



Kubernetes APIs Under the Hood

Who Am I?

Phillip Wittrock (@pwittrock)

Software Engineer at Google working on GKE and OSS Kubernetes

My mission is to make using Kubernetes simple and enjoyable

You might have come across me through...

- Kubectl
- Kubebuilder
- Kubernetes Steering Committee

Kubernetes Refresher

- **Nodes** are machines in a cluster that run Containers in **Pods**
- **Pods** are created and managed by higher level abstractions such as **ReplicaSets**
- **ReplicaSets** managed by higher level abstractions such as **Deployments**
- **Deployments** (and all other user owned objects) defined in files and created / updated with ``kubectl apply``
- **APIs** (**deployments**, **replicasets**, **pods**, **nodes**) == *Resource Types* and **Objects**
== *Resources*

Kubernetes APIs Are...

**Declarative, Asynchronous,
Level-Triggered,
Observable, Discoverable,
Versioned, Access
Controlled, Extensible, ...**

Kubernetes APIs are... Declarative

Configuration for a **Deployment** that manages 3 **Pods** each running an nginx container

Deployment resource is declared in a *file*

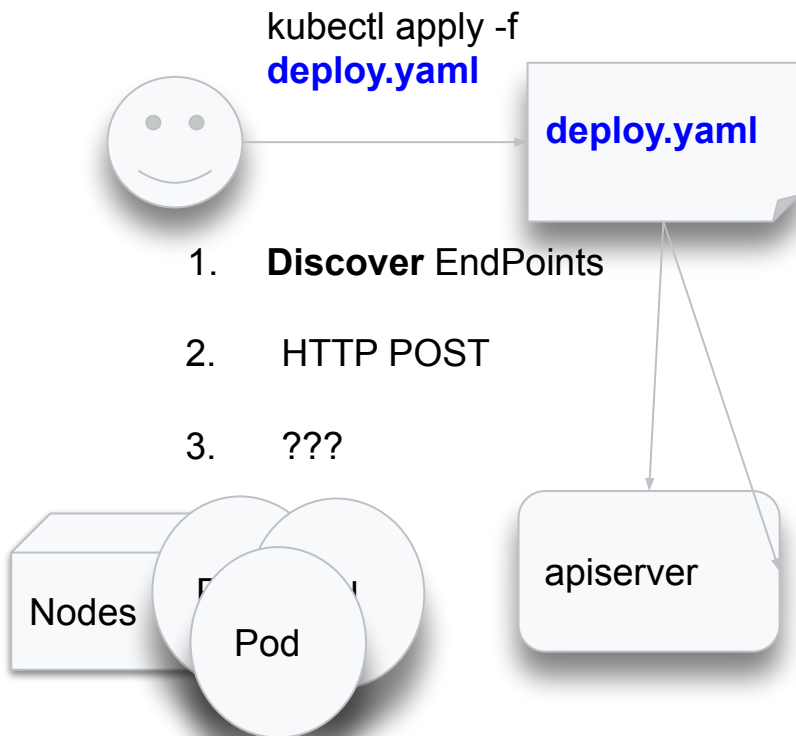
Create or update the resource in the cluster by run *kubectl apply* on a file or directory

```
kubectl apply -f deploy.yaml
```



```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: nginx-deployment
  labels: {app: nginx}
spec:
  replicas: 3
  selector:
    matchLabels: {app: nginx}
  template:
    metadata:
      labels: {app: nginx}
    spec:
      containers: [
        {name: nginx,
          image: 'nginx:1.7.9'}}]
```

Create Deployment Example



```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: nginx-deployment
  labels: {app: nginx}
spec:
  replicas: 3
  selector:
    matchLabels: {app: nginx}
  template:
    metadata:
      labels: {app: nginx}
    spec:
      containers: [
        {name: nginx,
          image: 'nginx:1.7.9'}]
```

Lifecycle - Resources, Controllers and Webhooks

- APIs declared as **Resources** - provide storage and endpoints
- APIs actuated by **Controllers** - execute the business logic
- APIs admitted by **Webhooks** - defaulting, validation, conversion

Kubernetes APIs are... Asynchronous & Observable

Resource:
Stores
Stuff

API endpoints
(CRUD storage)



Foo Resources
stored in etcd by the
apiserver

Asynchronous watch notification
on object create / update / delete

Controller:
Does Stuff

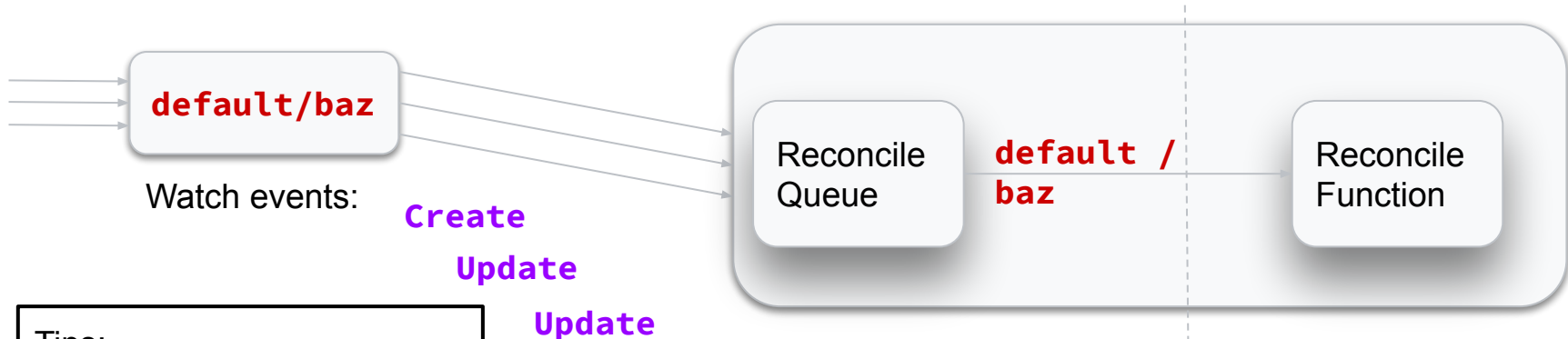
Foo Controller

Loose coupling between
Controller and API endpoints:
Storage doesn't know about
Controllers

Kubernetes APIs are... Level Triggered

Object

Controller



Tips:

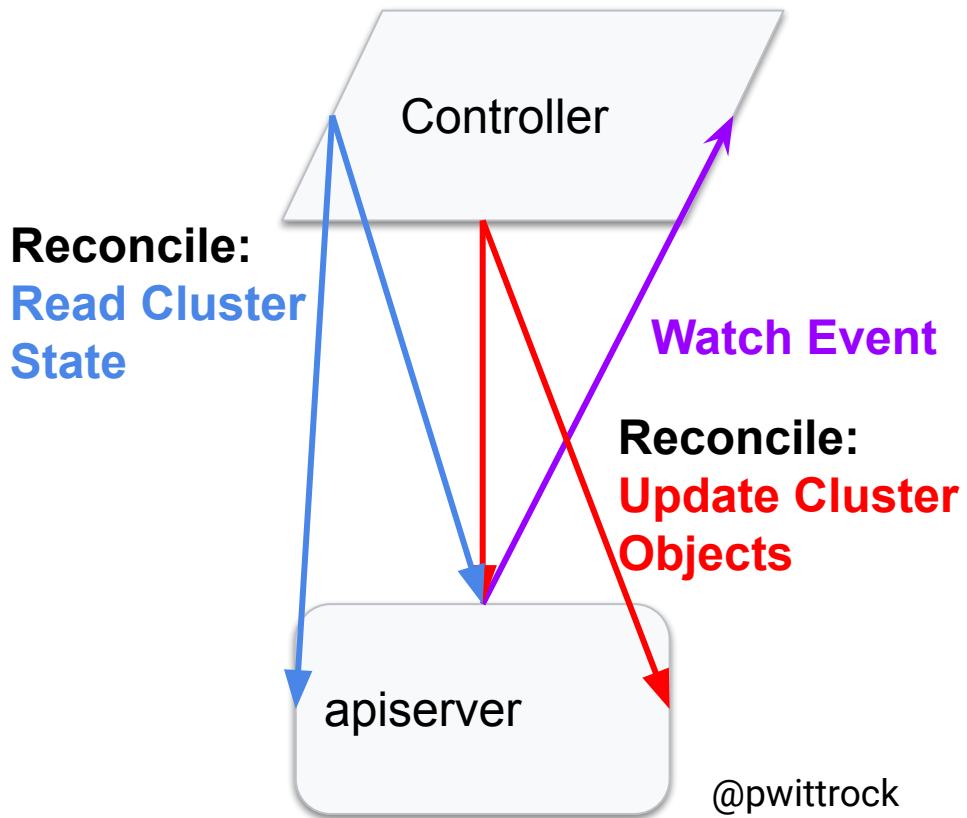
- Internal cleanup with `ownerReferences`
- External cleanup with `finalizers`

Batch events together
into single Reconcile call

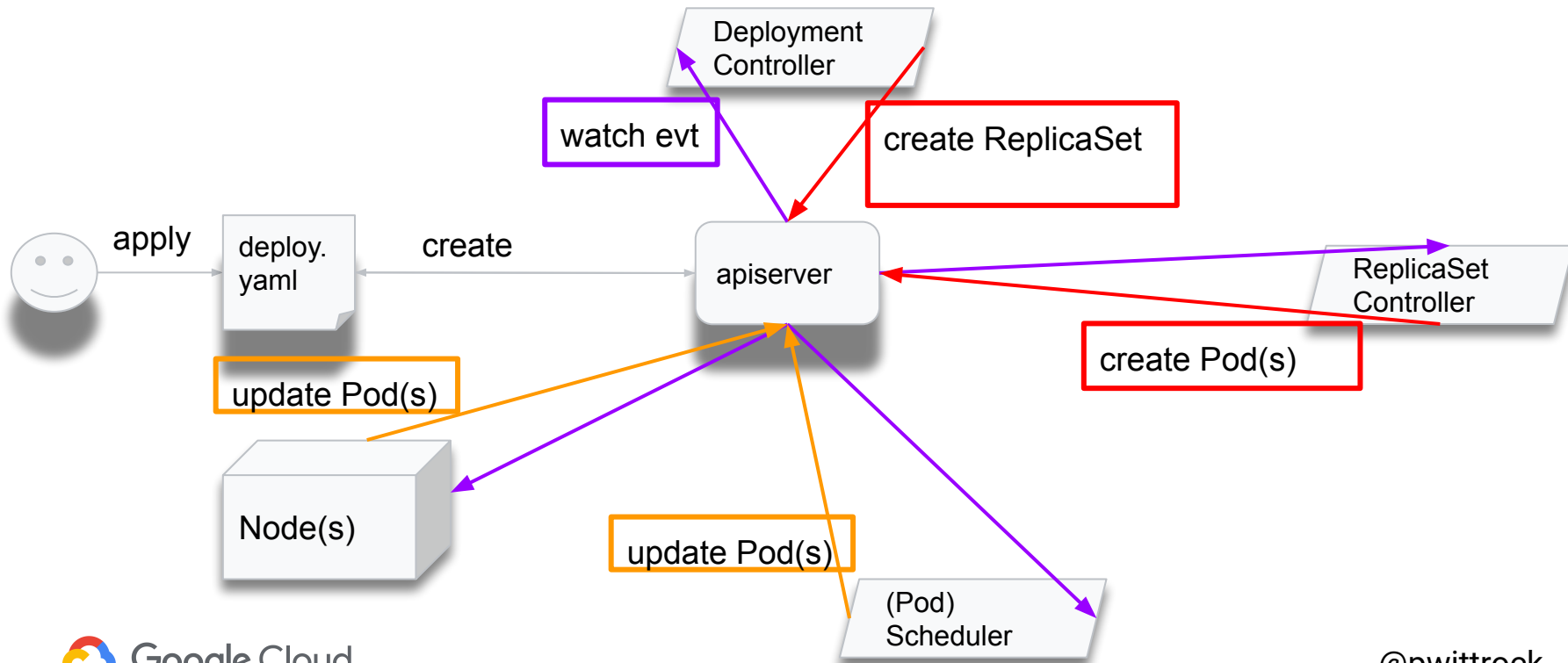
Reconcile on
namespace/name
only, not the event

Controller Workflow

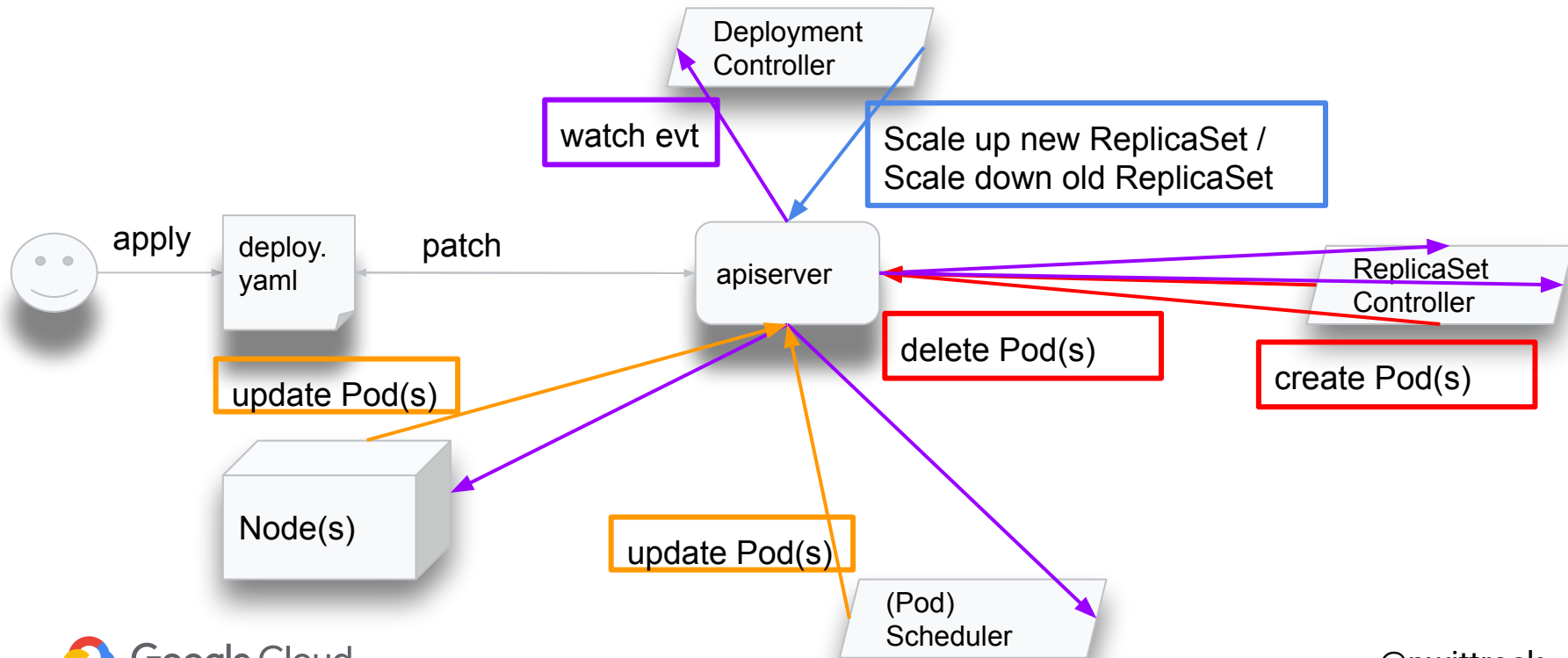
- apiserver streams **Watch Event** to Controller
- Controller **Reads Object + Related Objects** (e.g. Deployment + ReplicaSets)
- Controller **Creates new owned objects, updates owned objects, updates object status**



Kubectl Apply: Create Deployment



Kubectl Apply: Update Deployment



Resources

- Resource Types declare APIs
- Resources provide storage for objects
- Standardized schema structure
- Discoverable API endpoints and schema
- “Just work” with declarative tooling - e.g. `kubectl apply`

metadata	<code>apiVersion: v1</code> <code>kind: Pod</code> metadata: <code> name: my-app</code> <code> namespace: default</code> <code> ...</code>
spec	spec: <code> containers:</code> <code> - args: [sh]</code> <code> image: gcr.io/some-project/udptest</code> <code> imagePullPolicy: Always</code> <code> name: client</code> <code> ...</code> <code> dnsPolicy: ClusterFirst</code> <code> ...</code>
status	status: <code> podIP: 10.8.3.11</code> <code> ...</code>

TypeMeta

- *Kind (Deployment)*
 - Name of the API (e.g. Deployment)
- **Group (apps)**
 - Like a package in go, java, etc (e.g. *apps*)
- Version (v1)
 - Ensures backwards compatibility of: Defaulted Fields & Schema

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: nginx-deployment
  labels: {app: nginx}
spec:
  replicas: 3
  selector:
    ...
  template:
    ...
```

ObjectMeta

- Name and **Namespace* uniquely identify an object for a given Resource
- **Annotations** are arbitrary key-value pairs that cannot be queried
- **Labels** are key-value pairs that may be queried (selected)

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: nginx-deployment
  namespace: default
  labels: {app: nginx}
spec:
  replicas: 3
  selector:
    ...
  template:
    ...
```

Spec and Status

- Spec

- *Object Desired State* (e.g. how many replicas to run, template for Pods, etc)

- Status (not shown)

- Defines the observed state for an object (e.g. how many replicas are running)

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: nginx-deployment
  labels: {app: nginx}
spec:
  replicas: 3
  selector:
    matchLabels: {...}
  template:
    metadata:
      labels: {...}
    spec:
      ...
```


Resource Wiring

- **Labels/Selectors** locate objects
- **Label** - generated objects
- **Selector** - find labeled objects
- **OwnerReference** on generated objects

Tip: Objects with owner references are automatically garbage collected when all of their owners have been deleted

Deployment
name:nginx
labels:run=nginx
selector:run=nginx

ReplicaSet
name:nginx-65899c769f
labels:run=nginx
selector:run=nginx
owner: Deployment nginx

Pod
name:nginx-65899c769f-6slpx

Pod
name:nginx-65899c769f-fbgcv
labels: run=nginx
owner: ReplicaSet nginx-65899c769f

Synchronous Defaulting and Validation

- Unspecified optional fields may be defaulted by the apiserver before the object is stored
- Simple Schema validation performed through OpenAPI
- Complex validation performed by the apiserver before the object is stored

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: nginx-deployment
  labels: {app: nginx}

spec:
  # server defaults this value
  # to 1 if unset
  replicas: 1

  # make sure these match the
  # template labels
  selector:
    matchLabels: {...}
  template:
    metadata:
      labels: {...}
```

Kubernetes APIs are... Extensible

Mutating Webhook +
Service +
Deployment (or Pod)

CustomResourceDefinition
(CRD)

Deployment
(or Pod)

Foo
Admission

Foo
Resource

Foo
Controller

Defaulting, Validation, Version
Conversion

Storage, Schema, Display,
etc

Actuation:
level-triggered,
asynchronous

Updating Resources Gotchas

- Spec has shared ownership across multiple parts of the system
- Controllers or other actors may update the Spec with new fields which must be retained across updates to the object
- Both an issue for Controllers and for users managing Resources using config
- Need to either read-update-write or *apply*

```
apiVersion: v1
kind: Service
metadata:
  name: nginx
  labels: {app: nginx}
spec:
  selector:
    app: nginx
  ports:
    - protocol: TCP
      port: 80
    # not set by owner!
    # don't overwrite!
    clusterIp: 10.0.171.239
    # not set by owner!
    # don't overwrite!
    loadBalancerIp: 78.1124.19
  type: LoadBalancer
```

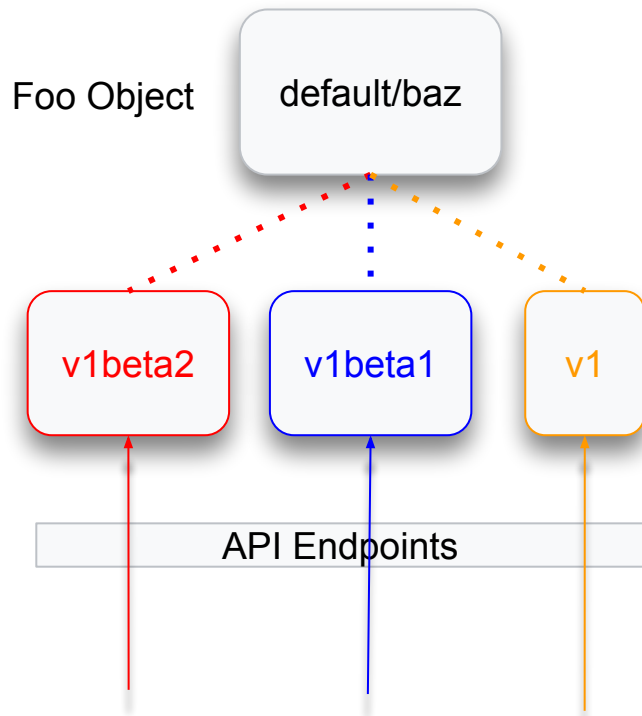
Observing Objects - Status and Events

- Actuation performed asynchronously
- Status published to users, tools and other controllers through Status field
- Conditions: key/value pairs that communicate status (current) to other tools (part of Status field)
- Events: separate objects that communicate past events to users

```
Kind: Pod
...
spec:
  readinessGates:
    - conditionType:
      "www.example.com/feature-1"
status:
  conditions:
    - type: Ready
      status: "False"
      lastProbeTime: null
      lastTransitionTime: ...
    - type: "www.example.com/feature-1"
      status: "False"
      lastProbeTime: null
      lastTransitionTime: ...
  containerStatuses:
    - containerID: docker://abcd...
      ready: true
```

Converting API versions

- Different versions of an API may have different representations
 - Changing default values and field names / field types requires a new version
- All versions of the same API are logically equivalent
- The same object may be read or written in any version -- the underlying object remains the same -- but the endpoints are different.



Classes of APIs

Composites

Operators

Spark, Airflow

Cloud Native Abstractions

Tekton, Knative

Decorators

Autoscalers, Resource
Tuners

Kubernetes APIs Are...

**Declarative, Asynchronous,
Level-Triggered, Observable,**
Discoverable, Versioned, Access
Controlled, **Extensible, ...**

Kubebuilder Workshop

<https://github.com/DirectXMan12/kubebuilder-workshops/tree/software-architecture-2019>

Rate today's session

Cyberconflict: A new era of war, sabotage, and fear

See passes & pricing

David Sanger (The New York Times)
9:55am-10:10am Wednesday, March 27, 2019
Location: Ballroom

Secondary topics: Security and Privacy

Rate This Session

We're living in a new era of constant sabotage, misinformation, and fear, in which everyone is a target, and you're often the collateral damage in a growing conflict among states. From crippling infrastructure to sowing discord and doubt, cyber is now the weapon of choice for democracies, dictators, and terrorists.

David Sanger explains how the rise of cyberweapons has transformed geopolitics like nothing since the invention of the atomic bomb. Moving from the White House Situation Room to the dens of Chinese, Russian, North Korean, and Iranian hackers to the boardrooms of Silicon Valley, David reveals a world coming face-to-face with the perils of technological revolution—a conflict that the United States helped start when it began using cyberweapons against Iranian nuclear plants and North Korean missile launches. But now we find ourselves in a conflict we're uncertain how to control, as our adversaries exploit vulnerabilities in our hyperconnected nation and we struggle to figure out how to deter these complex, short-of-war attacks.

David Sanger
The New York Times

David E. Sanger is the national security correspondent for the *New York Times* as well as a national security and political contributor for CNN and a frequent guest on *CBS This Morning*, *Face the Nation*, and many PBS shows.


Session page on conference website

✓ Attending Notes Remove

Cyberconflict: A new era of war, sabotage, and fear

9:55 AM - 10:10 AM, Wed, Mar 27, 2019


Speakers

 David Sanger
National Security Correspondent
The New York Times

📍 Ballroom

Keynotes

David Sanger explains how the rise of cyberweapons has transformed geopolitics like nothing since the invention of the atomic bomb. From crippling infrastructure to sowing discord and doubt, cyber is now the weapon of choice for democracies, dictators, and terrorists.

 **SESSION EVALUATION**

O'Reilly Events App

Questions?