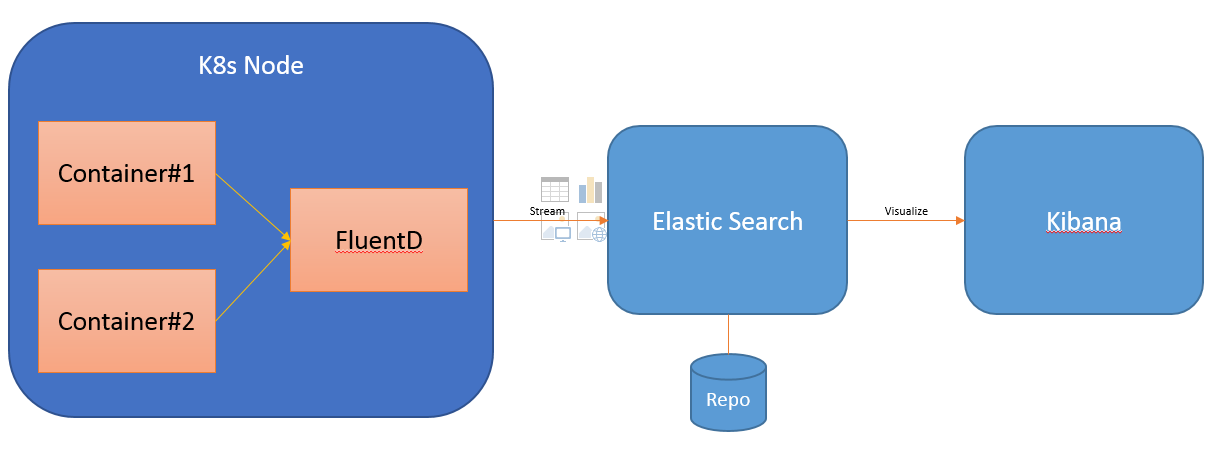
In this article, we will see how to use EFK for analyzing logs of applications deployed in K8s. I do believe that EFK stack does not require any introduction. It stands for Elastic Search, FluentD and Kibana.

There might be thousands of pods and each pod will have logs. We have already seen how to use distributed tracing using open tracing and jaeger. However, one more way of inspecting logs is EFK, which gives centralized logging system for K8s ecosystem. EFK also provides capabilities to investigate historical logs, search logs or filter logs based on the level – error, info, debug etc. Today, Elastic stack (ELK/EFK) is the standard way of managing the logs in the distributed systems.

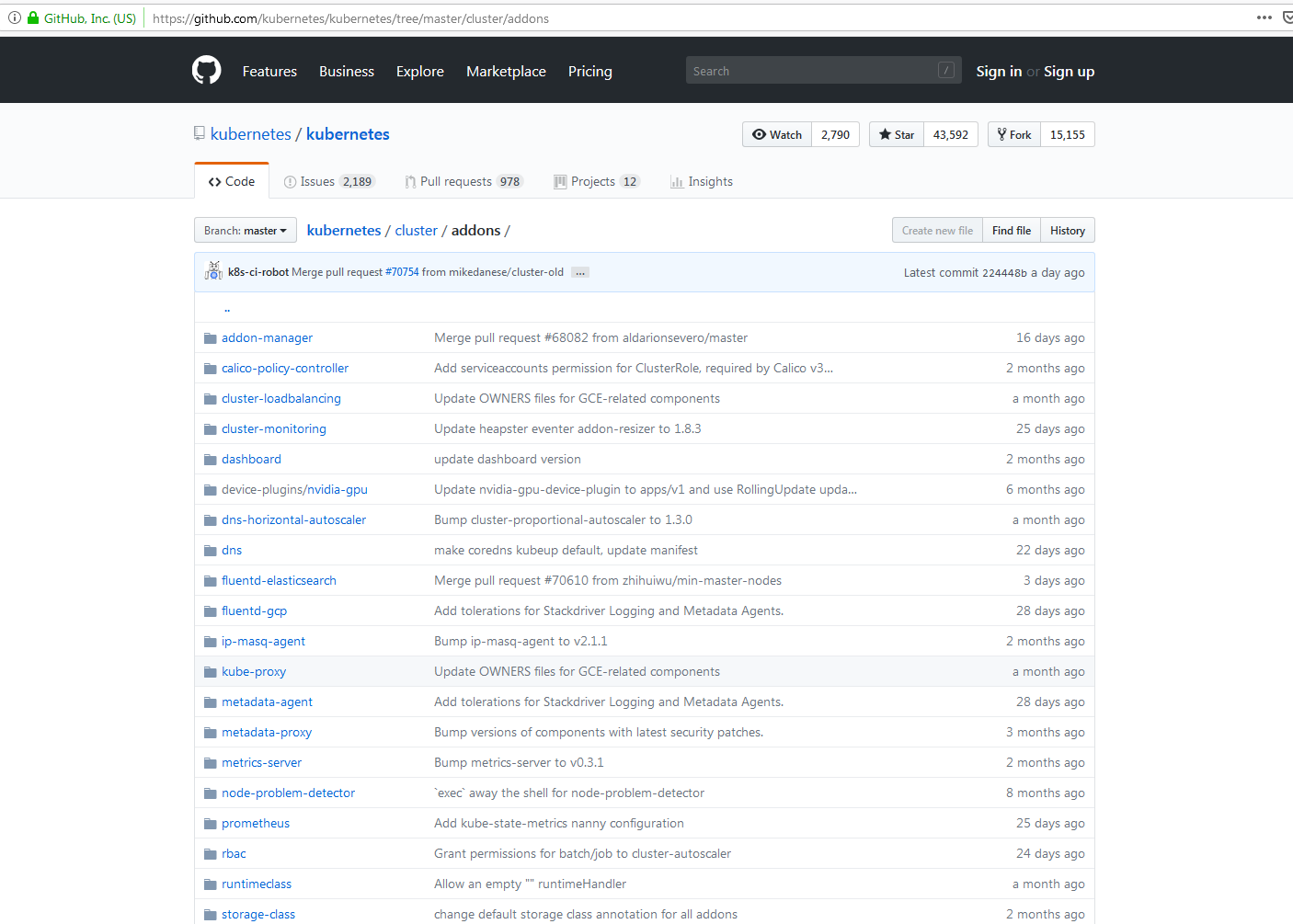
In addition, as ELK provides analytical and visualization capabilities, one can use EFK to monitor the K8s cluster. By monitoring, I refer to node health status, service performance, cpu, disk or network metrics an so on. However, I do not recommend to use EFK to do so as we have already used the best one in this space – Prometheus and Grafana. I do believe that one should always prefer to use right tools for right job.

We require few pods running in the K8s cluster. Those pods should log the traces and that’s it. Then we need to setup EFK for K8s and those who have worked with EFK know how intimidating it could to integrate EFK or ELK with other ecosystems be its Spark cluster or be its Hadoop cluster or be its K8s. That’s why in order to keep things simple, we are going to use standard configuration of EFK packaged only for K8s. FluentD (logstash equivalent) acts as the bridge of this integration and we are not going to change anything in the standard configuration. We are not going to experiement with log pattern, index name, refresh interval and so on. Once integration is done then its easy to customize the different configuration options.

Broadly, let me explain the overall working. In K8s cluster, we have nodes on which pods are running. Inside pods there could be multiple containers logging the traces. Now FluentD will be deployed in the K8s cluster. Docker container will be logging the traces and FluentD acts as data collector and will keep on collecting the logs as it works on pull model. FluentD pushes data to ElasticSearch. Elastic Search is a distributed search engine and it stores data into indexes and it works on a structure called as inverted index. Then the logs can be queried and visualized using Kibana.



Lets get started. Navigate to K8s Github page and explore the addon **fluentd-elasticsearch**.



Navigate to fluentd-elasticsearch. Now inside the repo, you will find different configuration files. There are three components – Elastic Search, Kibana and FluentD. For Elastic and Kibana, there are workload and service configuration yaml files. Whereas for FluentD, there are two files – one for configmap, which we will explore later and other one for the service.

I have done lot of talking. Let’s get into the action. I have collated different configuration files into two. These configuration files are available on my github - .The configmap file of FluentD is pretty much complicated and it’s tailored for K8s cluster. FluentD adds extra details to logs as we are instructing FluentD to add container name, node id, log pattern, pod name, namespace and so on. As I have described earlier, we need FluentD on each of the node. That’s why we also require a special configuration and in K8s that configuration is known as DaemonSet. In K8s, if any object is defined as DaemonSet, a special type of workload, then that entity is installed on each of the node.



Take a note of namespace. It’s kube-system. In addition, notice configuration of other two components – Elastic and Kibana. Here comes Elastic service and related configuration. Its deployed in the namespace kube-system and service name is elasticsearch-logging.



Now, I want you to pay special attention to type StatefulSet. It’s a special type in K8s and its akin to ReplicaSet; however StatefulSet provides one feature of managing the name of the pods created as a part of replicas. The name of the elastic pods will not be changed or no random generated suffix will be added to the name defined in the metadata that is elasticsearch-logging0. As we have specified two replicas, there will be two pods, one with a name elasticsearch-logging0 and other one with elasticsearch-logging1.

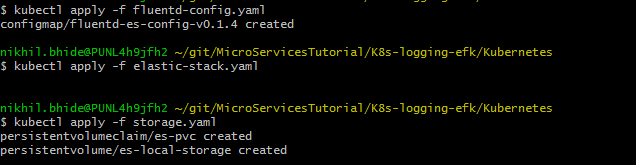


Similarly, Kibana is also deployed in the same namespace and deployment name is kibana-logging.

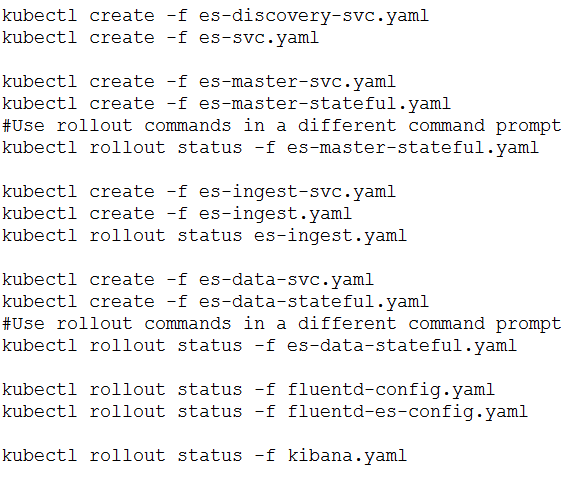


That is enough talking! I will be explaining the rest of the important aspects of the configuration in the last section of article. Let us now apply the configuration and get our hands dirty.

Now first that we will be doing is applying changes of FluentD. We will use standard command – kubectl apply. Then we will also create storage volumes so that elastic search pods can be created. Before we apply changes of elastic search and kibana, we need to create a volume storage so that elastic search can persist the data.



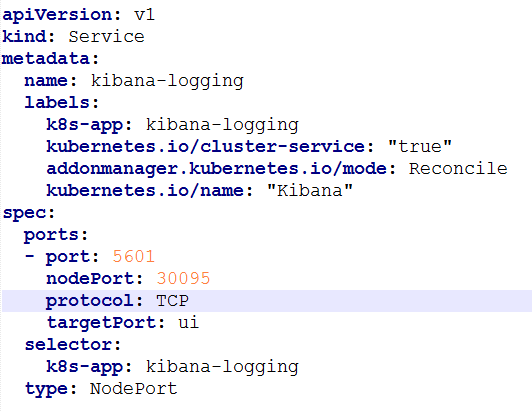
Then comes applying changes of elastic search configuration.



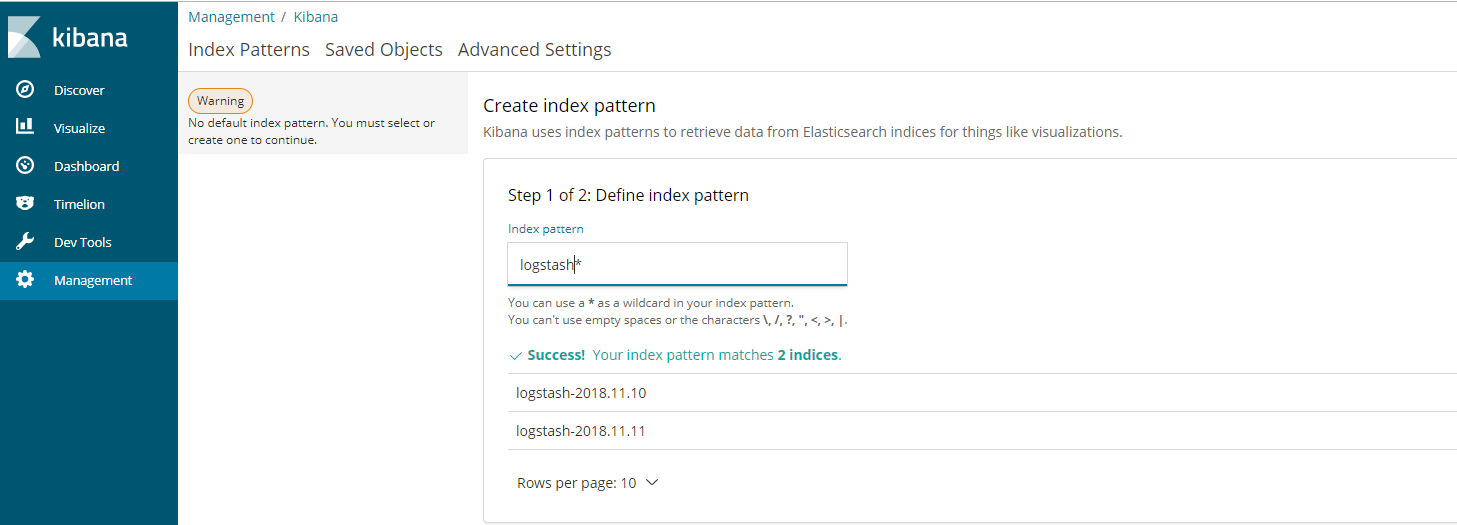
Once changes are applied, check different pods, services created in the K8s ecosystem by querying to kube-system namespace.

The standard configuration of elastic with K8s comes up without any persistence. What does it mean? If you are familiar with working of Elastic Search then its easy to understand. As I have described earlier in the flow that FluentD pushes data to Elastic and Elastic stores the data in the form of index. The data is stored in inverted index structure format on a file system and this file system can be local, nfs/gfs which are shared file systems or any data store such as mongo. Now when no configuration is provided, ElasticSearch stores the data on the local file system. Now there involves a risk as if elasticsearch pod is crashed or restarted because of any reason then data is also lost. In order to tackle this problem, we need to mount some kind of volume to Elastic Search deployment. In this example, I have demonstrated how to file system of a K8s node to a elastic search pod. K8s supports many volumes, and in the example, I have used Standard. StorageClasses use provisioners that are specific to the storage platform or cloud provider to give Kubernetes access to the physical media being use.

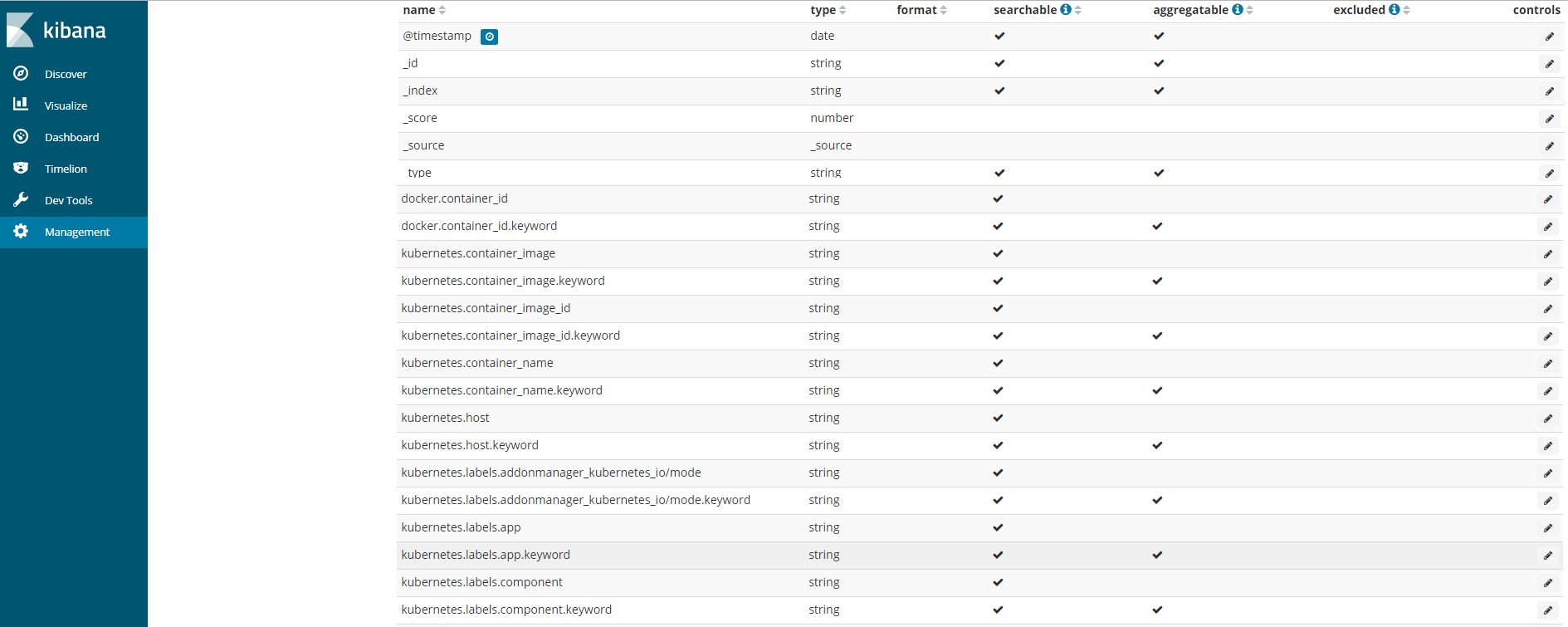
Now we are done with the configuration. Now we have exposed Kibana service as a NodePort so that it can be available outside of K8s cluster.



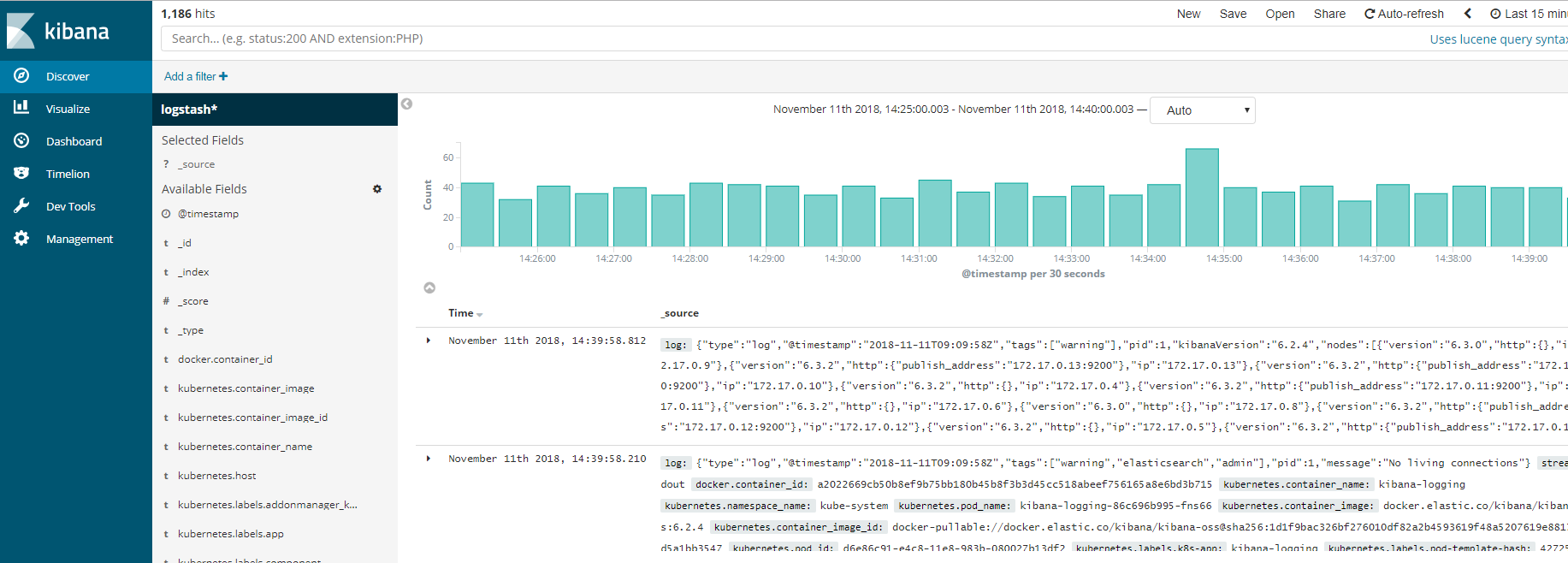
Now figure out ip address of minikube and navigate to Kibana dashboard at http://<MINIKUBE\_IP>:30095 and check whether you are able to see integration with elasticsearch is done. You can verify this by visiting discover page and on that page provide “logstash\*” an an index pattern.



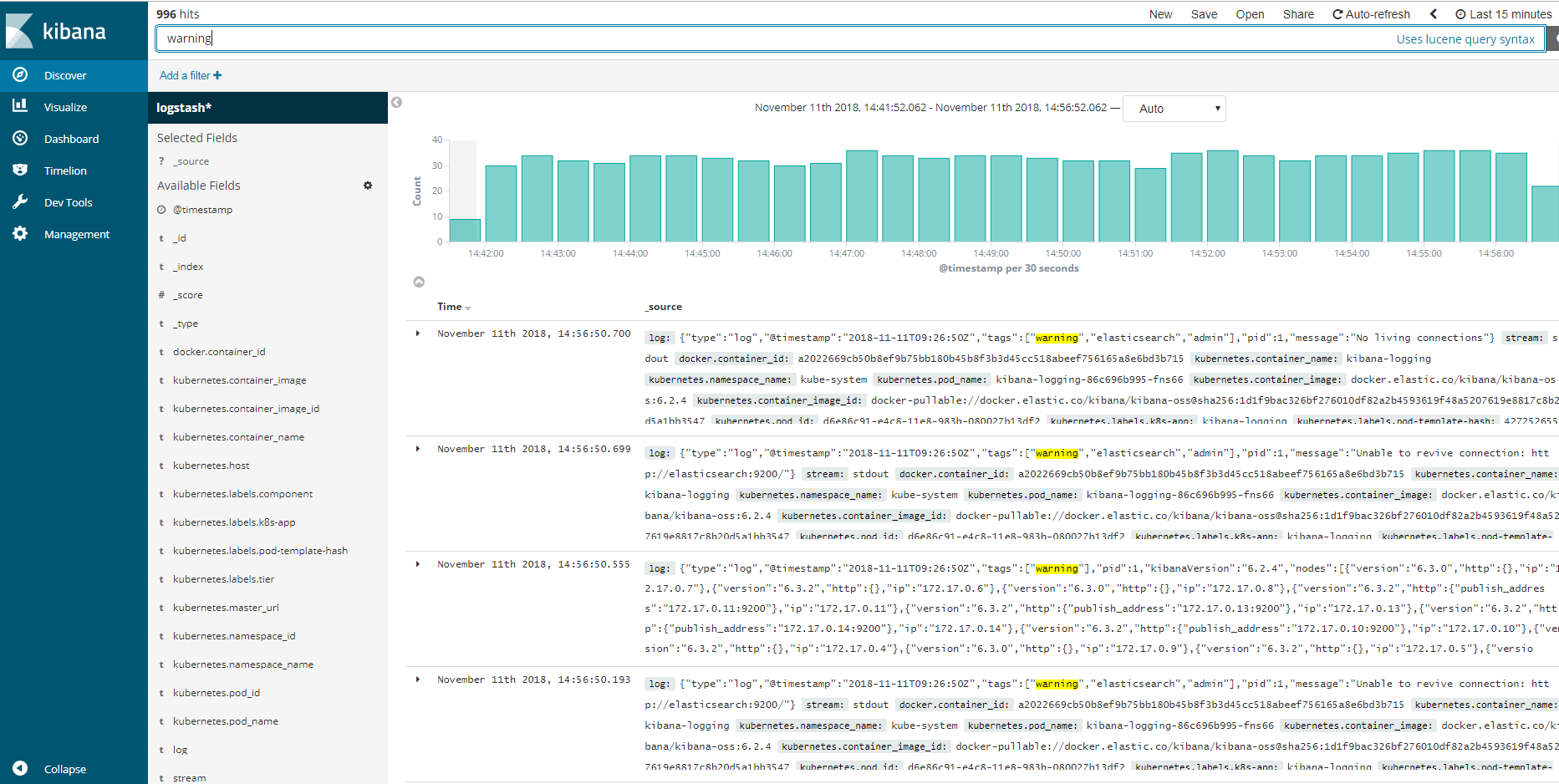
Click on next step to configure time filter field name. FluentD add @timestamp column to each of the log. Then create index pattern. Then you will stumble upon a very comprehensive index pattern, a pattern with 56 fields. During my initial days of K8s, this really did not intimidate me as I have used patterns with thousands fields in it. Its comprehensive as it covers all details of K8s cluster, right from docker container level attributes to K8s pod attributes and so on.



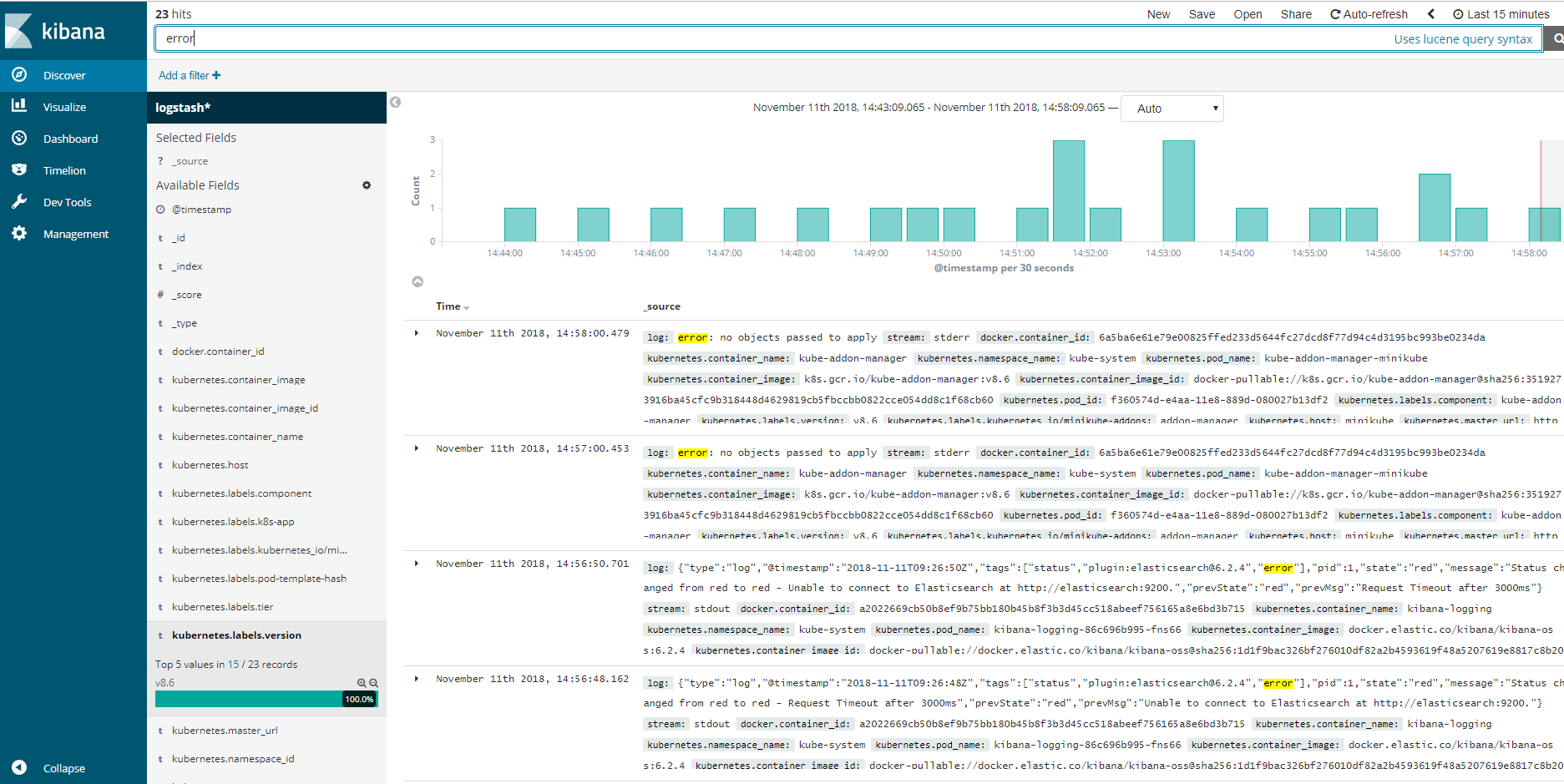
Then click on discover and then you will see the logs data.



Now this can be intimidating if you are new to ElasticSearch. Because this information is overwhelming. You may filter logs by making use of Elastic Search functionality. Just filter for critical warnings as follows-



Otherwise, you may want to filter for errors as follows-

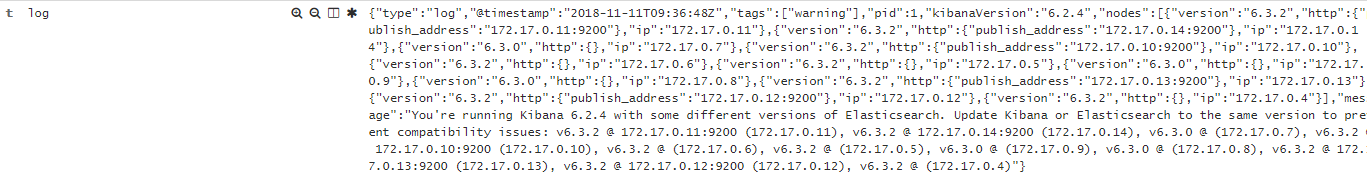


One of the important thing to do is to setup the refresh interval. Then expand any of the row and then you will notice table attributes and json.

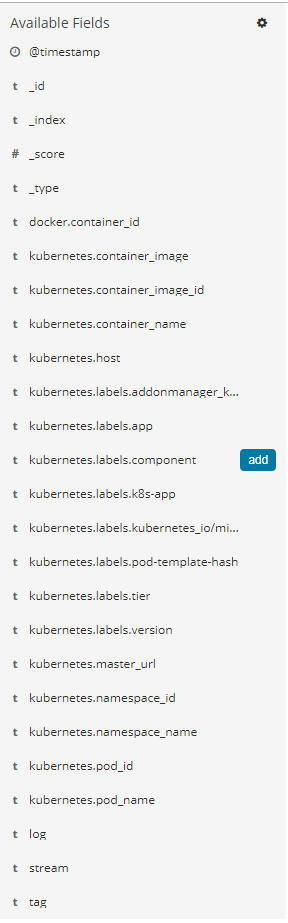




One of the important attribute is “log” and its listed in the Json and Table.



You can add columns by adding fields from the left hand side menu “Available Fields”.



This is how you can configure EFK to analyze logs of application deployed in K8s and logs of K8s cluster – different components of K8s ecosystem. That is it from side. You can explore further to find more details about using Elastic Search and Kibana.