**Beats:**

Let’s Suppose there is an E-commerce application which has different types of application like mobile, web-app, Desktop on different servers. I order to accumulate all these logs to Elastic search cluster, we need log stash.

We can configure Logstash on each of these edge servers, but it is bad architecture because log stash is research intensive application that comes various plugins. having these heavy weight log stash instances on the application servers might reduce the performance of the server.

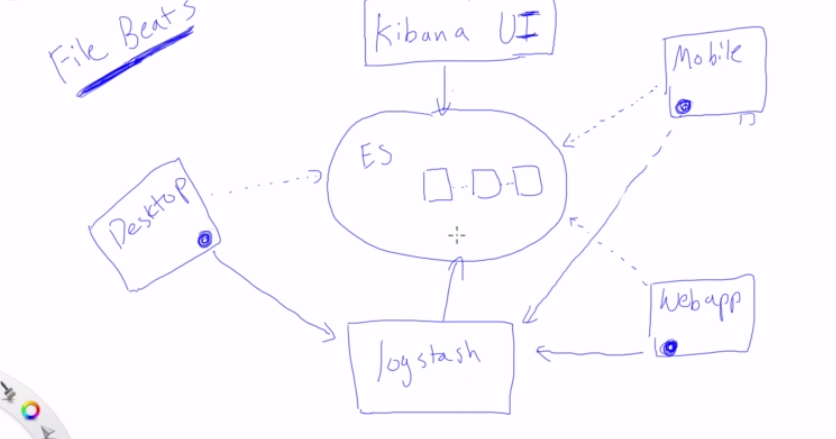
Log stash should be installed on a separate node and all we need is a light weight process that can ship these logs from respective servers to logstash so, that log stash can parse those logs and index them in to elastic search. We can use like scp or rsync but it has many disadvantages.

For this propose Elastic search introduced a application called beats.

Beats are light weight shippers that can be easily deployed on the edge servers. Here beats can tail the logs and send them as events to logstash and logstash consumes them.

File Beats: special kind of beats used to ship logs from edge servers.

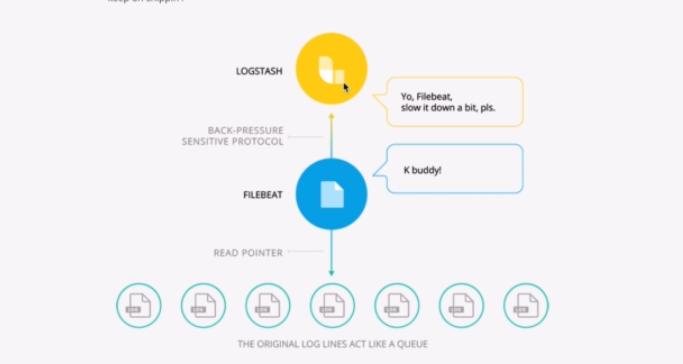
We can also use Kafka, mognoDB, MySQL to ship the logs.



Metric beats: shipper that collects metrics from operating system.

Packet beat: used for networking.

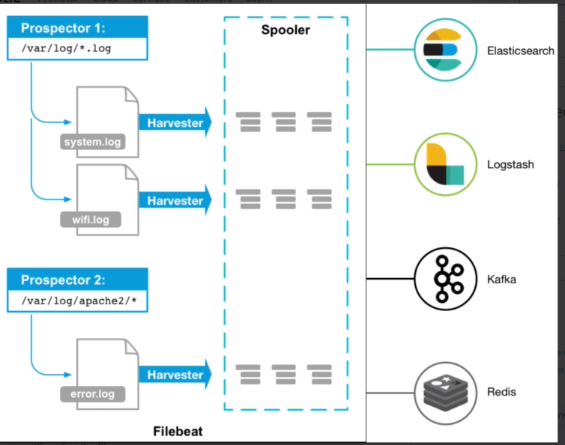
Heart beat: it tells whether the application is on or off.



File beat has a read pointer which reads the files and send them as events to logstash.

It has “Back-pressure sensitive protocol” which controls the rate of files transfer based on the logstash consuming rate.

We can also use File beats to directly index logs to elastic search, but log stash provides many features for doing that.



**Prospector**: It is responsible for identifying the list of files to read logs. You can specify the directory and kind of files you want, and prospector is configured to do that.

**Harvester**: It is responsible for reading the content of files. If the process starts off and starts again, Harvester knows where it left off in the file. So, it continues to read from next available line. **One harvester is started for each file. It is also responsible for opening and closing that file.**

**Spooler**: Here all the lines are aggregated together and sent over configured destinations.