Serverless computing endpoint based on AWS container cluster

Rafał Grzymkowski, INP PAS
The 8th DIRAC Users' Workshop
22-25 May 2018, CC-IN2P3, Lyon

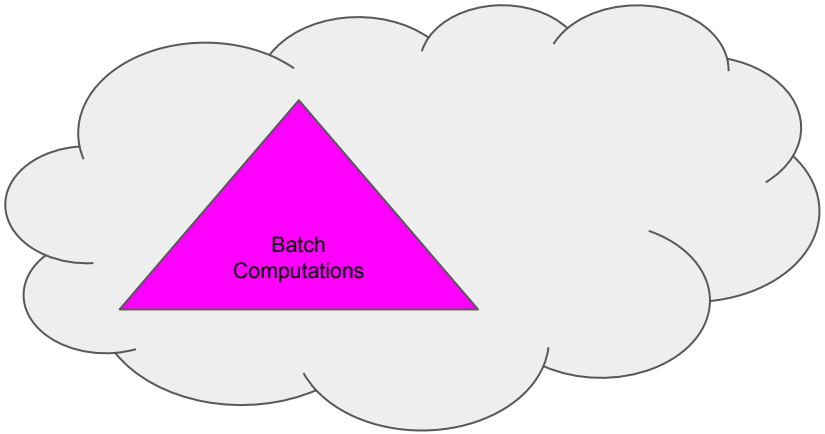
What is Serverless application?

Application based on cloud native services. We don’t need to wory about VMs management, security patches, scaling, all this is out of box.

- Function-as-a-Service (FaaS)
- HTTP API service
- Object Storage
- Job Scheduler based on Container Clusters

The Idea

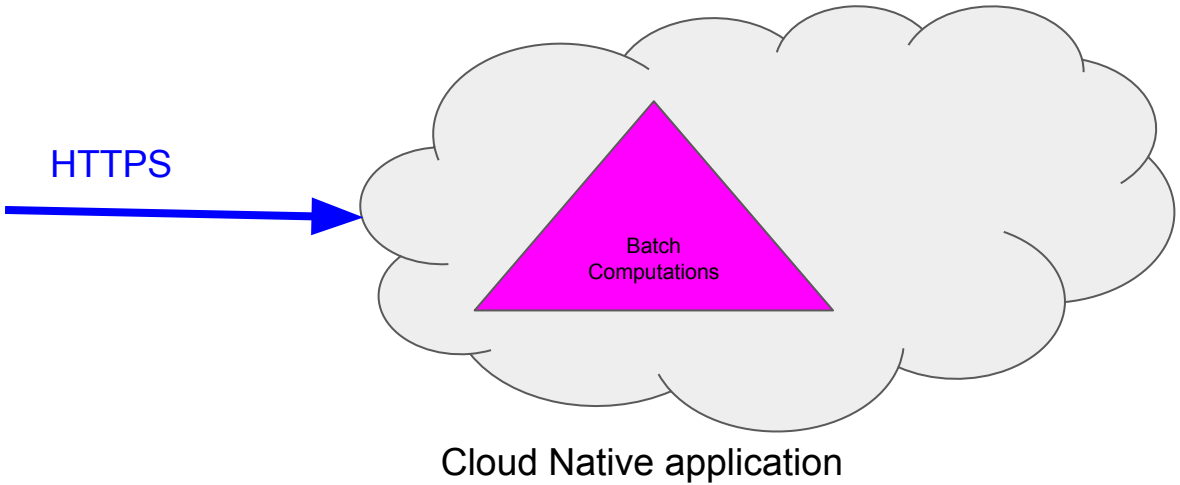
- To build a grid compatible computing endpoint based on cloud navite architecture.



Cloud Native application

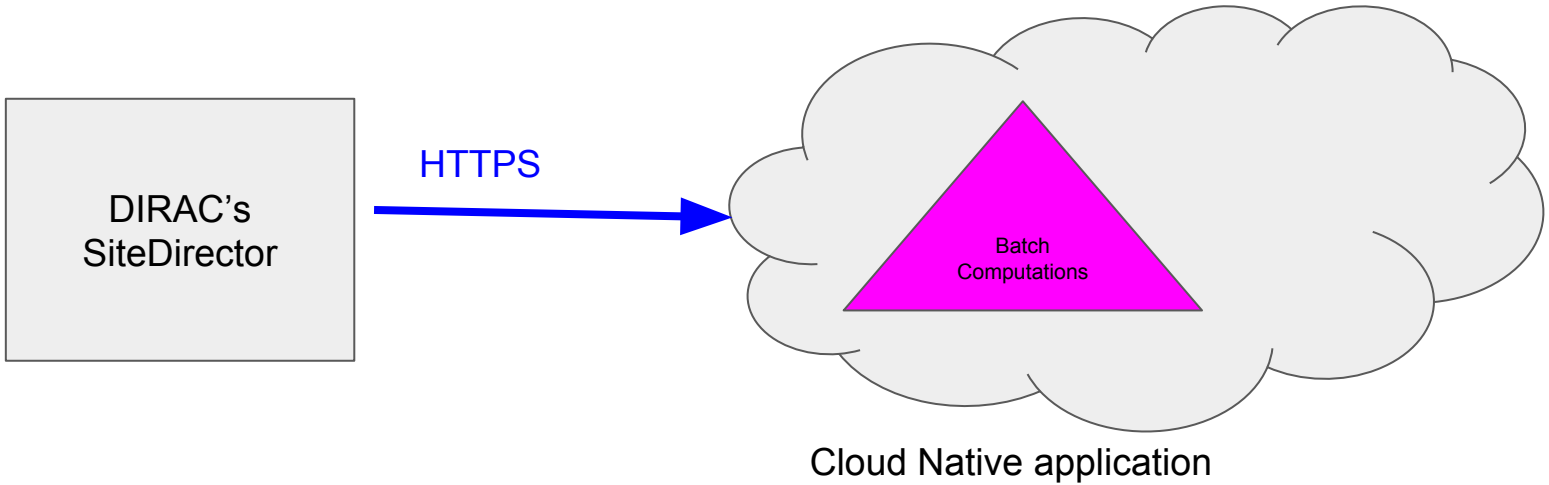
The Idea

- To build a grid compatible computing endpoint based on cloud native architecture.
- With REST interface.

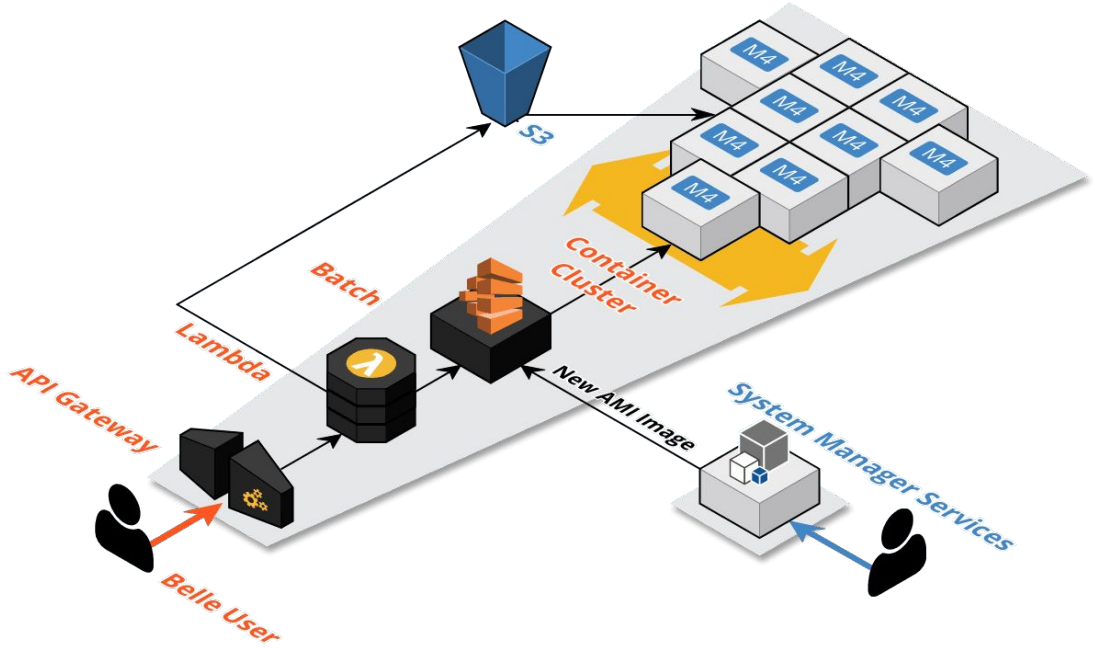


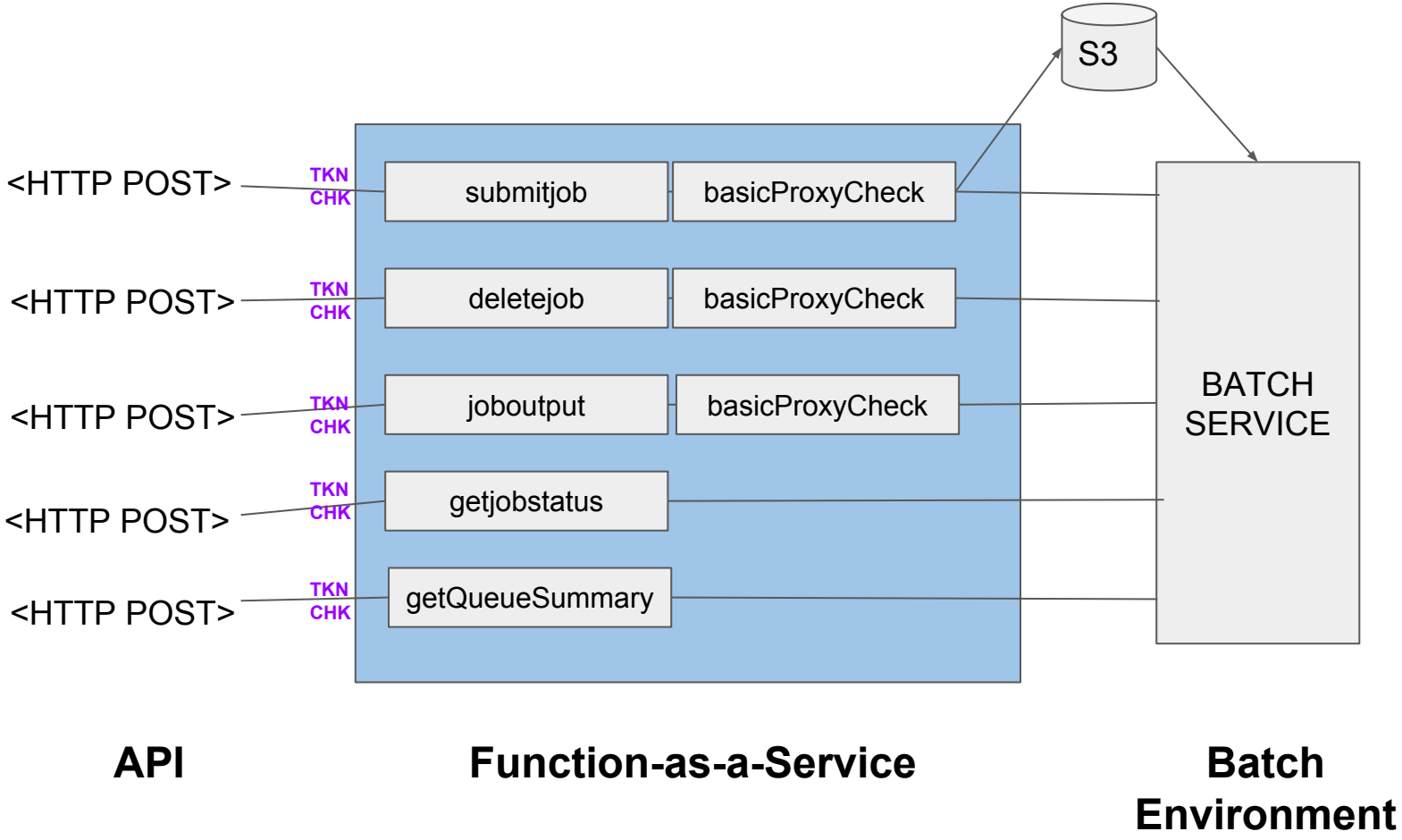
The Idea

- To build a grid compatible computing endpoint based on cloud native architecture.
- With REST interface.
- So we can map pilot jobs straight to AWS Batch jobs (scaling based on job queue, not on resources)



Serverless application architecture




<HTTP POST>
<HTTP POST>
<HTTP POST>
<HTTP POST>
<HTTP POST>
TKN CHK
TKN CHK
TKN CHK
TKN CHK
TKN CHK
submitjob
deletejob
joboutput
getjobstatus
getQueueSummary
basicProxyCheck
basicProxyCheck
basicProxyCheck
S3
BATCH SERVICE
API
Function-as-a-Service
Batch Environment

HTTP request example

```
curl
        -X POST
        -H "Content-Type: application/json"
        -H  "x-api-key: ZG4hUuDa56tOo8Rxu03o911"
        --data
                {
                  'proxyFileEncoded':encoded_proxy,
                  'wrapperFileEncoded':encoded_wrapper
                }
        https://api.cloud4belle2.net/submitjob?queue=belle2prod&job_name=17584eab645e4b5
```

Container bootstrapping

Bootstrap process:

- Install packages for the CentOS (gcc, sudo, awscli, …)
- Get proxy file
- Get pilot wrapper file
- Configure CVMFS
- Check proxy
- Run pilot wrapper as a sudo process

Security

- HTTPS as a transport layer.
- API Key (token in HTTP header).
- Grid proxy validation in the FaaS layer
- and in the container environment.

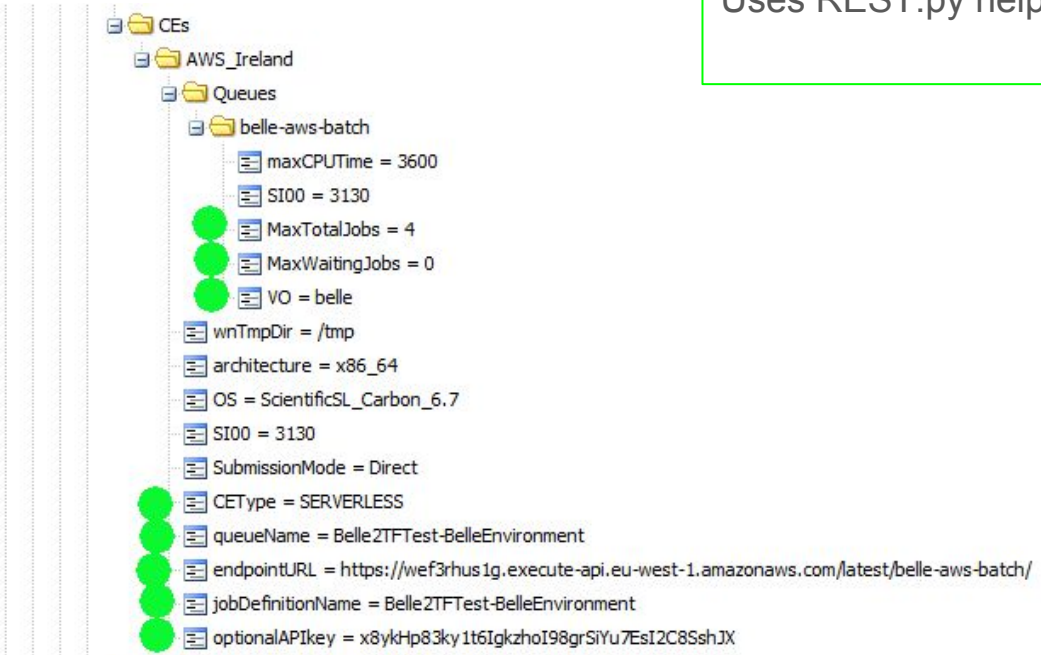
New ways of authentication can be easy applied!

ServerlessCE

ServerlessCE is a new type of ComputingElement for the DIRAC.

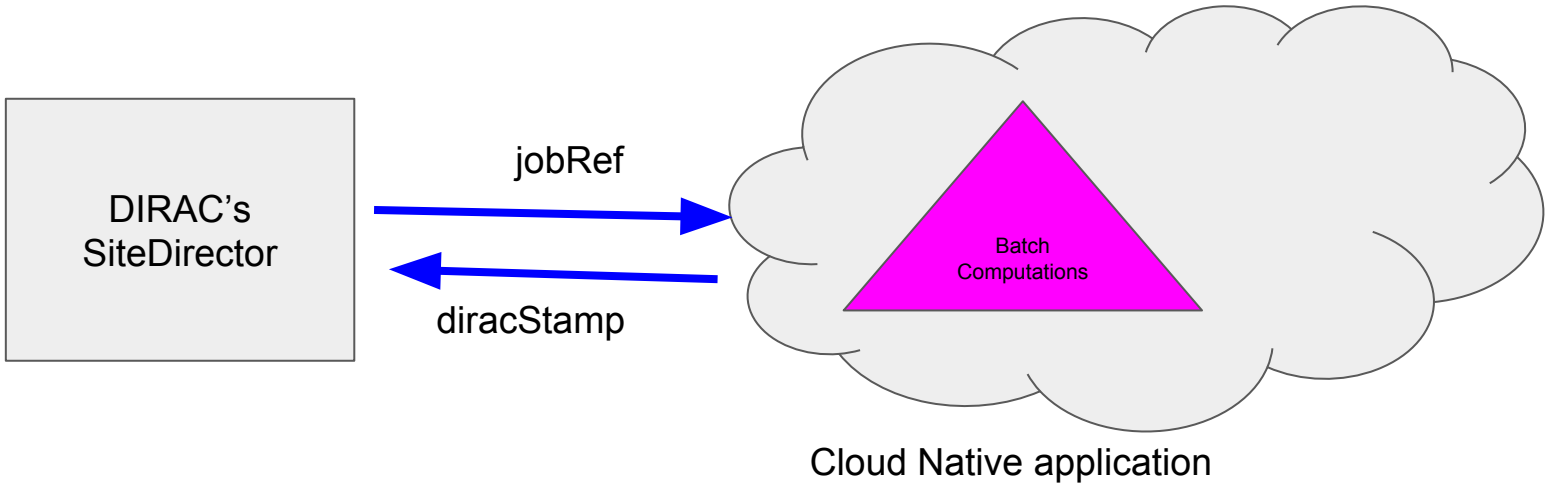
Uses REST.py helper based on ‘requests’ package.

Configuration in the DIRAC:



Passing IDs and limitations in DIRAC's DB

jobRef example: api.cloud4belle2.net/17584eab645e4b56a0a01a0ef29bf532
diracStamp example from AWS container: f57ccba4-61d9-42c6-a2a1-b538a9d2bb29



Costs of the serverless application

API Gateway - our case < $0.20 ($3.50 per million requests).

AWS Lambda - first million request for free, rest charged at $0.20 per million.

CloudWatch - optional, place to keep container output (pilot logs) and F-a-a-S output.

Route53 - optional, $0.50 per domain

Total cost < $1 per month

Summary

Container clusters can process Belle II **production jobs** smoothly.

We don’t need to use grid host cert but **proxy cert** like for standard pilot.

With the standard DIRAC’s site config we can manage of **number of running pilots** and **scaling-out speed** (cycle time for a SiteDirector agent).

Further steps: Use same DIRAC’s CE with **Google Cloud Enginge**.