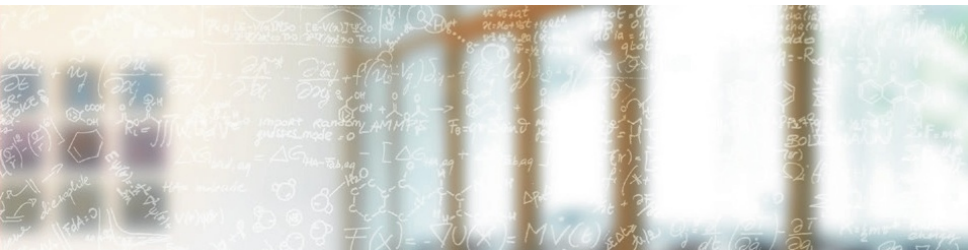




**CSCS**

Centro Svizzero di Calcolo Scientifico  
Swiss National Supercomputing Centre

**ETH** zürich



# FirecREST - a Web API for HPC at CSCS

CSCS User Lab Day - Meet the Swiss National Supercomputing Centre

Theofilos Manitaras - ETH Zurich/CSCS

September 9, 2019

# FirecREST a Web API for HPC at CSCS



- FirecREST in a Nutshell
- Microservice Architecture
- FirecREST Workflow
- API Endpoints & Demo client
- Why FirecREST?



**CSCS**

Centro Svizzero di Calcolo Scientifico  
Swiss National Supercomputing Centre

**ETH** zürich

# FirecREST in a Nutshell

---

# FirecREST Trivia



Firecrest in nature

# What is FirecREST?

FirecREST is a RESTful Services Gateway to HPC resources built with a microservices architecture.

- **Gateway to HPC:**

- Identity Access Management (IAM)
- HPC Workload Management
- Data Mover

- **RESTful interfaces:** Architecture abstraction allowing for simple, lightweight and fast applications.

- **Microservices:** Decouple functionalities found in monolithic server architecture.

# FirecREST Motivation

- Improve accessibility of HPC resources to scientific communities.
- Create an interface for the development of domain specific platforms.
- Develop a standard interface to Cray system resources.
- Take advantage of the CSCS Identity and Access Management so that users can use their credentials.



**CSCS**

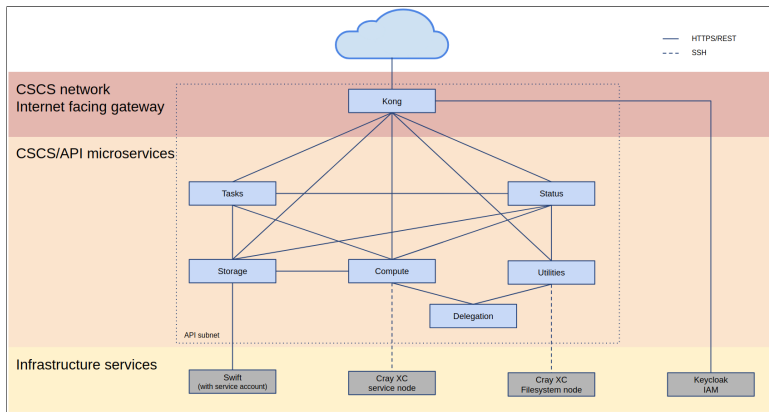
Centro Svizzero di Calcolo Scientifico  
Swiss National Supercomputing Centre

**ETH** zürich

# Microservice Architecture

---

# FirecREST Microservices Overview



FirecREST microservice architecture



# FirecREST Components

## Services

- Kong API gateway
- Compute
- Storage
- Utilities
- Support
  - Tasks
  - Status
  - Delegation

## Core CSCS Service Dependencies

- Workload Manager node
  - Compute
  - Xfer
- Utility node, filesystem utilities
- SWIFT, data transfer staging area
- Keycloak, authentication & authorization

# FirecREST Microservices

- **Kong:** Open-Source microservice API Gateway. Implements and enforces authentication, authorization, traffic control, analytics, logging.
- **Compute:** Non-blocking calls to workload manager for submitting/querying jobs. The service responds with a reference to a temporary task resource tracking the request state.
- **Storage:** Non-blocking calls to high-performance storage services. The service responds with a task reference as above.
- **Utilities:** Fundamental filesystem utilities. All calls are blocking with a timeout.
- **Support:**
  - Tasks: state of tasks
  - Delegation: OIDC token → SSH user-certificate
  - Status: information on services and infrastructure.



**CSCS**

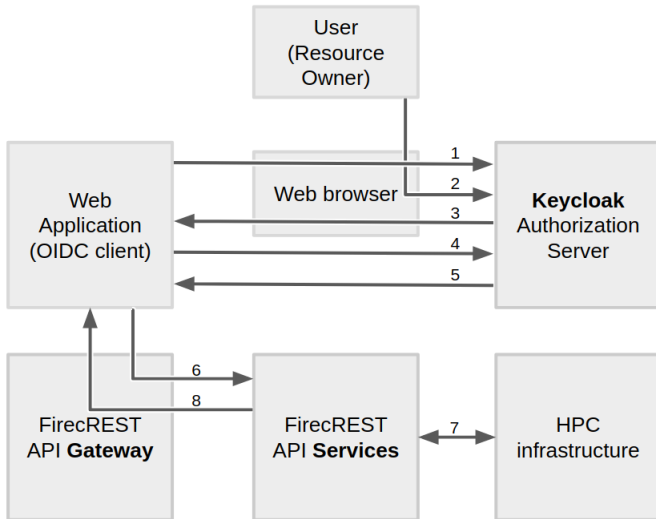
Centro Svizzero di Calcolo Scientifico  
Swiss National Supercomputing Centre

**ETH** zürich

# FirecREST Workflow

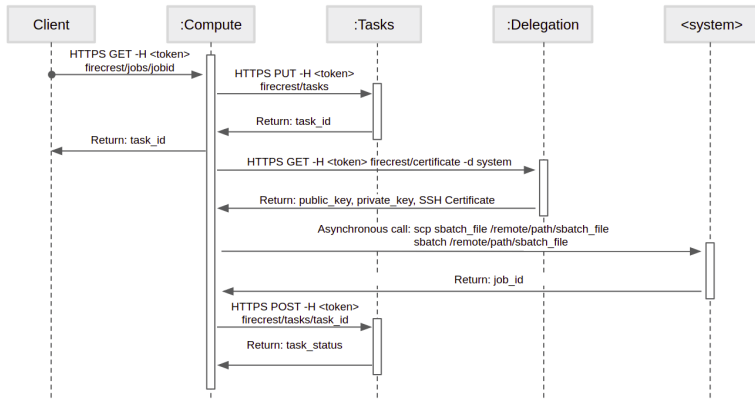
---

# Authentication and Authorization

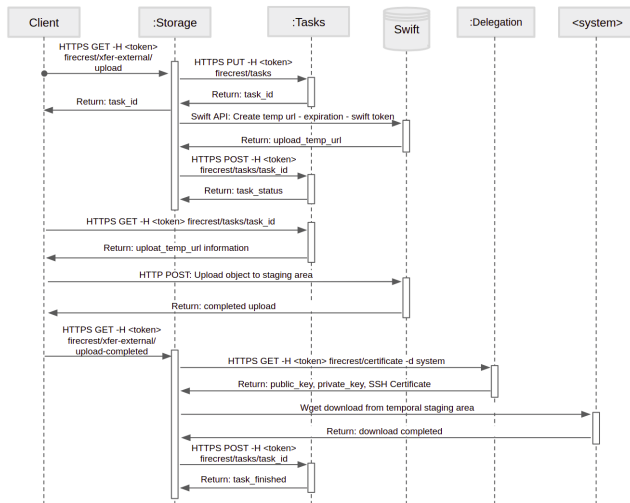


OIDC-based Authentication/Authorization

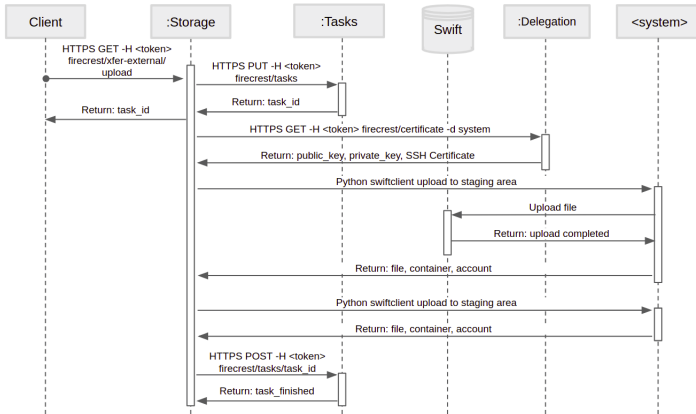
# Job Submission Workflow



# Data Upload Workflow



# Data Download Workflow





**CSCS**

Centro Svizzero di Calcolo Scientifico  
Swiss National Supercomputing Centre

**ETH** zürich

# API Endpoints

---



# The FirecREST API

## FirecREST API 0.9.9 QA53

[/firecrest-developers-api.yaml](#)

FirecREST platform, a RESTful Services Gateway to HPC resources, is a high-performance and reusable framework that integrates with existing HPC infrastructure, thus enabling the access to HPC resources to web-enabled services.

FirecREST provides a REST API that defines a set of functions through which developers can interact with using the HTTP/REST protocol architecture. Calls to the REST API received by the services gateway are translated into the appropriate infrastructure requests. Among the most prominent services that FirecREST exposes we find authentication and authorization, system status, file-system access, data mover, execution of parallel jobs, accounting information, etc.

Servers

▾

**Status** Status information of infrastructure and services. >

**Utilities** Basic system utilities. All calls are blocking and low-latency operations, maximum operation duration is limited by a timeout. >

**Compute** Non-blocking calls to workload manager to submit and query jobs. The service responds with a reference to the temporary task resource that tracks the state of the request. >

**Storage** Non-blocking calls to high-performance storage services. The service responds with a reference to the temporary task resource that tracks the state of the request. >

**Tasks** Access status and response of compute and storage tasks. >

## Overview of the FirecREST API

# Status

<b>Status</b> Status information of infrastructure and services. <span>▼</span>	
GET	/status/services List of services
GET	/status/services/{servicename} Get service information
GET	/status/systems List of systems
GET	/status/systems/{machinename} Get system information.

The **Status** endpoints

# Displaying the status of the systems

Home <span>Log Out</span>	
Displaying status of the CSCS systems	
System	Status
dom	available
daint	available

System status using endpoint: `/status/system/{machinename}`

# Utilities

Utilities <small>Basic system utilities. All calls are blocking and low-latency operations, maximum operation duration is limited by a timeout.</small>			▼
GET	/utilities/{machinename}/ls	List directory contents	
POST	/utilities/{machinename}/mkdir	Creates a directory	
POST	/utilities/{machinename}/rename	Rename/move a file, directory, or symlink	
POST	/utilities/{machinename}/chmod	Change file mode bits	
POST	/utilities/{machinename}/chown	Change file owner and group	
GET	/utilities/{machinename}/file	determine file type	
POST	/utilities/{machinename}/symlink	Create a symlink	
GET	/utilities/{machinename}/download	Download a small file	
POST	/utilities/{machinename}/upload	Uploads a small file	

## The **Utilities** endpoints

# Upload small file

[Home](#)[Log Out](#)

In this page you can upload your files to CSCS

**destination**

/users/manitart/my\_dir

**file**

my\_file

**machine**

daint ▾

Upload a small file using: `/utilities/{machinename}/upload`

# List User Files

Home <span>Log Out</span>	
Listing files for /users/manitart/reframe_dev_public	
Name	Size
CONTRIBUTING.md	423
Jenkinsfile	9708
LICENSE	1543
README.md	4227
bin	4096
ci-scripts	4096
config	4096
cscs-checks	4096
docs	4096
reframe	4096
reframe.log	60735
reframe.py	101
requirements.txt	34
setup.cfg	68
setup.py	1045
test_reframe.py	769
tutorial	4096

List user directory using: `/utilities/{machinename}/ls`

# Compute

<b>Compute</b>	Non-blocking calls to workload manager to submit and query jobs. The service responds with a reference to the temporary task resource that tracks the state of the request.	✓
POST	/jobs/{machine}	Submit Job
GET	/jobs/{machine}	Retrieves information from all jobs
GET	/jobs/{machine}/{jobid}	Retrieves information from a job
DELETE	/jobs/{machine}/{jobid}	Delete Job
GET	/jobs/{machine}/acct	Job account information

## The **Compute** endpoints

# Submitting a Job

[Home](#)[Log Out](#)

In this page you can upload sbatch files to CSCS

file

simple\_sbatch.sh

machine

daint

Submit an sbatch script using: `/jobs/{machine}`



# Monitor active jobs

Home <span>Log Out</span>		
Listing Active Jobs		
Jobid	State	Name
16427851	PENDING	FirecREST_test
16427854	PENDING	FirecREST_test
16427861	PENDING	FirecREST_test

Monitor status of active jobs using: `/jobs/{machine}`

# Storage

<b>Storage</b> Non-blocking calls to high-performance storage services. The service responds with a reference to the temporary task resource that tracks the state of the request. <span>▼</span>	
GET	/storage/xfer-internal/rsync rsync
GET	/storage/xfer-internal/mv move (rename) files
GET	/storage/xfer-internal/cp copy files and directories
GET	/storage/xfer-internal/rm remove files or directories
GET	/storage/xfer-external/upload Upload a file
GET	/storage/xfer-external/upload-finished/{task_id} Upload to Object Storage finished notification
GET	/storage/xfer-external/download Download a file

## The **Storage** endpoints

# Upload a large file

[Home](#)[Log Out](#)

## In this page you can upload your files to CSCS

Enter the target path

Enter the source path

Upload file using: `/storage/xfer-external/upload`

# Tasks

Tasks Access status and response of compute and storage tasks. <span>▼</span>		
GET	/tasks/tasks	returns all tasks
PUT	/tasks/tasks	Creates a task
GET	/tasks/tasks/{id}	task status information
POST	/tasks/tasks/{id}	Updates a task
DELETE	/tasks/tasks/{id}	Delete task

## The **Tasks** endpoints

# Displaying FirecREST tasks

Home Log Out	
Displaying FirecREST tasks	
Task ID	Content
001b4db8c445f85a0526088bec51e0e5	{'data': 'No queue slurm jobs', 'description': 'Finished sucessfully', 'hash_id': '001b4db8c445f85a0526088bec51e0e5', 'last_modify': '2019-08-26T13:20:58', 'service': None, 'status': '200', 'user': 'manitart'}
03ed1d0c651cd23e85f34e4321e42a31	{'data': {'hash_id': '03ed1d0c651cd23e85f34e4321e42a31', 'msg': 'curl -i -X POST -F max_file_size=5368709120 -F max_file_count=1 -F expires=1569592916 -F -F redirect= -F file=@/home/manitart/my_large_file', 'system': '127.0.0.1:2222', 'target': '/users/manitart/my_dir', 'user': 'manitart'}, 'last_modify': '2019-08-26T13:20:58', 'service': None, 'status': '111', 'user': 'manitart'}
03fade1dd511afc1067feb6b00dbe5b	{'data': 'Submitted batch job 764657', 'description': 'Finished sucessfully', 'hash_id': '03fade1dd511afc1067feb6b00dbe5b', 'last_modify': '2019-08-26T13:20:58', 'service': None, 'status': '200', 'user': 'manitart'}
05f97642c2c7c5b939ab4fcfb2f54760	{'data': 'No queue slurm jobs', 'description': 'Finished sucessfully', 'hash_id': '05f97642c2c7c5b939ab4fcfb2f54760', 'last_modify': '2019-08-26T13:20:58', 'service': None, 'status': '200', 'user': 'manitart'}
087460e13635a0abca721792a280f4e2	{'data': 'No queue slurm jobs', 'description': 'Finished sucessfully', 'hash_id': '087460e13635a0abca721792a280f4e2', 'last_modify': '2019-08-26T13:20:58', 'service': None, 'status': '200', 'user': 'manitart'}

Display all tasks using: `/tasks/tasks`



**CSCS**

Centro Svizzero di Calcolo Scientifico  
Swiss National Supercomputing Centre

**ETH** zürich

# Why use FirecREST?

---

# FirecREST Advantages

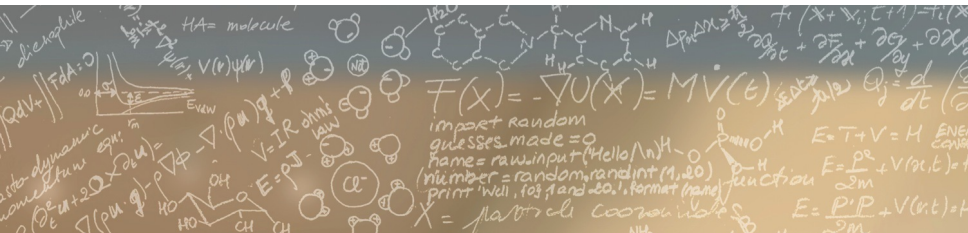
- Common, stable, maintainable API
- Enforce that all API requests are authenticated
- Applications never manipulate user credentials
- Only allow requests from registered applications
- User-managed access permissions per application
- Stateless security module by use of tokens (OIDC)
- Enables managing of execution workloads
- Enables external data transfer from/to HPC filesystems
- Soon to be open-sourced



**CSCS**

Centro Svizzero di Calcolo Scientifico  
Swiss National Supercomputing Centre

**ETH** zürich



**Thank you for your attention.**