AVOLENS

CLOUD NATIVE COMPUTING FOUNDATION

TESTING AND FUZZING THE KUBERNETES ADMISSION CONFIGURATION

AVOLENS

CLOUD NATIVE COMPUTING FOUNDATION

$WHOAMI

- Benjamin Koltermann
- CEO of AVOLENS
- Cloud/Kubernetes Security Engineer
- CTF player @fluxfingers
- @p4ck3t0 on Twitter

AVOLENS

CLOUD NATIVE COMPUTING FOUNDATION

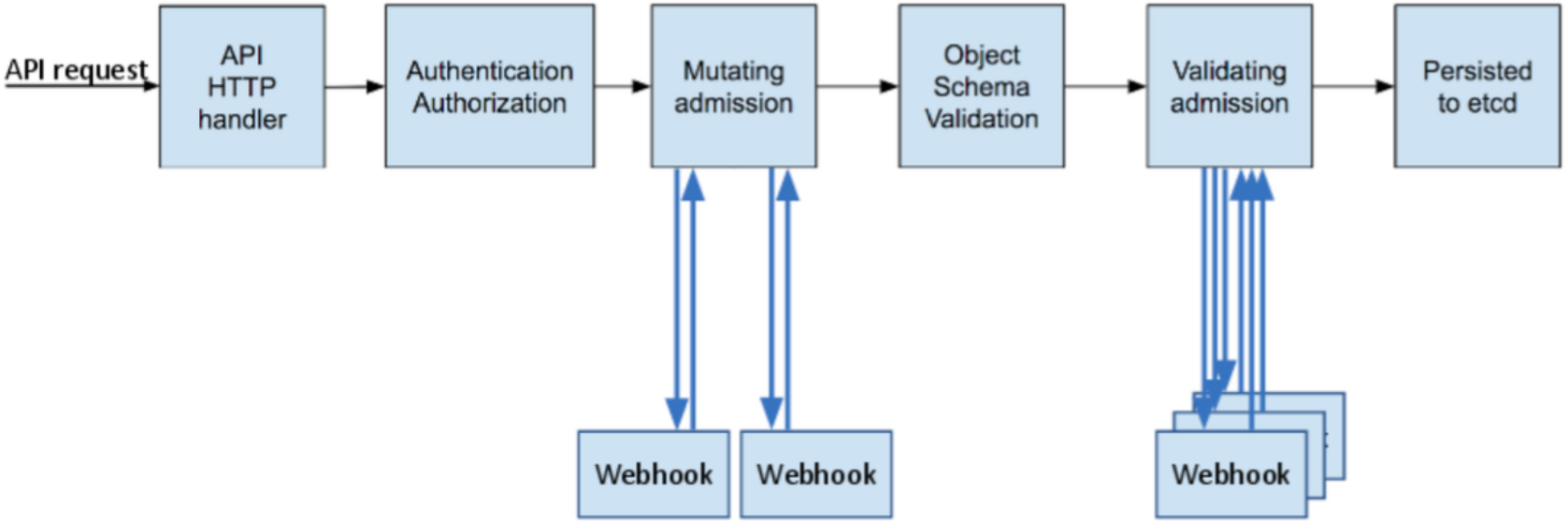
AGENDA

- Kubernetes Intro
- Admission controllers
- Testing Admission Controllers
  - Problem definition
  - Kubernets specific challenges
- KubeFuzz
  - Overview
  - Use cases
  - Demo

KUBERNETES INTRO

- Container orchestration system
- Open and extensible (create your own API/CRDs)
- Everything is an API object

ADMISSION CONTROLLERS



Admission Controllers work like a bouncer for Kubernetes API calls

ADMISSION CONTROLLERS - WHY?

- Mutating and Validating Admission Controllers are a powerful tool to enforce policies
  - Don't allow privileged containers
  - Don't allow containers to run as root
- Validate Kubernetes Objects for CRDs

ADMISSION CONTROLLERS IN THE WILD

- GKE Autopilot Cluster restricts the use of privileged containers
- Elasticsearch Cloud on K8s Validating Admission Controller validates objects used by the Elasticsearch operator
- KernelModule Operator manages out-of-tree kernel modules

TESTING ADMISSION CONTROLLERS

- as shown before, we want our ACs to work as intended
- how do we ensure this?
- not many mature and established methods exist

COMMON APPROACHES

- manual testing
- unit testing
- => fuzzing

AVOLENS

CLOUD NATIVE COMPUTING FOUNDATION

K8S SPECIFIC CHALLENGES

INPUT GENERATION - SCHEMAS

```
<snip>
     "path": {
       "description": "Path to access on the HTTP server.",
       "type": "string"
      },
      "port": {
       "description": "IntOrString is a type that can hold an int32 or a string.  When
       "type": "string",
       "format": "int-or-string"
      },
      "scheme": {
       "description": "Scheme to use for connecting to the host. Defaults to HTTP.\n\n
       "type": "string",
       "enum": [
         "HTTP",
         "HTTPS"
       ]
      }
<snap>
```

- => generate syntactically and semantically correct inputs

K8S SPECIFIC CHALLENGES

EXECUTION

- authenticate against cluster (automatically done thanks to kube-rs)
- select namespace
- submit resource to API
- use dryrun!

K8S SPECIFIC CHALLENGES

COVERAGE/FEEDBACK

- Inputs that trigger yet unseen behavior are interesting
- Api accept/reject
- Denial of service / errors
- AC return codes
- AC messages

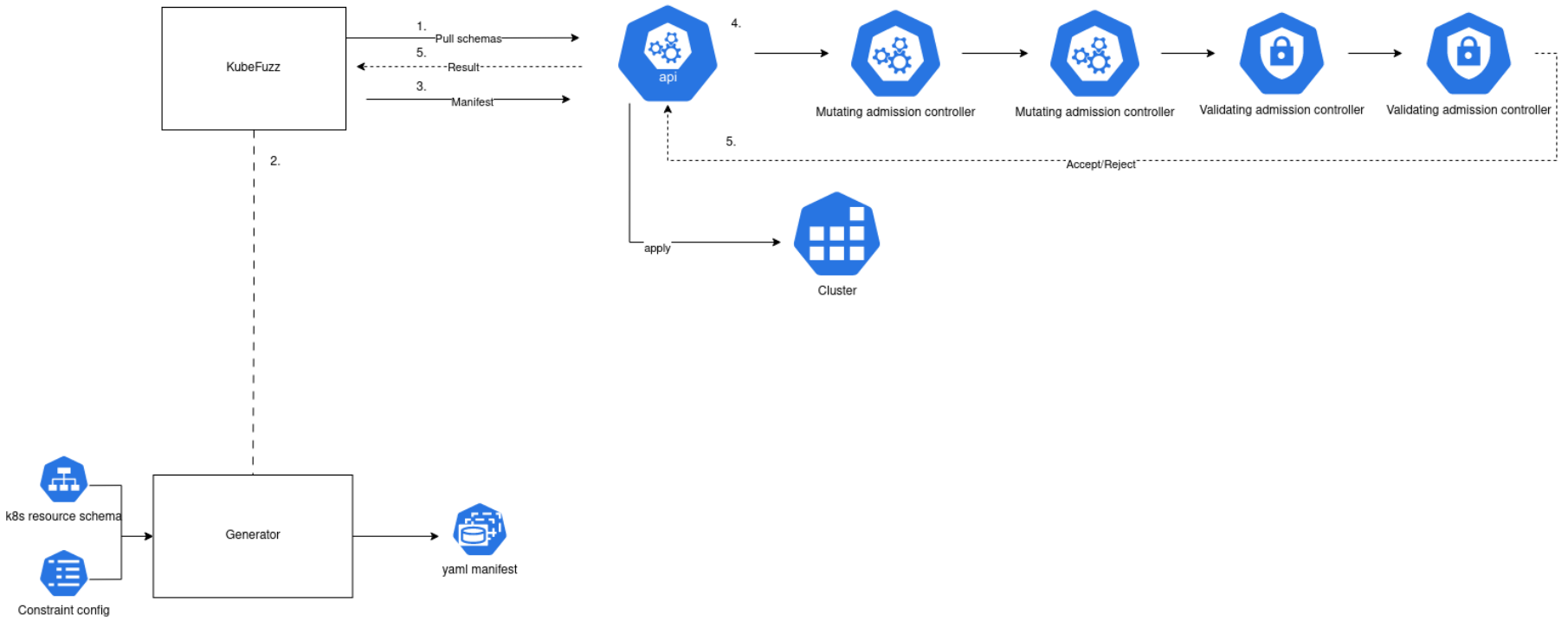
SAMPLE COVERAGE

```
DEBUG kubefuzz::runtime> new coverage: 400 : ... "container privileged"
...
DEBUG kubefuzz::runtime> new coverage: 400 : ... "port forbidden"
...
DEBUG kubefuzz::runtime> new coverage: 400 : ... "image not on whitelist"
```

AVOLENS

CLOUD NATIVE COMPUTING FOUNDATION

KUBEFUZZ



AVOLENS

KUBEFUZZ

CLOUD NATIVE COMPUTING FOUNDATION

CONSTRAINING SPECS

- often, we only care about specific fields
- we might want to more precisely control formats of fields

```
fields:
  - $.status.message

  - path: $.spec.containers
    minmax: [1,5]

  - path: $.spec.containers.securityContext.privileged
    values:
      - true
    required: true

  - path: $.spec.containers.name
    regex_values:
      - 'test-[0-9]{3}'
  - path: .*IP.*
    regex: true
    values:
      - 127.0.0.2

group: ""
version: "v1"
```

KubeFuzz will obey formats, types and enums and try to guess formats

KUBEFUZZ

USE CASES

- test existing AC configuration for unexpected allowed manifests and errors
- test stability of new ACs (developer scenario)
- differential testing of differnet ACs

KUBEFUZZ

DEMO

AVOLENS

CLOUD NATIVE COMPUTING FOUNDATION

THANK YOU!

Github: https://github.com/avolens/kubefuzz

Website: https://kubefuzz.io