LOG Long-term Backlog

Under consideration, but not currently planned for inclusion in a release.

Persistent Log Volumes

Logs are written to files and shipped by Filebeat. There are two main reasons for this:

1. Files provide resilience. If indexing stops for any reason (including if Logstash or Elasticsearch aren't running) then logs will be spooled.
2. Logfiles exist for purposes other than indexing. People examine them, read them, scrape them.

However logfiles are written to a Kubernetes emptyDir{} volume in most cases. This means:

1. The size of the volume is unspecified. No minimum is guaranteed. This can result in the loss of logs, and potentially other errors, including in other components that happen to be using the same disk.
2. The contents of the volume are lost on container restart, which may also result in the loss of logs.

There was some concern that Kubernetes volumes are unsuited to capturing logs, presumably due to IO demands. This may be true of certain storage types, but it's not true of volumes in general.

All logfiles should be captured and safeguarded on persistent volumes.

Filebeat Daemonsets

Filebeat is injected into every pod. (In practice not all, since pods are continually being added, and that is also a problem). Logs written to persistent volumes (and as noted all logs ought to be) are amenable to shipping by Filebeat containers running as Kubernetes daemonsets.

This would allow us to configure a scalable set of Filebeat shippers, without modifying each and every pod.

~~LOG-169~~ - Document alternatives to the filebeat agent as etherial PV sidecar container for the ELK stack CLOSED

Streams Transports

Notwithstanding that we're expected to retain logfiles for various reasons, Filebeat is one of many options for shipping logs to Logstash.

Filebeat is complex. In its current configuration, it requires the Filebeat container in each pod. So far there have been no complaints, but it's invasive.

We've tested syslog and TCP transports, and they work very well.

Issues:

1. Native Logstash TCP transports require Logstash classes in the classpath, and this is a non-trivial rollout. Syslog should not, but there has been no testing for older providers, and it made sense to improve standardization first.
2. Streams require persistence, so that log entries aren't lost, even if Logstash is down or deconfigured. There are many options, but all are somewhat complicated.
3. Filebeat worked everywhere, and needed to happen first.

Streams transports would likely be an optional configuration, perhaps on a per-container basis.

Probes

This idea has been discussed a number of times, such as here. Most commonly it has been with reference to Zipkin. HuabingZhao recently raised OpenTracing.

~~LOG-104~~ - Investigate Jaeger / opentracing / zipkin distributed tracing agent/server CLOSED

There is no doubt that this is an idea with enormous potential. What's held us back so far is:

1. The practicality of rolling out probes to every ONAP component. ONAP is not heterogeneous. There are a great number of components. There are many owners. There are many architectures. There is considerable diversity. Achieving consensus and executing the rollout are both formidable challenges.
2. The approach taken by the existing (and pre-existing) guidelines is somewhat orthogonal, and still a work-in-progess. It is based on explicit logging of MDCs and other attributes, each with clearly defined semantics.
3. Competition for resources, and more immediate (albeit less exciting) issues to deal with during Amsterdam and Beijing.
4. We haven't explored what data it will provide, nor proposed what we propose to do with it.

Of those, #1 is the most daunting, and #4 is where we'd need to begin.