Review Open Call SME-2

Extended Stress tests for asvin.io

Rahul Karade

asvin.io

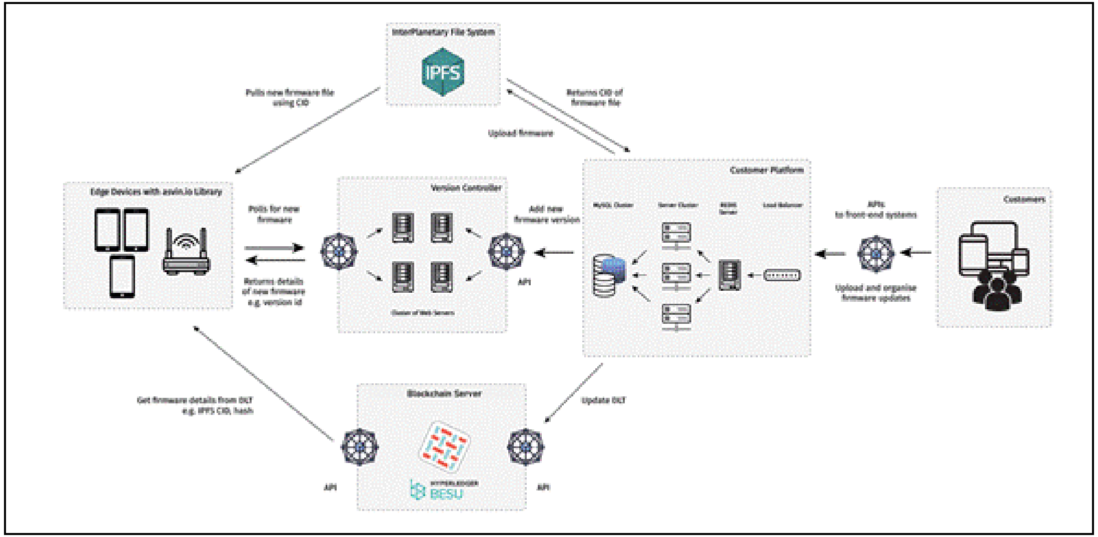
Virtual Review Meeting Open Call Experiments

19th November 2020

asvin.io

HEALING THE INTERNET OF THINGS

Concept


InterPlanetary File System
IPFS
Pulls new firmware file using CID
Returns CID of firmware file
Upload firmware
Edge Devices with asvin.io Library
Polls for new firmware
Returns details of new firmware e.g. version id
Version Controller
Cluster of Web Servers
Add new firmware version
API
Customer Platform
MySQL Cluster
Server Cluster
REDIS Server
Load Balancer
APIs to front-end systems
Upload and organise firmware updates
Customers
Get firmware details from DLT e.g. IPFS CID, hash
Blockchain Server
HYPERLEDGER
BESU
API
API
Update DLT

Comprized of 4 components

- Customer Platform
- Blockchain
- IPFS
- Version Controller

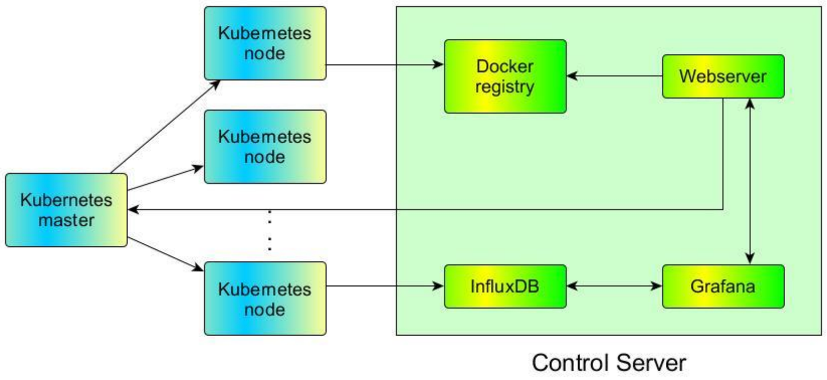
Objectives

- Evaluation of scalability, resilience and performance
- Generate stressed conditions for the platform
- Examine feasibility of PUF, Homomorphic and Functional encryption
- Find pain points of the platform
- Optimize platform configurations by iterative experiments

Background & Motivation

- Generate high traffic for the platform using IoT device Simulation
- Prepare asvin platform for the market release
- Verification and validation of the architecture
- The success of the stage 1 experiment
- Fed4Fire+ experiments are cost effective
- Diversified resources on Fed4FIRE+ testbeds

Experiment Setup



FEATURES

- Kubernetes cluster of 150 nodes
- Control server to build and deploy docker images
- Grafana to visualize analytics
- InfluxDB to store time series

https://lib.ugent.be/catalog/rug01:002494719

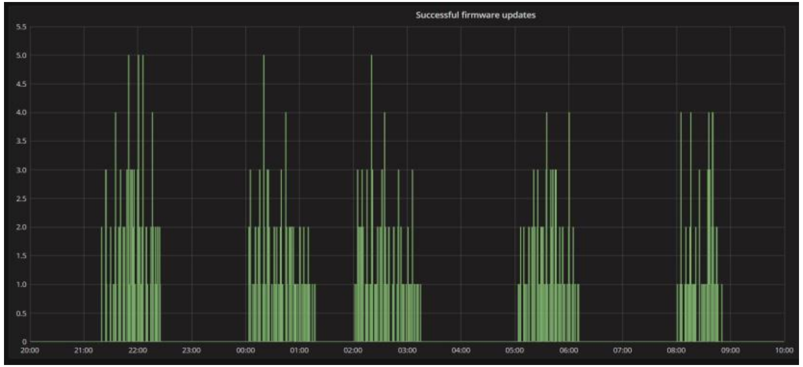
Results

- The asvin platform performed exceptionally well in stressed conditions.
- The device registrations on the Blockchain, rollout management on customer platform and firmware distributing on IPFS were smooth and seamless.
- The SRAM based PUF solution is reliable for generating secured keys for cryptographic applications. The Homomorphic and functional encryptions need improvements.
- The asvin platform is adaptable to network latency and bandwidth limitation

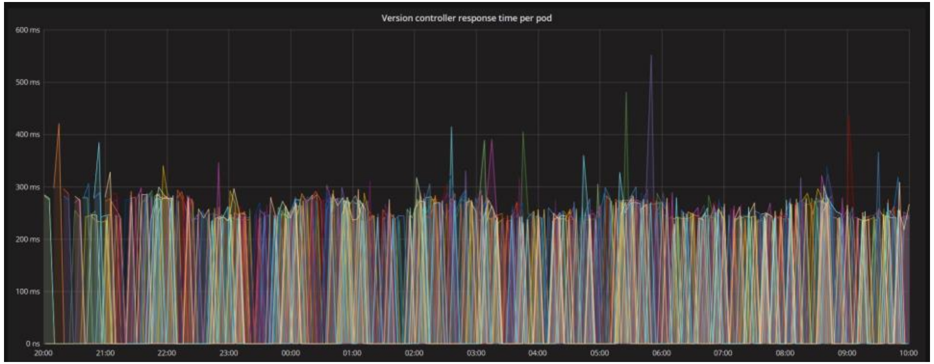
Analytics

FIRMWARE UPDATE

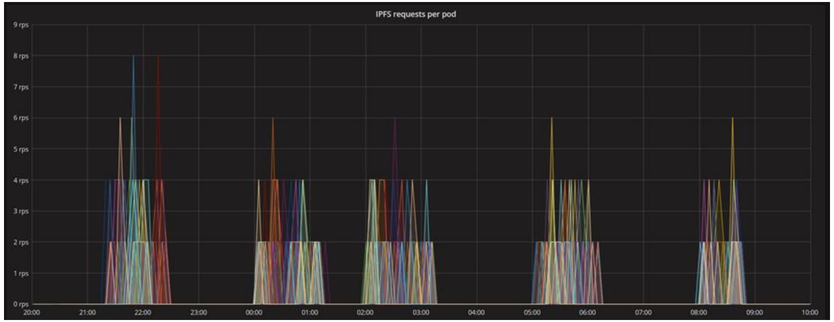
RESPONSE TIME



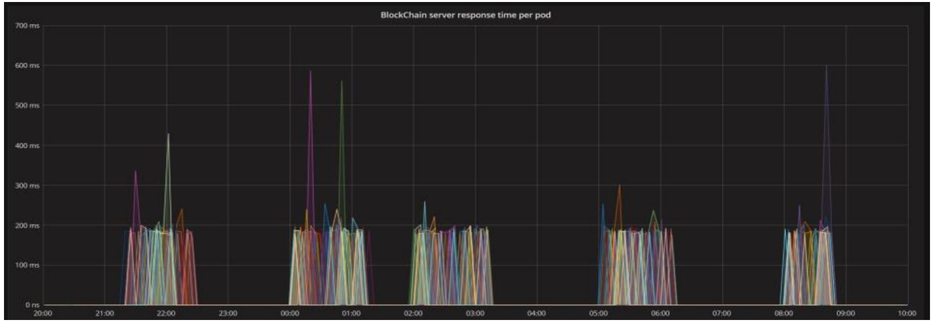
Successful firmware updates



Version controller response time per pod



IPFS requests per pod



BlockChain server response time per pod

Lessons Learned

- The distributed and decentralized nature of the asvin platform make it highly scalable and resilient.
- The asvin platform can handle burst of requests without congestion.
- The PUF based cryptographic solution is a value addition.
- The horizontal scaling of the platform is very productive for load balancing.
- Fed4FIRE+ experiments can be tailored for specific needs

Business Impact

- Acquired multiple pilot projects
- Received several startup awards
- Streamlined time to market
- Improved brand value and market presence
- Backing of EU sponsored project
- Overall a better platform

How did it Help?

- The iterative experiments generated humongous data
- The data was converted into actionable insights
- A credible validation of scalability
- Fed4FIRE+ experiments gave practical knowledge
- Fed4FIRE+ experiments reinforced trust in our solution.
- Acquired new skills, e.g. Kubernetes

Value Perceived

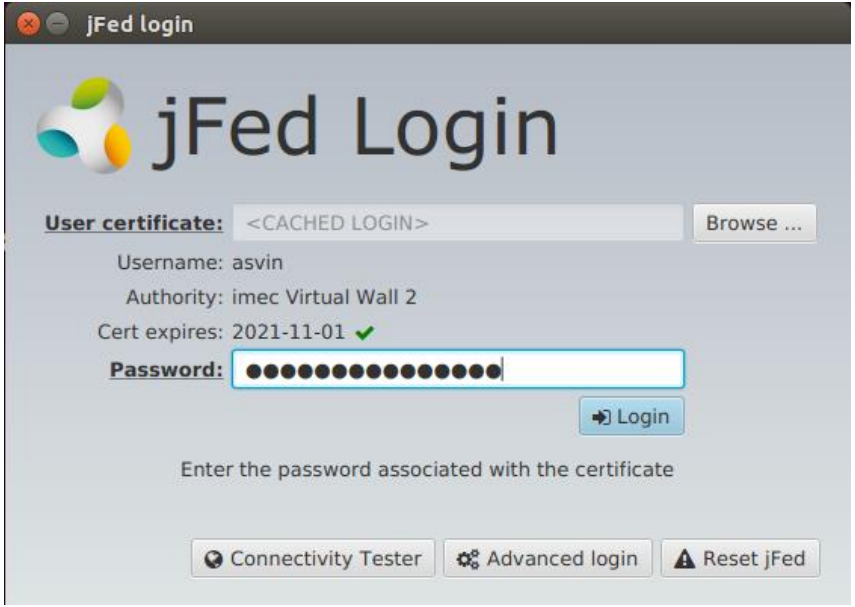
- Increased knowledge about the architecture
- Practical experience
- Proof of scalability and resilience
- Acquired new competence
- Confidence to run experiments on Fed4FIRE+ in future
- Edge over competitors

Why Fed4FIRE+?

- Open, reliable and highly accessible
- Credibility of European Union
- Diversity of available resources
- Simple, efficient and cost-effective experimental process
- Excellent technical support and documentation
- Combining infrastructures
- Cost effective

Resources and Tools

JFED



User certificate: <CACHED LOGIN>

Username: asvin

Authority: imec Virtual Wall 2

Cert expires: 2021-11-01 ✓

Password: ●●●●●●●●●●●●●●●●

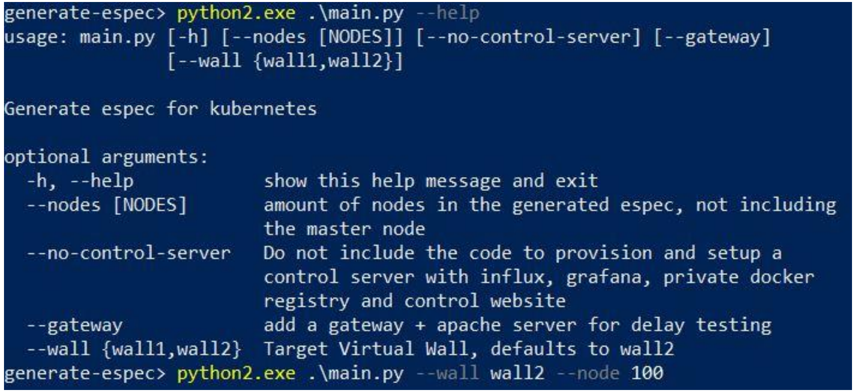
Enter the password associated with the certificate

USAGE

- Provision and manage experiment on testbeds
- RSpec
  - Network and resource configuration
- ESpec
  - Bootstrap an experiment
- Testbed
  - Virtual Wall 1

Resources and Tools

GENERATE ESPEC



```
generate-espec> python2.exe .\main.py --help
usage: main.py [-h] [--nodes [NODES]] [--no-control-server] [--gateway]
               [--wall {wall1,wall2}]

Generate espec for kubernetes

optional arguments:
  -h, --help            show this help message and exit
  --nodes [NODES]       amount of nodes in the generated espec, not including
                        the master node
  --no-control-server   Do not include the code to provision and setup a
                        control server with influx, grafana, private docker
                        registry and control website
  --gateway             add a gateway + apache server for delay testing
  --wall {wall1,wall2}  Target Virtual Wall, defaults to wall2
generate-espec> python2.exe .\main.py --wall wall2 --node 100
```

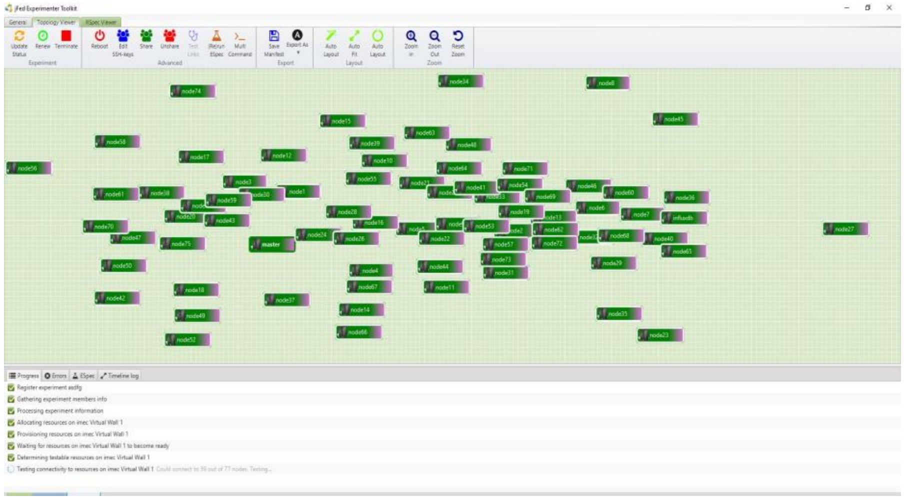
USAGE

- Tool written in python
- Generate ESpec for Virtual Wall1 and Wall2
- Easy to create and deploy Kubernetes cluster on testbeds

https://github.ugent.be/jlemaes/generate-espec

Resources and Tools

KUBERNETES



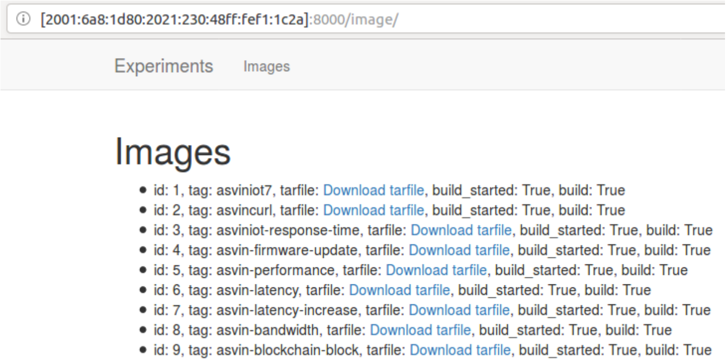
```
☑ Register experiment asdfg
☑ Gathering experiment members info
☑ Processing experiment information
☑ Allocating resources on imec Virtual Wall 1
☑ Provisioning resources on imec Virtual Wall 1
☑ Waiting for resources on imec Virtual Wall 1 to become ready
☑ Determining testable resources on imec Virtual Wall 1
○ Testing connectivity to resources on imec Virtual Wall 1 Could connect to 39 out of 77 nodes. Testing...
```

USAGE

- Container orchestration system
- Used to deploy, scale and manage container applications

Experiment Setup

CONTROL SERVER



Images

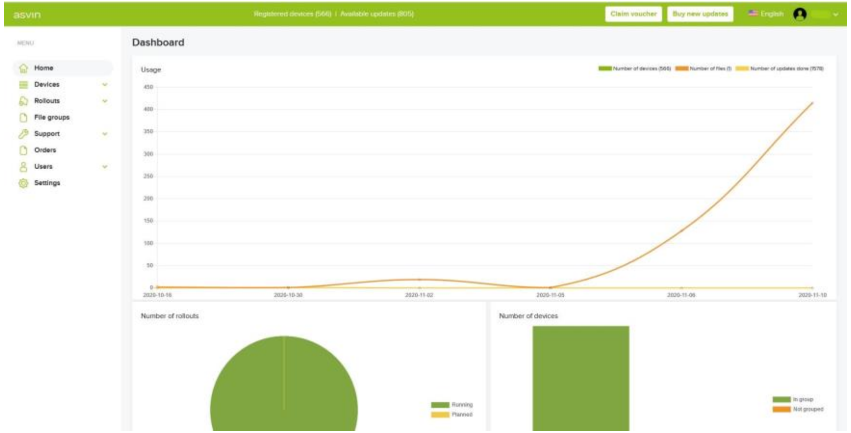
- id: 1, tag: asviniot7, tarfile: Download tarfile, build_started: True, build: True
- id: 2, tag: asvincurl, tarfile: Download tarfile, build_started: True, build: True
- id: 3, tag: asviniot-response-time, tarfile: Download tarfile, build_started: True, build: True
- id: 4, tag: asvin-firmware-update, tarfile: Download tarfile, build_started: True, build: True
- id: 5, tag: asvin-performance, tarfile: Download tarfile, build_started: True, build: True
- id: 6, tag: asvin-latency, tarfile: Download tarfile, build_started: True, build: True
- id: 7, tag: asvin-latency-increase, tarfile: Download tarfile, build_started: True, build: True
- id: 8, tag: asvin-bandwidth, tarfile: Download tarfile, build_started: True, build: True
- id: 9, tag: asvin-blockchain-block, tarfile: Download tarfile, build_started: True, build: True

USAGE

- Build and deploy docker images.
- Start and control an experiment on cluster
- Scale containers on the cluster
- Utilize InfluxDB and Grafana for visualization

Asvin Platform

DASHBOARD



USAGE

- Firmware management
- Device management
- Rollout management
- User management

Added Values

- User friendly interface of jFed experimenter
- Around the clock technical support
- Abundant nodes on testbeds, Wall1 206 and Wall2 159
- High speed internet connectivity on testbeds
- Network impairment e.g. delay, packet loss and bandwidth limitation possible

THANK YOU FOR YOUR ATTENTION

WWW.FED4FIRE.EU

Co-funded by the
European Union

Co-funded by the
Swiss Confederation

This project has received funding from the European Union’s Horizon 2020 research and innovation programme, which is co-funded by the European Commission and the Swiss State Secretariat for Education, Research and Innovation, under grant agreement No 732638.