# Future Cloud Applications #2

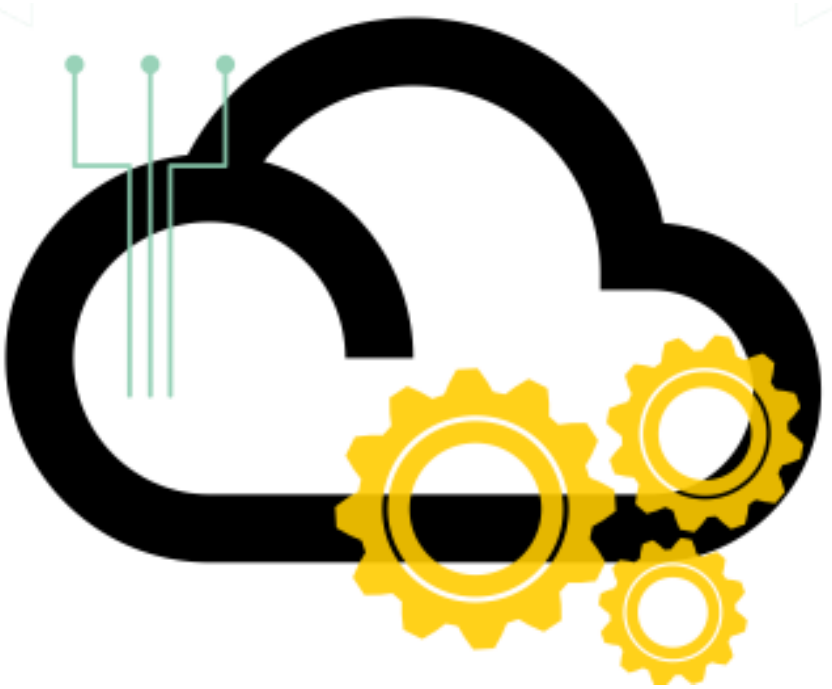
# More on FaaS: The Swiss Army Knife of Serverless Computing

Josef Spillner <josef.spillner@zhaw.ch>
Service Prototyping Lab (blog.zhaw.ch/icclab)

Apr 27, 2017 | Future Cloud Applications

# This Event

Future Cloud Applications
- link academic research ← → industry
- highly technical
- no specific technology preference
  - for this, other meetups exist
- ~every 2-3 months
  - to be calibrated



Participation
- meetup group (55+ people)
- open for all
- open for talk proposals!

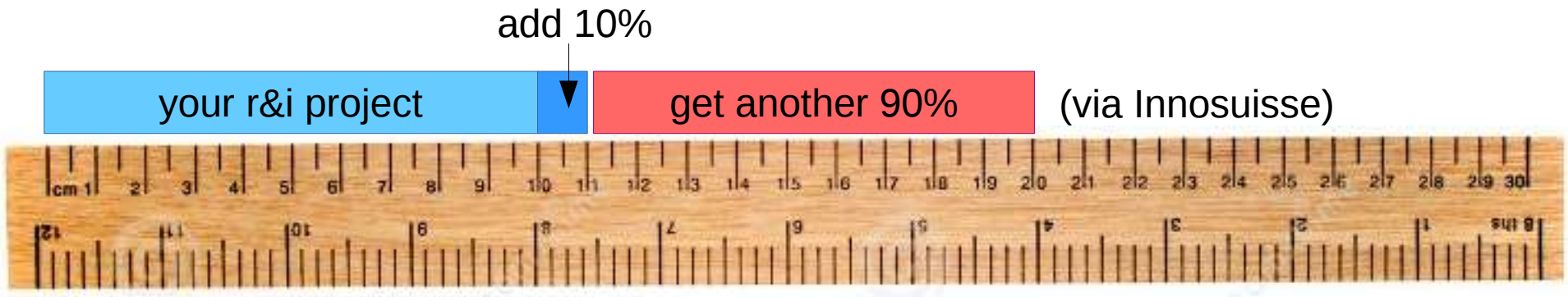

[cyprus-property-buyers.com (no, really)]

ICCLAB SPLAB Service Engineering Research Area

# Service Prototyping Lab

Research approach

- ambitious long-term research initiatives
  - cloud-native applications, service tooling, cloud accounting & billing
- transfer of results into Swiss companies




Publishing approach

- preprint-first - rapid dissemination
- open source, Labsite
- blog posts, events

ICCLAB SPLAB

# Service Prototyping Lab + ICCLab


Service Engineering Research Area
ICCLAB SPLAB

# The FaaS Space

AWS Lambda

Google Cloud Platform
Functions

OpenWhisk

Azure

Webtask.io

Hook.io

PyWren

Zappa

Serverless Framework

Apex

Kubeless

Funktion

Fission

Picasso

Effe

Docker-LambCI

OpenLambda

Lever OS

X-Ray

Step Functions

ICCLAB SPLAB

# FaaS Examples

AWS Lambda:

```
def lambda_handler(event, context):
    “‘
    event: dict
    context: meta information object
    returns: dict, string, number, ...
    “‘
    # ...
    return “result“
```

OpenWhisk:

```
def handler(input):
    “‘
    input: dict
    returns: dict
    “‘
    # ...
    return {}
```

Fission:

```
def main():
    “‘
    input: via flask.request.get_data()
    returns: str
    “‘
    # ...
    return “result“
```

Azure:

```
def main():
    from AzureHTTPHelper import\
    HTTPHelper
    input = HTTPHelper().post
    # ...
    open(os.environ[“res“], “w“).write(\
    json.dumps({“body“: “...“}))
main()
```

Further differences:

- function scoping (e.g. with/without export in JavaScript)
- function naming (mangling on client or service side)

ICCLAB SPLAB Service Engineering Research Area

# Our Tools for FaaS

Podilizer
(Java)

Lambada
(Python)

today

Web2Cloud
(JavaScript)


Lambackup
(file backups)

Lama
(relational data)

Snafu
(FaaS host)

today

ICCLAB SPLAB

# Lambada

Definition of “FaaSification“

→ Process of automated decomposition of software application into a set of deployed and readily composed function-level services.

FaaSification := code analysis + transformation + deployment + on-demand activation

Integration Categories:
- generic (code/function unit generation)
- single-provider integration
- multi-provider integration

Decomposition Categories:
- static code analysis
- dynamic code analysis

→ Lambada: FaaSification for Python

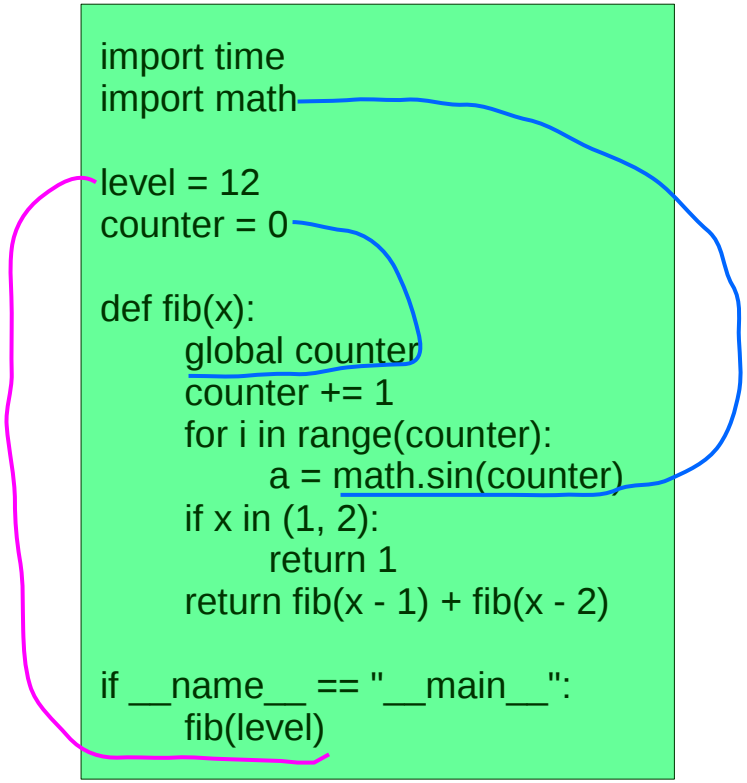
ICCLAB SPLAB Service Engineering Research Area

# Lambada

Code Analysis

Dependencies
- imported modules
- global variables
- dependency functions
  - defined in other module
  - defined in same module

Input/Output
- printed lines
- input statements
  - tainting
  - stateful function splitting



# Lambada

Code Transformation

Rewrite rules, via AST:

```
return 9                  print(“hello“)                    local_func()
------------------        return 9                          ---------------------
return {“ret“: 9}         ----------------------------------- local_func_stub()
                          return {“ret: 9“, “stdout“: “hello“}
```

Stubs, via templates:

```
def func_stub(x):
    input = json.dumps({“x“: x})
    output = boto3.client(“lambda“).invoke(FN=“func“, Payload=input)
    y = json.loads(output[“Payload“].read().decode(“utf-8“))
```

# Lambada

Code Transformation

Stateful proxies for Object-Oriented Programming:

```
class Test:                  →   class Proxy:
    def __init__(self):              def __new__(cls, clsname, p=True):
        self.x = 9                       if p: # __new__ must return callable
                                             return lambda: Proxy(clsname, False)
    def test(self):                  else:
        return self.x * 2                    return object.__new__(cls)

                                     def __init__(self, clsname, ignoreproxy): ...
                                     def __getattr__(self, name): ...
```

→ Test becomes Proxy(“Test“), Test() then invokes proxy
→ test() becomes remote_test({“x“: 9}) through network proxy class
→ automatically upon import of class

# Lambada

Code Deployment + Activation

```
$ lambada [--local] [--debug] [--endpoint <ep>] <file.py>
$ python3 -m lambada <file.py>

>>> import lambada
>>> lambada.move(globals() [, endpoint="..."])
```

Local mode: source code modified locally as copy

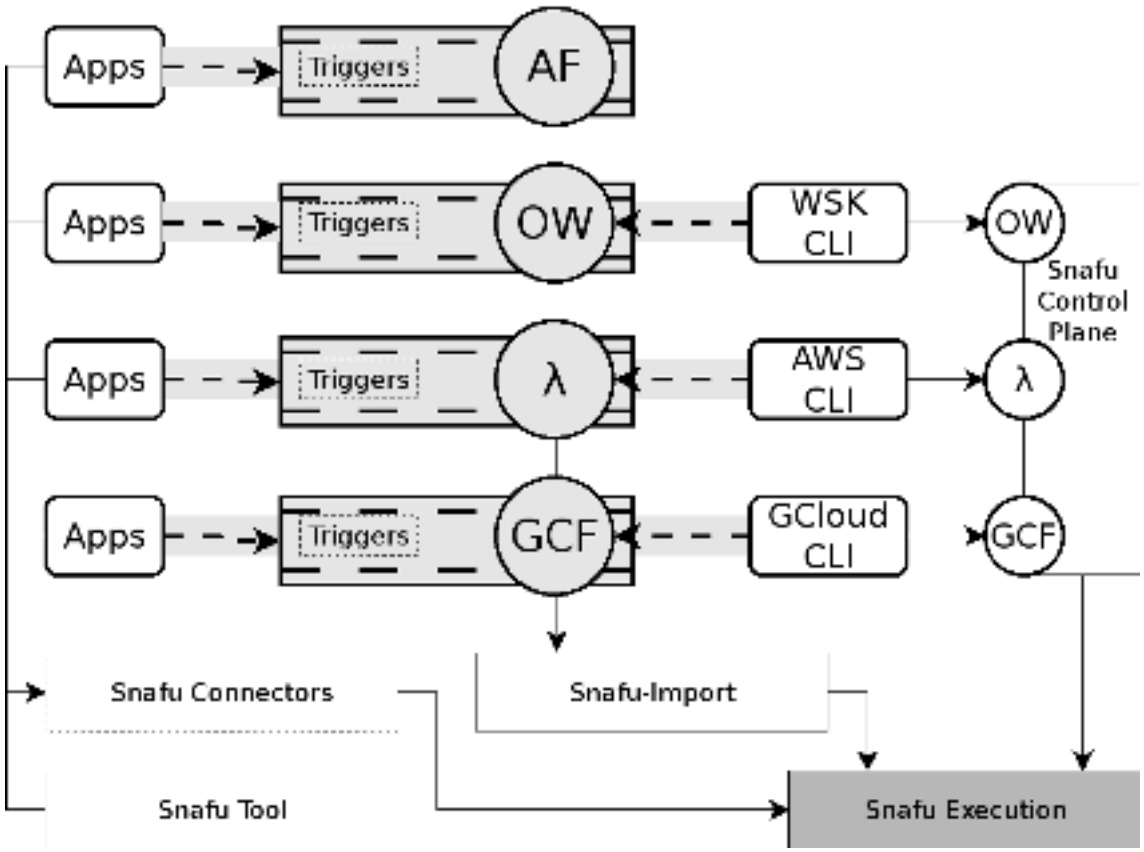
Remote mode: rewritten source code deployed and invoked

# Lambada - Demo Time!

[d0wn.com]

# Snafu

The Swiss Army Knife of Serverless Computing



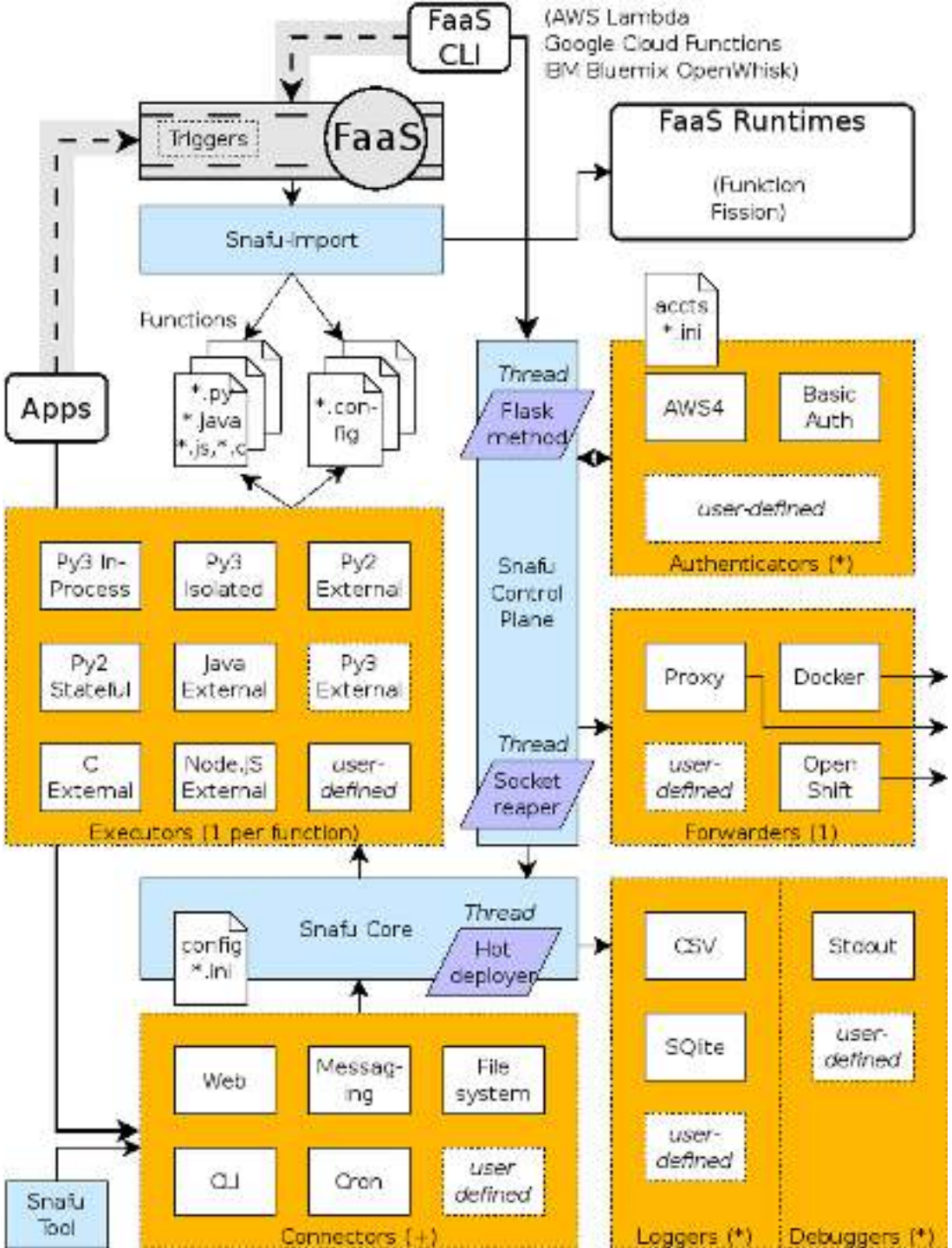
# Snafu

Current Implementation

- scalable from developer single instance to multi-tenant deployments
- executes Python 2 & 3, Java, JavaScript, C
- integrates with FaaS ecosystem at-large
- extensible subsystems

SLOC: ~1800
(including subsystems: ~800)

$ pip install snafu
$ docker run -ti jszhaw/snafu



# Snafu

Standalone mode

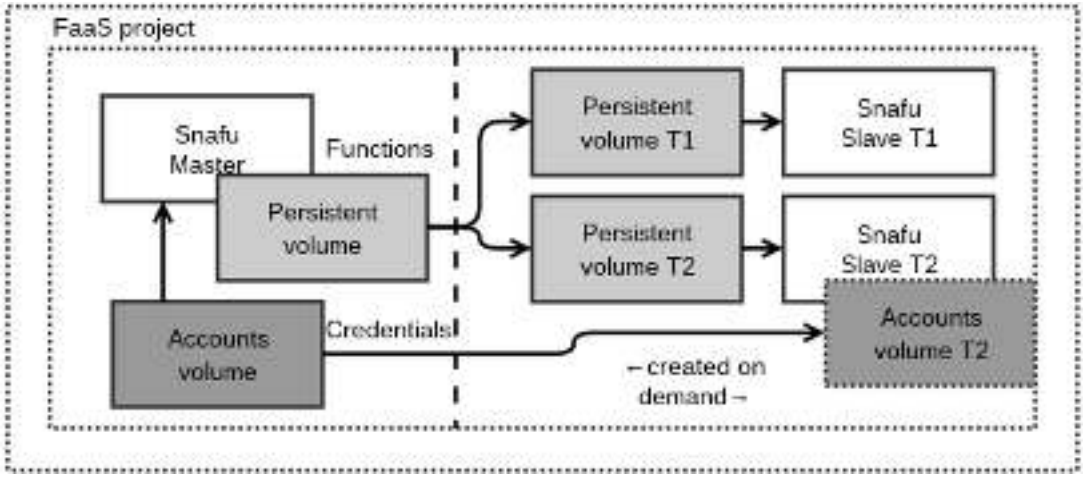
- call functions interactively
- batch mode with/without input pipe
- performance, robustness & correctness tests
- development

```
$ snafu

$ snafu -x <function> [<file/dir>]

$ snafu -l sqlite -e java -c lambda -C messaging
```

# Snafu

Daemon mode

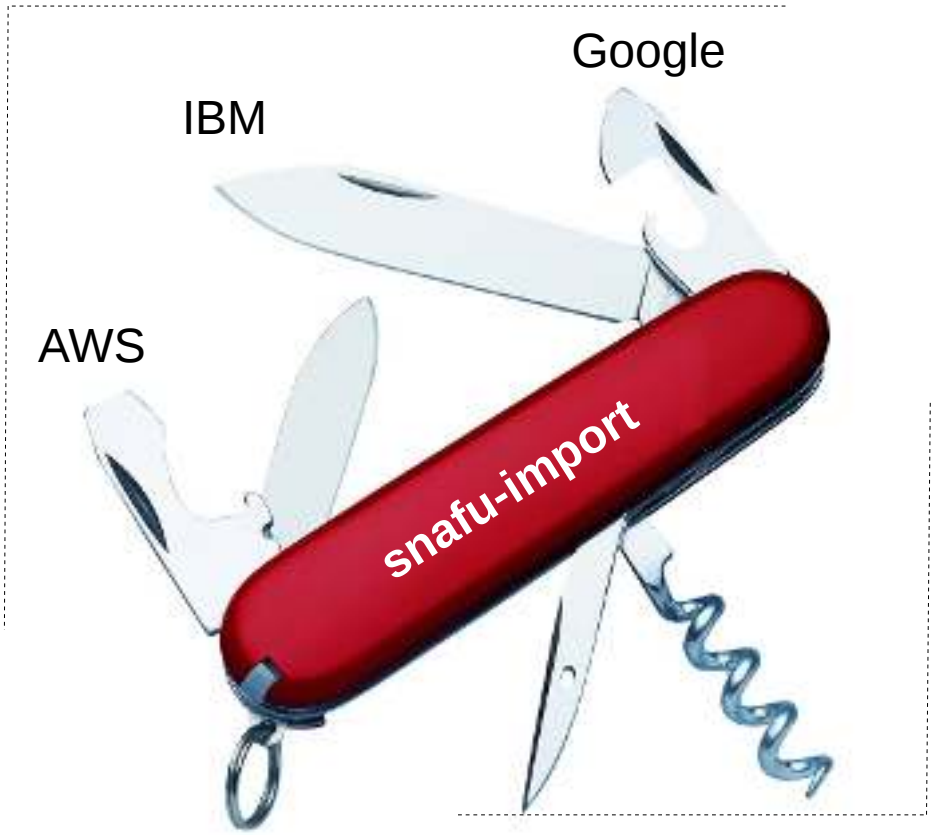
- hosted functions
- multi-tenant provisioning
- per-tenant isolation
- compatibility with existing client tools




```
$ snafu-control

$ snafu-control -a aws -r -d -e docker

# snafu-accounts --add -k <k> -s <s> -e <ep>
```

# Snafu

Integration into the wider FaaS ecosystem



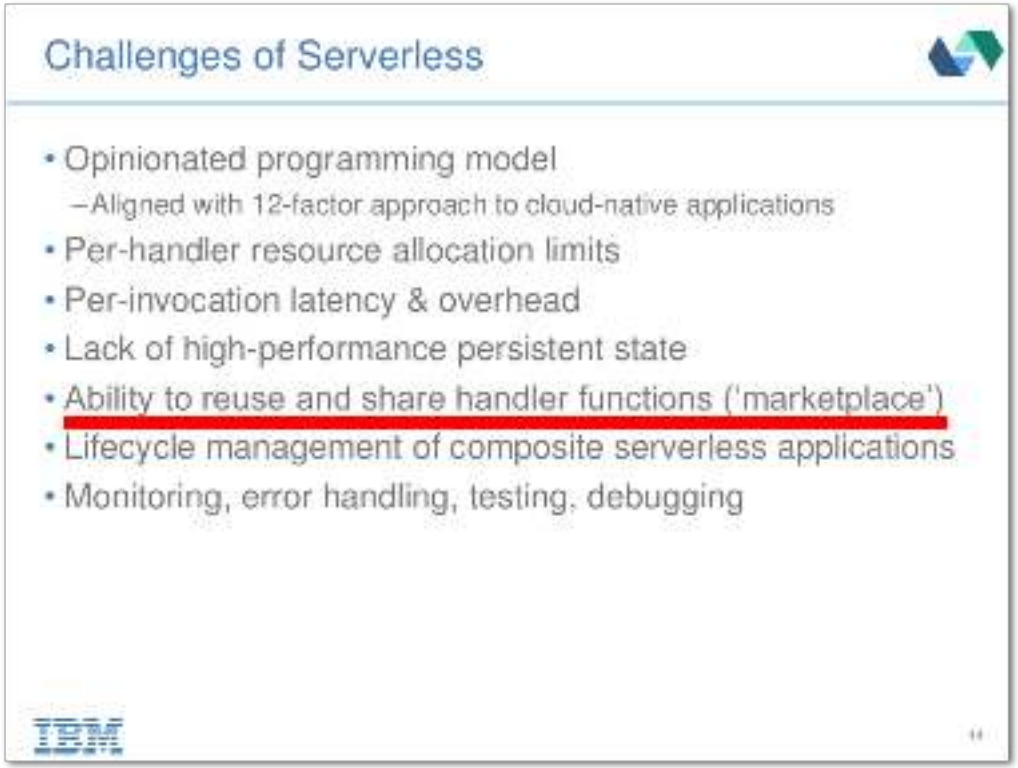

```
$ snafu-import \
--source <s> \
--target <t>
```

```
$ alias aws="aws \
--endpoint-url \
http://localhost:10000"

$ wsk property set \
--apihost \
localhost:10000

$ ./tools/patch-gcloud
```

# Snafu - Demo Time!


[pinterest.com]

# One more thing...




[slideshare.net/Alex Glikson]

:-)

*Full demo @ Open Cloud Day 14.06.2017 in Bern*

# Further Reading and FaaS Fun

Lama, Lambackup:

- https://arxiv.org/abs/1701.05945

Podilizer:

- https://arxiv.org/abs/1702.05510

Snafu:

- https://arxiv.org/abs/1703.07562

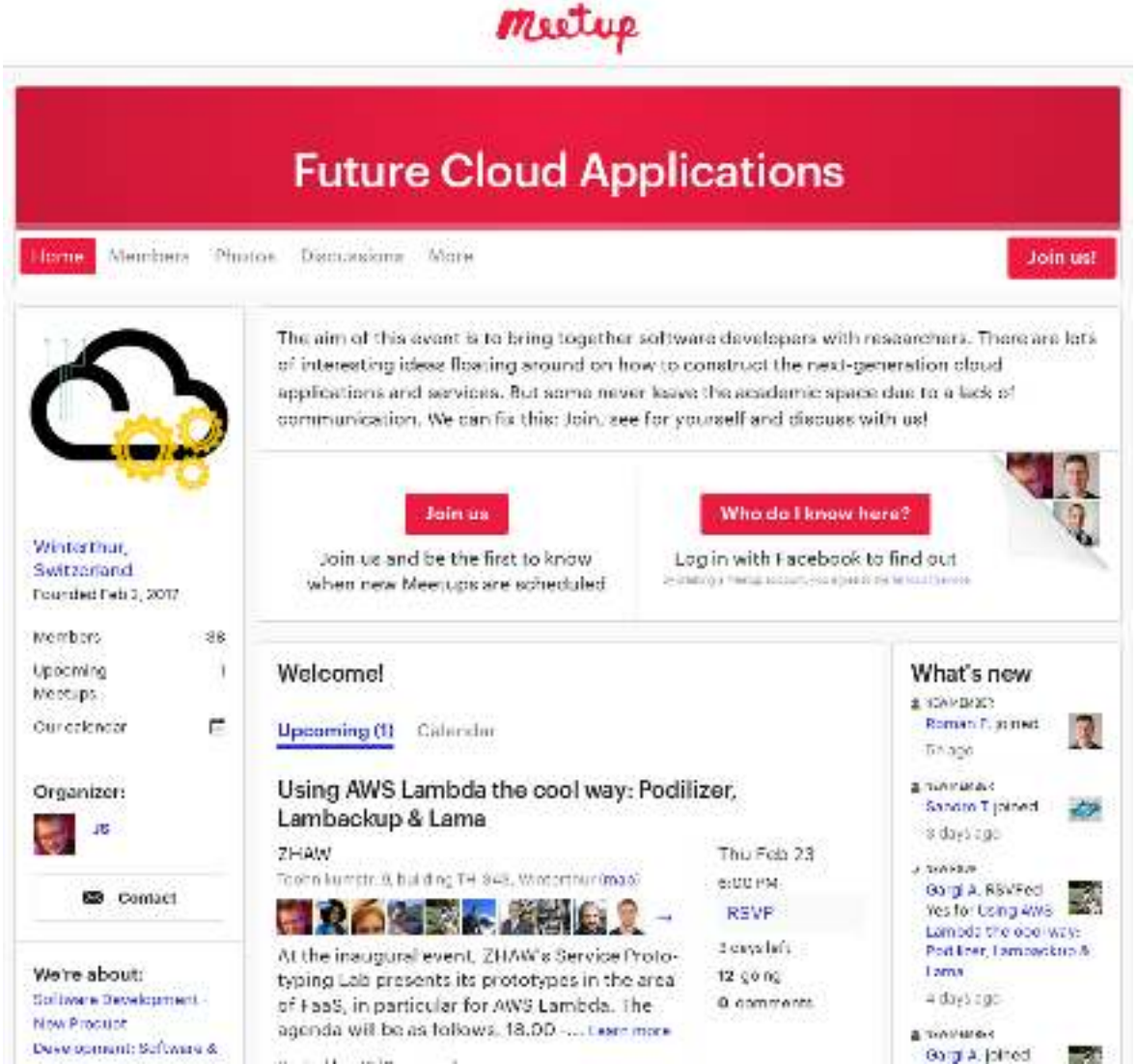
On arXiv Analytics:

On GitHub:

[github.com/
serviceprototypinglab]

# Next Future Cloud Applications Event!

Our suggestion: around July 2017... topic: elasticity boundaries for compositions of stateful and stateless microservices




ICCLAB SPLAB Service Engineering Research Area