Project documentation

Fabian Zaremba
Course: TIF20A
Lecturer: Dr. Stephan Laage-Witt
Lecture: Programming Languages – Introduction To Python
Deadline: Source code and documentation until 30.06.21

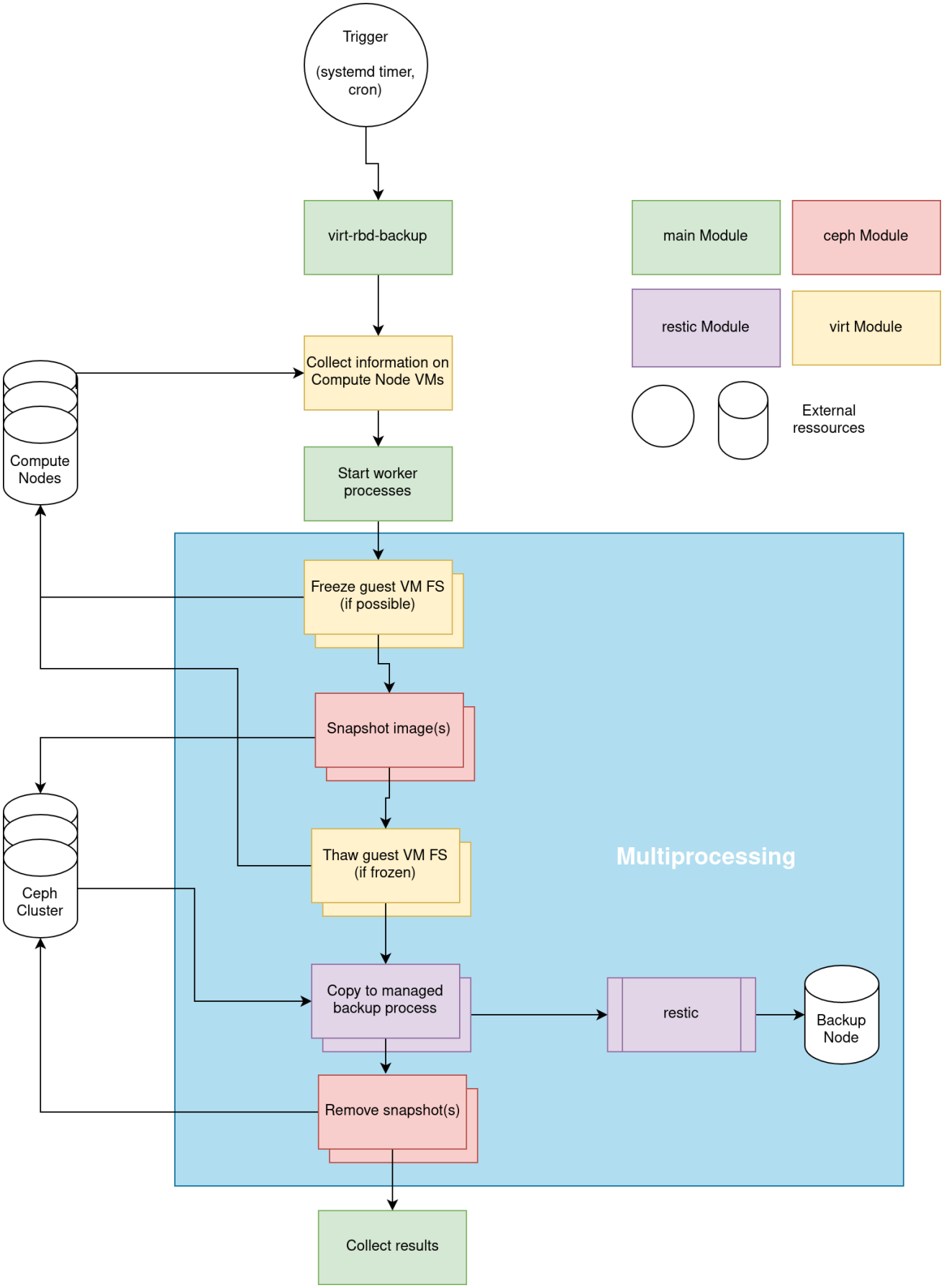
Introduction

Name	virt-rbd-backup – backup solution for libvirt / Ceph RBD
Repository	https://github.com/fabian-z/virt-rbd-backup
Short description	virt-rbd-backup makes automatic backups of virtual machines that are managed with libvirt (Hypervisor QEMU) and stored using Ceph RBD images. Python APIs from libvirt and Ceph are used to list the relevant virtual machines, create a coordinated RBD snapshot and save the snapshot to an output module. The snapshot is removed afterwards.

Features / Scope	• Dynamic list of relevant virtual machines (libvirt API)
	• Create / Process / Delete RBD snapshots via Ceph RBD API
	• Support multiple RBD images per VM
	  • Optimizations for consistency applied in multiprocessing
	• Output modules for external applications or backup targets
	  • Currently a single module is implemented: Restic
	• If supported by guest operating system: Pause filesystem activity during snapshot (using libvirt API and QEMU guest agent)

Out of scope	• Backup management • Scheduling / Automation of Backups (Example with systemd in res/) • Ceph cluster configuration or management

Architecture

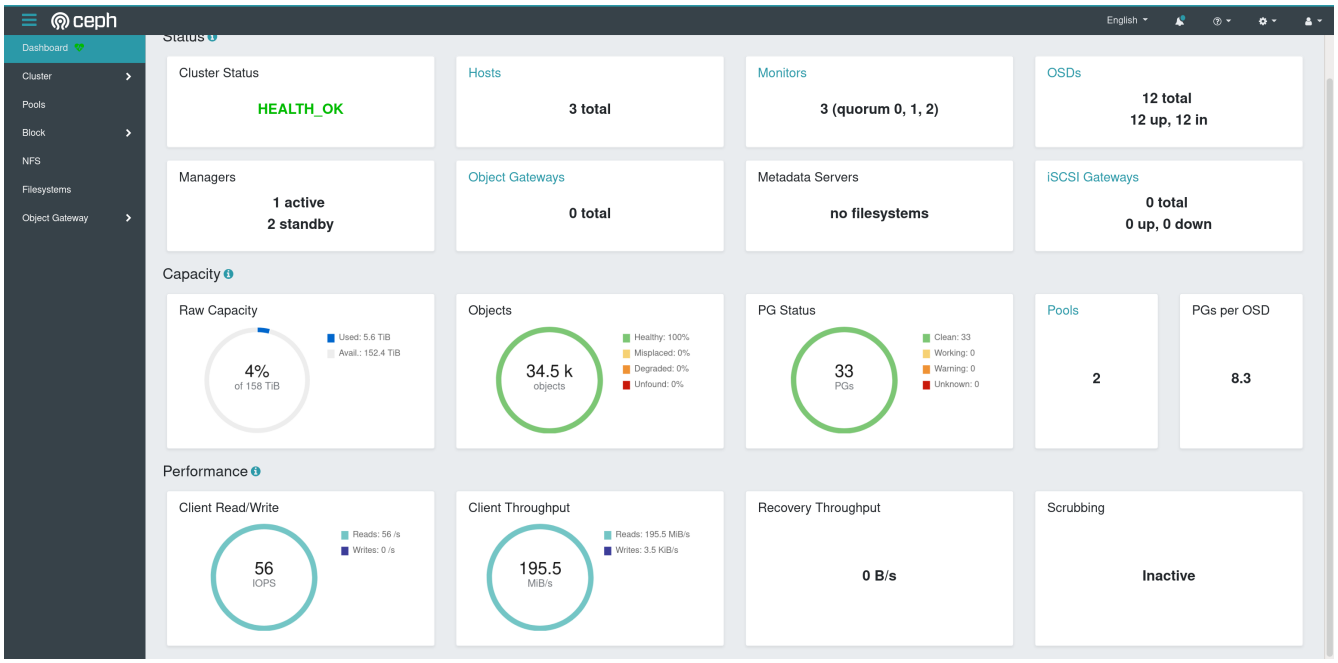


Implementation / Development

- Interpreter version used for development, testing and deployment: Python 3.9
- External modules only for required APIs (RADOS, RBD, libvirt)
  - Installation has to be done via OS / distribution
  - No PyPi packages available, has to be matched with running software version
- Linting with pylint, formatting and naming convention from PEP8 (also using autopep8)
- Configuration with exported Python Constants (config module)

Results

Screenshot Ceph Cluster Dashboard
(Backup with 3 worker processes)



Example output

```
root@node1:~/virt-rbd-backup# python3 main.py
Ignoring non-network disk for domain: TERMSRV
Processing 1 images for domain 7c104238-a949-4c1b-a244-9bcd43de9a21
Processing 1 images for domain e5de2bad-34d8-4171-b676-f6ba03330497
Processing 1 images for domain 7e7ebf0c-f302-44cb-a56f-491ae9967328
Processing 1 images for domain 76e895c8-52ea-4b8f-94dc-0cab169ca4bc
Backup successful: No error occurred for domain e5de2bad-34d8-4171-b676-f6ba03330497
Backup successful: No error occurred for domain 76e895c8-52ea-4b8f-94dc-0cab169ca4bc
Processing 1 images for domain c2813f16-0250-4be1-8fbb-11a26f9ab227
Backup successful: No error occurred for domain 7e7ebf0c-f302-44cb-a56f-491ae9967328
Processing 1 images for domain 41abe680-68b0-45d0-ab6d-7003a729fd8d
Backup successful: No error occurred for domain 41abe680-68b0-45d0-ab6d-7003a729fd8d
Processing 1 images for domain 639a55e4-d0b1-47d1-b5a9-62983a13f0db
Backup successful: No error occurred for domain c2813f16-0250-4be1-8fbb-11a26f9ab227
Backup successful: No error occurred for domain 639a55e4-d0b1-47d1-b5a9-62983a13f0db
Backup successful: No error occurred for domain 7c104238-a949-4c1b-a244-9bcd43de9a21
```

Time and resource usage

```
        Command being timed: "python3 main.py"
        User time (seconds): 4546.00
        System time (seconds): 654.25
        Percent of CPU this job got: 466%
        Elapsed (wall clock) time (h:mm:ss or m:ss): 18:35.41
        Average shared text size (kbytes): 0
        Average unshared data size (kbytes): 0
        Average stack size (kbytes): 0
        Average total size (kbytes): 0
        Maximum resident set size (kbytes): 387360
        Average resident set size (kbytes): 0
        Major (requiring I/O) page faults: 1
        Minor (reclaiming a frame) page faults: 11233559
        Voluntary context switches: 38772816
        Involuntary context switches: 4200646
        Swaps: 0
        File system inputs: 16528
        File system outputs: 223257168
        Socket messages sent: 0
        Socket messages received: 0
        Signals delivered: 0
        Page size (bytes): 4096
        Exit status: 0
```

When calculating with complete image sizes (including sparse space), virt-rbd-backup has reached speeds of up to 550 MiB/s in integration tests with 3 worker processes. Connection between servers was made with Ethernet 10 GBASE-T.

The restic output module create a backup snapshot for each processed RBD image, which allows to restore the original (raw) image content of the RBD snapshot.

Project state

All planned features have been implemented. Additionally, a worker pool is used for concurrent and parallel processing of backups. virt-rbd-backup is currently on production use on multiple servers.

Additional features / Prospect

- Additional output modules
- Progress indication using additional queues and modified I/O processing – large implementation complexity (needs extensive testing)