MB/s   00:01
12.csv                                     100%  146MB  73.2MB/s   00:02
13.csv                                     100%  215MB 107.3MB/s   00:02
14.csv                                     100%  242MB 120.9MB/s   00:02
```

Note- *I gave permission to user to copy files to the directory on elkprod*
chown -R aasingh:aasingh ./mnt/ssd2/rawdata/

```
exit
```

Then, ssh the elk server

```
ssh 156.56.6.121
```

Take root permission

```
sudo su
```

Check in the `/mnt/ssd2/rawdata/2017/NonStitchedFlows/toStitch2017` **to see if files are in right folder, or else move the files to dedicated month folder.**

Once, we have the raw data in desired location, we need to change the location in

```
vim /mnt/ssd/Stitching/FolderWalker.java
```

```
public static void main(String[] args){
    final long startTime = System.nanoTime();
    // in this new filewalker object, the first path is the folder you 
want to stitch together
    // the second path is where you want the stitched files to go
   FolderWalker fw = new FolderWalker(/mnt/ssd2/rawdata/2017/NonStitchedFlows/toStitch2017",
                                       "/mnt/ssd2/rawdata/2017/StitchedFlows/Stitched2017");

    fw.run();
    //System.out.println(fw.getLastPath());
```

So, vim into the java file, find the main function and add the source and destination locations.

NOTE- make sure to have source as /mnt/ssd2/rawdata/2017/NonStitchedFlows and ensure there are only folders in this location i.e. it has folders like 08,09,10 etc and no files like something.csv. Files have to be inside each respective folder.

Note – Changed the directory structure a bit as follows:

- **Added folder '***/mnt/ssd2/rawdata/2017/NonStitchedFlows/**toStitch2017***'* **to '***/mnt/ssd2/rawdata/2017/NonStitchedFlows***' folder. This is important because we have to make sure the new data stays in an isolated folder. Old data is saved in '**/mnt/ssd2/rawdata/2017/NonStitchedFlows/**Archive' folder in the same directory location. Ensure you move the data to Archive folder once the work is done.**

- **Added folder '***/mnt/ssd2/rawdata/2017/StitchedFlows/Stitched2017*' & '*/mnt/ssd2/rawdata/2017/StitchedFlows/Stitched2017/Archive*'. **This is again important that folder Stitched2017 is always empty for new data**

and old data is moved to Archive folder at
'/mnt/ssd2/rawdata/2017/StitchedFlows/Archive'

Now navigate to the directory-

```
cd /mnt/ssd/Stitching/
```

Compile FolderWalker.java with javac command –

```
javac FolderWalker.java
```

Then, run the java file as -

```
java FolderWalker
```

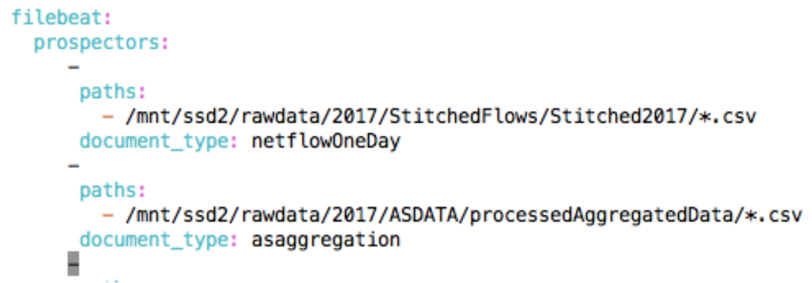
Note- If you are getting a `String index out of range` exception, it's usually because your source or destination already has some partially processed file. Remove them and retry.

This should stitch the files. Please ensure the files are being stitched in order by looking at the first few outputs of this program.

Once the stitching scipt is complete, make sure that filebeat is monitoring this folder by checking the config at –

```
vim /etc/filebeat/filebeat.yml
```

Here, you will see config for netflowOneDay, asaggregation,& flowprofile. Since we are working with Netflow data, we check netflowOneDay config and set it accordingly –



```
filebeat:
  prospectors:
      -
       paths:
         - /mnt/ssd2/rawdata/2017/StitchedFlows/Stitched2017/*.csv
       document_type: netflowOneDay
      -
       paths:
         - /mnt/ssd2/rawdata/2017/ASDATA/processedAggregatedData/*.csv
       document_type: asaggregation
      -
```

Just restart the filebeat service with this command –

```
      sudo service filebeat restart
```

And check status with –

```
      sudo service filebeat status
```

That is all with Netflow data. It should be making its way into ELK at this moment. Once done, generally after a couple hours, you can stop filebeat to stop accidental data transfers due to change by

```
      sudo service filebeat stop
```

AS-AGGREGATION DATA

On flowproc navigate to the directory specific to the year (not month)-

```
cd /media/netflow/profiles-data/live/rtr_seat_transpac/2017/
```

Once in the right folder, run the following script with the correct parameters-

```
day_start=01 day_end=31 month=03 year=2017; for day in
$(seq -w $day_start $day_end); do nfdump -R $month/$day  -o
csv -n 0 -A srcas,dstas -s record/bytes >
/home/aasingh/temp.csv; cut --complement -d, -f 1-12,14-
17,20-48 /home/aasingh/temp.csv >
/home/aasingh/aggregateflows/$month$day$year.csv ;wc -l
/home/aasingh/aggregateflows/$month$day$year.csv; done
```

Note – here, *day_end* changes on number of days in that month. For eg- 31,30,39

The new aggregated data is saved at – '/home/aasingh/aggregateflows/'

Now move this data over to 121 (elkprod). We can send this in any order but the right order is good practice –

NOTE: I added an extra layer of directory structure under ASDATA. Now we have 'rawAggregatedData' and 'processedAggregatedData' which have an extra layer containing, 'Archive' and 'Processed' folders. Archive folder is being monitored by 'Filebeat' while 'Processed' is for preparing the data. Once the data is processed send it from 'Processed' folder to 'Archive' folder and it will be shipped to elasticsearch by Filebeat.

```
cd /home/aasingh/aggregateflows/
```

```
scp -r $(ls -rt 03*2017.csv)
aasingh@156.56.6.121://mnt/ssd2/rawdata/2017/ASDATA/rawAggregatedData/toProce
ss
```

Now, the data is on 121, so go to elkprod (with root access) and navigate to the directory –

```
cd
```

```
/mnt/ssd2/rawdata/2017/ASDATA/rawAggregatedData/toProcess
```

We will remove the header and tail information from the aggregate files with this script –

```
    for file in ./03*2017.csv; do tail -n +2 $file > ./temp.csv
    ; sudo rm $file; head -n -3 ./temp.csv > $file ; sudo rm
    temp.csv;  done;
```

Now move the data you want to process in 'toProcess' **folder. Anything old should be in** 'Archive' folder.

Time for python. Let's setup the environment with this script

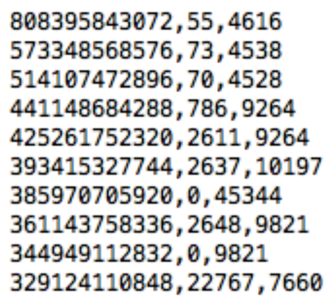
```
   source /home/aasingh/environments/root_env/bin/activate
```

Now run the python script `astable.py` **at location /**`mnt/ssd2/rawdata/2017/ASDATA` **with this command –**

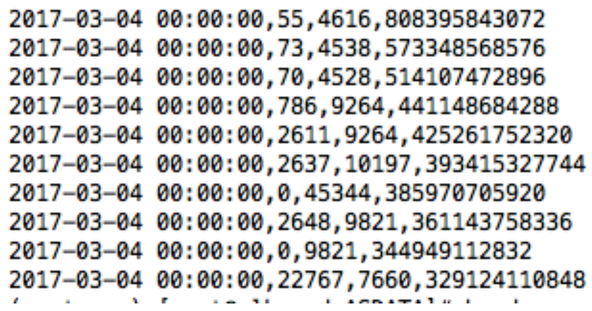
```
   python astable.py
```

This script will change the format of the aggregate files as follows:

- Add datetime field
- Change column order to DATETIME,SAS,DAS,INPUTBYTE



```
808395843072,55,4616
573348568576,73,4538
514107472896,70,4528
441148684288,786,9264
425261752320,2611,9264
393415327744,2637,10197
385970705920,0,45344
361143758336,2648,9821
344949112832,0,9821
329124110848,22767,7660
```



```
2017-03-04 00:00:00,55,4616,808395843072
2017-03-04 00:00:00,73,4538,573348568576
2017-03-04 00:00:00,70,4528,514107472896
2017-03-04 00:00:00,786,9264,441148684288
2017-03-04 00:00:00,2611,9264,425261752320
2017-03-04 00:00:00,2637,10197,393415327744
2017-03-04 00:00:00,0,45344,385970705920
2017-03-04 00:00:00,2648,9821,361143758336
2017-03-04 00:00:00,0,9821,344949112832
2017-03-04 00:00:00,22767,7660,329124110848
```

Now move the processed files to respective Archive folders as follows:
This is one complete script-

```
   mv rawAggregatedData/toProcess/*.csv ./rawAggregatedData/Archive/ ; mv
processedAggregatedData/Processed/*.csv processedAggregatedData/Archive/
```

Now restart filebeat if not already running with –

```
	sudo service filebeat restart
```

It will automatically start sending new files to ELK. You can check that with -

```
	sudo service filebeat status
```

And off course, same as before, if you need to change config –

```
	vim /etc/filebeat/filebeat.yml
```

FLOWPROFILE DATA

1- Run python on 121 –

```
	source /home/aasingh/environments/root_env/bin/activate
```

2- Go to the directory –

```
	cd /home/aasingh/flowpro/
```

3- Run the python script (in the background) to get flows above 500 mb –

```
python pyes2csv.py &
```

Troubleshooting notes -

- Change the size parameter in the script inside python file to control the number of results returned.
- Fixing the problem of window size in elastic search leading to only showing 10000 results at once. It was fixed with running the following command as root on 121 -

```
curl -XPUT "http://localhost:9200/filebeat-*/_settings" -d
'{ "index" : { "max_result_window" : 500000 } }'
```

4- **Run another python script which will create a new text data file with date columns in a specific format.**

```
	python py2adddates.py
```

Go to flow proc now

5- Next, run the script that will go to raw data for every row in input file-

```
nohup cd /media/REMOTE_SHARES/flow-
store/rtr.losa.transpac/2016/;IFS=","; cat
/home/aasingh/flowpro/outputfilefinal.csv | while read sip sport dip
dport proto input_bytes date_start date_end year_start month_start
day_start year_end month_end day_end ; do if [ $month_start -eq
$month_end ]; then for days in $(seq $day_start $day_end) ; do nfdump
-R /media/REMOTE_SHARES/flow-
store/rtr.losa.transpac/$year_start/$month_start/$day -o csv "proto
$proto and src ip $sip and src port $sport and dst ip $dip and dst
port $dport" | cut --complement -d, -f 20-48 | tail -n +2 | head -n -
3; done ; else for months in $(seq $month_start $month_end) ; do
nfdump -R /media/REMOTE_SHARES/flow-
store/rtr.losa.transpac/$year_start/$month -o csv "proto $proto and
src ip $sip and src port $sport and dst ip $dip and dst port $dport" |
cut --complement -d, -f 20-48 | tail -n +2 | head -n -3; done; fi;
sleep 2; done >> /home/aasingh/flowpro/flowout.csv &
```

6- Copy prepared files from flowproc to ELKPROD in folder - /mnt/ssd2/rawdata/year/flowpro

Once the data is prepared on flowproc in the desired folder, we have to send it to elkprod. Again order here doesn't matter, so we can use any copy command to send to elkprod.

7- **Run flowmerge.py and flowmerge2.py** in this order to add metadata and change column order.

8- **Filebeat import:** Once the data is in elkprod under the right directory, and filebeat is running (restart it if not already running), the data will start flowing into elk automatically. You can check with filebeat's status command.

Files and Directory Structure :

1. **LOGSTASH CONFIG FILES STORED AT /etc/logstash/conf.d/**
    - **02-beats-input.conf** – Beats configuration including index names and types. No updates needed usually.
    - **15-netlog-filter.conf** – Logstash configuration including input, output, filter, transformation, and grok pattern. Very important file. Updates required if data pattern changes.
    - **30-elasticsearch-output.conf** – Elasticsearch config like port, output, nodes, etc. No updates needed usually.

2. **NGINX Config:**
    - **kibana.conf :** This is an nginx config file specific for kibana dashboard. Contains major kibana configurations such as certificates, proxy, ports, dashboard files, authentication, etc.

3. **DASHBOARD:**
    - **index.html:** Contains HTML and javascript used to connect dashboard with Kibana with all visual customizations like buttons, colours, etc.

4. **BEATS/FILEBEAT:**
    - **filebeat.template.json:** Mappings file. The contents of this file describe the schema of elasticsearch content. Here you can define the datatypes and other mapping features of values you are importing into ELK.
    - **filebeat.yml:** Filebeat config file. Very important file containing locations where filebeat is monitoring.

5. **ELASTICSEARCH:**
    - **elasticsearch.yml:** Contains elasticsearch config information. Very important file for setting up, optimizing and scaling Elasticsearch. Describes major properties like Cluster, nodes, Heap size, etc.

6. **PYTHON SCRIPTS:**
    - **astable.py :** Used in importing ASDATA. This file goes through each record of AS aggregation imported from flowproc and updates the order of columns and adds datetime field.

- **flowmerge.py:** Used in flowprofiling. It parses the flowprofile data and removes AP flag, converts bytes to bits per second.

- **flowmerge2.py :** Removes unnecessary columns from raw data.

- **ips.py:** File for Abhinandan’s task. It filters IPV6 and V4 records into separate files.

- **pyes2csv.py:** Very important file for flowprofiling. It gets records/flows which are greater than a certain size. Example– flows above 500 Mb. It also adds certain fields for readability.

- **py2adddates.py:** Run this after pyes2csv.py. It adds datetime to the records and makes them eligible to be imported to ELK.

Directory structure of Data Stored:

```
/mnt/ssd2/rawdata/
/mnt/ssd2/rawdata/2017
/mnt/ssd2/rawdata/2017/ASDATA
/mnt/ssd2/rawdata/2017/ASDATA/processedAggregatedData
/mnt/ssd2/rawdata/2017/ASDATA/processedAggregatedData/Archive
/mnt/ssd2/rawdata/2017/ASDATA/processedAggregatedData/Processed
/mnt/ssd2/rawdata/2017/ASDATA/rawAggregatedData
/mnt/ssd2/rawdata/2017/ASDATA/rawAggregatedData/toProcess
/mnt/ssd2/rawdata/2017/ASDATA/rawAggregatedData/Archive
/mnt/ssd2/rawdata/2017/flowpro
/mnt/ssd2/rawdata/2017/StitchedFlows
/mnt/ssd2/rawdata/2017/StitchedFlows/Archive
/mnt/ssd2/rawdata/2017/StitchedFlows/Stitched2017
/mnt/ssd2/rawdata/2017/NonStitchedFlows
/mnt/ssd2/rawdata/2017/NonStitchedFlows/Archive
/mnt/ssd2/rawdata/2017/NonStitchedFlows/Archive/01
/mnt/ssd2/rawdata/2017/NonStitchedFlows/Archive/02
/mnt/ssd2/rawdata/2017/NonStitchedFlows/toStitch2017
/mnt/ssd2/rawdata/2017/NonStitchedFlows/toStitch2017/03
/mnt/ssd2/rawdata/2015
/mnt/ssd2/rawdata/2015/ASDATA
/mnt/ssd2/rawdata/2015/ASDATA/rawAggregatedData
/mnt/ssd2/rawdata/2015/ASDATA/processedAggregatedData
/mnt/ssd2/rawdata/2015/flowpro
```

```
/mnt/ssd2/rawdata/2015/NonStitchedFlows
/mnt/ssd2/rawdata/2015/NonStitchedFlows/02
/mnt/ssd2/rawdata/2015/NonStitchedFlows/03
/mnt/ssd2/rawdata/2015/NonStitchedFlows/04
/mnt/ssd2/rawdata/2015/NonStitchedFlows/05
/mnt/ssd2/rawdata/2015/NonStitchedFlows/06
/mnt/ssd2/rawdata/2015/NonStitchedFlows/07
/mnt/ssd2/rawdata/2015/NonStitchedFlows/08
/mnt/ssd2/rawdata/2015/NonStitchedFlows/09
/mnt/ssd2/rawdata/2015/NonStitchedFlows/10
/mnt/ssd2/rawdata/2015/NonStitchedFlows/11
/mnt/ssd2/rawdata/2015/NonStitchedFlows/12
/mnt/ssd2/rawdata/2015/StitchedFlows
/mnt/ssd2/rawdata/2016
/mnt/ssd2/rawdata/2016/ASDATA
/mnt/ssd2/rawdata/2016/ASDATA/rawAggregatedData
/mnt/ssd2/rawdata/2016/ASDATA/rawAggregatedData/toProcess
/mnt/ssd2/rawdata/2016/ASDATA/rawAggregatedData/Archive
/mnt/ssd2/rawdata/2016/ASDATA/processedAggregatedData
/mnt/ssd2/rawdata/2016/ASDATA/processedAggregatedData/Archive
/mnt/ssd2/rawdata/2016/ASDATA/processedAggregatedData/Archive2
/mnt/ssd2/rawdata/2016/ASDATA/processedAggregatedData/Processed
/mnt/ssd2/rawdata/2016/flowpro
/mnt/ssd2/rawdata/2016/StitchedFlows
/mnt/ssd2/rawdata/2016/StitchedFlows/Archive
/mnt/ssd2/rawdata/2016/StitchedFlows/Archive/beforeSeattle
/mnt/ssd2/rawdata/2016/StitchedFlows/Stitched2017
/mnt/ssd2/rawdata/2016/NonStitchedFlows
/mnt/ssd2/rawdata/2016/NonStitchedFlows/Archive
/mnt/ssd2/rawdata/2016/NonStitchedFlows/Archive/01
/mnt/ssd2/rawdata/2016/NonStitchedFlows/Archive/02
/mnt/ssd2/rawdata/2016/NonStitchedFlows/Archive/03
/mnt/ssd2/rawdata/2016/NonStitchedFlows/Archive/04
/mnt/ssd2/rawdata/2016/NonStitchedFlows/Archive/07
/mnt/ssd2/rawdata/2016/NonStitchedFlows/Archive/08
/mnt/ssd2/rawdata/2016/NonStitchedFlows/Archive/09
/mnt/ssd2/rawdata/2016/NonStitchedFlows/Archive/10
/mnt/ssd2/rawdata/2016/NonStitchedFlows/Archive/11
/mnt/ssd2/rawdata/2016/NonStitchedFlows/Archive/12
/mnt/ssd2/rawdata/2016/NonStitchedFlows/toStitch2017
```

Future work and incomplete modules as of May 4, 2017

1- **DNS Hostname in ELK:** Currently, we only have AS names going in to ELK with the help of dictionary plugin in logstash. This dictionary has 'ASNUMBER':'ASNAME'. Similarly, we want to extend this concept to IPAddresses too such that we have a dictionary containing 'IPADDRESS':'HOSTNAME'. This dictionary will be restricted only to top IP addresses (in terms of input bytes). We also have to setup a provision so that this dictionary is updated with new entries as we see more top flows in data.

   Another good way to approach this problem is using DNS plugin in Logstash itself. It worked in the past. But it doesn't seem to be working as of now. Also, this might be an inefficient approach because we will be sending way too much queries to DNS servers, which might block us thinking a DDOS perpetrator. So, we can explore this option but dictionary approach is more feasible as of now.

2- **Last Day Stitching:** Our stitching script went through some modification and is currently working fine as long as we provide it 'data in right order to read'. For a flow that goes on for multiple days, the script will stitch all of the days together, except the last day. This is a bug. For example- a flow that goes on for 11 days will have stitched with 10 days as one record and another record of the last day.

   Most probable cause is in the java file Stitcher.java.

3- **Logstash as Service:** Currently, we are running logstash manually as a background process. In order to stop logstash, we search for it's PID and manually kill it. You can find PID by using top and looking at the process with name java and usually taking a lot of memory and CPU after elasticsearch itself.

   This is a bug since we should be able to run logstash as a service like –' service logstash start' or ' systemctl start logstash'. Most probable cause is file permissions. There seems to be some logstash folder or files which are not given permission to the user 'logstash'.

4- **Flow-profiling:** Although the entire pipeline for generating and importing flowprofiles is in place and well documented, the process itself is inefficient. The problem is when we run nfdump command to get flowprofile for a flow with 5 tuples, it fetches all of its raw records from nfdump. If the flow is really long and goes over multiple days, nfdump process closes automatically. For example, while working with flows which were larger than 500MB, i.e flows exceeding TB's, I had to restart nfdump process multiple times for example at 500flows, 900flows, 1600flows. But once the flow started to be smaller, the process ran fine and didn't close automatically. This is a bug either with my bash script which we run on flowproc or nfdump as a process, or flowproc server which kills a process if it runs over a certain time threshold.

5- **Updates to netflow Dashboard**:The dashboard is a simple html file with javascript. You can find it -
/var/www/nginxsite.com/public_html/

------END OF FORMAL DOCUMENTATION - PRACTICE QUERIES BELOW------

Query to aggregate with ips for africa data

PREPARE AGGREGATED DATA FROM RAW DATA-

```
day_start=01 day_end=30 month=06 year=2015; for day in $(seq -w
$day_start $day_end); do nfdump -R $month/$day  -o csv -n 0 -A
srcas,dstas -s record/bytes > /home/aasingh/temp.csv; cut --complement
-d, -f 1-12,14-17,20-48 /home/aasingh/temp.csv >
/home/aasingh/aggregateflows/$month$day$year.csv ;wc -l
/home/aasingh/aggregateflows/$month$day$year.csv; done
```

AFTER PREPARING AGGREGATED DATA COPY TO FOLDER ON 121 SERVER -

REMOVE THE TOP AND BOTTOM LINES FROM AGGREGATED DATA-

```
location=/mnt/ssd2/rawdata/2016/ASDATA/;for file in $location/*.csv;
do tail -n +2 $file > $location/temp.csv ; sudo rm $file; head -n -3
$location/temp.csv > $file ; done
```

RUN THE PYTHON CODE TO ADD DATE AND PROCESS IT

```
cd /home/aasingh/environments/data/
```

```
python astable.py 31 12
```

that is - python astable.py date month (only if filename pattern is monthNameYear.csv. Change python code if file name pattern changes)

a quick solution to some months -

```
for month in {03..12}; do python astable.py 30 $month; done
```

```
for day in {05..30}; do python astable.py $day 06; done
```

Paste it to elk location for filebeat to automatically send these logs

to ELK-

```
cp /mnt/ssd/2015/ASDATA/processedAggregatedData/10*_mod.csv
/home/elkdata/files/asdata/
```

Query for everything (while, if, for, enumeration) working on demo dataset

```
IFS=","; cat /home/aasingh/environments/outputfile.csv | while read
sip sport dip dport proto input_bytes date_start date_end year_start
month_start day_start year_end month_end day_end ; do if [
$month_start -eq $month_end ]; then for n in "flowpro" "root_env" ; do
ls $n; done ; else echo "run for both months entirely"; fi; done
```

Query for if else with csv reading and dates

```
IFS=","; cat /home/aasingh/environments/outputfile.csv | while read
sip sport dip dport proto input_bytes date_start date_end year_start
month_start day_start year_end month_end day_end ; do if [
$month_start -eq $month_end ]; then echo "run for just the days the
flow happened"; else echo "run for both months entirely"; fi; done
```

Query to end to the end of line

```
sed "/^/ s/$/,$test/" /home/aasingh/flowpro/flowsreplica.csv
```

Query for if else

```
if [ $(echo $TEST |cut -d'-' -f 1) -eq 2016 ]; then echo "yo"; else
echo "nada"; fi
```

Query to access directory using date start/end

```
ls /media/REMOTE_SHARES/flow-store/rtr.losa.transpac/"$(echo $TEST
|cut -d'-' -f 1)"/"$(echo $TEST |cut -d'-' -f 2)"/"$(echo $TEST |cut -
d'-' -f 3| cut -d' ' -f 1)"
```

Query to get day month year in csv

```
IFS=",";cat /home/aasingh/flowpro/flows.csv | while read sip sport dip
dport proto input_bytes date_start date_end; do echo "$date_start" |
cut -d'-' -f 3| cut -d' ' -f 1; done
```

Query to get result above 10000

-

f you need big deep pagination, I think only one variant of solution is to increase value max_result_window

```
curl -XPUT "http://localhost:9200/my_index/_settings" -d '{ "index" :
{ "max_result_window" : 500000 } }'
```

The increase in memory usage, I is not found for values of ~ 100k