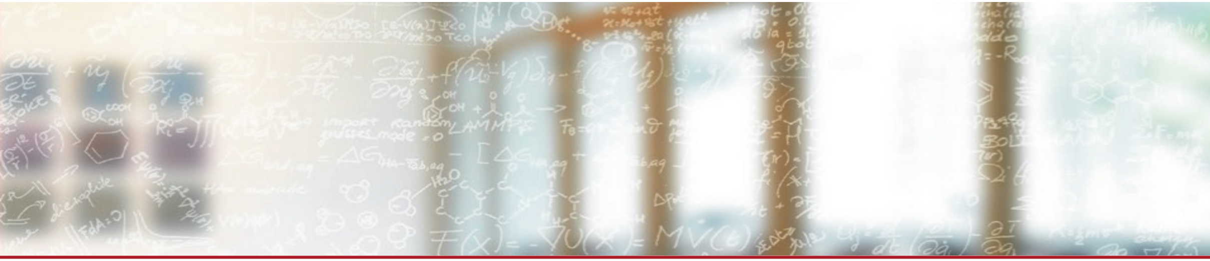
CSCS
Centro Svizzero di Calcolo Scientifico
Swiss National Supercomputing Centre

ETH zürich



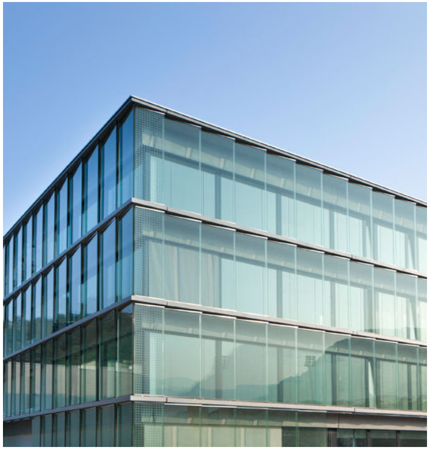
Introduction to FirecREST: a REST interface for programming access to HPC resources

User Lab Day

Eirini Koutsaniti, CSCS/ETH Zurich

2 September 2022

FirecREST: a RESTful API to HPC systems



- Firecrest in a Nutshell
- Microservice Architecture
- Advanced FirecREST Workflows
- Demo

Motivation

- Users wanted to develop applications and platforms that take advantage of the HPC resources.
- Need for a standard modern interface to the HPC resources:
  - HPC clusters
  - Job scheduler
  - Filesystem operations
  - Internal and external data transfers
- Need to integrate with the existing infrastructure

Firecrest in a Nutshell

FirecREST is a **RESTful Web API infrastructure**.

- Provides advanced HPC functionality for modern web-enabled portals and applications. It gives access to
  - HPC Workload Management
  - Data Mover
- Enforces integration with Identity Access Management (IAM) of the HPC center.

Concrete examples of the API

How to list the contents of a directory:

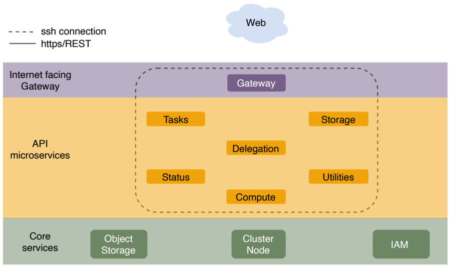
```
$ curl -X GET "firecrest.cscs.ch/utilities/ls?targetPath=<targetpath>" \
       -H "Authorization: Bearer <token>" \
       -H "X-Machine-Name: daint"
```

How to submit a job:

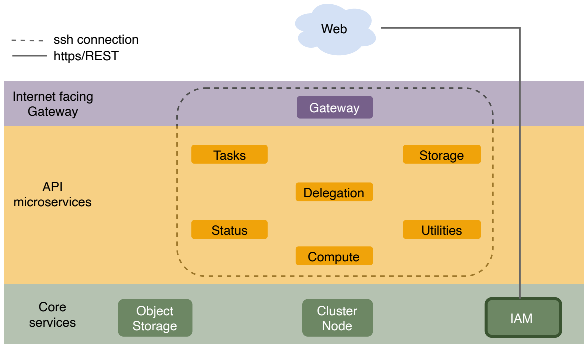
```
$ curl -X POST "firecrest.cscs.ch/compute/jobs/upload" \
       -H "Authorization: Bearer <token>" \
       -H "X-Machine-Name: daint" \
       -F "file=@/path/to/script.sh"
```

Microservice Architecture

- FirecREST is a collection of loosely coupled services.
- This architecture provides maintainability, security and stability.



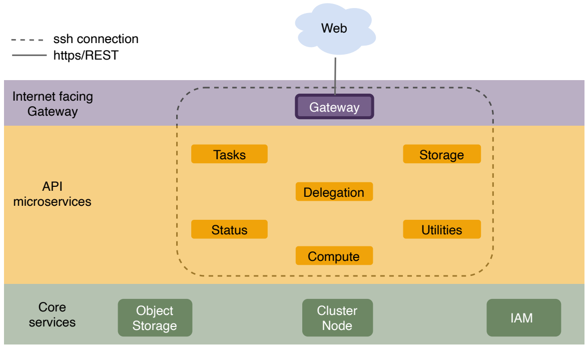
Microservice Architecture



IAM Layer

- Each FirecREST request has to include an OIDC token in the header.
- The first thing a client would have to do is to aquire a valid token from the OIDC server.

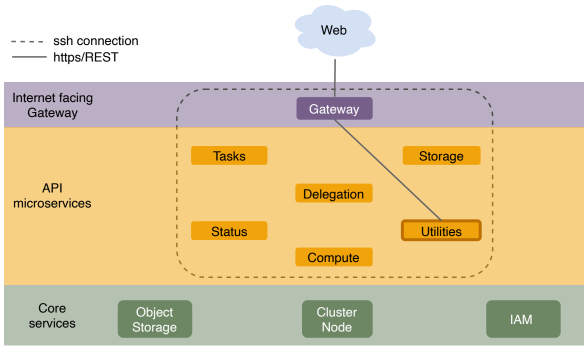
Microservice Architecture



Gateway

- It should be the only service that is open to the internet.
- It is the responsible microservice that will implement and enforce:
  - authentication
  - authorization
  - traffic control
  - analytics and logging of requests

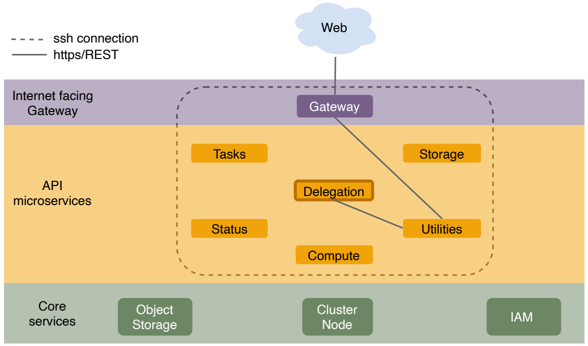
Microservice Architecture



Utilities microservice

- Provides filesystem utilities.
- Checks the validity of the parameters passed with the request.
- All calls are blocking operations.

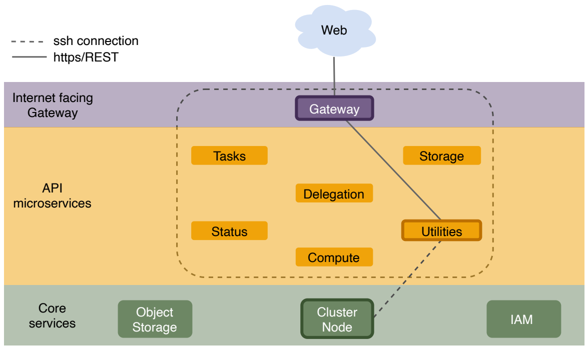
Microservice Architecture



Delegation microservice

- Creates a short-lived SSH certificate to be used for user authentication.
- These certificates are created for the given combination of username, shell command and arguments.

Microservice Architecture



- The Utilities microservice uses the SSH certificate to log in to a **Cluster node**.
- Parses the output of the command.
- Returns a json object to the client.

CSCS

ETH zürich

Microservice Architecture

Other microservices of FirecREST:

- **Compute:** Non-blocking calls to workload manager for submitting/querying/canceling jobs.
- **Storage:** Non-blocking calls to high-performance storage services.
- **Tasks:** Keeps track of the tasks that are created during asynchronous calls.
- **Status:** Provides information on services and infrastructure.

Advanced FirecREST Workflows

Compute Microservice

Every time FirecREST interacts with the scheduler, it is creating a task resource.

- To submit/query/cancel a job the client makes the appropriate request to the Compute microservice.
- It gets a response immediately with the newly created task.
- The task can be used to track the status of the request in an asynchronous way.

Advanced FirecREST Workflows

Storage Microservice - External transfers

- A staging area is used: Object Storage.
- The client will upload/download the file to/from this area.
- The requests from the client to FirecREST aim to get the url to this staging area.
- This allows FirecREST to be responsive and lightweight, since it delegates the large transfers to a service that is more suitable for this.

Advanced FirecREST Workflows

Storage Microservice - Internal transfers

- For small files' transfers you can simply use the Utilities Microservice.
- The maximum file size for data transfers through Utilities is configurable and you can get it from the Status Microservice.
- FirecREST has configurable time limit for all commands, so for larger files you will have to use the dedicated WLM queue for internal data transfers.
- FirecREST will create the job script and submit it based on the request's arguments to Storage Microservice.

Demo

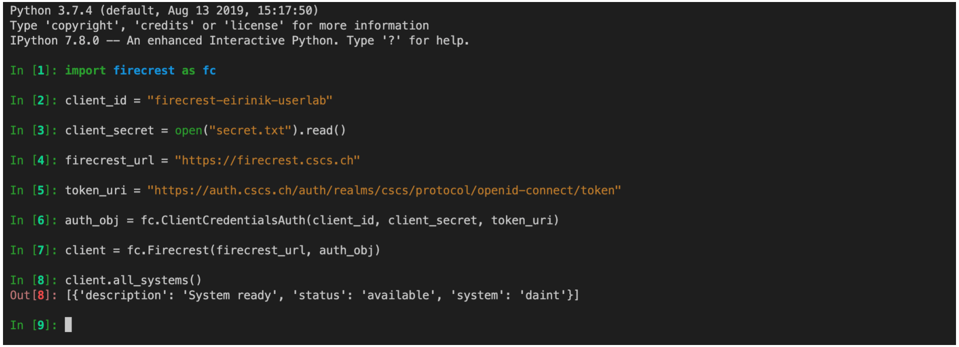
Register a client

- Every request to FirecREST requires an access token, that will be obtained by Keycloak.
- You can register, modify and delete your personal clients in https://oidc-dashboard-prod.cscs.ch/

Demo

Python wrapper - pyfirecrest

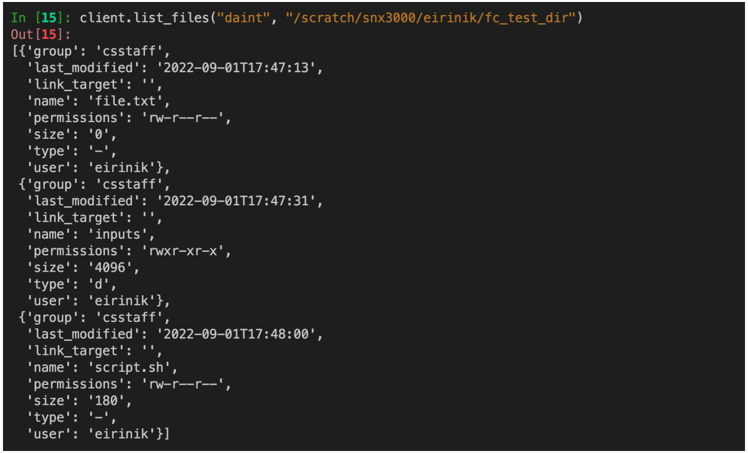
```
python3 -m pip install pyfirecrest
```



```
Python 3.7.4 (default, Aug 13 2019, 15:17:50)
Type 'copyright', 'credits' or 'license' for more information
IPython 7.8.0 -- An enhanced Interactive Python. Type '?' for help.

In [1]: import firecrest as fc

In [2]: client_id = "firecrest-eirinik-userlab"

In [3]: client_secret = open("secret.txt").read()

In [4]: firecrest_url = "https://firecrest.cscs.ch"

In [5]: token_uri = "https://auth.cscs.ch/auth/realms/cscs/protocol/openid-connect/token"

In [6]: auth_obj = fc.ClientCredentialsAuth(client_id, client_secret, token_uri)

In [7]: client = fc.Firecrest(firecrest_url, auth_obj)

In [8]: client.all_systems()
Out[8]: [{'description': 'System ready', 'status': 'available', 'system': 'daint'}]

In [9]: 
```

Examples

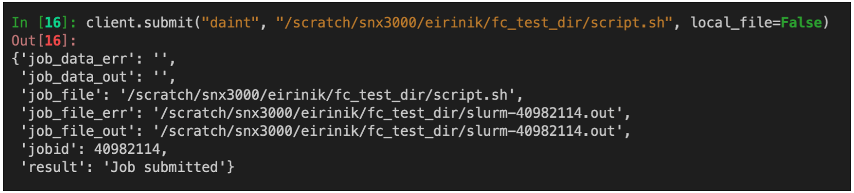
Python wrapper - pyfirecrest



```
In [15]: client.list_files("daint", "/scratch/snx3000/eirinik/fc_test_dir")
Out[15]:
[{'group': 'csstaff',
  'last_modified': '2022-09-01T17:47:13',
  'link_target': '',
  'name': 'file.txt',
  'permissions': 'rw-r--r--',
  'size': '0',
  'type': '-',
  'user': 'eirinik'},
 {'group': 'csstaff',
  'last_modified': '2022-09-01T17:47:31',
  'link_target': '',
  'name': 'inputs',
  'permissions': 'rwxr-xr-x',
  'size': '4096',
  'type': 'd',
  'user': 'eirinik'},
 {'group': 'csstaff',
  'last_modified': '2022-09-01T17:48:00',
  'link_target': '',
  'name': 'script.sh',
  'permissions': 'rw-r--r--',
  'size': '180',
  'type': '-',
  'user': 'eirinik'}]
```

Examples

Python wrapper - pyfirecrest



```
In [16]: client.submit("daint", "/scratch/snx3000/eirinik/fc_test_dir/script.sh", local_file=False)
Out[16]:
{'job_data_err': '',
 'job_data_out': '',
 'job_file': '/scratch/snx3000/eirinik/fc_test_dir/script.sh',
 'job_file_err': '/scratch/snx3000/eirinik/fc_test_dir/slurm-40982114.out',
 'job_file_out': '/scratch/snx3000/eirinik/fc_test_dir/slurm-40982114.out',
 'jobid': 40982114,
 'result': 'Job submitted'}
```

CSCS

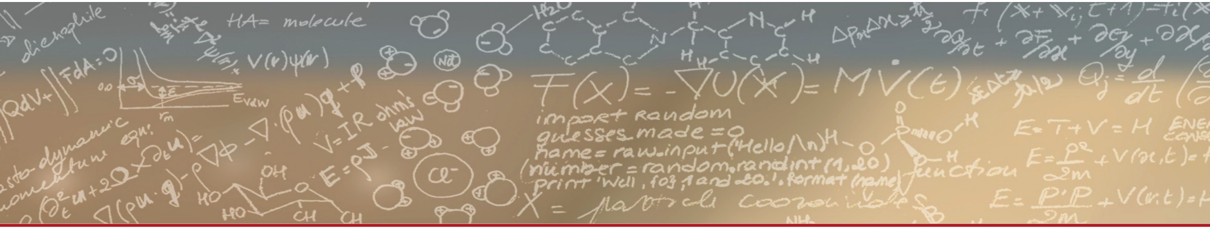
ETH zürich

Where to find more information

- CSCS User Portal: https://user.cscs.ch/tools/firecrest/
- The complete API: https://firecrest-api.cscs.ch/
- Source on Github: https://github.com/eth-cscs/firecrest/
  It includes a template client in Python and a demo environment in Docker.
- Documentation page and examples: https://firecrest.readthedocs.io
- Python library for the API: https://github.com/ekouts/pyfirecrest
- Product page: https://products.cscs.ch/firecrest/

CSCS
Centro Svizzero di Calcolo Scientifico
Swiss National Supercomputing Centre

ETH zürich



Thank you for your attention