Introduction

Functional requirements

The following requirements should be fulfilled:

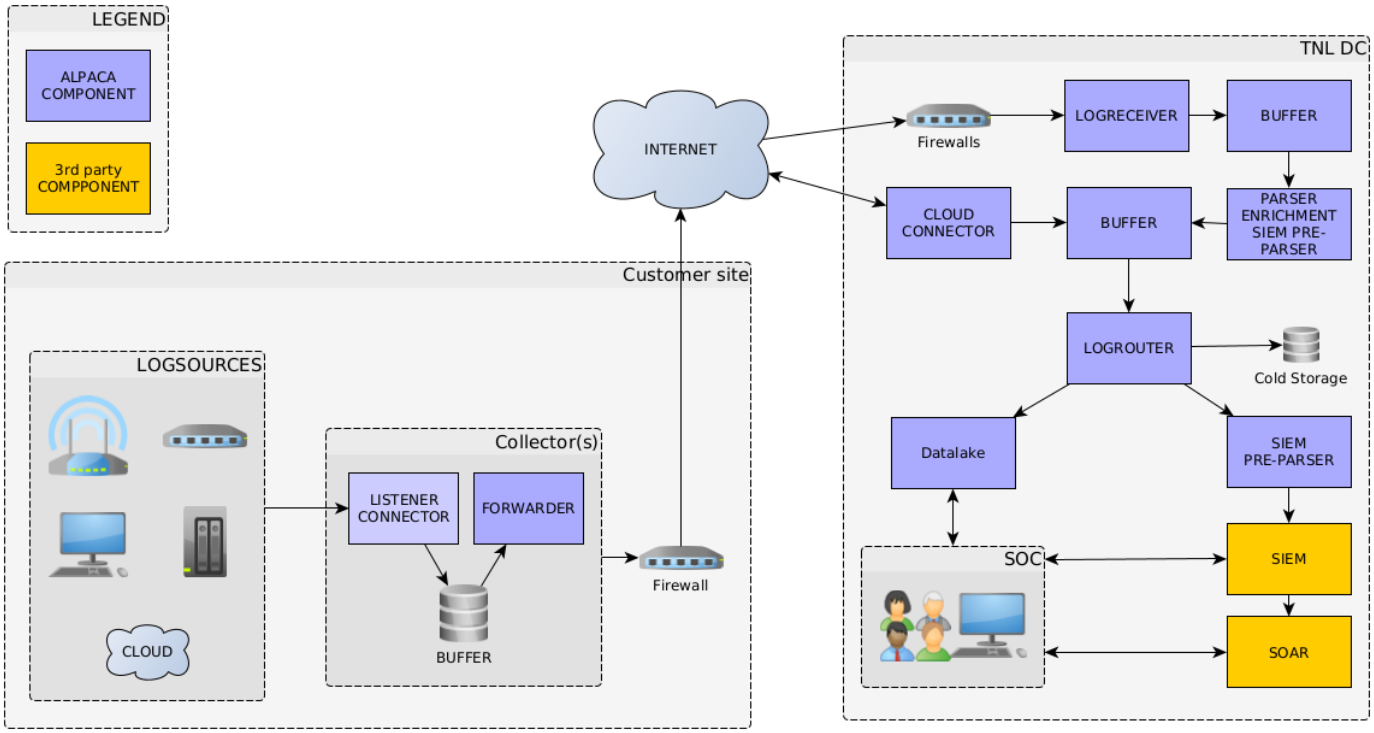
- Fully featured logmanagement-system
  - receive logs in ANY logformat
  - Converting ogformats to support SIEM requirements
  - filtering unwanted logs
  - Protecting pipelines from overflooding
- Should fully integrate with Qradar and other SIEMs
- The following functionality should be available
  - buffering (in case of congestion/network outage/component failures)
  - filtering (should be possible anywhere in the pipeline)
  - logs should be searchable in a database-like datalake
  - logs should be stored to cold storage
  - encryption of data in transit AND data at rest should be supported
  - high availability, system should be able to fully recover from any type of intermittent failure
  - Redundancy: components should be replacable without service-degradation.
  - Solution should be platform-independent (OS/Hardware agnostic)
  - Components must be supported on latest OS/patchlsevels.
  - Components should be in active development/support.
  - platform should support log-transformation to meet Qradar log-standards
  - Each part of the data-pipeline should be auditable/monitorable.
  - Multi tenancy
  - Proven technology

Design Principles

To limit the possibilities we also decided on some principles the solution should follow:

- No java unless thoroughly tested
- No docker/containers
- No fancy-schmancy python-code.
- Run on Linux
- both X86/ARM support for key components
- Deployable using industry standard deployers (ansible/chef/puppet)

Basic pipeline architecture



During each (critical) step data will be written to storage (which will be HA/redundant) to ensure no data will be lost when a critical failure occurs.
The amount of data in memory will be limited as much as possible

Base components

During extensive research and experiences from the past, the following software-stacks have been selected as the preferred components to build the new solution.

- Vector (https://vector.dev) As the core log-management core.
- Kafka as an high-speed buffering solution
- OpenSearch for core data-lake, metrics and dashboarding and datalake alternative
- Ansible (deployer for configuration and setup)

Other packages will be selected depending on need or to handle specific use-cases.