SAND2020-6430PE

CAPSEC

Containerized Application Security for Real Time Software Upgrade & Patching

Adrian Chavez

SANDIA NATIONAL LABORATORIES

DOE PACT VIRTUAL SHOWCASE

Business Problem

- Software patching disrupts uptime, resulting in increased costs and lost revenue.
- Many industries require near continuous uptimes, delaying patching.
- Delays in patching result in increased vulnerabilities to cyber attack

Technical Challenges

Thesis: Use of Containerization eliminates costs and downtime associated with updating/patching software.

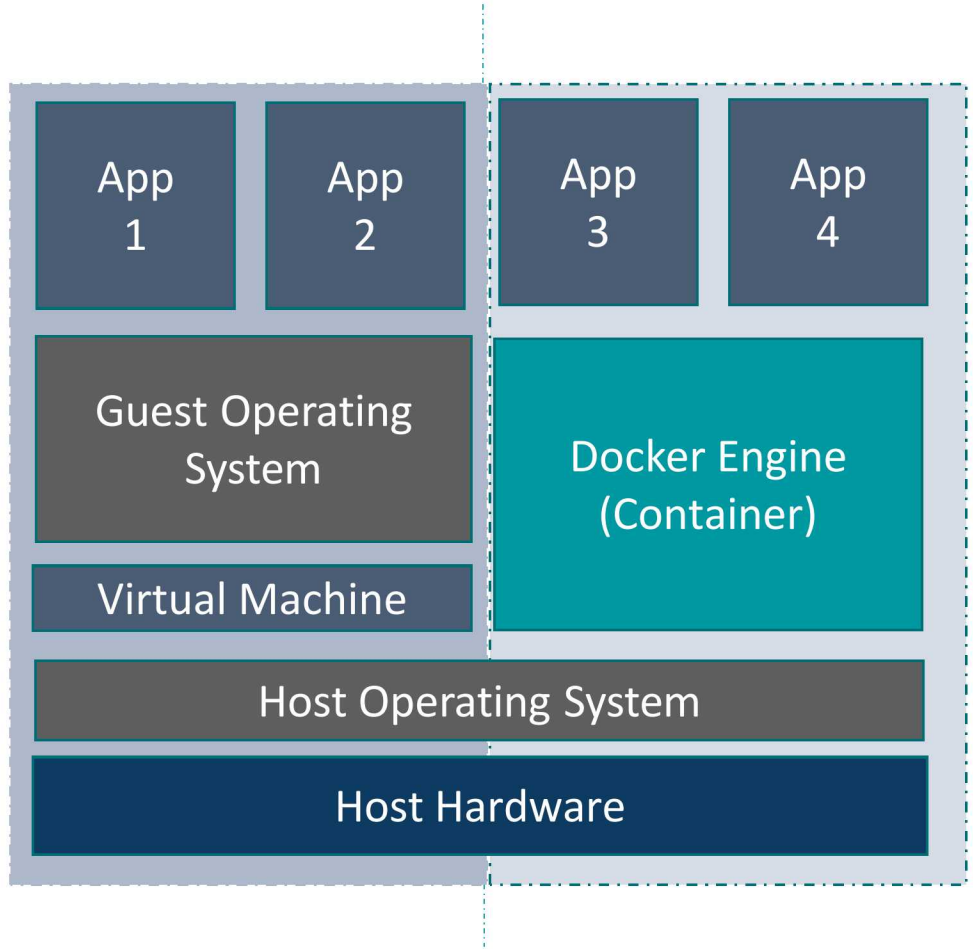
- Increase resiliency to cyber attack
- Detect and isolate cyber attacks

Challenges:

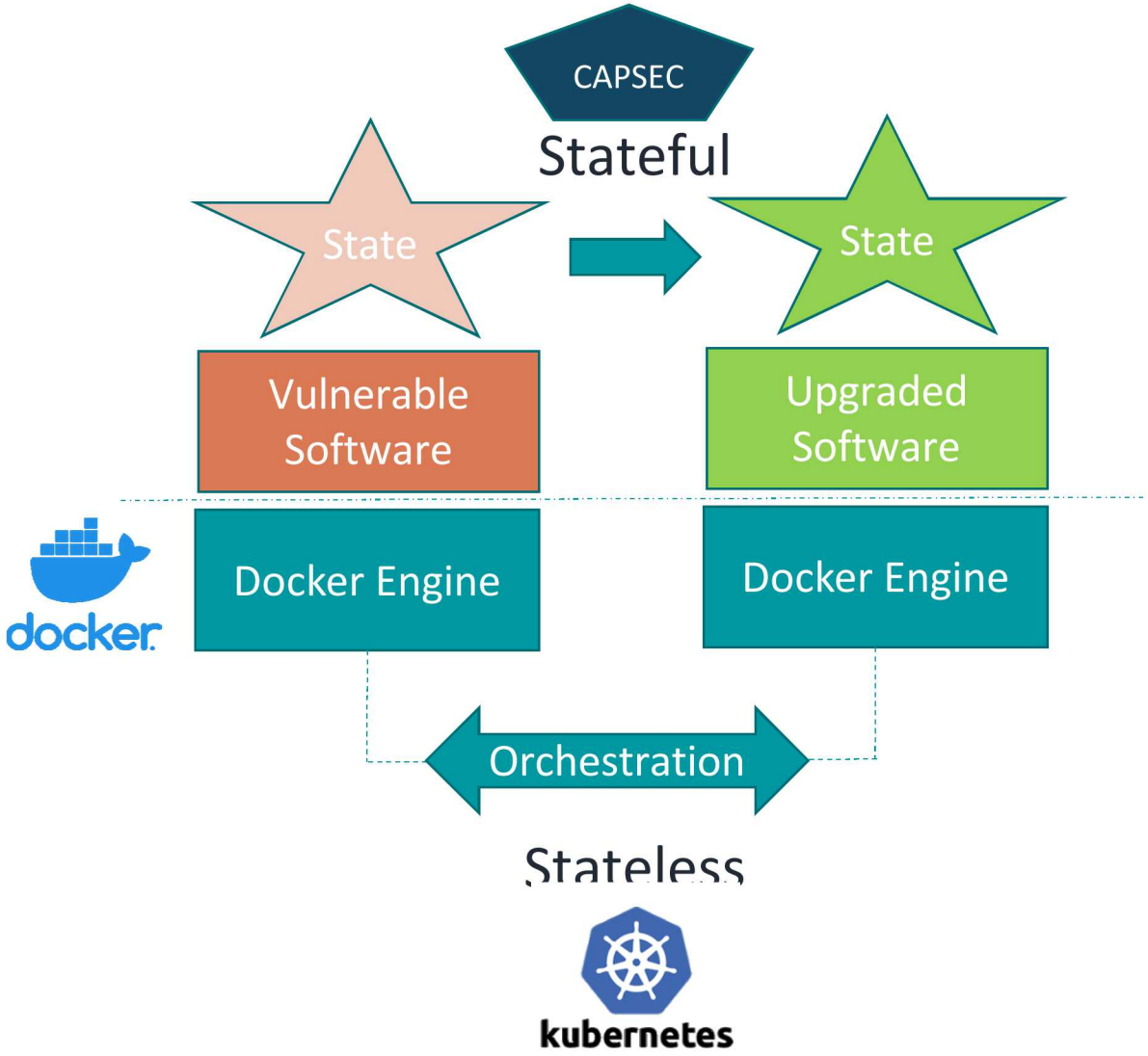
- Accounts for Stateful (i.e.: ModBus) and Stateless (i.e.: Web Applications)
- The computing resource limitations of Stateful devices prevent use of IT continuous patching solutions
- Orchestration and containerization of existing software
- Fault tolerance

Containerization

- Containers bundle all software, libraries, environment variables and configuration files
- Use less resources than VMs due to independence from full OS.
- Most common:
  - Open source Docker Engine
  - Open source container-orchestration system Kubernetes



Solution Architecture

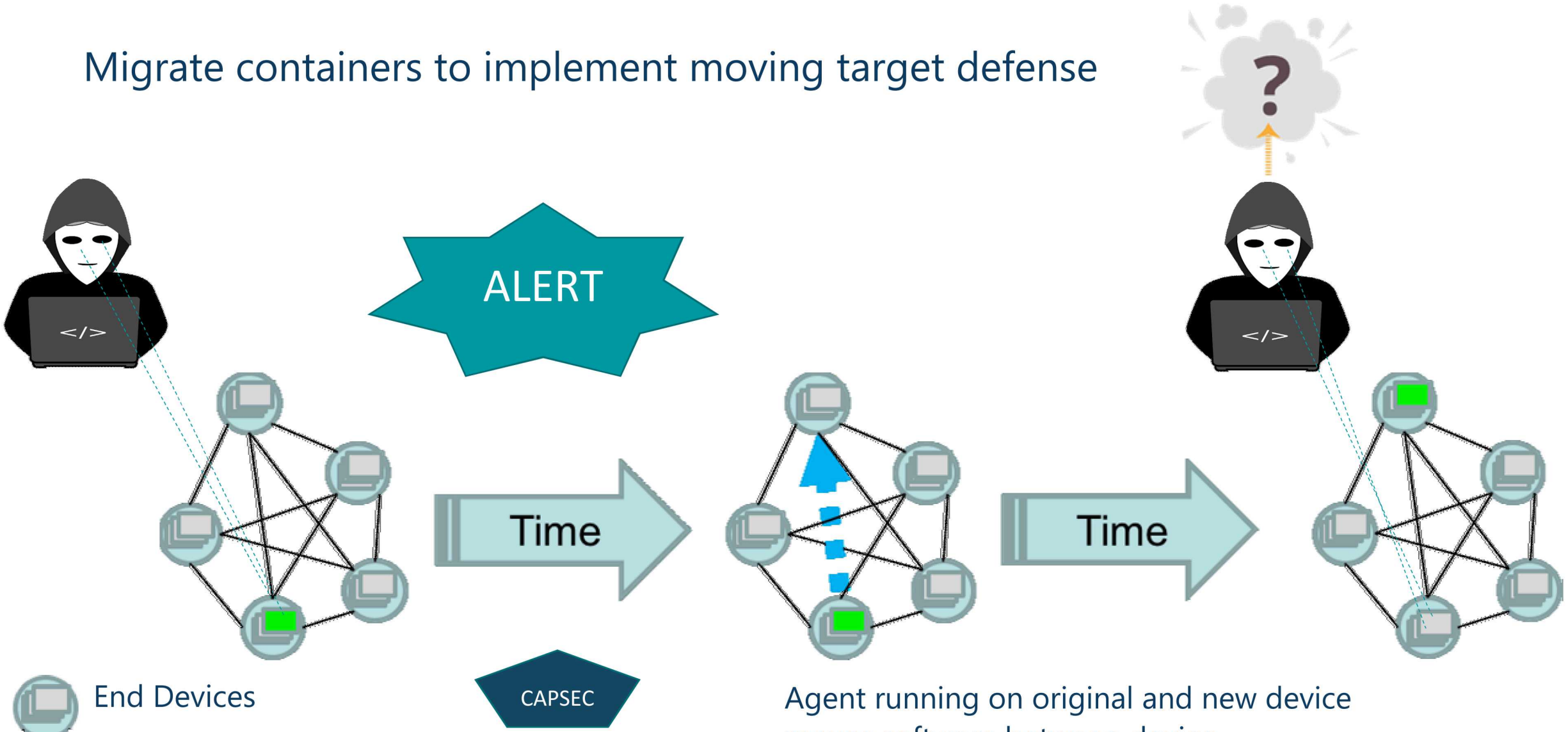


- Containerized architecture
- Replicates Stateful environments
- TLS Agents on parallel containers

Solution Architecture

CAPSEC

Migrate containers to implement moving target defense



Development History & Results

CAPSEC

Principal Investigator
Adrian Chavez

- Ph.D. Computer Science, University of California, Davis
- Principal Member of Technical Staff, Cybersecurity R&D

Developmental History

2016 First filing (July)
2018 Project kick-off (May)
US patent granted (July)
2020 Established proof-of-concept (Feb)

Funding History: 2018-Present
$2.5M, DOE CESER Office

Research Team
Ryan Birmingham
Jasenko Hosic
Jaykumar Patel
Kandy Phan
William Stout

Technology Readiness Level (TRL):
TRL 4 as of June 2020

IP Protection
US Patent No. 10,037,203
Real Time Software Upgrade

Market Validation

Lead: Sandia National Laboratories

Partners: SEL Schweitzer Engineering Laboratories; Pacific Northwest National Laboratory; GRIMM; Chevron; Fort Belvoir United States Army Virginia; GRIMM; Chevron

Technical Requirements & Benefits

CAPSEC

TECHNICAL REQUIREMENTS

- Live Upgrade & Migration Software
- System of Multiple Units of Hardware
- -Or- Software Replicates Nodes

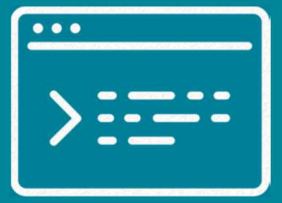
BENEFITS

- Improved resilience
- Active patching & remediation during cyber attack
- Decreased maintenance time

Technology Roadmap

CAPSEC

Expand application management support and fault tolerance



Test and evaluate CAPSec within representative environments

Interoperate CAPSec with vendor commercial products

Collect data and metrics

Pilot test CAPSec with OT partners

Bringing CAPSec to Market

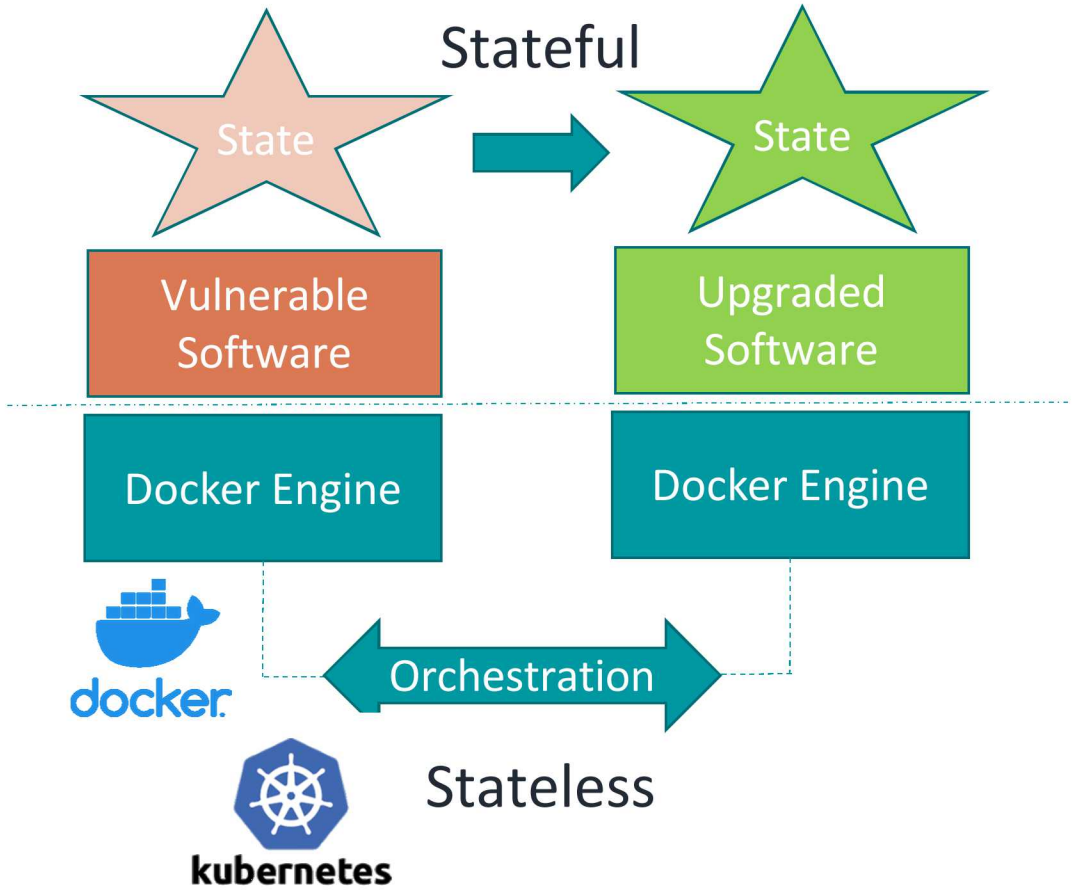
CAPSEC

DOE PACT VIRTUAL SHOWCASE

DEMONSTRATION

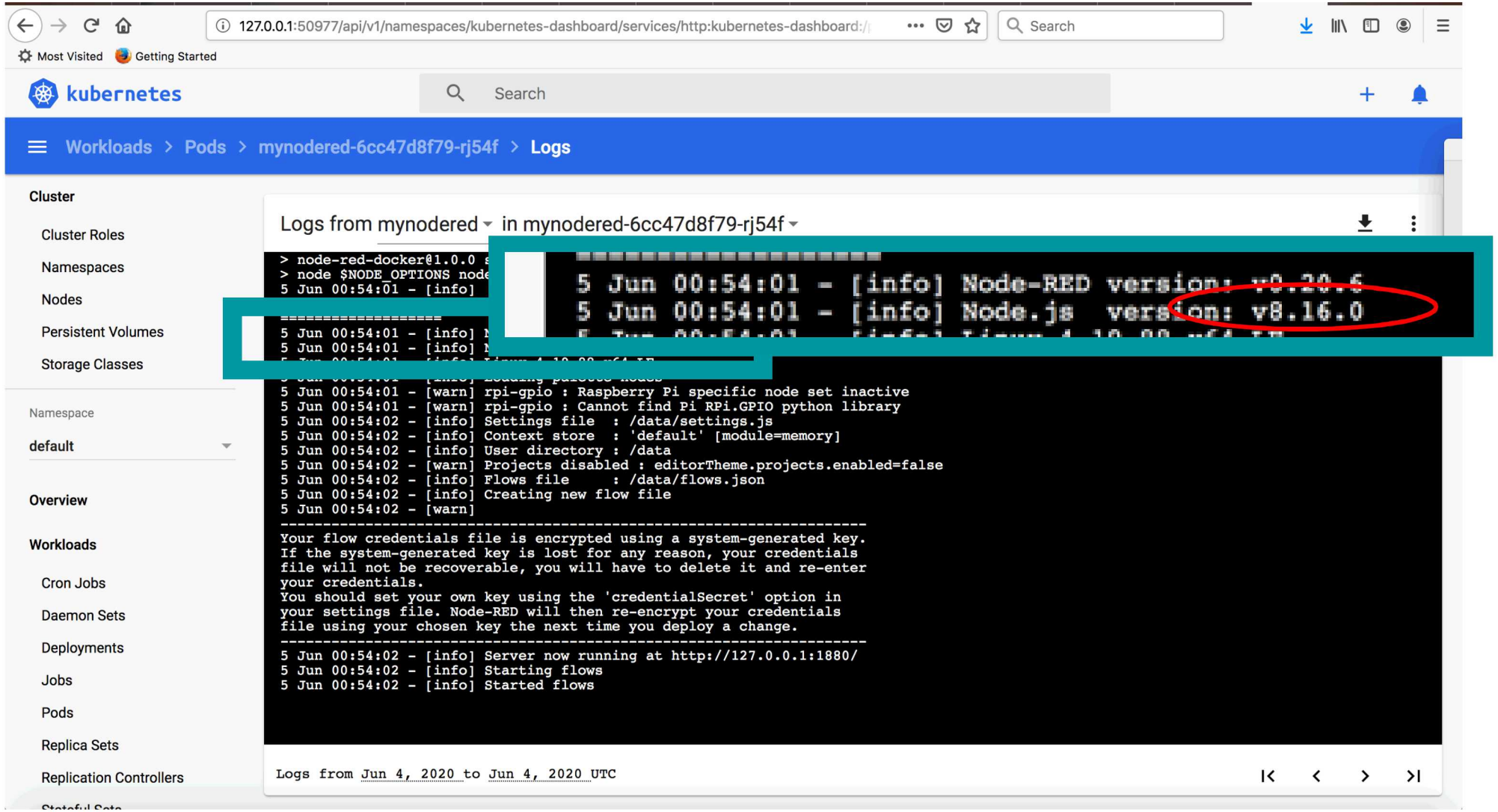
Solution Architecture

CAPSEC

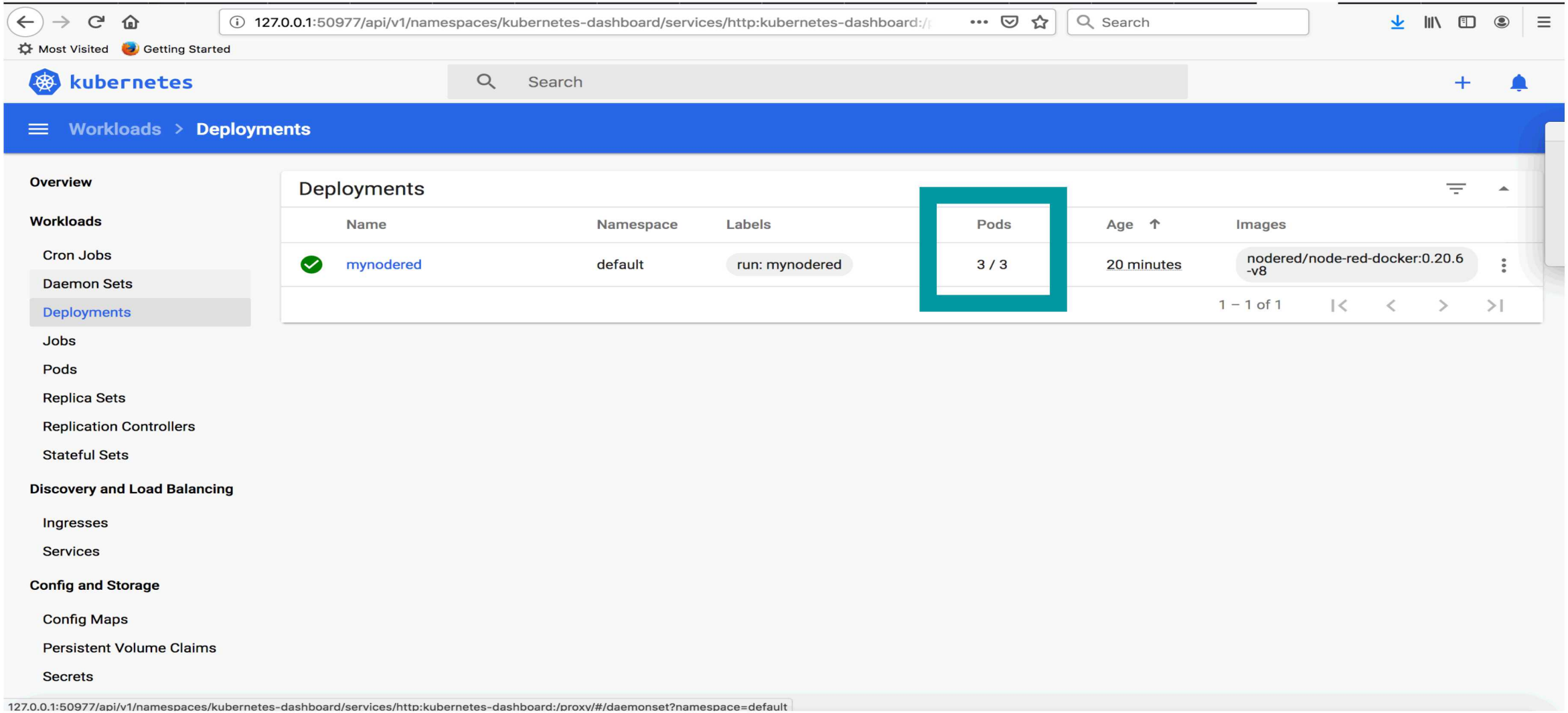


- Containerized architecture
- Replicates Stateful environments
- TLS Agents on parallel containers

NodeRed Software Version BEFORE Upgrade



NodeRed Running Multiple Instances to Upgrade Software without Downtime



Upgrade NodeRed Software One Instance at a Time

```
S1006074:~ adrchav$ kubectl run mynodered --port=1880 --image=nodered/node-red-docker:0
.20.6-v8
kubectl run --generator=deployment/apps.v1 is DEPRECATED and will be removed in a futur
e version. Use kubectl run --generator=run-pod/v1 or kubectl create instead.
deployment.apps/mynodered created
S1006074:~ adrchav$
```

```
S1006074:~ adrchav$ kubectl set image deployment mynodered mynodered=nodered/node-red-d
ocker:latest --record
deployment.apps/mynodered image updated
S1006074:~ adrchav$
```

```
S1006074:~ adrchav$ kubectl set image deployment mynodered mynodered=nodered/node-red-d
ocker:latest --record
deployment.apps/mynodered image updated
S1006074:~ adrchav$ kubectl rollout status deployment mynodered
deployment "mynodered" successfully rolled out
S1006074:~ adrchav$
```

NodeRed Version Patched