KubeCon | CloudNativeCon

Europe 2019

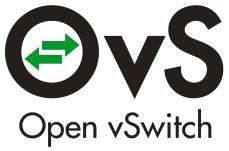
Using eBPF to bring Kubernetes-aware Security to the Linux Kernel

Dan Wendlandt – Isovalent

@danwendlandt @ciliumproject

Who am I?

KubeCon
CloudNativeCon
Europe 2019


OvS
Open vSwitch

@


nicira


NSX

@


vmware


cilium

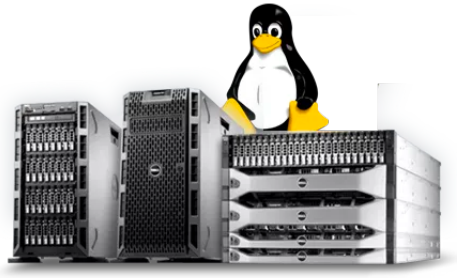
@


ISOVALENT

Linux – A General Purpose Operating System

General Purpose OS Abstractions:

Processes, Files, IP Addresses, TCP ports



What would it mean to:

Optimize Linux for securely running Kubernetes-based microservices?

Runtime Attacks Happen When….

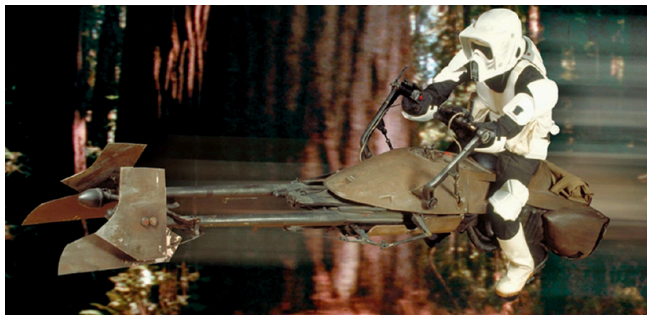
Existing set of software systems (application services, databases, external APIs)

Application team has an expected path of execution and data flows.

Attacker finds an alternate but still permitted path of execution and data flow.

Runtime Security is About…

Enabling apps to run,
and developers to move
as fast as possible…



… while preventing execution
and dataflow paths not
intended by the app developers

What is BPF?

Berkeley Packet Filter

Highly efficient sandboxed virtual machine in the Linux kernel.

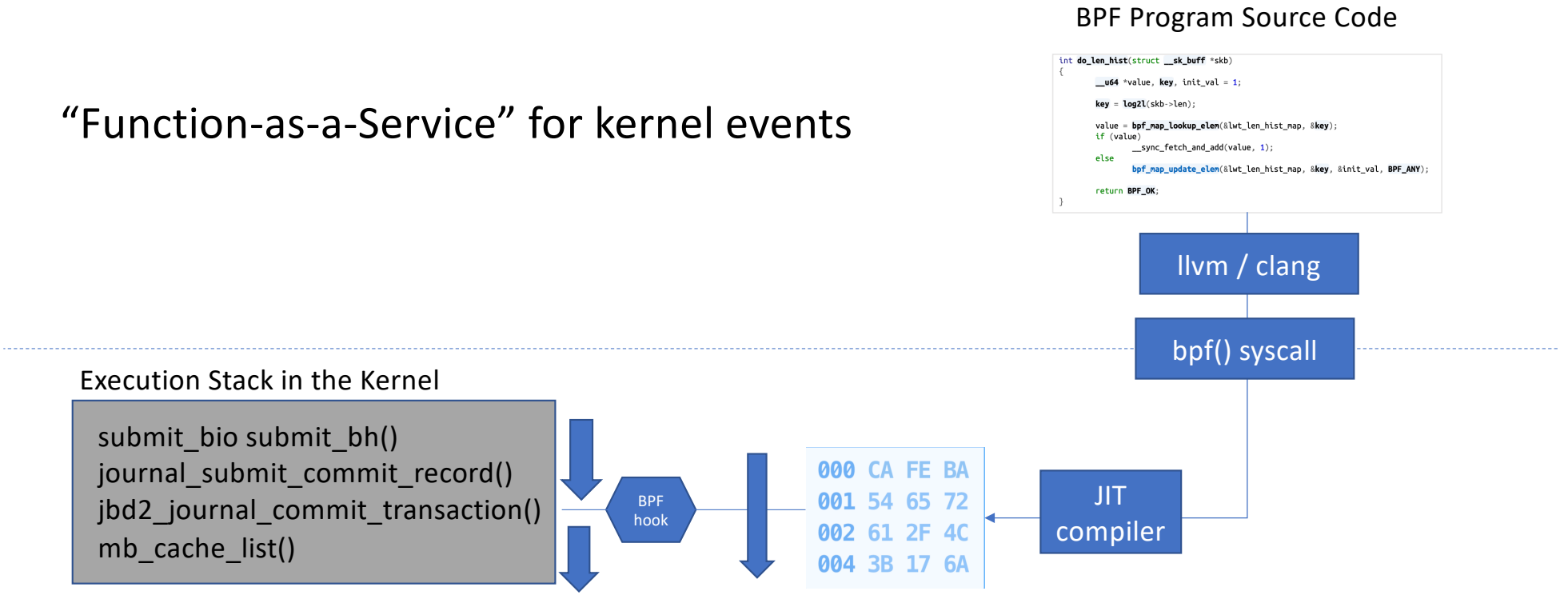
Making the Linux kernel programmable at native execution speed.

Origins in the humble “tcpdump”:

```
tcpdump -n dst host 192.168.1.1
```

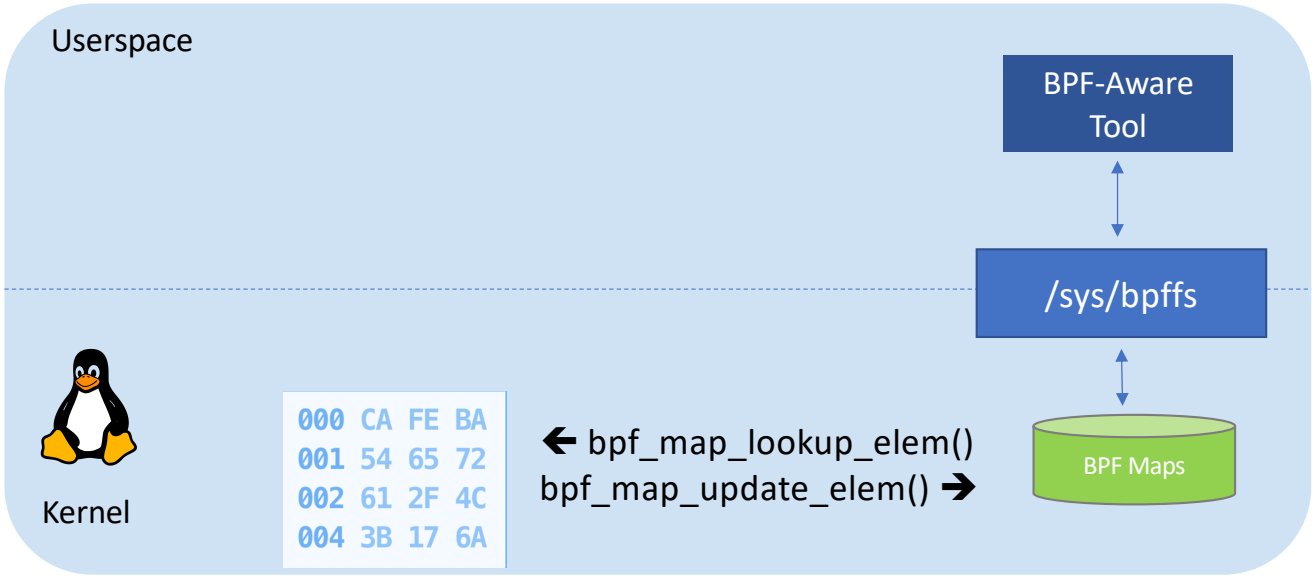
BPF Concepts #1: Programs and Hook Points

"Function-as-a-Service" for kernel events



BPF Concepts #2: Maps

Efficient data structures that persist across function invocation.

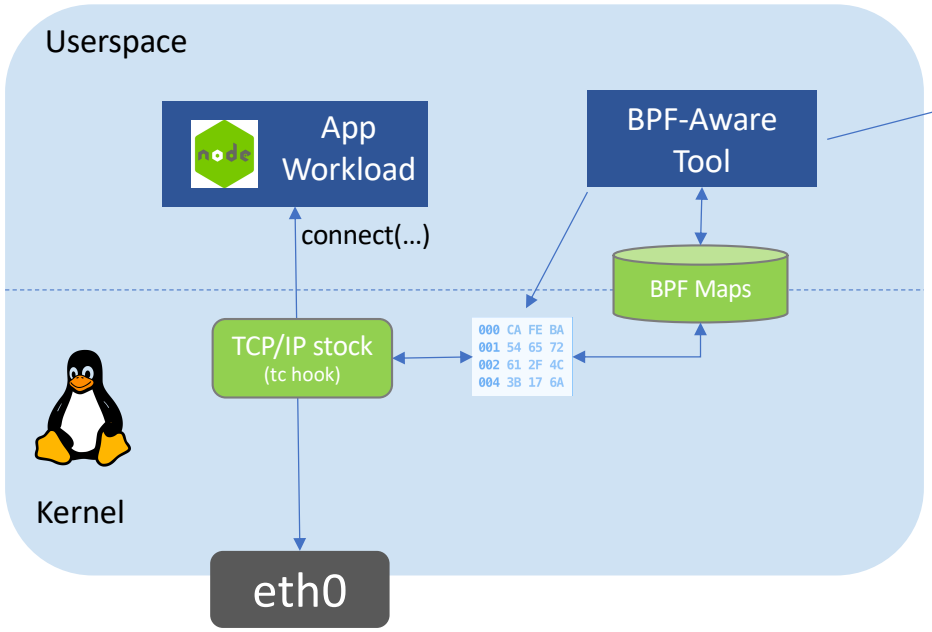


Highly Efficient:

- Fine-grained update of BPF program config data (e.g., policy/load-blancing rules)
- Accumulation of visibility data in-kernel, with only summaries exported to userspace.

https://lwn.net/Articles/664688/

BPF: Putting it All Together



1. Creates custom logic as pseudo C code, to be run at a specific BPF trace point.
2. Compiles to BPF byte code
3. Loads byte code into kernel using bpf() syscall.
4. Kernel verifies safety of code, JIT-compiles for native perf.
5. BPF program executes each time trace point is invoked.

5. Highly efficient communication of data between kernel + userspace using BPF maps.

BPF Tech Adoption

- L3-L4 Load balancing
- Network security
- Traffic optimization
- Profiling

https://code.fb.com/open-source/linux/

- Replacing iptables with BPF
- NFV & Load balancing (XDP)
- Profiling & Tracing

https://goo.gl/6JYYJW

Google

- QoS & Traffic optimization
- Network Security
- Profiling
- http://vger.kernel.org/lpc-bpf2018.html#session-1

NETFLIX

- Performance Troubleshooting
- Tracing & Systems Monitoring
- Networking
  http://www.brendangregg.com/blog/2016-03-05/linux-bpf-superpowers.html

Learn More: http://docs.cilium.io/en/latest/bpf

How You Can Use BPF

Toolkits for writing & running arbitrary BPF programs / traces

https://github.com/iovisor/bcc

https://github.com/iovisor/bpftrace

https://github.com/iovisor/kubectl-trace

Multi-use
BPF directly exposed

Platforms built on / using BPF

Targeted Use Cases,
BPF under the covers

Runtime Attacks Happen When….

Existing set of software systems (application services, databases, external APIs)

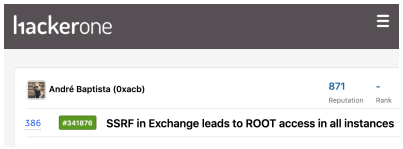
Application team has an expected path of execution and data flows for normal behavior.

Attacker finds an alternate but still permitted path of execution and data flow.

K8s Microservices Runtime Attack Vectors

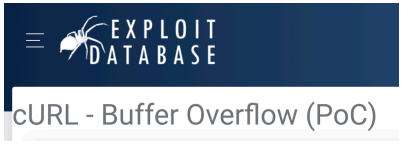
KubeCon CloudNativeCon Europe 2019

Buggy or Malicious
Main Service



https://hackerone.com/reports/341876

Buggy or Malicious
Sidecar / Init Container



https://www.exploit-db.com/exploits/24487

Insider with “kubectl exec”
for prod troubleshooting.

```
kubectl exec -it jobposting  /bin/bash
/root:#
```

https://kubernetes.io/docs/tasks/debug-application-cluster/get-shell-running-container/

Degrees of Freedom == Paths for Exploit

General Purpose OS leaves many degrees of freedom for malicious execution paths + data flows….

VS.

BPF lets us build an OS security model tailored to K8s microservices apps

Securing Microservices…

What unique attributes of Kubernetes microservices can we leverage?

Micro Services

Single service per-container, launched as pid 0.

Additional code run as init/sidecar containers.

Service code updated by deployment of new container.

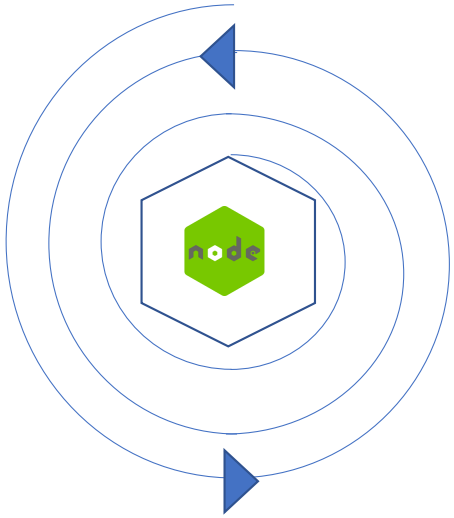
Identity tied to service being implemented, not IP address

Service offers an API (HTTP, gRPC, Kafka, Redis, etc) with rich semantics well beyond TCP/UDP port.

Identifying and Stopping Runtime Attacks

Measure

expected behavior



Monitor

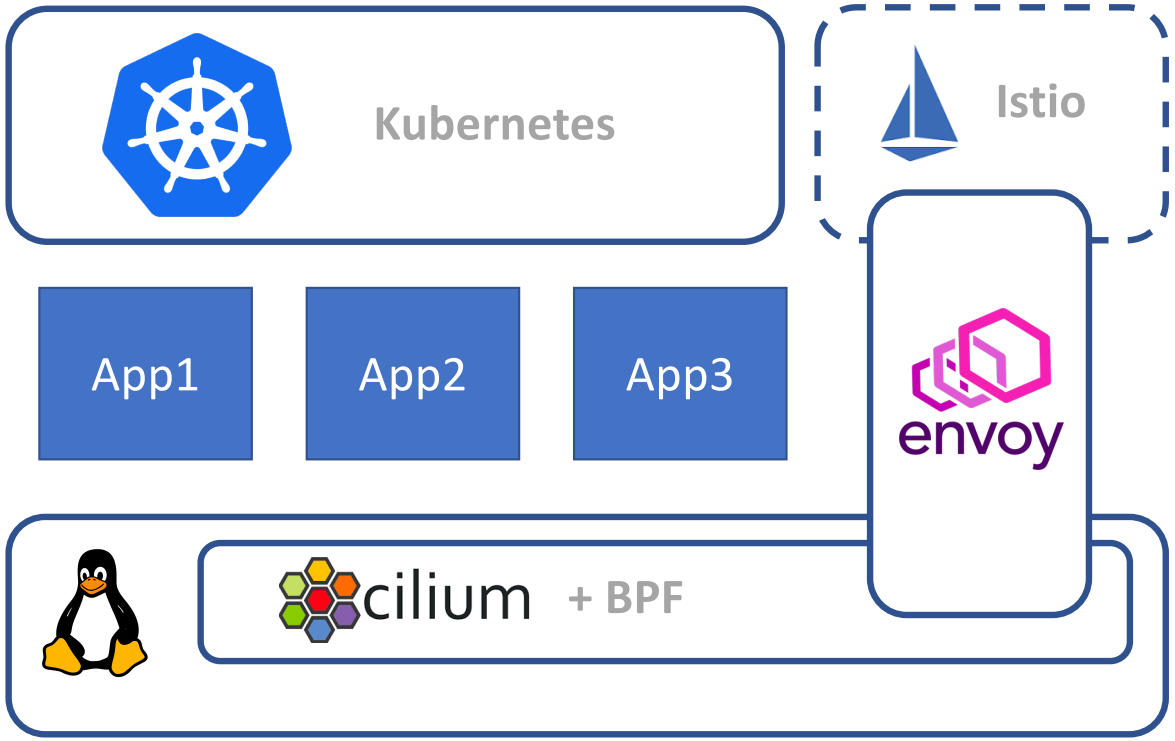
possible deviations

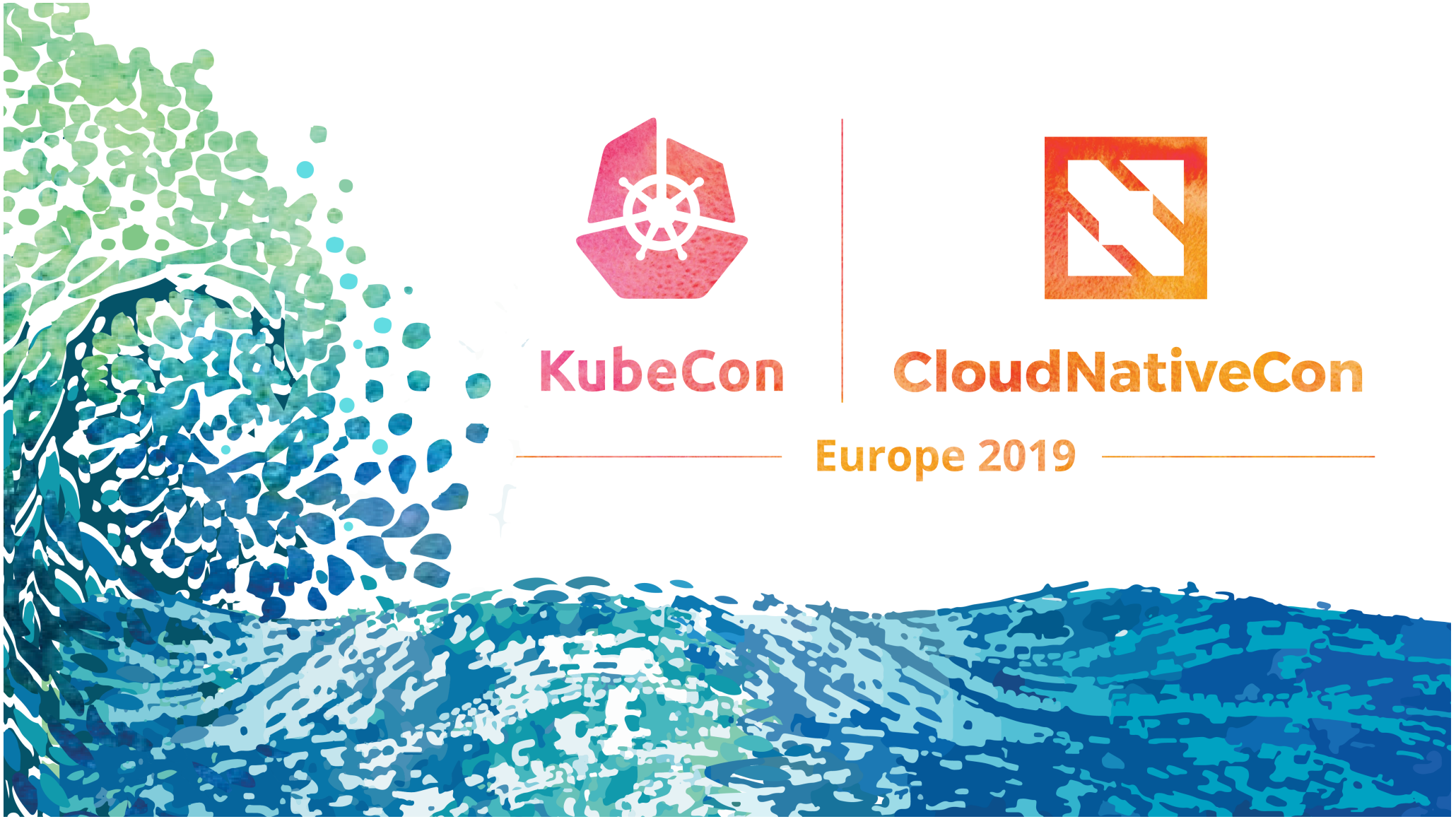
Constrain

to expected behavior

Demo Time…

A New Microservices Stack is Emerging




KubeCon
CloudNativeCon
Europe 2019