Improving availability in Industrial Control Systems using Software-Defined Networking

By: Marios Andreou & Joris Jonkers Bøth

Supervisors: Dominika Rusek and Pavlos Lontorfos

Industrial Control Systems

Deloitte.

- Mission critical systems
- Need reliable network
- Downtime → Issues
- Problem: network failures cause long downtimes due to manual intervention

Related Work

- Kalman et. al. (2016)
    - SDN used for traffic segmentation
    - SDN-based IDS
- Zhou et. al. (2017)
    - SDN + NFV used to mitigate DDoS attacks
    - No hardware failure detection implemented
- Pavlos Lontorfos (2020)
    - SDN can be used for hardware failover in an ICS environment
    - No automatic hardware replacement implemented in case of a failure

Research question

How could Software Define Networking combined with Network Function Virtualization enhance availability in an Industrial Control Systems in case of a network hardware failure?

Research subquestions

- How can SDN combined with NFV provision backup network equipment to maintain availability during a network failure?

- What are the consequences of provisioning backup network equipment in an ICS environment for the manageability and connectivity of the network and its connected PLCs?

- What are the limitations of using SDN combined with NFV in an ICS environment regarding the availability of the connected PLCs?

Methodology

- Set up a virtualized ICS environment using:
  - OpenPLC
  - Open vSwitch
  - Faucet

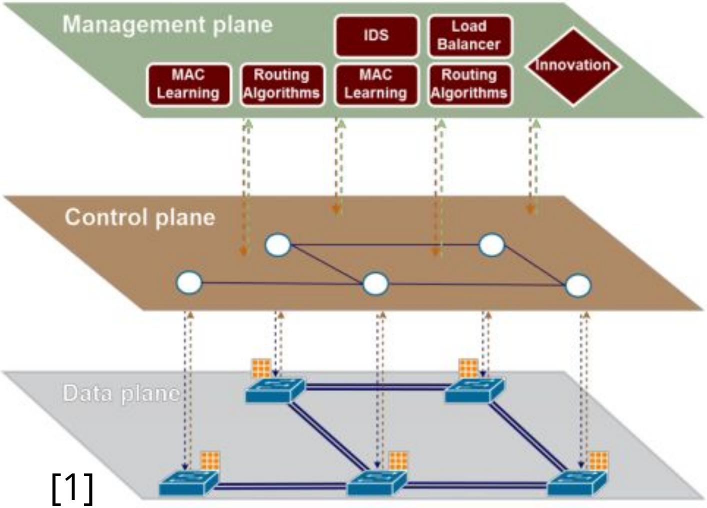
- Implement different NFV solutions to detect unreported failures
  - Re-route traffic in case of a hardware failure
  - Redeploy a new network hardware in case of a failure
  - Redeploy a new interface in case of an interface failure

- Benchmark difference between solutions
  - Run ping with interval 10 ms
  - Measure amount of packets dropped
  - Calculate downtime
  - Repeat 10 times
  - With more deployed hosts and switches on the network

- Research advantages and limitations of solutions

Background: Software Defined Networking

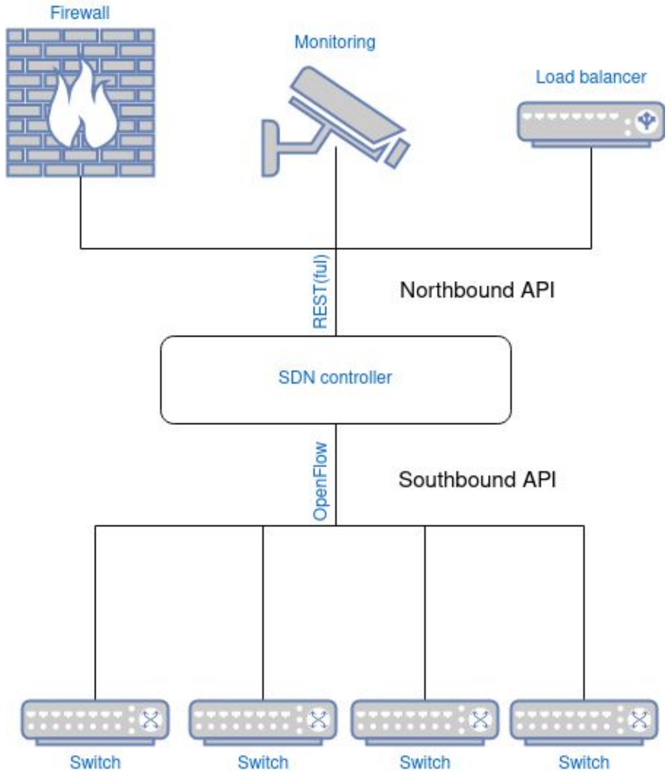
- Separation of Control Plane from Data plane
  - Management Plane → Routing, MAC Learning, etc.
  - Control Plane → Centralized/Distributed Controller
  - Data Plane → Forwarding Switches
- Vendor independent



[1]

Background: Software Defined Networking (Cont.)

- Northbound interfaces
  - Used to connect the control plane to the management plane
    - Communication Between Applications and Controller
    - Allow for monitoring applications (metrics)
    - REST(ful) API
- Southbound interfaces
  - Used for communication between the SDN controller and the underlying network devices
    - OpenFlow API



Background: Faucet

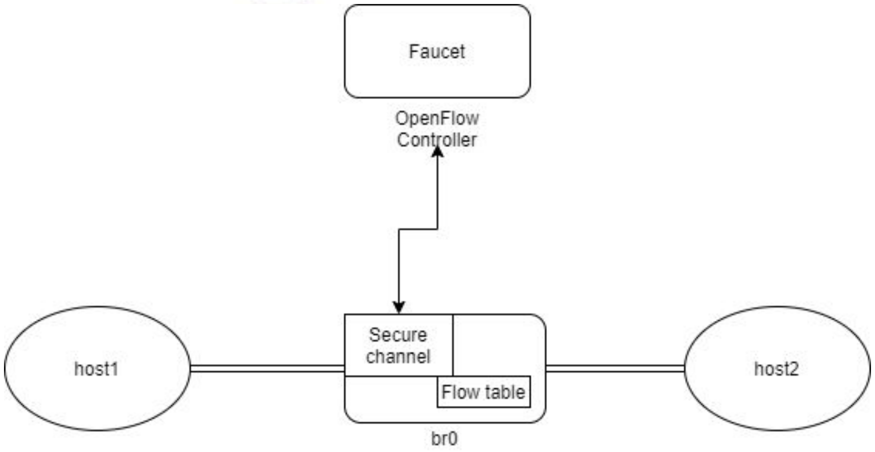
- Open source controller using OpenFlow 1.3
- Designed for High Availability (through idempotency)
- Built-in support for Open vSwitch (OVS)
- Supports:
    - Layer 2 switching
    - VLANs
    - BGP
    - Layer 3 and 4 routing
    - ACLs
    - And more
- Release v1.9.53 (December 8, 2020)

[2]

Background: Faucet

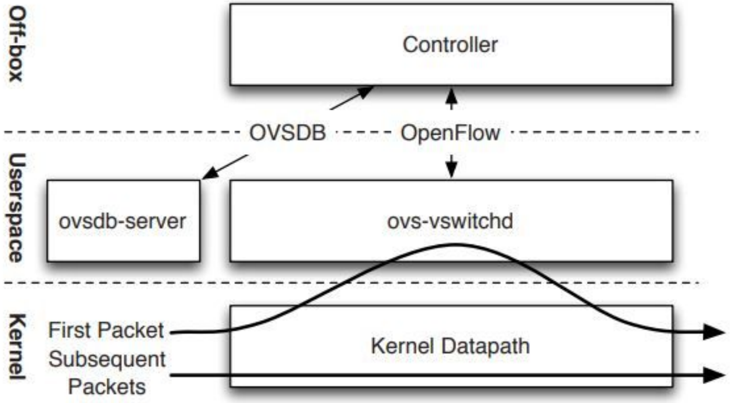
```yaml
vlans:
    office:
        vid: 100
        description: "office network"

dps:
    br0:
        dp_id: 0x1
        hardware: "Open vSwitch"
        interfaces:
            1:
                name: "host1"
                description: "host2 network namespace"
                native_vlan: office
            2:
                name: "host2"
                description: "host2 network namespace"
                native_vlan: office
```



Background: Open vSwitch

- Virtual switches
- Support for OpenFlow
- Main components:
    - ovs-vswitchd → communication with OpenFlow controller
    - kernel datapath → handles packets



Background: Network Function Virtualization

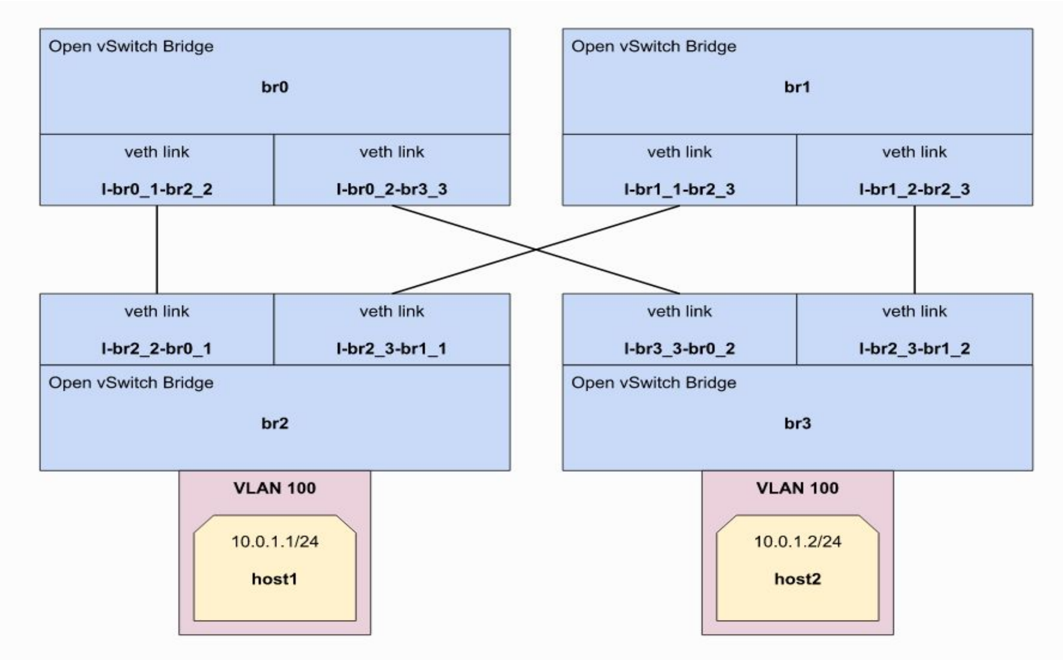
- Virtual Machines offer network services
  - Intrusion Detection System (IDS)
  - DNS
  - DHCP
  - NAT
  - Firewall
  - Load Balancer
  - Virtual Switches

- Can be used to extend SDN

- Dynamic

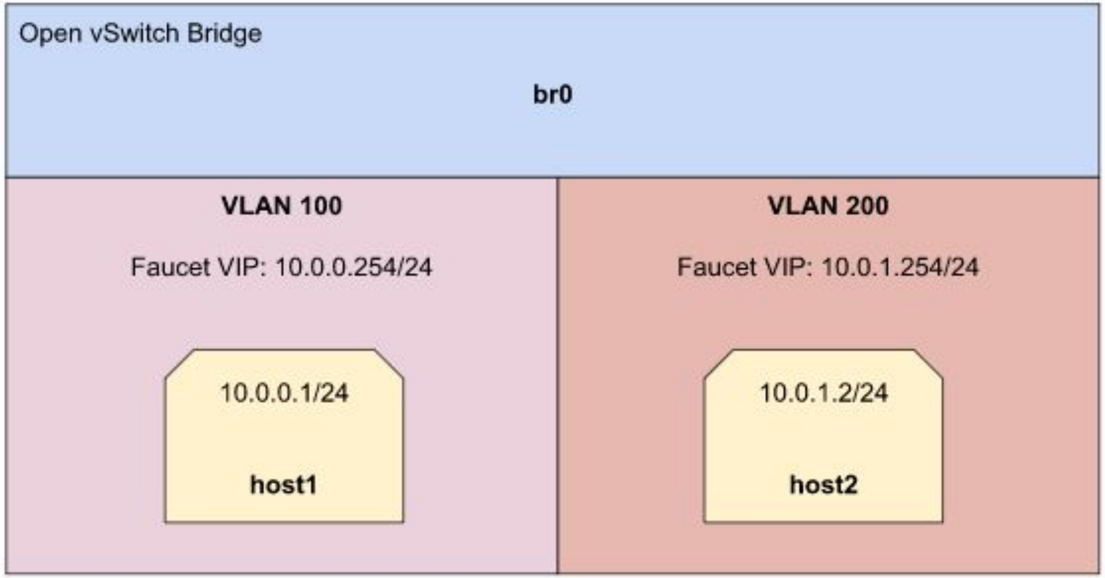
Scenario (1)

- Two bridges for redundancy (br0 and br1)
- Two intermediate switches (br2 and br3)
- One bridge goes down (br0 or br1)
- Traffic would be rerouted to other bridge (br0 or br1)



Scenario (2)

- One bridge (br0)
- br0 goes down → br0 will be redeployed
- Connection re-established



Scenario (3) - NFV

- Used same topology as scenario 2
- Write code for NFV to look which ports are connected to bridge (1)
- Get TX value of each port every two seconds and compare them to previous values (2)
    - Two seconds needed to perform evaluation on 104 bridges
- Interface fails → tx value stops increasing → interface recreated

```
Bridge br0
    Controller "tcp:145.100.111.132:6653"
    fail_mode: secure
    Port veth-host1
        Interface veth-host1
    Port veth-host2
        Interface veth-host2
    Port br0
        Interface br0
            type: internal
```

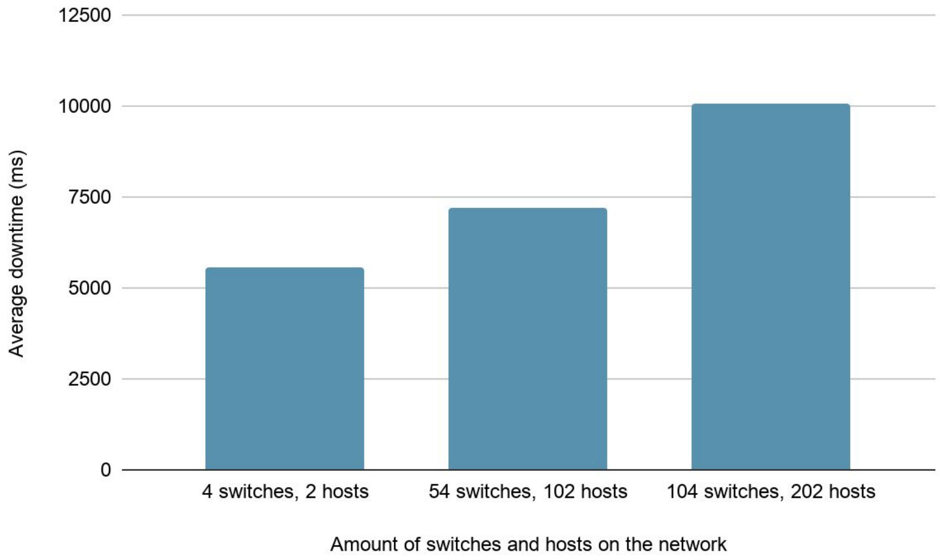
(1)

```
root@RP2-SDN-ICS:/etc/faucet# ovs-ofctl dump-ports br0
OFPST_PORT reply (xid=0x2): 3 ports
  port  "veth-host1": rx pkts=6449, bytes=629158, drop=0, errs=0, frame=0, over=0, crc=0
           tx pkts=3457, bytes=335522, drop=0, errs=0, coll=0
  port LOCAL: rx pkts=0, bytes=0, drop=0, errs=0, frame=0, over=0, crc=0
           tx pkts=0, bytes=0, drop=0, errs=0, coll=0
  port  "veth-host2": rx pkts=3435, bytes=333814, drop=0, errs=0, frame=0, over=0, crc=0
           tx pkts=3458, bytes=335592, drop=0, errs=0, coll=0
```

(2)

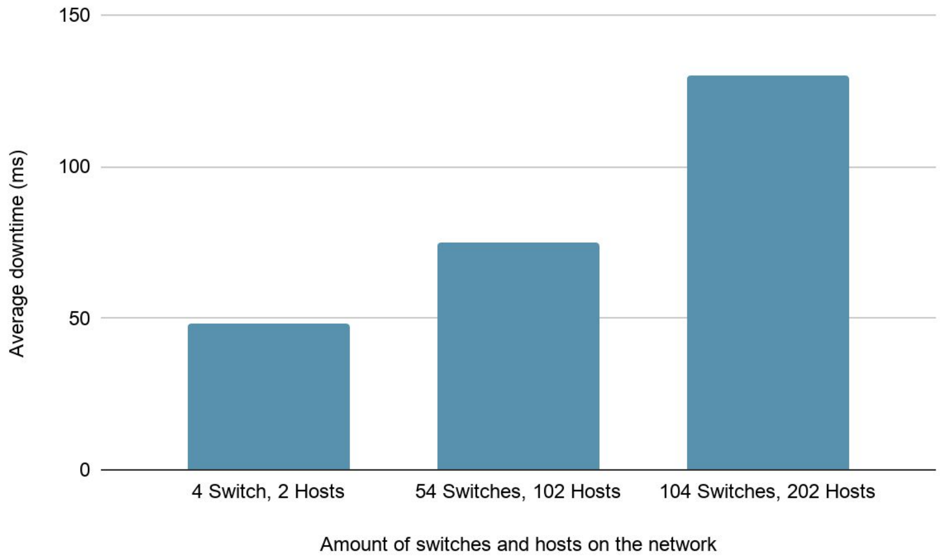
Results: Scenario (1)

- 4 switches - 2 hosts → Mean: 5576ms
- 54 switches - 102 hosts → Mean: 7217ms
- 104 switches - 202 hosts → Mean: 10050ms



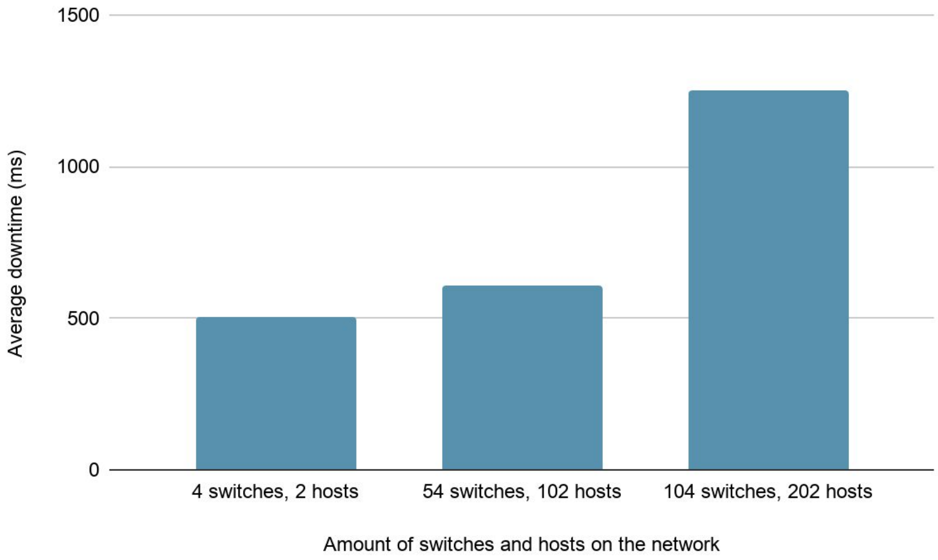
Results: Scenario (2)

- 4 switches - 2 hosts → Mean: 48ms
- 54 switches - 102 hosts → Mean: 75ms
- 104 switches - 202 hosts → Mean: 130ms



Results: Scenario (3)

- 4 switches - 2 hosts → Mean: 507ms
- 54 switches - 102 hosts → Mean: 608ms
- 104 switches - 202 hosts → Mean: 1254ms



Results: Comparison

- Redeployment < Recreation < Rerouting

Discussion

- Virtualization Architecture
  - Xen hypervisor type 2 used
  - Container-based
  - Native hypervisor (Type 1)
- SDN controller
  - Faucet written in Python → High-level language
  - Floodlight (Java), Nox (C++) or Trema (Ruby and C)
- Network Functions
  - Scenario (3) limitation on 2 seconds interval for every check
  - Scenario (2) and (3) not feasible on hardware
- Topology
  - Single-point of failure for intermediate switches
    - Scenario (2) could be used (48-130 ms average downtime)
- Downtime measured as ICMP packets dropped
  - Bidirectional traffic
  - Short interval

Conclusion

How can SDN combined with NFV provision backup network equipment to maintain availability during a network failure?

- Monitor health of switch and its ports
- Instruct system to redeploy switch or port if failure detected

Conclusion (Cont.)

What are the consequences of provisioning backup network equipment in an ICS environment for the manageability and connectivity of the network and its connected PLCs?

- Reduced downtime
  - Redeployment < Recreation < Rerouting
- Restored connectivity
  - After redeployment, ICMP packets arrived at destination
- Restored manageability
  - Device re-established connection to Faucet dynamically

Conclusion (Cont.)

What are the limitations of using SDN combined with NFV in an ICS environment regarding the availability of the connected PLCs?

- Additional load on the controller and network
  - Checks require bandwidth and CPU power
- Reaction delays present
  - NFV interval to check every 2 seconds
- Limited effectivity in case of using hardware
  - Hardware cannot be re-deployed dynamically

Conclusion (Cont.)

How could Software Define Networking combined with Network Function Virtualization enhance availability in an Industrial Control Systems in case of a network hardware failure?

- Dynamic decision
  - Automatic deployment of virtualized hardware in case of a failure
  - Automatic port recreation in case of a failure
  - Automatic rerouting

Future Research

- Run experiments on hardware

- Look into different SDN controllers
  - E.g. Nox (C++), Floodlight (Java)

- Research NFV function efficiency
  - Code efficiency
  - Shorter check intervals
  - Network size limitations

Summary

- Reduce downtime in ICS environments
  - Redeployment < Recreation < Rerouting

- SDN combined with NFV has shown to be an effective solution
  - Improve availability by reducing downtime
    - Re-routing
    - Redeployment of a switch
    - Recreation of a port

- Detecting failures and dynamically take action according to the scenario
  - No human intervention

References

[1] Rui Miguel da Concei ̧c ̃ao Queiroz.Integration of SDN technologies in SCADA Indus-trial Control Networks. 2017.url:https://estudogeral.sib.uc.pt/bitstream/10316/83367/1/Relat%c3%b3rio%20de%20Estagio%20-%20versao%20FINAL_pos%20correcoes_v3.pdf

[2] Faucet Foundation. url:https://www.faucet.org.nz/