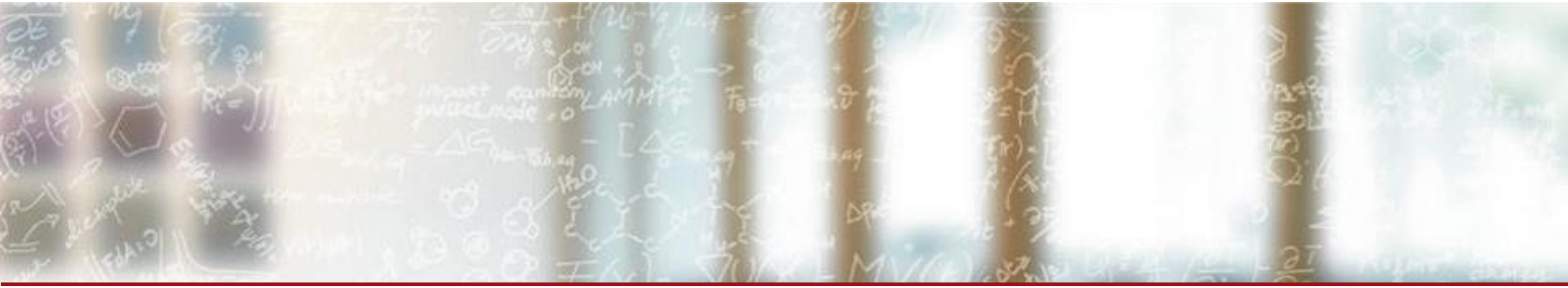
CSCS
Centro Svizzero di Calcolo Scientifico
Swiss National Supercomputing Centre

ETH zürich



FirecREST-v2, the next generation API for HPC

Introduction

Lugano, CH
May 28th, 2025

Agenda

FirecREST-v2 Training – Agenda

Time	Topic
10:00 – 10:15	Welcome coffee and registration
10:15 – 11:00	FirecREST-v2 introduction
11:00 – 11:05	Break
11:05 – 12:00	pyFirecREST and FirecREST CLI hands on
12:00 – 12:15	FirecREST UI showcase
12:15 – 13:15	Lunch break
13:15 – 14:30	Hackathon hands on
14:30 – 14:45	Coffee break
14:45 – 16:00	Hackathon hands on (cont.) and Closure

FirecREST-v2 Training – Repository

- Course repository: https://github.com/eth-cscs/firecrest-training

- In this repository you can find this presentation, documentation of FirecREST, and training material

- Clone the repository in your laptop:

```
> git clone https://github.com/eth-cscs/firecrest-training
> cd firecrest-training
```

FirecREST-v2 Training – Tell us about you

- Tell us who you are and what you do
- Did you know about FirecREST? Do you use it?
- What are your expectations for today’s hackathon?

Motivation

What's FirecREST?

- FirecREST is an API for managing and accessing HPC resources
  - Works as an HTTP proxy for HPC resources
  - Provides a standard development layer for users and web developers
  - Exposes abstractions for workload schedulers (SLURM, PBS, etc), large data transfer, and access to POSIX filesystems operations
  - Enables a single point of access for automated workflows on HPC, which facilitates support

What's FirecREST?

- You can use FirecREST for
  - Automated workflows on HPC (cron jobs, periodic testing, etc)
  - CI/CD Pipelines on HPC (Gitlab CI, Github Actions, Jenkins CI, etc)
  - Workflow managers for AI/ML (Apache Airflow, Snakemake, Nextflow, etc)
  - Create HPC User Dashboards (GUI, Browser apps, Mobile apps, etc)

Why do we need a version 2?

- FirecREST-v1 has been developed since 2018, open-source and public since 2019, and in Production at CSCS since 2021.
  - Since late 2023 we've started the evaluation on new API, and in 2024 a review on the performance of version 1 to identify which are the main issues
  - FirecREST team identified 2 main bottlenecks
    - HTTP server issues: due to the usage of **sequential** HTTP server and web application framework technologies
    - Latency issues: derived from connection to backend services, such as the clusters and storage using **sequential** libraries

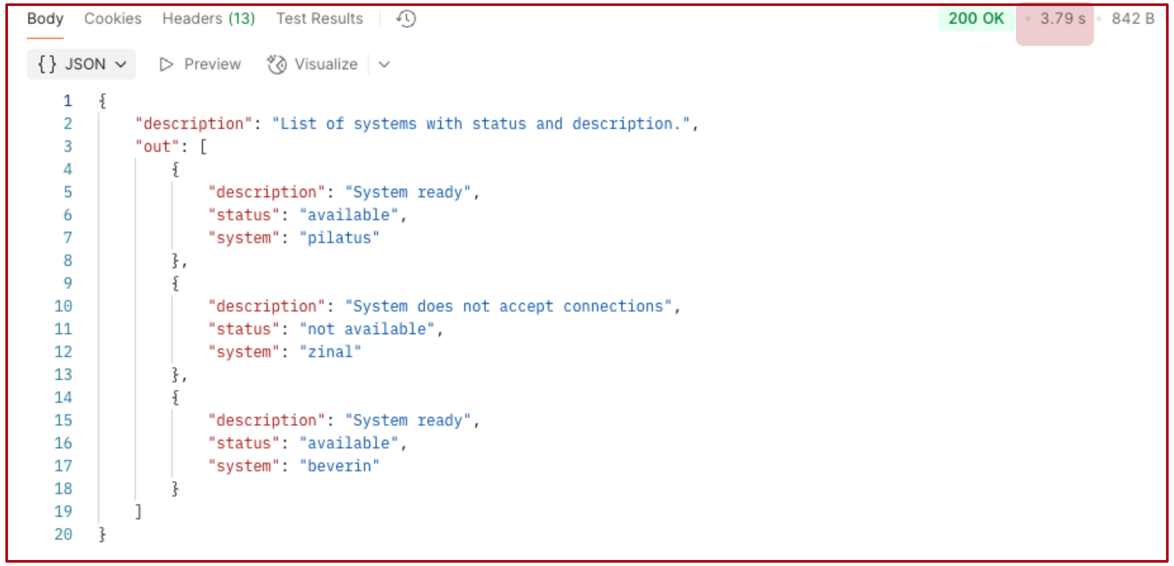
Why do we need a version 2?

- Version 2 improvements
  - Technologies: end-to-end asynchronous communication
    - Web server: `FastAPI`
    - HTTP server: `Uvicorn` (ASGI)
    - Backend services request using `AsyncIO` and `AsyncSSH`
  - Enhancement in connection to backend service
    - Introducing the Health Checker
    - Periodically checks that the backend services can handle requests efficiently, increasing response times
    - *v1 checks availability on requests, which leads to timeouts and longer response times*

Why do we need a version 2?

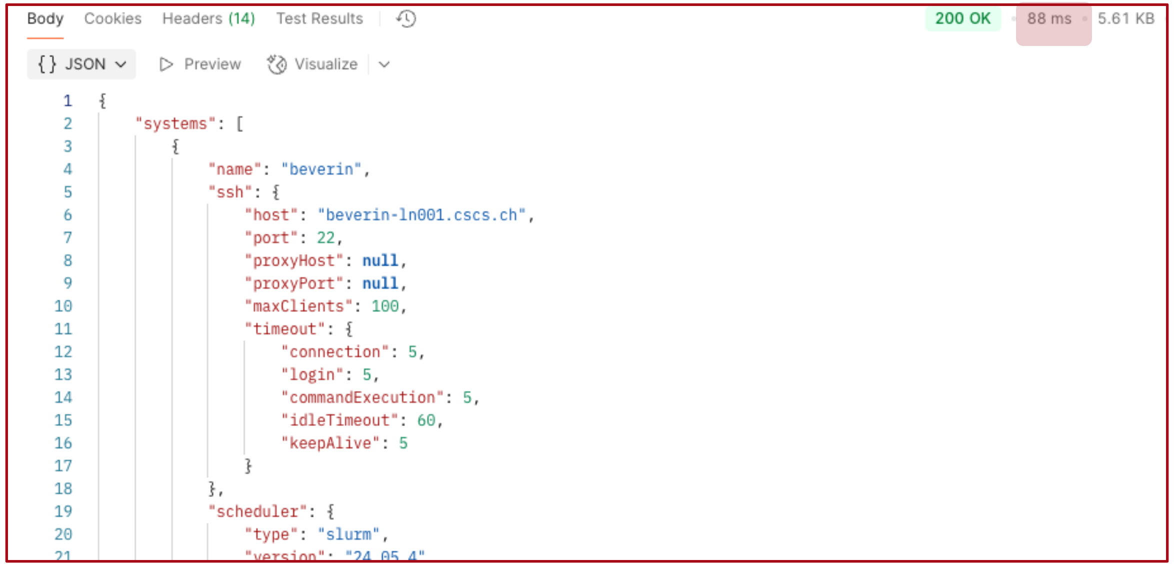
- Version 2 improvements
  - ○ *Example 1: systems’ status performance*

Version 1



```
{
    "description": "List of systems with status and description.",
    "out": [
        {
            "description": "System ready",
            "status": "available",
            "system": "pilatus"
        },
        {
            "description": "System does not accept connections",
            "status": "not available",
            "system": "zinal"
        },
        {
            "description": "System ready",
            "status": "available",
            "system": "beverin"
        }
    ]
}
```

Version 2

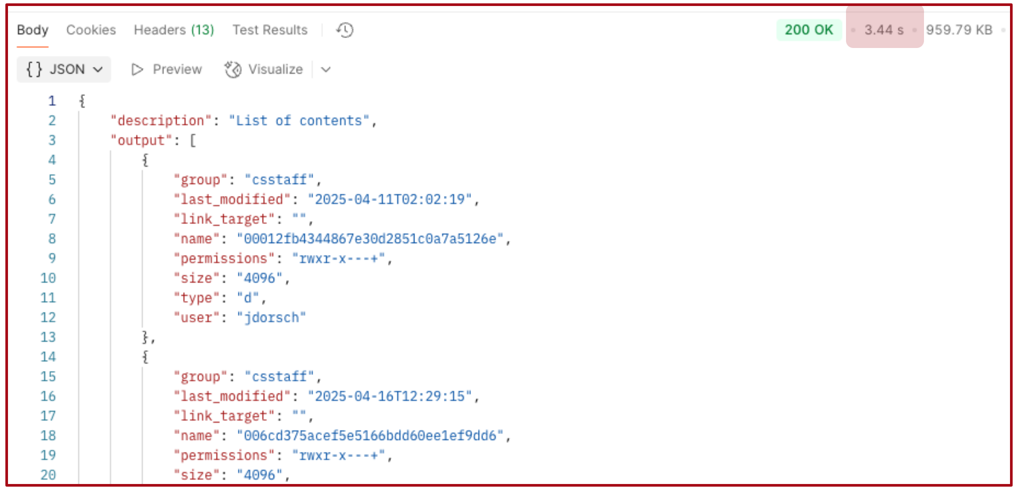


```
{
    "systems": [
        {
            "name": "beverin",
            "ssh": {
                "host": "beverin-ln001.cscs.ch",
                "port": 22,
                "proxyHost": null,
                "proxyPort": null,
                "maxClients": 100,
                "timeout": {
                    "connection": 5,
                    "login": 5,
                    "commandExecution": 5,
                    "idleTimeout": 60,
                    "keepAlive": 5
                }
            },
            "scheduler": {
                "type": "slurm",
                "version": "24.05.4"
```

Why do we need a version 2?

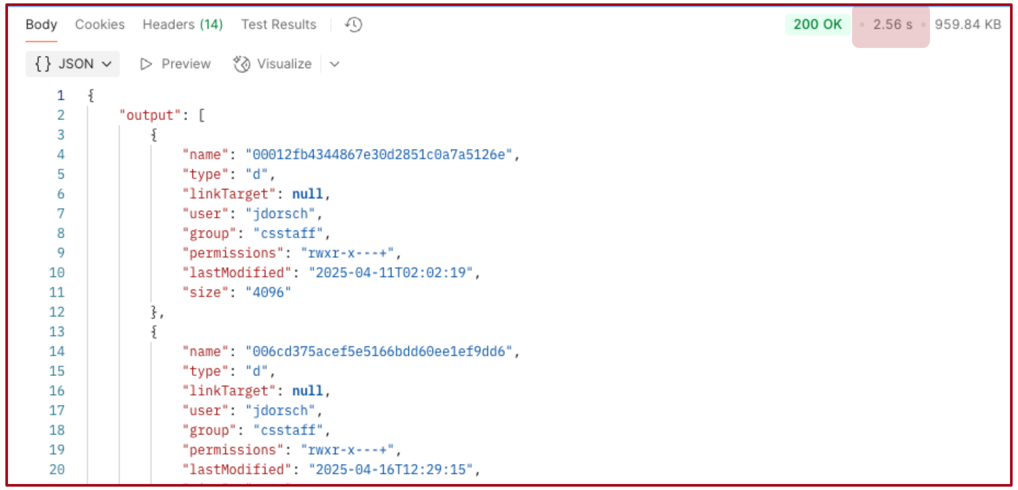
- Version 2 improvements
  - *Example 2: listing filesystems performance (5300 items)*

Version 1



```
{
    "description": "List of contents",
    "output": [
        {
            "group": "csstaff",
            "last_modified": "2025-04-11T02:02:19",
            "link_target": "",
            "name": "00012fb4344867e30d2851c0a7a5126e",
            "permissions": "rwxr-x---+",
            "size": "4096",
            "type": "d",
            "user": "jdorsch"
        },
        {
            "group": "csstaff",
            "last_modified": "2025-04-16T12:29:15",
            "link_target": "",
            "name": "006cd375acef5e5166bdd60ee1ef9dd6",
            "permissions": "rwxr-x---+",
            "size": "4096",
```

Version 2

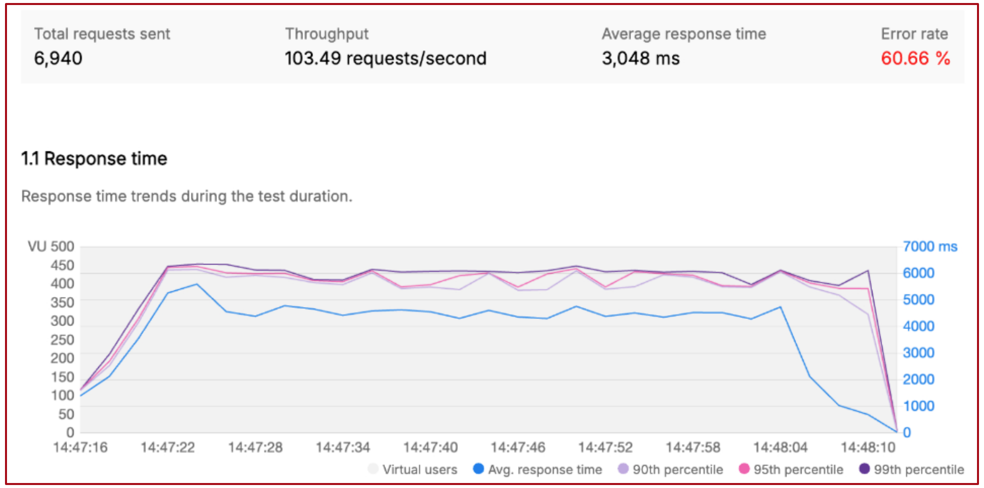


```
{
    "output": [
        {
            "name": "00012fb4344867e30d2851c0a7a5126e",
            "type": "d",
            "linkTarget": null,
            "user": "jdorsch",
            "group": "csstaff",
            "permissions": "rwxr-x---+",
            "lastModified": "2025-04-11T02:02:19",
            "size": "4096"
        },
        {
            "name": "006cd375acef5e5166bdd60ee1ef9dd6",
            "type": "d",
            "linkTarget": null,
            "user": "jdorsch",
            "group": "csstaff",
            "permissions": "rwxr-x---+",
            "lastModified": "2025-04-16T12:29:15",
```

Why do we need a version 2?

- Version 2 improvements
  - *Example 3: high-throughput regime (500 concurrent clients)*

Version 1

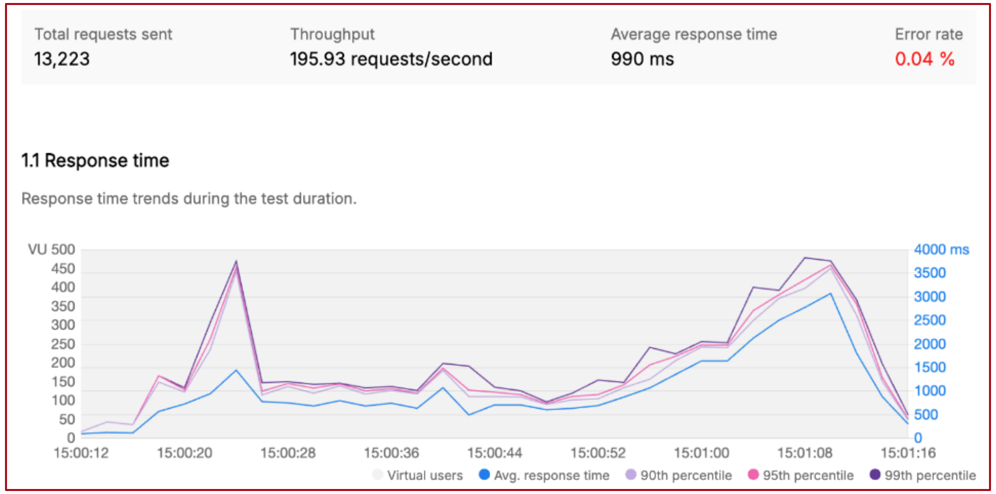


Total requests sent	Throughput	Average response time	Error rate
6,940	103.49 requests/second	3,048 ms	60.66 %

1.1 Response time

Response time trends during the test duration.

Version 2



Total requests sent	Throughput	Average response time	Error rate
13,223	195.93 requests/second	990 ms	0.04 %

1.1 Response time

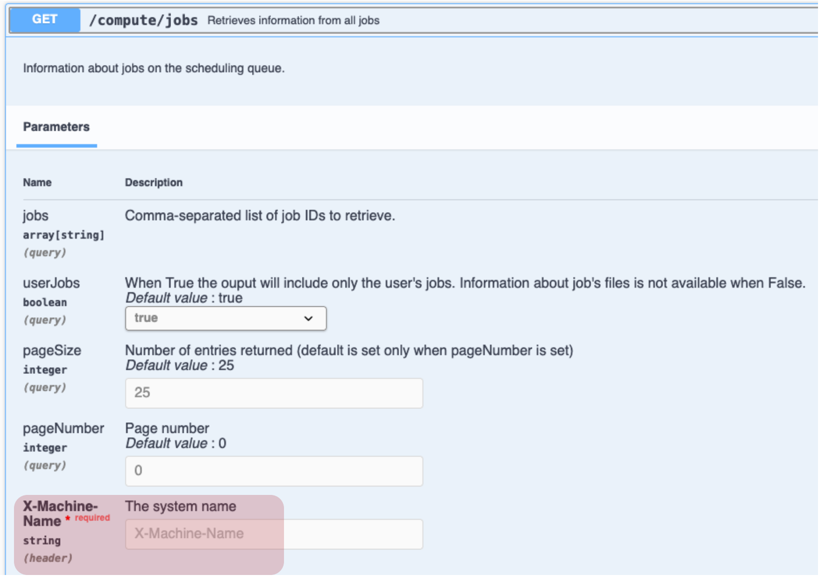
Response time trends during the test duration.

Differences between v1 and v2

Differences between v1 and v2

- Interface
  - v2 API specification at https://eth-cscs.github.io/firecrest-v2/openapi
  - Parameters
    - System name
      - In v1, the parameter was passed in the `X-Machine-Name` **header**
      - In v2, this is passed as a **path** parameter

Version 1



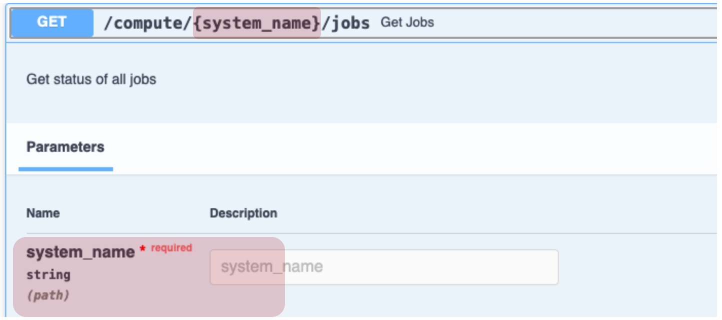
GET `/compute/jobs` Retrieves information from all jobs

Information about jobs on the scheduling queue.

Parameters

Name	Description
jobs `array[string]` *(query)*	Comma-separated list of job IDs to retrieve.
userJobs `boolean` *(query)*	When True the ouput will include only the user's jobs. Information about job's files is not available when False. *Default value* : true true
pageSize `integer` *(query)*	Number of entries returned (default is set only when pageNumber is set) *Default value* : 25 25
pageNumber `integer` *(query)*	Page number *Default value* : 0 0
X-Machine-Name * required `string` *(header)*	The system name X-Machine-Name

Version 2



GET `/compute/{system_name}/jobs` Get Jobs

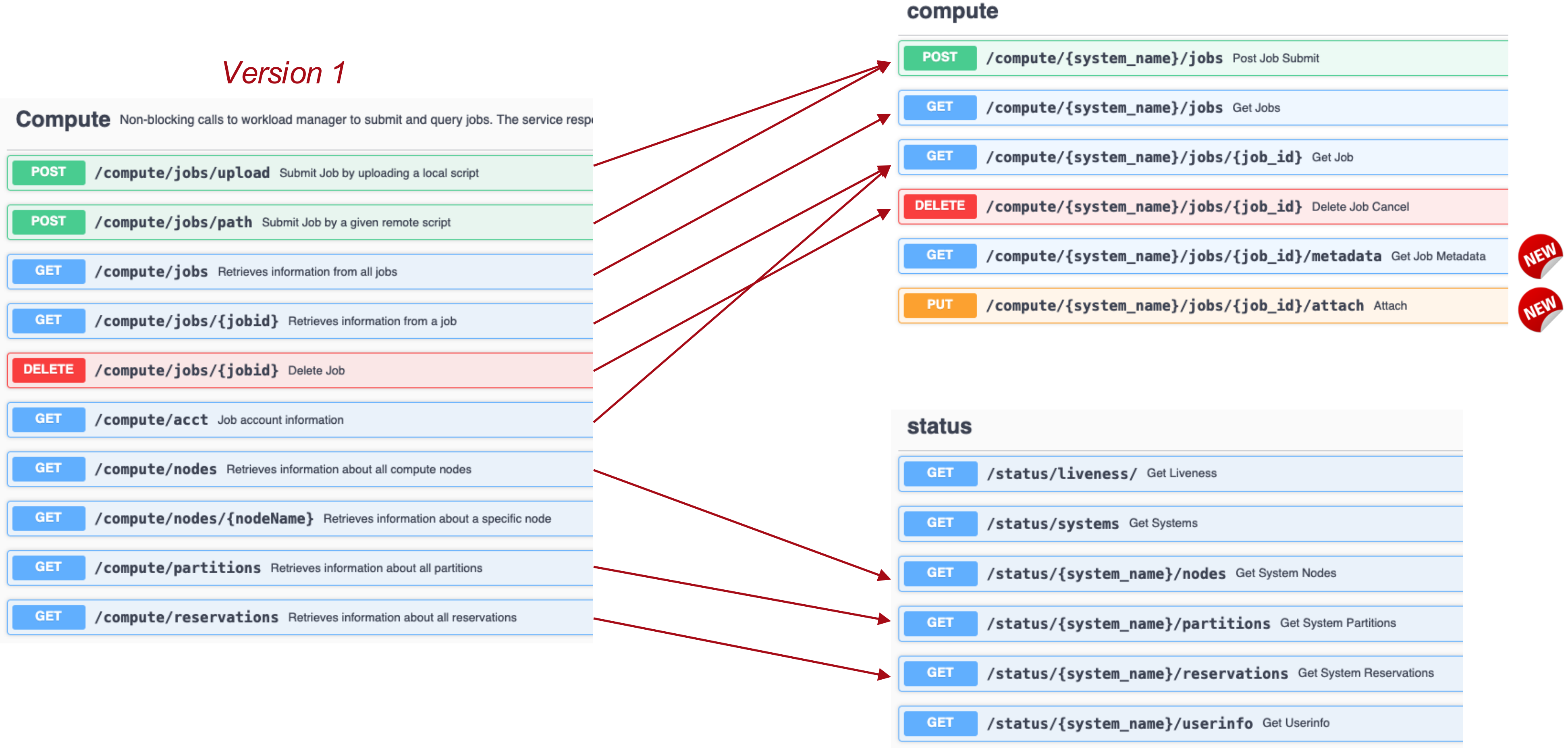
Get status of all jobs

Parameters

Name	Description
system_name * required `string` *(path)*	system_name

Differences between v1 and v2

- Scheduler



Version 1

Compute Non-blocking calls to workload manager to submit and query jobs. The service resp

- POST `/compute/jobs/upload` Submit Job by uploading a local script
- POST `/compute/jobs/path` Submit Job by a given remote script
- GET `/compute/jobs` Retrieves information from all jobs
- GET `/compute/jobs/{jobid}` Retrieves information from a job
- DELETE `/compute/jobs/{jobid}` Delete Job
- GET `/compute/acct` Job account information
- GET `/compute/nodes` Retrieves information about all compute nodes
- GET `/compute/nodes/{nodeName}` Retrieves information about a specific node
- GET `/compute/partitions` Retrieves information about all partitions
- GET `/compute/reservations` Retrieves information about all reservations

Version 2

compute

- POST `/compute/{system_name}/jobs` Post Job Submit
- GET `/compute/{system_name}/jobs` Get Jobs
- GET `/compute/{system_name}/jobs/{job_id}` Get Job
- DELETE `/compute/{system_name}/jobs/{job_id}` Delete Job Cancel
- GET `/compute/{system_name}/jobs/{job_id}/metadata` Get Job Metadata NEW
- PUT `/compute/{system_name}/jobs/{job_id}/attach` Attach NEW

status

- GET `/status/liveness/` Get Liveness
- GET `/status/systems` Get Systems
- GET `/status/{system_name}/nodes` Get System Nodes
- GET `/status/{system_name}/partitions` Get System Partitions
- GET `/status/{system_name}/reservations` Get System Reservations
- GET `/status/{system_name}/userinfo` Get Userinfo

Differences between v1 and v2

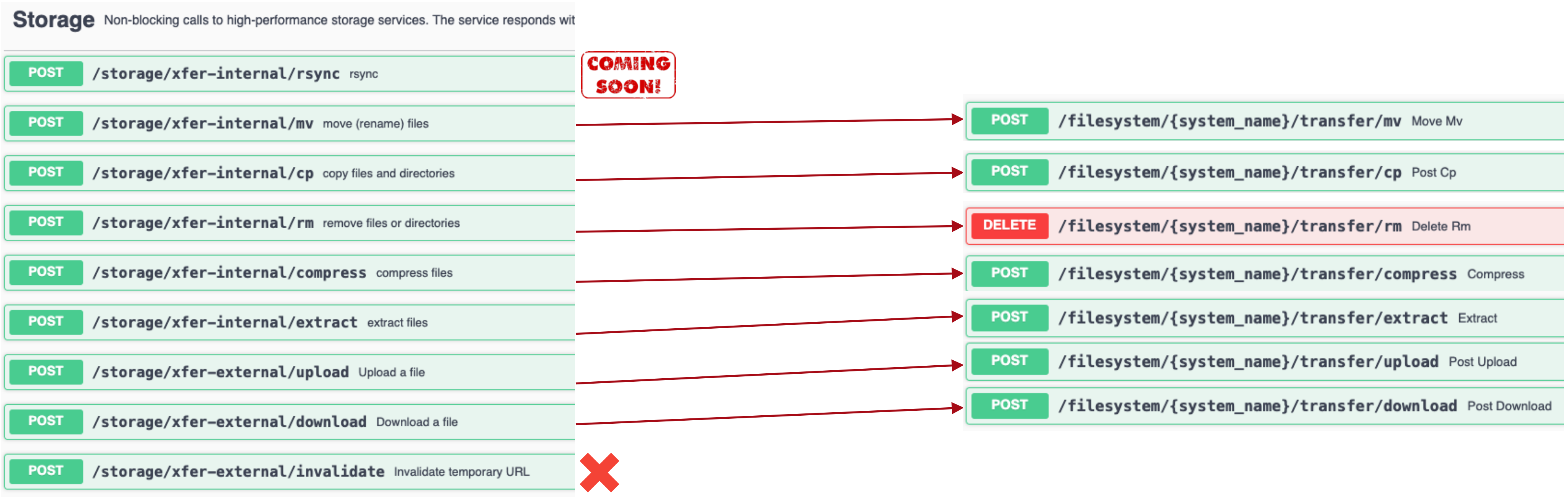
- Scheduler
  - All operations are now blocking (submission, querying, accounting, etc)
  - Submitting or querying the status of a job now returns an immediate response
  - There’s no need for the `/tasks` endpoint (removed)

```
> GET compute/{system_name}/jobs
{
    "jobs": [
        {
            "jobId": 1000,
            "name": "Job-1000",
            "status": {
                "state": "COMPLETED",
                "stateReason": "ReqNodeNotAvail",
                "exitCode": 0,
                "interruptSignal": 0
            },
            "tasks": [
                {
                    "id": "1000.batch",
                    "name": "batch",
                    "status": {
                        "state": "COMPLETED",
                        "stateReason": null,
                        "exitCode": 0,
                        "interruptSignal": 0
                    },
                    "time": {
                        "elapsed": 4,
                        "start": null,
                        "end": null,
                        "suspended": 0,
                        "limit": null
                    }
                },
                {…}
            ],

    ]
}
```

Differences between v1 and v2

- Interface
  - Data transfer and data mover



Storage Non-blocking calls to high-performance storage services. The service responds wit

Version 1	*Version 2*
POST `/storage/xfer-internal/rsync` rsync	COMING SOON!
POST `/storage/xfer-internal/mv` move (rename) files	POST `/filesystem/{system_name}/transfer/mv` Move Mv
POST `/storage/xfer-internal/cp` copy files and directories	POST `/filesystem/{system_name}/transfer/cp` Post Cp
POST `/storage/xfer-internal/rm` remove files or directories	DELETE `/filesystem/{system_name}/transfer/rm` Delete Rm
POST `/storage/xfer-internal/compress` compress files	POST `/filesystem/{system_name}/transfer/compress` Compress
POST `/storage/xfer-internal/extract` extract files	POST `/filesystem/{system_name}/transfer/extract` Extract
POST `/storage/xfer-external/upload` Upload a file	POST `/filesystem/{system_name}/transfer/upload` Post Upload
POST `/storage/xfer-external/download` Download a file	POST `/filesystem/{system_name}/transfer/download` Post Download
POST `/storage/xfer-external/invalidate` Invalidate temporary URL	✖

Differences between v1 and v2

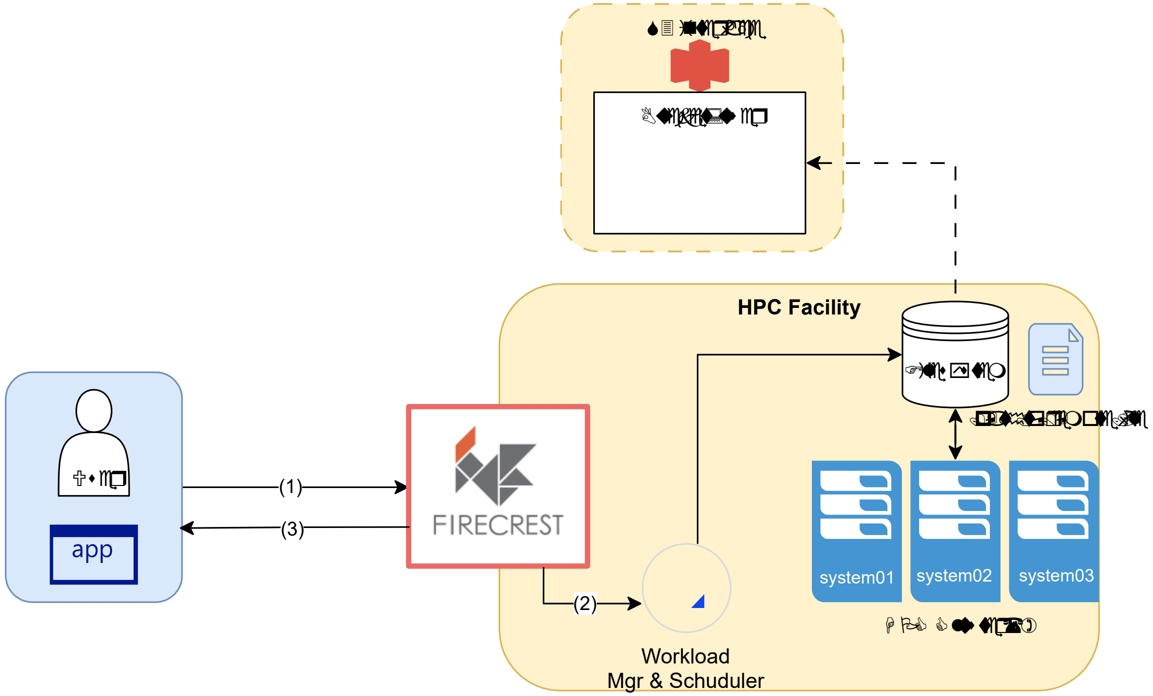
- Interface
  - Data Transfer and data mover
    - External download and upload for large files is now is managed via the workload scheduler.

  - Download process

    (1) User requests to download a file

    (2) FirecREST creates a job
    that moves the file to an S3 bucket

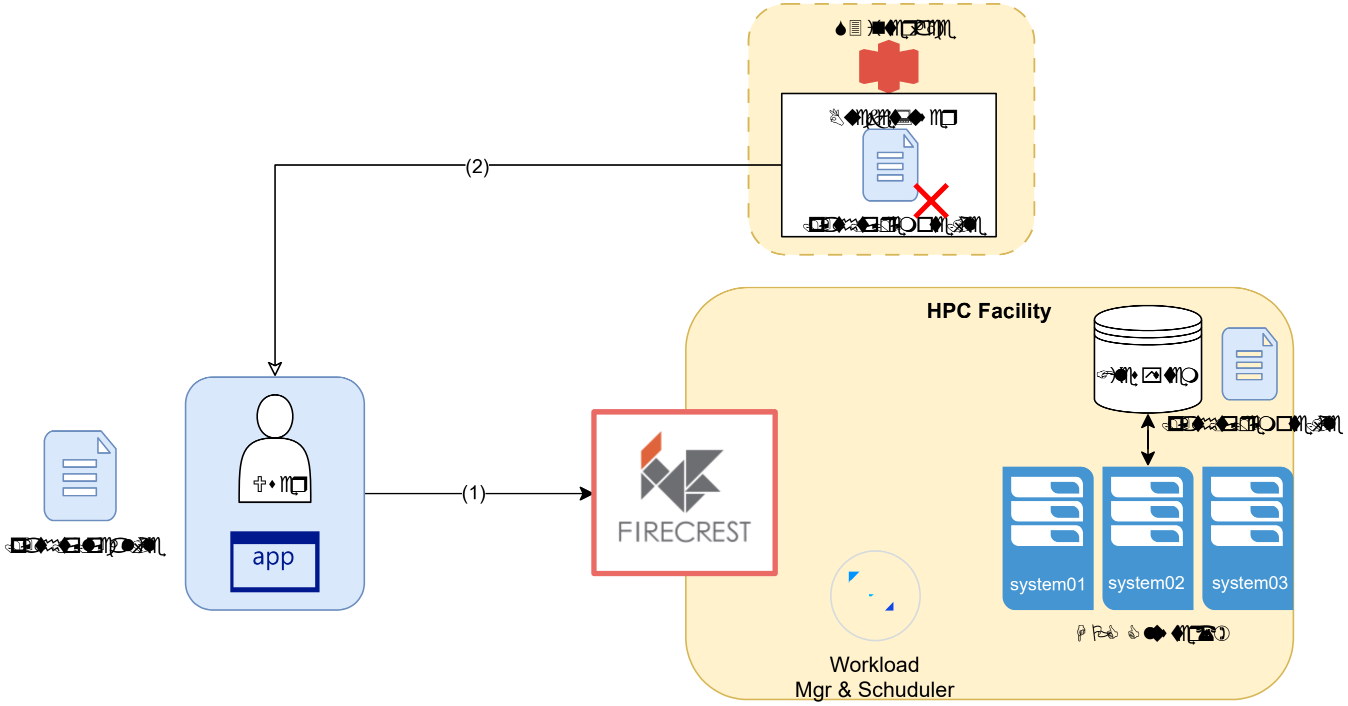
    (3) FirecREST returns to the user a
    self-signed URL to download the file



Differences between v1 and v2

- Interface
  - Data Transfer and data mover
    - External download and upload for large files is now is managed via the workload scheduler.
  - Download process (cont.)
    1. User can check the status of the transfer using the scheduler
    2. Once the file is on S3, the user can use the URL to download the file

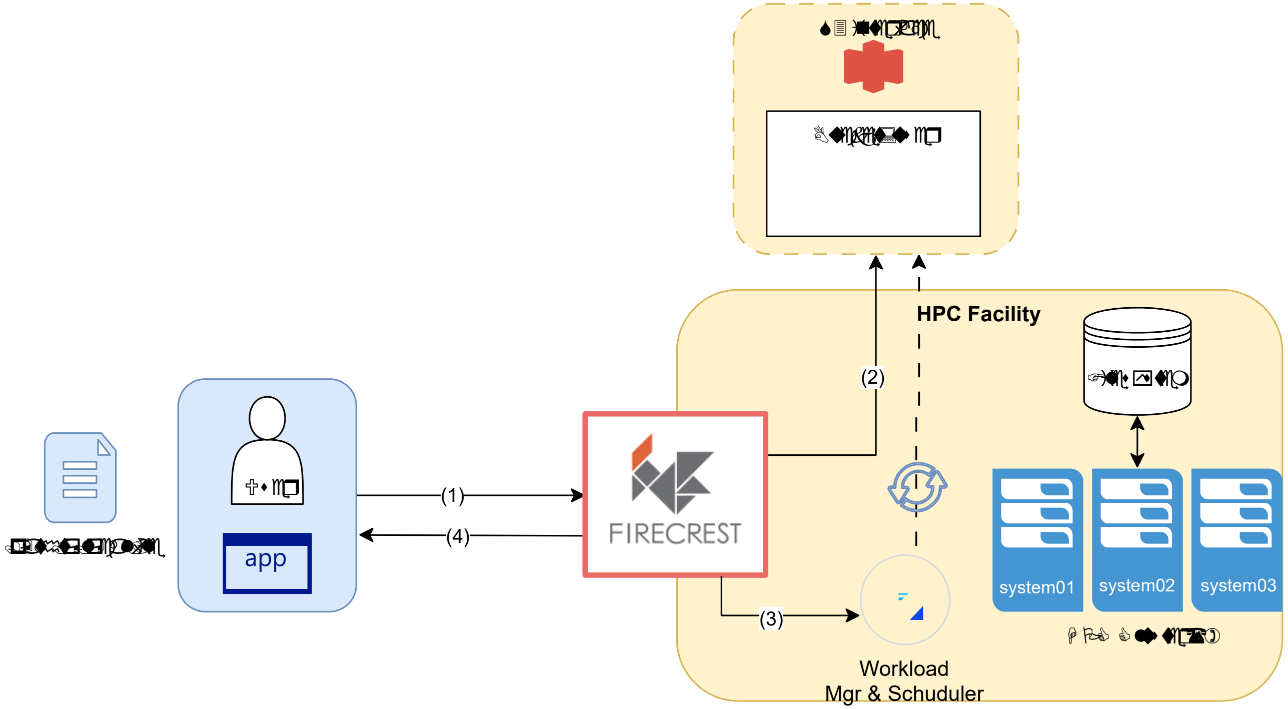
The object is removed from S3 automatically



Differences between v1 and v2

- Interface
  - Data Transfer and data mover
    - External download and upload for large files is now is managed via the workload scheduler.

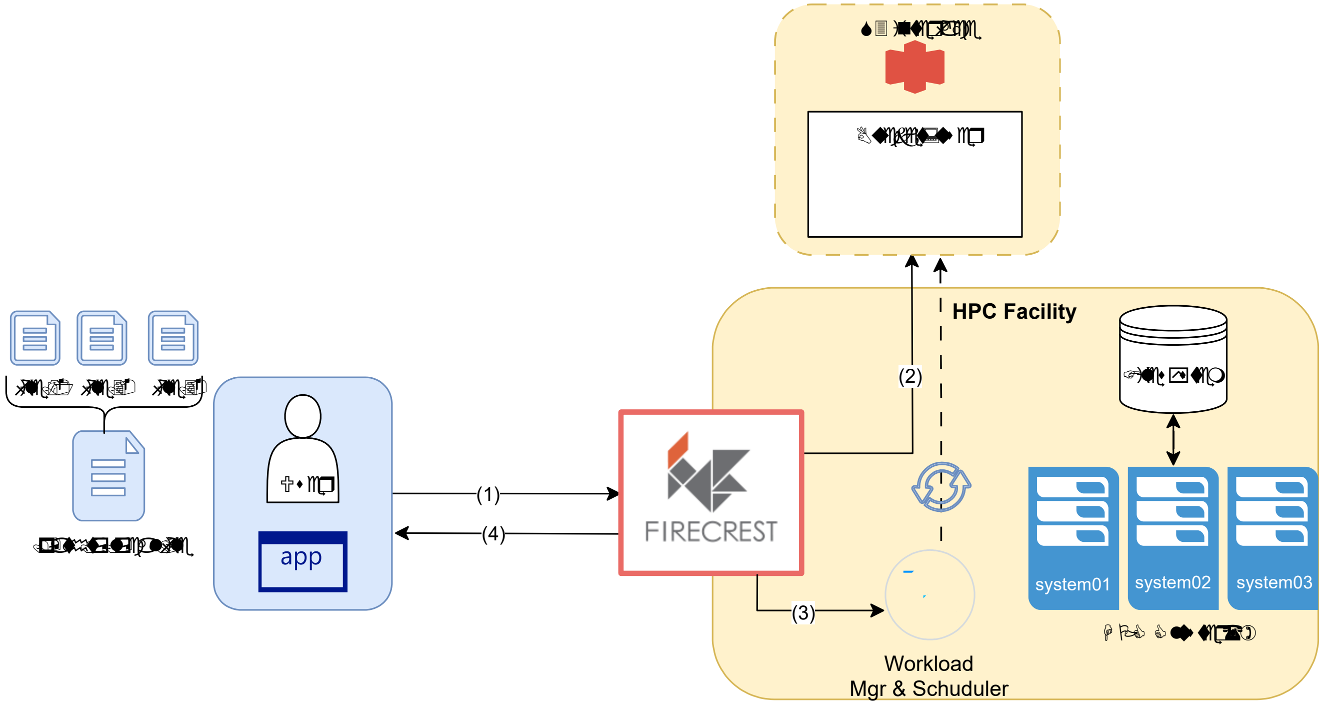
  - Upload process
    1. User requests to upload a file to a remote directory in the target system
    2. FirecREST will create a bucket on S3 server
    3. FirecREST creates a job in the scheduler waiting for the file to be uploaded
    4. FirecREST reports to the user the self-signed upload form



Differences between v1 and v2

- Interface
  - Data Transfer and data mover
    - External download and upload for large files is now is managed via the workload scheduler.

  - Upload process (cont.)
    1. If the file to upload is >5GB,
       S3 requires the file to be uploaded
       in chunks of <= 5GB
    2. FirecREST is able to handle that
       request and provide a list of URLs
    3. The user must split the file in the
       number of chunks required



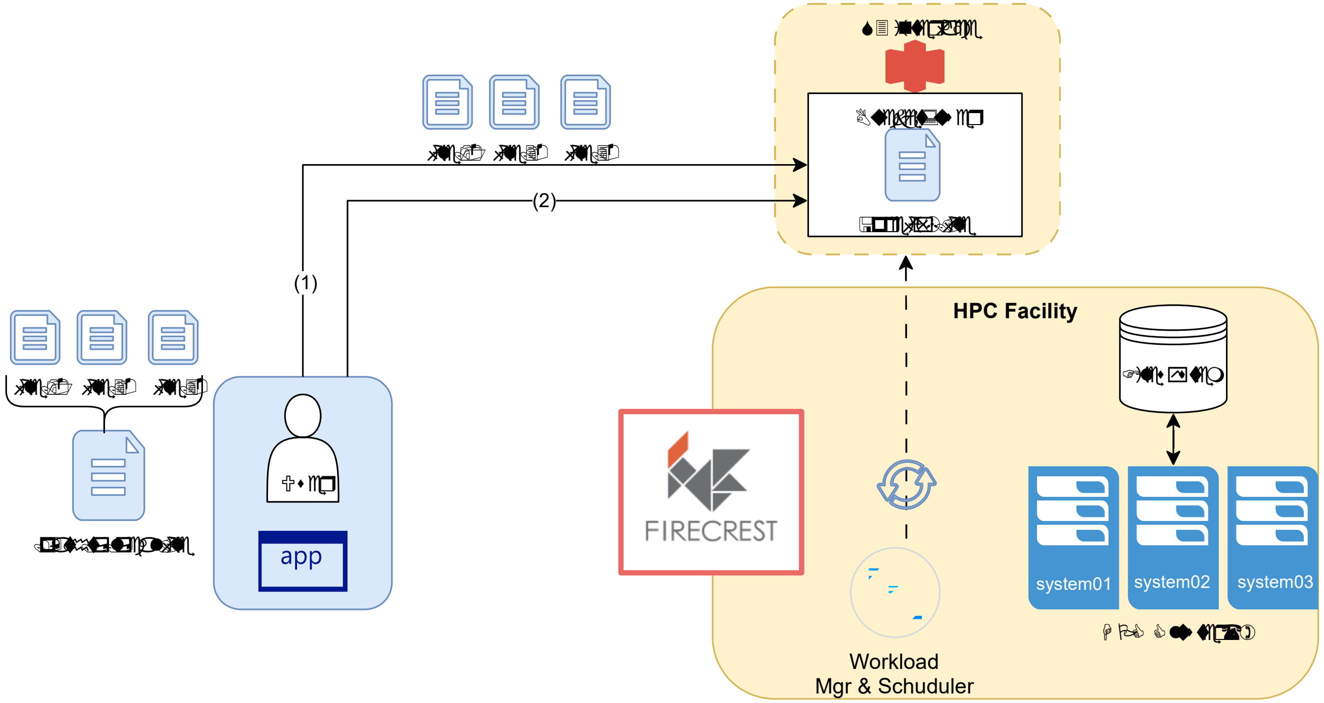
Differences between v1 and v2

- Interface
  - Data Transfer and data mover
    - External download and upload for large files is now is managed via the workload scheduler.

  - Upload process (cont.)

    1. User upload the parts using the URLs provided

    2. To close the multipart upload, user provides the URL to join the chunks



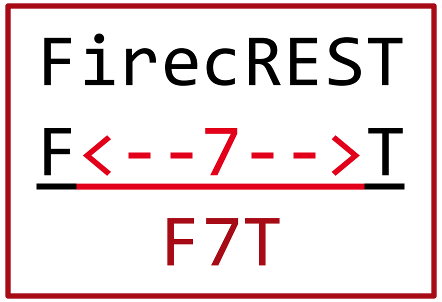
Documentation

Documentation

- Useful links
    - Official page at CSCS: https://products.cscs.ch/firecrest/
    - Official Docs: https://eth-cscs.github.io/firecrest-v2/
    - Python Library and CLI Docs: https://pyfirecrest.readthedocs.io

Requirements for today's training

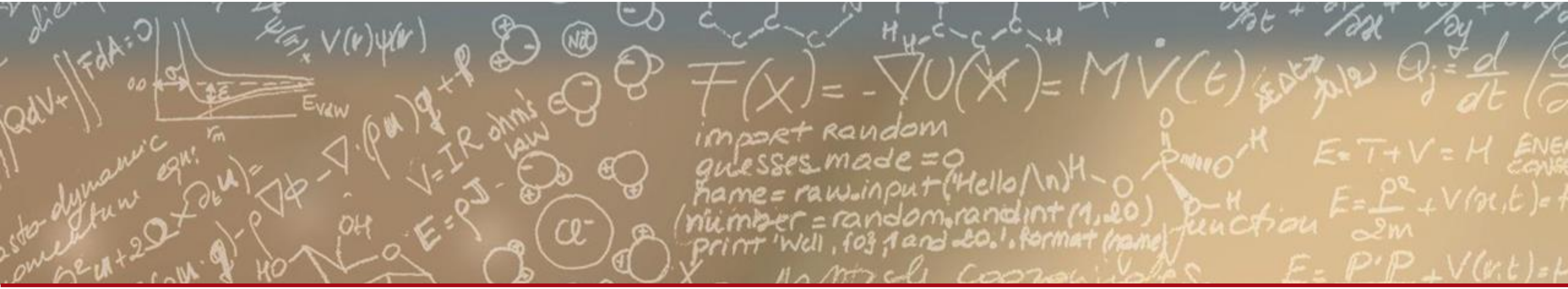
- GitHub repository: https://github.com/eth-cscs/firecrest-training
- Developer Portal: https://developer.cscs.ch
- Training account for `daint` (username and password provided by your trainers)
- When submitting jobs use `--reservation=firecrest` in your sbatch file
- Slack channel: firecrest-community
- You might see today the naming "**`f7t`**" several times: it is called ”numeronym”



FirecREST

F<--7-->T

F7T


F(X) = -∇U(X) = MV̇(t)
import random
guesses_made = 0
name = raw_input('Hello\n')
number = random.randint(1, 20)
print 'Well, {0}, 1 and 20.'.format(name)
V=IR ohm's law
E = T + V = H
function

Thank you for your attention.