EPJ Web of Conferences **337**, 01052 (2025) https://doi.org/10.1051/epjconf/202533701052
CHEP 2024

The Canadian Belle II Tier1 Raw Data Centre

Marcus Ebert[1,*], *Colson* Driemel[1], *Randall* Sobie[2], and *Tristan* Sullivan[1]

[1]Department of Physics and Astronomy, University of Victoria, BC, Canada
[2]Institute of Particle Physics, University of Victoria, BC, Canada

Abstract. We present our innovative design to hosting Canada's share of the Belle II raw data and the necessary computing infrastructure for its processing. Our approach involves disk-only grid storage solution that utilizes XRootD and ZFS, complemented by a compute framework that deploys specialized virtual machines to process the raw data. We discuss the advantages of our approach as well as the challenges encountered during deployment and commissioning.

1 Introduction

The Belle II [1] experiment is an electron-positron collider experiment that takes place at the KEK Laboratory in Tsukuba, Japan [2]. Physics data taking began in 2021, and it is expected to continue for about 10 years. The Belle II member states form a collaboration with institutions around the world. To analyze the data of the experiment, collected data need to be processed to be usable for analyses. Belle II adopted the computing model of the World Wide LHC Computing Grid [3], with the data distributed worldwide.

In addition, a copy of the collected raw data is distributed between six sites, called Raw Data Centers (RDC), in five countries: USA, Italy, France, Germany, and Canada, as an additional backup in case of catastrophe at the main raw data storage facility at KEK. Other than storing the raw data, the RDCs are also required to have computing resources to process the raw data periodically.

The Canadian site is hosted at the University of Victoria (UVic) [4]. It will store and process 15% of the collected raw data. This paper describes the data flow in the next section, followed by a description of the UVic RDC, the challenges in commissioning and operating the center, and a brief summary.

2 Data Flow from KEK to Raw Data Centers

First, raw data is copied from KEK to the site's temporary storage area. In a second step, it is transferred from the temporary area to a dedicated raw data storage area on the same site. At most sites, this raw data storage area is associated with a tape system, and the data copied to it is eventually moved to tape. These tape systems are usually accessible through standard grid storage interfaces and protocols.

When the raw data needs to be processed, it is copied from the raw data area - typically staged in from tape - to the site's temporary storage area. From there, it becomes accessible

*e-mail: *mebert@uvic.ca*

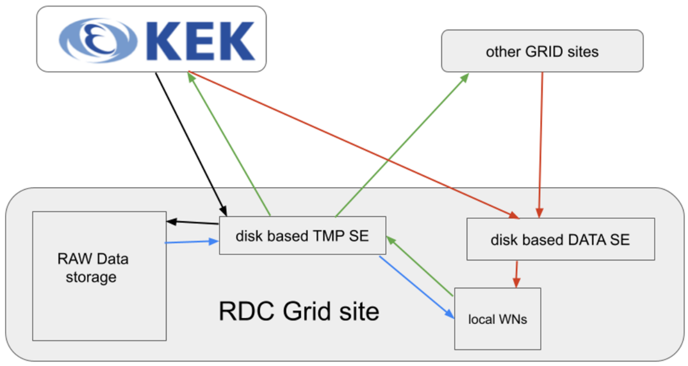

Figure 1. Raw data flow out of KEK (black), for data processing on-site (blue) and storing output off-site (green), and for making processed data available (red). Data is copied to a temporary (TMP) storage area of an RDC site first, and from there to the permanent raw data area. To process the data it gets first copied to the temporary storage area again and from there worker nodes (WN) can access it. The processed data is then copied from the WN to the temporary area and from there to KEK or to other grid sites. Processed data gets then copied to the DATA area of the site to be used by user analysis jobs running on the WNs.

to jobs running on local worker nodes, either by copying the file from the temporary area to the worker node or streaming the file directly to the analysis process on the worker node. In a last step, the processed data is then copied from the worker node to the temporary storage area and from there back to KEK or to other grid sites. To make the processed data available for user analyses, it is copied from KEK, or from other grid sites, to the data area of the SE from where it can be accessed by the worker nodes.

The data flow is shown in Figure 1. A more detailed overview of the Belle II raw data management system can be found in [5].

3 UVic Raw Data Center (RDC) site infrastructure

3.1 General Overview

At the UVic RDC site, there is no grid accessible tape system as well as very limited manpower (less than 1 FTE) available. On the other hand, we have access to systems managed by the University or the Digital Research Alliance of Canada. We will describe how an RDC site can be built and managed under such conditions. A comparison between the UVic RDC site and a typical RDC site configuration is shown in Figure 2.

We use a disk-based system for long term storage of the raw data. Disk based storage systems are often considered an unreliable long-term storage medium due to the uncertain lifetime of individual disks. We will show how we assure the data is secure. The raw data storage area (DATA SE) and the temporary storage area (TMP SE) share a common infrastructure, and both are accessible through the same endpoint. To manage this storage endpoint, we use a standard XRootD [6] installation. Currently, we use six data servers each with twenty-eight 20 TB disks for data storage. To extend the storage in the future, we plan to add external disk enclosures to the existing machines.

To ensure that data on disk is safely stored, we use ZFS for the data disks rather than hardware RAID systems. For disaster recovery, the data is also backed up to tape.

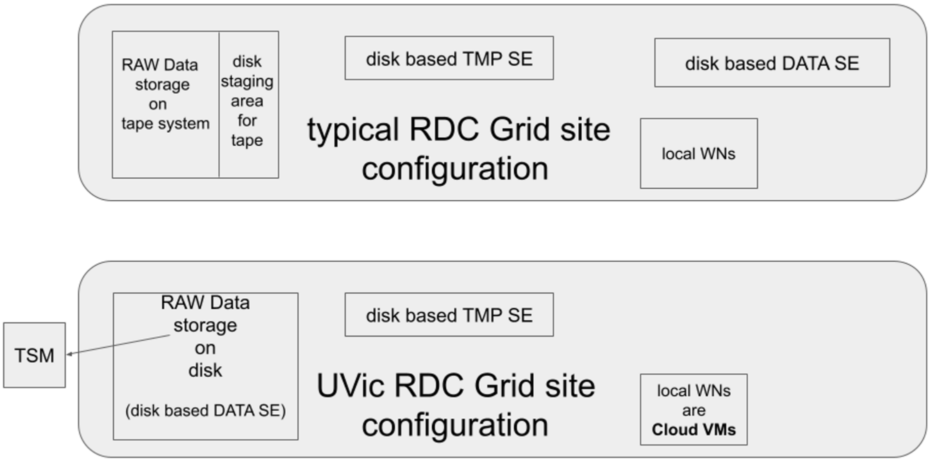

Figure 2. Comparison of a typical RDC site configuration (top) with the UVic RDC site (bottom). The disk based TMP and DATA SEs are usually accessible through the same endpoint. The UVic RDC site uses the disk based DATA SE to keep all raw data on disk. Raw data within the DATA area is backed up on tape (TSM) for disaster recovery. Data processing is done on cloud VMs.

For the processing of the data, we use virtual machines (VM) as described in subsection 3.5. These VMs are started and removed on demand, depending on the number of jobs in the batch system. We use HTCondor as our batch system.

3.2 Data Access

Belle II uses the *davs://* and *root://* protocols to access the data. At UVic, we use XRootD for data access, which supports both protocols natively. XRootD offers the advantage of a very simple configuration and does not rely on additional infrastructure, such as a database.

Our setup consists of one redirector and multiple servers. The redirector is the system's access point and directs all requests to a server appropriate for a specific request. Since data transfers are handled by the servers and do not go through the redirector, the redirector is a very lightweight system without storage requirements. We deploy the redirector system in a VM.

The redirector and the servers share a common configuration file, which simplifies the setup. Files are stored in full on individual servers using their full logical filename. For example, if a file is received with the URL

```
davs://<REDIRECTOR>/TMP/belle/test/testfile.root
```

then it is stored on one of the servers on the data disk under

```
TMP/belle/test/testfile.root
```

The logical filename is preserved in the physical filename and no database is needed. In addition, any Linux tool can be used to copy files between storage locations on the same server or between servers. This makes future migration to other hardware trivial.

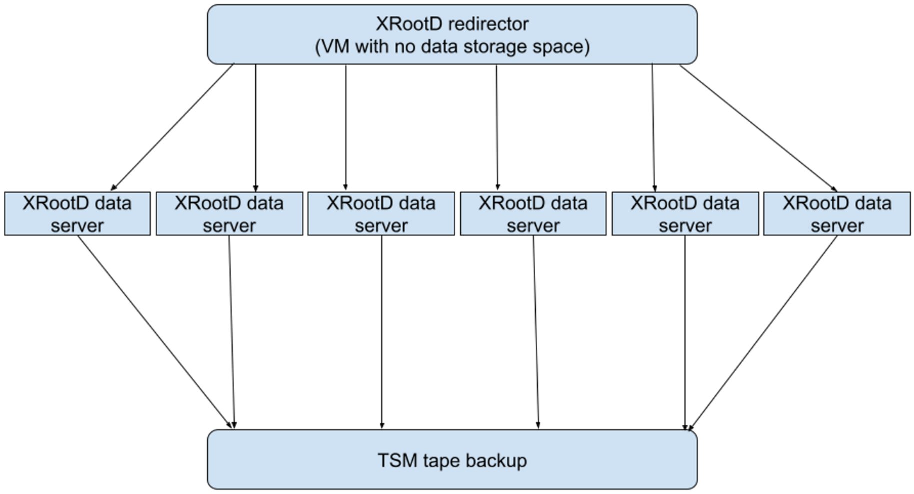

Figure 3. Overview of the XRootD system with connection to TSM. The XRootD redirector is the access point for data requests and redirects those to the XRootD data servers. Raw data areas on the XRootD data servers are backed up to tape using TSM.

3.3 Data safety

Although we are not required to store the data on tape, we make a second copy on the local tape system at UVic. Since all files are stored using the experiment's full logical filename, we use the IBM Storage Protect system (TSM) [7], which preserves the directory structure. The TSM system is managed by University Services at UVic.

Because the full logical filename is used to store a file on tape, files can also be restored to different servers and become instantly available via XRootD once on disk. This simplifies file retrieval in the event of a complete machine failure. Additionally, if a file becomes inaccessible on disk, XRootD can execute scripts to automatically retrieve files from tape. An overview of the system is shown in Figure 3.

To ensure data safety on disk, we use ZFS with raidz3 on servers that currently have twenty-eight data disks. ZFS ensures that a checksum is calculated for each block when a file is stored on disk. During data access, checksums are recalculated and compared with the original ones to verify that no data has been altered since the file was stored. If the checksum of a block differs or if a disk is inaccessible, the data is restored from redundancy blocks. Raidz3 calculates three redundancy blocks to ensure that the data remains save and correctly accessible even if the equivalent of three disks fail simultaneously.

3.4 Data access performance

All our servers have network interfaces that allow transfer rates of up to 25 Gbps. However, in tests we only achieved transfer rates of up to 10 Gbps. We found that the transfer rates are limited by disk access. While we could use a different disk configuration to speed up data access on disks, we prefer data safety over speed by using raidz3. Also, the current transfer rate is fast enough for any transfers between our site and KEK.

By not using the full network bandwidth for data access to the internal disks, we have the possibility of increasing the storage later on by adding external disk enclosures. The external disks can be configured as an independent ZFS and integrated into the current XRootD configuration. We expect that access to the data area on the external disk enclosure would also yield a transfer rate of about 10 Gbps, resulting in a total data transfer rate from a server that is close to the maximum network transfer rate of the interfaces.

3.5 Data processing

Belle II uses the DIRAC [8] workload management system. Jobs are submitted via DIRAC site-director systems. At UVic, we have run a local site-director for many years. The UVic site-director queries the central DIRAC system at KEK for jobs and then submits them to a local HTCondor [9] batch system. For the RDC, we use the same site-director and batch system.

All our HTCondor worker nodes for Belle II are VMs on OpenStack systems that are started on demand. To be able to start VMs on demand, we use Cloudscheduler [10][11], as shown in Figure 4. Cloudscheduler's HTCondor poller monitors the batch system and periodically reports its status to the main Cloudscheduler instance. The OpenStack installations are managed by the Digital Research Alliance of Canada [12]. We have successfully used Cloudscheduler for ATLAS, Belle II, and DUNE for more than 10 years. We also run Cloudscheduler as a service for other ATLAS and Belle II sites, as well as for *BABAR* [13].

Using clouds allows the system to be independent of physical hardware. In the event of outages, worker node VMs can be temporarily started on any other cloud. All VMs are started from a single image, which Cloudscheduler makes available to all clouds the project can access. Using a single image file also makes updating the system trivial since only one file needs to be changed. The updated image is then automatically distributed to the clouds, and any new VM will be based on it. In addition, using clouds allows us to start VMs with different resource flavors that allocate specific resources to each VM. Cloudscheduler can automatically choose the minimal flavor required based on the needs of a job in the queue.

3.6 Infrastructure services

At UVic, we operate a raw data system which requires we also run infrastructure services including Cobbler, Kickstart, perfsonar, and an NFS server. The perfsonar system runs on a dedicated machine while the other services are deployed as VMs. To run these VMs, we utilize two identically configured hypervisors. One of the hypervisors runs all the infrastructure VMs as well as the XRootD redirector VM. The filesystem of this active hypervisor is mirrored via ZFS snapshots to the second hypervisor. The second hypervisor is used for redundancy and can start the same VMs if the first hypervisor fails. All VMs run via libvirt and QEMU [14].

We use Ganglia [15] and Nagios [16] for general server and service monitoring. In addition, we have developed scripts and web pages to monitor the consistency between files on tape and disk, as well as the tape backup process.

4 Challenges during deployment

During deployment, we came across two issues with XRootD: one involving third party copy (TPC) transfers and another one involving directory listings using *davs://*. We will discuss both issues.

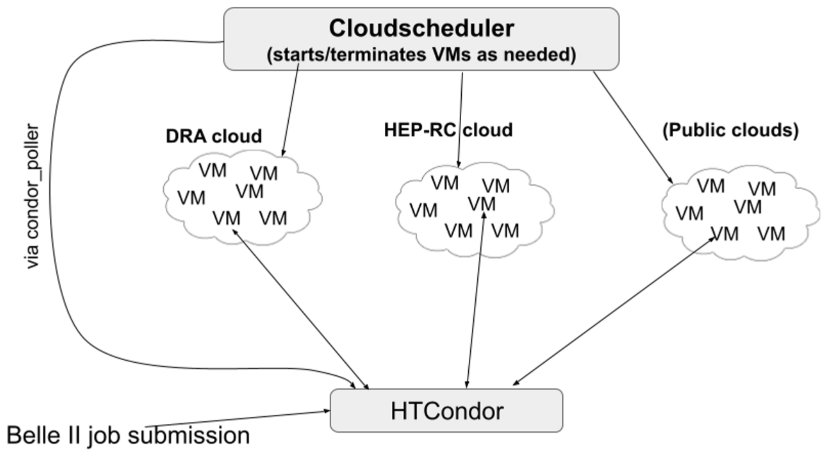

Figure 4. Overview of the on-demand usage of cloud resources. Belle II jobs get submitted to HTCondor which is monitored by Cloudscheduler. Depending on the number of jobs and the job requirements, VMs are booted or shutdown on demand on clouds of the Digital Research Alliance (DRA) of Canada. In the event of an outage, other clouds can be used to start those VMs, including public clouds.

4.1 Issues with TPC transfers

For a TPC transfer, the following steps are usually invoked by a client:

1. Ask if path exists on site, if not then create it
2. If file does not exists yet then a request is sent to the active site in the TPC transfer to start the copy process

In the first step, the path may exists on one of the connected XRootD data servers, or it will be created on one. At the end of this step, the redirector informs the client that the path exists. When the copy request is received in the second step, the redirector directs the transfer to a server that can store the file. However, this server may not be the one where the directory was created in the first step. In that case, it is expected that the directory is automatically created on the server that needs to store the file before the copy is initiated.

Instead, when XRootD is the active party in a *davs://*-based TPC transfer and the file path does not exist on the server responsible for storing the file, the transfer failed. Once this issue was brought to the attention of the XRootD developers, it was fixed very quickly. This issue is resolved in XRootD version 5.7 and above.

4.2 Issues with directory listing

This issue also affects only requests using the *davs://* protocol. When a "directory listing request" comes in, first it reaches the redirector. The redirector then sends a request to all servers to determine which server has the directory on disk. Once a server confirms that the directory is on disk, the redirector directs the client's request to this server. When the server responds to the request, the directory content is sent directly to the client rather than to the redirector. Therefore, the client receives the listing from only a single server, even if the directory exists with files in it on multiple servers. The client may also receive a different listing each time, depending on which server responds first to the redirector's request. This is not a bug but the expected behavior of responding to *davs://*-based listing requests.

A workaround is available, involving redirecting the listing request to a proxy server, which then queries the redirector using the *root://* protocol and returns the full list of files to the client via *davs://*. We have not implemented this workaround so far because listing directory contents is not part of the normal workflow for Belle II's TPC transfers and is of lower priority. Listing of a specific file is not affected by this issue.

5 Summary

In conclusion, we found that building and operating an on-disk only RDC can be done in a safe and secure manner. We ensure that the data on disk is secure and an off-site backup exists for disaster recovery.

We found that XRootD, with its native support for *davs://* and *root://*, is an efficient WLCG storage element. Combined with on-demand HTCondor cloud worker nodes managed via Cloudscheduler, it provides a full grid site solution. The use of ZFS checksums for each on-disk block makes ZFS the preferable choice over hardware RAID solutions [17]. In addition, sending ZFS snapshots between machines offers a simple way to mirror infrastructure services and maintain redundant hardware systems in case of failures or outages.

XRootD's use of the experiment's full logical filename for on-disk storage, combined with TSM for disaster recovery, enables files to be restored on demand to any server through XRootD's internal staging mechanism. We also found it is easy to contact XRootD developers and the community when issues arose. Thanks to the developers' responsiveness, issues that came up were resolved in a very short time.

References

[1] The Belle II Experiment, accessed: Jan. 2025, `https://belle2.org`

[2] KEK, High Energy Accelerator Research Organisation, accessed: Jan. 2025, `https://www.kek.jp/en/`

[3] J.F. Molina, A. Forti, M. Girone, A. Sciaba, Operating the Worldwide LHC Computing Grid: current and future challenges, Journal of Physics: Conference Series **513**, 062044 (2014). 10.1088/1742-6596/513/6/062044

[4] The University of Victoria, Canada, accessed: Jan. 2025, `https://www.uvic.ca`

[5] Hernández Villanueva, Michel, Ueda, Ikuo, The Belle II Raw Data Management System, EPJ Web Conf. **245**, 04005 (2020). 10.1051/epjconf/202024504005

[6] A. Dorigo, P. Elmer, F. Furano, A. Hanushevsky, XROOTD - A highly scalable architecture for data access, WSEAS Transactions on Computers **4**, 348 (2005).

[7] IBM Storage Protect, accessed: Jan. 2025, `https://www.ibm.com/products/storage-protect`

[8] Stagni, Federico, Tsaregorodtsev, Andrei, Sailer, André, Haen, Christophe, The DIRAC interware: current, upcoming and planned capabilities and technologies, EPJ Web Conf. **245**, 03035 (2020). 10.1051/epjconf/202024503035

[9] D. Thain, T. Tannenbaum, M. Livny, Distributed computing in practice: the Condor experience., Concurrency - Practice and Experience **17**, 323 (2005).

[10] F. Berghaus, K. Casteels, C. Driemel, M. Ebert, F.F. Galindo, C. Leavett-Brown, D. MacDonell, M. Paterson, R. Seuster, R.J. Sobie et al., "High-Throughput Cloud

Computing with the Cloudscheduler VM Provisioning Service", Computing and Software for Big Science **4**, 4 (2020), `https://doi.org/10.1007/s41781-020-0036-1`.

[11] Cloudscheduler software GitHub repository, accessed: Jan. 2025, `https://github.com/hep-gc/cloudscheduler`

[12] Digital Research Alliance of Canada, accessed: Jan. 2025, `https://www.alliancecan.ca`

[13] Status page of the HEP-RC Cloudscheduler instance, Accessed: Jan. 2025, `https://csv2.heprc.uvic.ca`

[14] The qemu machine emulator and virtualizer, accessed: Jan. 2025, `https://www.qemu.org`

[15] Ganglia monitoring software, accessed: Jan. 2025, `http://ganglia.info`

[16] Nagios software, accessed: Jan. 2025, `https://www.nagios.org/`

[17] M. Ebert, A. Washbrook, Evaluation of ZFS as an efficient WLCG storage backend, Journal of Physics: Conference Series **898**, 062054 (2017). 10.1088/1742-6596/898/6/062054