# Getting camlogs(s) in a seperate index

To create an index, an index mapping is needed - In general, I think for collecting filebased logs - the filebeat template suits me.

Make a copy of filebeat.json from the zip package at https://download.elastic.co/beats/dashboards/beats-dashboards-1.1.0.zip and change filebeat.json name and the content likewise.

Then upload and create index.

```
root@elkserver:curl -XPUT http://localhost:9200/.kibana/index-pattern/camlog-* -d @camlog.json
{"_index":".kibana","_type":"index-pattern","_id":"camlog-*","_version":2,"_shards":{"total":2,"successful":1,"
failed":0},"created":false}
root@elkserver:
```

Then, copy filebeat-index-template.json to camlog-index-template.json (and change the content likewise)

```
root@elkserver:~# curl -XPUT 'http://localhost:9200/_template/camlog?pretty' -d@camlog-index-template.json
{
  "acknowledged" : true
}
root@elkserver:~#
```

The collection on moserver is (this is a part of it)

**/etc/filebeat/filebeat.yml**

```
      paths:
        - /data/camera-data/Fordor.log
        - /data/camera-data/Baghus.log
      document_type: camlog
      input_type: log
```

This is shipped to Logstash, where output is configured for ElasticSearch- notice the if for type "camlog":

**/etc/logstash/conf.d/30-elasticsearch-output.conf**

```
output {
  if [type] == "pinglog" {
    elasticsearch {
      hosts => ["localhost:9200"]
      sniffing => true
      manage_template => false
      index => "pinglog-%{+YYYY.MM.dd}"
      document_type => "%{[@metadata][type]}"
    }
  }
  else
  {
    if [type] == "camlog" {
      elasticsearch {
        hosts => ["localhost:9200"]
        sniffing => true
        manage_template => false
        index => "camlog-%{+YYYY.MM.dd}"
        document_type => "%{[@metadata][type]}"
      }
    }
    else
    {
      elasticsearch {
        hosts => ["localhost:9200"]
        sniffing => true
        manage_template => false
        index => "%{[@metadata][beat]}-%{+YYYY.MM.dd}"
        document_type => "%{[@metadata][type]}"
      }
    }
  }
}
```

Test and restart:

```
root@elkserver:/etc/logstash/conf.d# /etc/init.d/logstash configtest
Configuration OK
root@elkserver:/etc/logstash/conf.d# /etc/init.d/logstash restart
Killing logstash (pid 16262) with SIGTERM
Waiting logstash (pid 16262) to die...
Waiting logstash (pid 16262) to die...
Waiting logstash (pid 16262) to die...
Waiting logstash (pid 16262) to die...
logstash stopped.
logstash started.
root@elkserver:/etc/logstash/conf.d#
```

As I have ensured logs are pretty much key=value for the values I need to collect, I use kv to get the fields, and a mutate to ensure the string value for "pingtime=" becomes a float (otherwise it cant be used in a Visualization):

**/etc/logstash/conf.d/02-beats-input.conf**

```
input {
  beats {
    port => 5044
    ssl => true
    ssl_certificate => "/etc/pki/tls/certs/logstash-forwarder.crt"
    ssl_key => "/etc/pki/tls/private/logstash-forwarder.key"
  }
}
filter {
  if [type] == "apache" {
    grok {
      match => { "message" => "%{COMBINEDAPACHELOG}" }
    }
    date {
      match => [ "timestamp" , "dd/MMM/yyyy:HH:mm:ss Z" ]
    }
    geoip {
      source => "clientip"
      target => "geoip"
      database => "/etc/logstash/GeoLiteCity.dat"
      add_field => [ "[geoip][coordinates]", "%{[geoip][longitude]}" ]
      add_field => [ "[geoip][coordinates]", "%{[geoip][latitude]}"  ]
    }
    mutate {
      convert => [ "[geoip][coordinates]", "float"]
    }
  }
  else
  {
    if [type] == "pinglog" {
      kv {}
      mutate {
        convert => { "pingtime" => "float" }
      }
    }
  }
}
```

This should bring pinglogs in the index "pinglogs"