Evaluation and Implementation of Various Persistent Storage Options for CMSWEB Services in Kubernetes Infrastructure at CERN

Muhammad Imran[1,2], Valentin Kuznetsov[3], Panos Paparrigopoulos[1], Spyridon Trigazis[1] and Andreas Pfeiffer[1]

[1] CERN, Geneva, Switzerland
[2] National Centre for Physics, Islamabad, Pakistan
[3] Cornell University, New York, USA

E-mail:
`muhammad.imran@cern.ch`,`vkuznet@protonmail.com`,`panos.paparrigopoulos@cern.ch`,
`spyridon.trigazis@cern.ch`,`andreas.pfeiffer@cern.ch`

Abstract. This paper summarizes the various storage options that we implemented for the CMSWEB cluster in Kubernetes infrastructure. All CMSWEB services require storage for logs, while some services also require storage for data. We also provide a feasibility analysis of various storage options and describe the pros/cons of each technique from the perspective of the CMSWEB cluster and its users. In the end, we also propose recommendations according to the service needs. The first option is the CephFS which can be mounted multiple times across various clusters and VMs and works very well with k8s. We use it both for data and the logs. The second option is the Cinder volume. It is the block storage that runs the filesystem on top of it. It can only be attached to one instance at a time. We use this option only for the data. The third option is S3 storage. It is object storage that offers a scalable storage service that can be used by applications compatible with the Amazon S3 protocol. It is used for the logs. For S3, we explored two mechanisms. For the first scenario, we consider fluentd that runs as a sidecar container in the service pods and sends logs to S3 bucket. For the second scenario, we considered filebeat that runs as a sidecar container in the service pod and scaps those logs to fluentd which runs as a daemonset in each node and sends those logs to S3 in the end. The fourth option is EOS. We configured EOS inside the pods of the CMSWEB services. The fifth option that we explored is to use dedicated VMs that have Ceph volume attached to them. In EOS and VM, the logs from the service pods are sent to EOS/VM using the rsync approach. The last option is to send service logs to Elasticsearch. It has been implemented using fluentd that runs as a daemonset in each node. In parallel to the sending logs to S3 fluentd also sends those logs to the Elasticsearch infrastructure at CERN.

1. Introduction

The Compact Muon Solenoid (CMS) is a general-purpose detector at the Large Hadron Collider (LHC) at CERN, Geneva, Switzerland [1]. The CMS experiment runs hundreds of thousands of jobs daily on its distributed computing system to simulate, reconstruct and analyse the data taken during collision runs. A dedicated cluster ("CMSWEB") is used to host essential CMS central services which are responsible for the CMS data management, data discovery, and various data bookkeeping tasks. The cluster was based on virtual machines (VMs) on the CERN

OpenStack cloud infrastructure. Recently, we migrated CMSWEB VM cluster to Kubernetes (k8s) infrastructure to enhance the sustainability of CMSWEB services [2, 3].

This paper gives brief summary of the various storage options that we implemented and explored for the CMSWEB cluster in k8s infrastructure. All CMSWEB services require storage for logs, while some services also require storage for data. We provide a feasibility analysis of various storage options and describe the pros/cons of each technique from the perspective of the CMSWEB cluster and its users. As an outcome of this analysis, we also propose recommendations of the storage option according to the service needs.

The remainder of this paper is organized as follows. Section 2 describes various persistent storage options that we implemented for CMSWEB services in k8s infrastructure. Section 3 presents the feasibility analysis of these storage options. Finally, we conclude in Section 4.

2. Persistent Storage Options for CMSWEB Services

In this section, we briefly describe the various storage options that we evaluated for the CMSWEB cluster in k8s infrastructure. These storage options are described below:

2.1. EOS

EOS Open Storage (EOS) is a software solution that aims to provide fast and reliable multi-PB disk-only storage technology for both LHC and non-LHC use-cases at CERN [4]. In the VM cluster, we used EOS for archiving logs while in the k8s cluster, we tested it by configuring EOS inside the pods of the CMSWEB services. Logs from the pods are sent to EOS using the *rsync* approach. The rsync is a utility for efficiently transferring and synchronizing files across networked computers by comparing the modification times and sizes of files at predefined interval as shown in Figure 1. Logs between intervals are lost if the service container restarts.

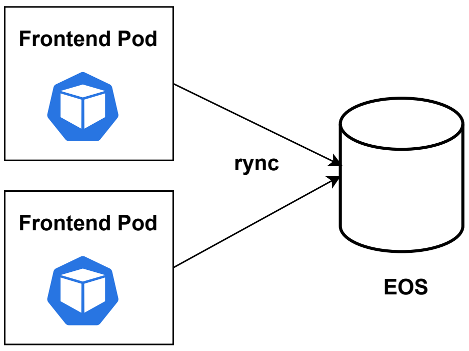

Figure 1. Logs from services pods are transferred to EOS using rsync approach at interval

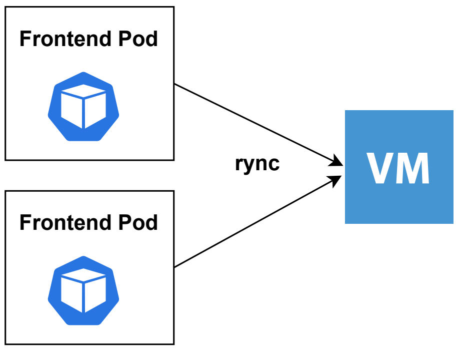

Figure 2. Logs from services pods are transferred to dedicated VM using rsync approach at interval

2.2. Dedicated VM

In the VM cluster, we used a dedicated VM to provide few days of logs to users while in the k8s cluster, we implemented the same technique using *rsync* approach to a dedicated VM. Logs from the pods are sent to the dedicated VM at predefined interval as shown in Figure 2. Similar to EOS, logs between intervals are lost if the service container restarts.

2.3. CephFS

The Ceph File System (CephFS), is a POSIX-compliant file system built on top of Ceph's distributed object store, RADOS. CephFS endeavors to provide a state-of-the-art, multi-use,

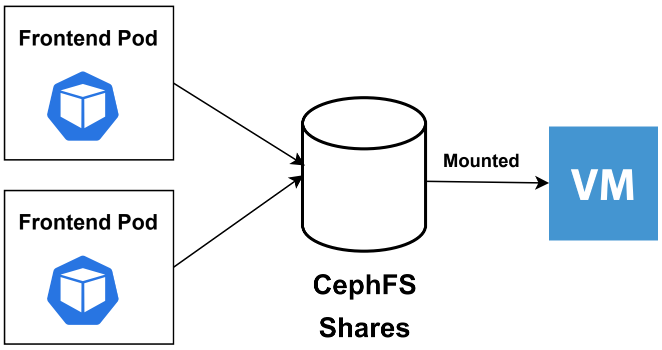

Figure 3. Data/logs from services pods are stored in CephFS shares which are mounted on VM and become accessible to users.

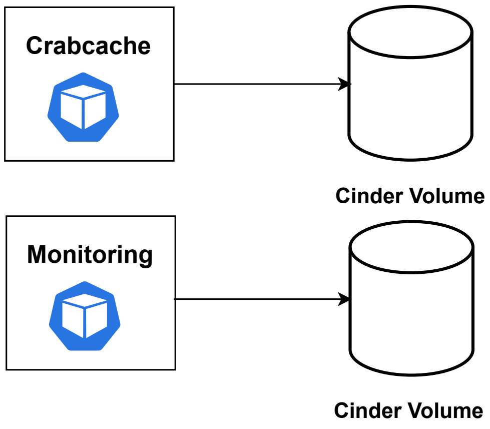

Figure 4. Cinder volumes are attached to service pods in one to one relationship and can only be accessed within a pod.

highly available, and performant file store for a variety of applications [5]. CephFS shares can be mounted multiple times across various clusters/VMs, and works very well with k8s. We can use it both for data and the logs. The data/logs are available to the users in real-time as shown in Figure 3 where Apache logs and Filebeat data are stored. However, the service availability depends upon CephFS shares. The other drawback is the slow grep operation for searching logs.

2.4. Cinder
Cinder is a Block Storage service for OpenStack that runs the filesystem on top of it. Cinder virtualizes the management of block storage devices and provides end users with a self service API to request and consume those resources without requiring any knowledge of where their storage is actually deployed or on what type of device [6]. It can only be attached to one instance at a time as shown in Figure 4 but it can be detached/reattached. We use this option where a large amount of storage for the data is needed for singleton services. We cannot mount it to any VM and it is accessible via services pod only.

2.5. S3 using Fluentd
S3 is an object storage that offers a scalable storage service that can be used by applications compatible with the Amazon S3 protocol [7]. We tested it with the combination of fluentd which

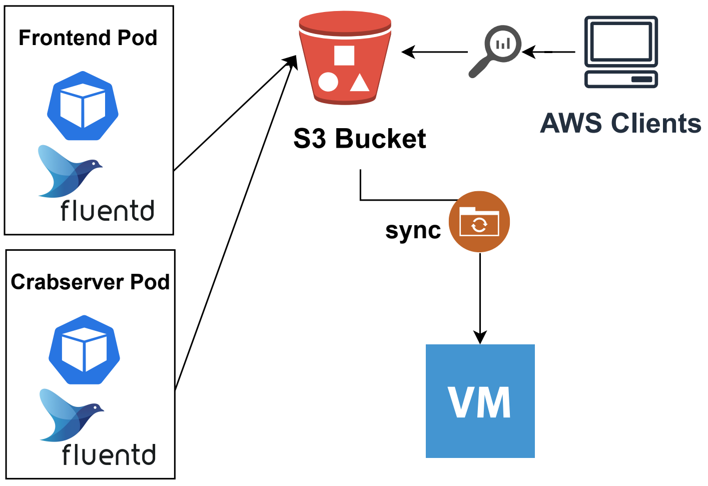

Figure 5. Services pods use fluentd to scrap logs to S3 bucket from where these become available via AWS clients or synced to VM

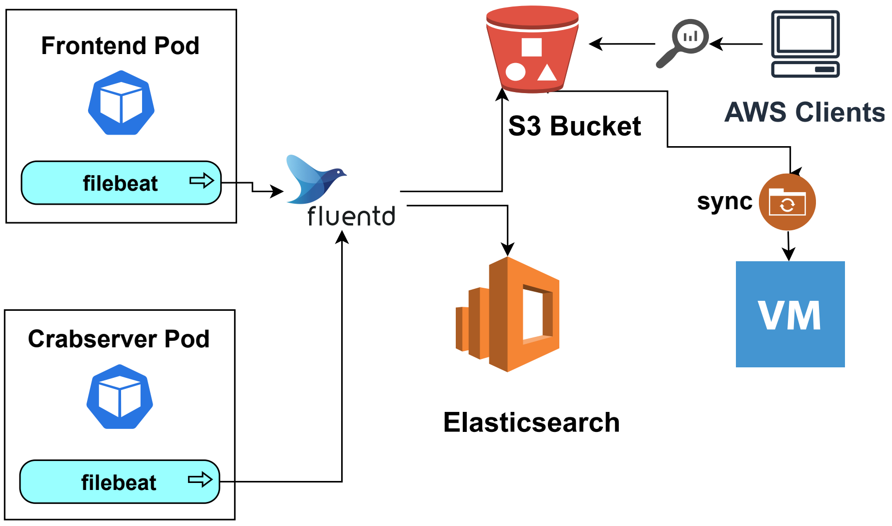

Figure 6. Services pods use filebeat to scrap logs to fluentd which aggregates and sends them to S3 bucket and elastic search in parallel fashion. Logs are accessed using AWS clients or are synced to a VM.

we run as a sidecar container as shown in Figure 5. The fluentd is an open source data collector for unified logging layer [8]. The fluentd scraps logs from container and sends them to the S3 bucket at predefined interval. The logs from S3 bucket are then copied to the dedicated VM using sync or these can be fetched via AWS clients as shown in Figure 5. We tested this scheme for the logs. This scheme has the drawback of data lost for the interval timespan if the service container restarts.

2.6. S3 and Elasticsearch using Filebeat and Fluentd

The mechanism of this scheme is shown in Figure 6. Here services pod run filebeat as a sidecar container. The filebeat is a lightweight shipper for logs [9]. The filebeat scraps logs to fluentd in real time. The fluentd daemon sends logs in parallel to S3 bucket and elasticsearch at predefined interval. The logs are then synced to VM from S3 bucket using AWS clients. Unlike the previous S3 using fluentd technique, the logs are not lost with a container restart. Users can access logs from Elasticsearch as well using a GUI. However, the main shortcoming of this strategy is an extra metadata in the log entry, i.e. a podname and complex workflow which requires proper setup and maintenance

3. Feasibility Analysis

In this section, we discuss about the feasibility analysis of various techniques that we explored/implemented for CMSWEB services. This analysis is summarized in Table 1. The logs are accessible in real time in case of CephFS and Cinder storage while for all other schemes these are available after interval. We can use Cinder only for data as it can not be mounted on any other VM while we can use CephFS for both data/logs as it can be mounted on separate VMs from where users can access data/logs. All other schemes can be used only for logs. Searching is slow in CephFS while in case of Cinder we can get access to data within pod only. For all other cases, searching is efficient. Data can be lost in the scheme where container restarts for some reason within predefined intervals. The services have dependencies for CephFS and Cinder storages i.e., in case of any issues with the storage, service become unavailable.

From above analysis, we recommend the use of *S3 and Elasticsearch using Filebeat and Fluentd* for maintaining logs, *Cinder* for large data volumes e.g. DB back-end, and *CephFS* for service data storage e.g. cache.

Table 1. Feasibility analysis of various techniques with respect to different criteria

Storage Options	**Accessibility**	**Usage**	**Searching**	**Data Lost**	**Service Dependency**
EOS	At interval	Logs	Easy	Yes	No
Dedicated VM	At interval	Logs	Easy	Yes	No
CephFS	Real time	Data/Logs	Slow grep	No	Yes
Cinder	Real time	Data	Only in pod	No	Yes
S3 using Fluentd	At interval	Logs	Easy	Yes	No
S3 and Elasticsearch using Filebeat and Fluentd	At interval	Logs	Easy	No	No

4. Conclusions

In this paper, we investigated the feasibility of various persistent storage options for logs and data for the CMSWEB k8s cluster. A number of different storage options such as file-system-like (EOS), block storage (Cinder), shares (CephFS) and object-store-like (S3) were evaluated and outline the benefits and drawbacks of each strategy from the CMSWEB cluster's and users' perspectives. From our analysis, we recommend the use of *S3 and Elasticsearch using Filebeat and Fluentd* for maintaining logs, *Cinder* for large data volumes e.g. DB backend, and *CephFS* for service data storage e.g. cache.

References

[1] CMS Collaboration 2008 *Jinst* **803** S08004
[2] Imran M and et al 2021 *PoS* **ICHEP2020** 911
[3] Imran M and et al 2021 *Cluster Computing* 1–15
[4] EOS Storage at CERN Website: `https://github.com/cern-eos/eos` last accessed: 29.11.2021
[5] Cephfs storage Website: `https://docs.ceph.com/docs/master/cephfs` last accessed: 04.14.2020
[6] Cinder storage in openstack Website: `https://wiki.openstack.org/wiki/Cinder` last accessed: 04.14.2020
[7] Mascetti L and et al 2020 *EPJ Web of Conferences* vol 245 (EDP Sciences) p 04038
[8] Fluentd Website: `https://www.fluentd.org/` last accessed: 29.11.2021
[9] Filebeat middleware Website: `https://www.elastic.co/beats/filebeat` last accessed: 29.11.2021