ONAP Application Logging Specification - Post Dublin

Going with the casablanca spec for Dublin - for the library to implement this spec - retrofit portal/sdk (or SO's library) - need to have an AOP enabled library like our reference - 3rd choice is to AOP enable AAI's

Active Logging Specifications

deprecated - ~~*20181025: update: originally we decided to continue the casablanca version v111 of ONAP Application Logging Specification v1.2 (Casablanca) in Dublin but after constructive feedback/pushback from one small part of the AT&T based Acumos community and feedback/pushback from part of the AT&T based ECOMP community based on initial pulls into ECOMP and some feedback from 2 high level projects in ONAP - I recommend refocusing on only one requirement - a common ONAP first spec. Creating a 3 way spec that satisfies Acumos, ECOMP and ONAP can no longer be our focus due to the limited time and very limited resources of the logging spec/ELK deployment work a couple of us need to do. I have received tremendous support and spec/JIRA work from Lorraine Welch, Spondon Dey, Shishir Thakore and Luke Parker (all the marker/mdc work and slf4j library - and originating the project spec/elk stack). Creating a combined specification that meets the needs of private AT&T ECOMP that consumes Open Source ONAP, public Open Source Acumos that consumes parts of ONAP and even ONAP itself with it's several Logging implementations across Portal, SO, AAI is not really working or will take too long to accept a new library that everyone can accept. Since I am the current PTL, I have decided that an "ONAP first" direction is required. Any implementation/spec changes will need to be prototyped in public first and accepted directly into affected projects as we change the spec - not as an exercise after we have finalized the spec. I recommend that the way forward as suggested by the core of my coworkers in Acumos is to take an existing library and slowly iterate that library to match an "ONAP" specific spec that includes markers and MDC's and only if possible keep the older formats by the ONAP consuming projects. I will start from the bottom up for Dublin - using the portal/sdk library and adjust it in sync with any spec changes in parallel (no spec changes without first prototyping them in existing use cases by users of the portal/sdk library). For the E release we can increase adoption to the rest of the components not currently using portal/sdk. What is critical this time around is working with the team(s) affected by any adjustment to their library first. Secondary consideration will be given to non-opensource consumers of this specification inside corporations that privately wrap ONAP. This will mean becoming part of the public portal and policy weekly meetings as a de-facto sub-project. The first order of work will be infrastructure (shipping logs to ELK) and then an alignment to the existing portal /sdk logback.xml spec - before we start changing things. so we can start mining the ELK logs before we adjust them.*~~

Statement of Work

Detail any changes to the existing casablanca spec (implementing in Dublin) - Active Logging Specifications - for the upcoming E release.

Each specification is created in the release prior to where it is implemented

Work to align the logs, ship them, consolidate, provide for analysis, backup and restore them will be done from a developer perspective for the Dublin release.

see Logging Dublin Scope

Features

Key words are: out of the box, reuse components, iterative changes - not major spec changes.

There is one primary use case and one enabler use case

UC1: Provide for transaction tracing via elasticsearch or dashboard for distributed transactions across ONAP components - enabled by the logs shipped to the ELK stack - in real time.
(the ELK stack is there - we just need to adjust log content - amount and format)

UC2: Provide for UC1 tracing via standardized logs - ideally via marker/mdc (label/key:value pair) markup

Logging Alignment before enhancement

Plan is to take the portal/sdk library - align the rest of onap to this library and then add marker/mdc (labels and key/value pair) support as required in phase 2

Work will include 1) verify pipeline 2) verify logback format

Open Standards

Look at a standard that will allow us to use standard parsing libraries all the way to eventual AI/ML by adopting a standard like open tracing.

Infrastructure Improvements

including filebeat sidecar anti-pattern/replacement, elk stack upgrade, deployment template, library language scope (python, swift...)

Monitoring

Investigate out of the box prometheus monitoring

S3P

backup/restore, security, log format monitoring, elk dashboards

3rd party consumption alignment

Lower in priority will be to align onap with requirements of Acumos and ECOMP for example - the focus will be on alignment of ONAP components only - to the primary goal of providing a transaction tracing system via the ELK stack.