AVOLENS

TROOPERS

TESTING AND FUZZING THE KUBERNETES ADMISSION CONFIGURATION

AVOLENS

TROOPERS

$WHOAMI

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

AVOLENS

TROOPERS

>WHOAMI

- Maximilian Rademacher
- Vulnerability Researcher at AVOLENS
- CTF player @fluxfingers
- @_tunn3l on Twitter

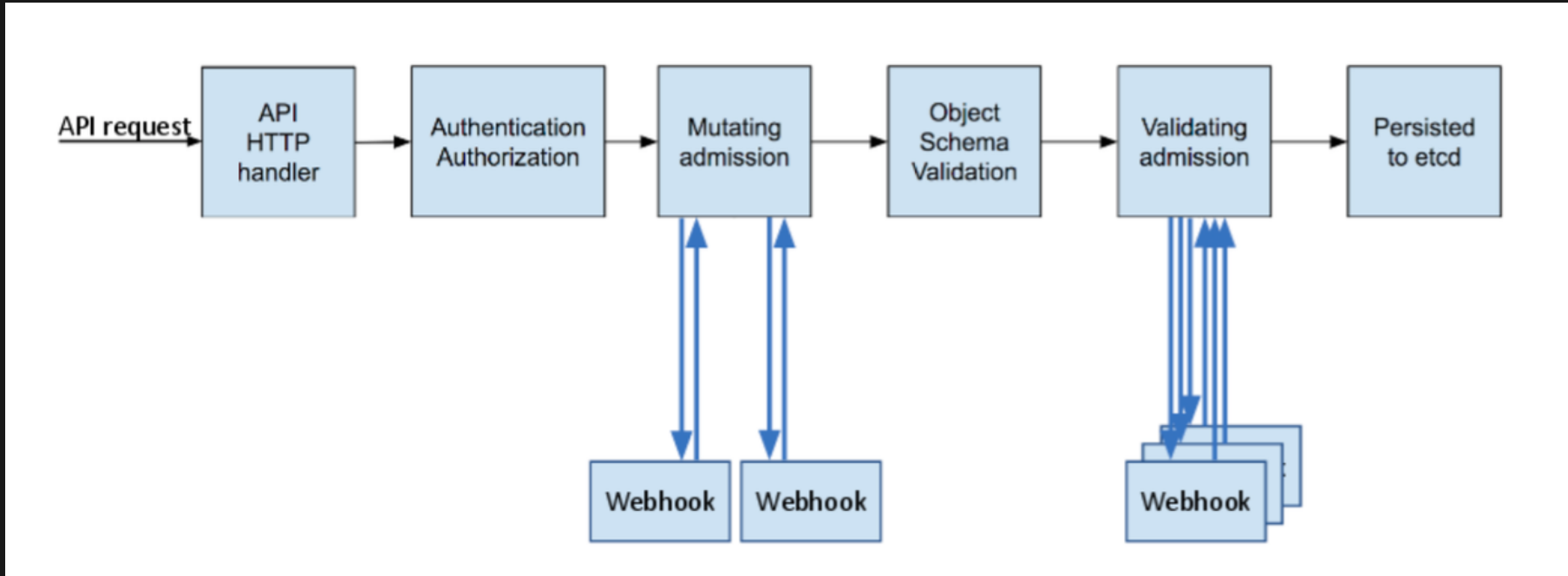
AGENDA

- Kubernetes Intro
- Admission controllers
  - Overview
  - Best Practices
  - Real world examples
- Testing Admission Controllers
  - Problem definition
  - Fuzzing as a solution
  - Kubernets specific hurdles

KUBERNETES INTRO

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

ADMISSION CONTROLLERS



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 - BEST PRACTICES

- Running inside the Kubernetes Cluster
- Failure policy is set to Fail
- Written in Go/Rust
  - Use the standard libraries

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

FUZZING

- Automatically generate inputs and see if program behaves unexpectedly

```
CORPUS = []

while True:
    sample = generate_input(CORPUS)


    result = run_program(sample)

    if is_intersting(result):
         CORPUS.append(sample)
```

  - typical challenges:
    - how to generate inputs?
    - how to feed samples to the program?
    - how to determine if result is interesting?

K8S SPECIFIC CHALLENGES

INPUT GENERATION

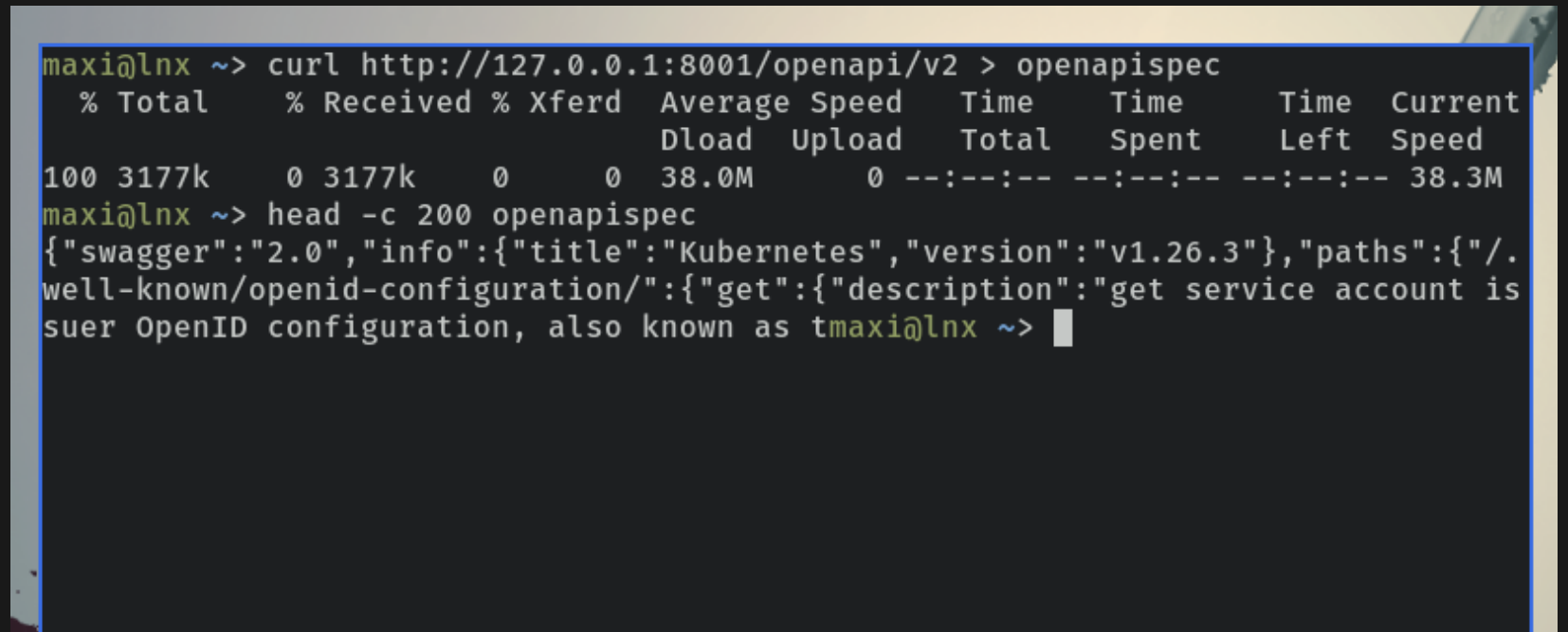
- How do we know what resources are available?
- How do we know how resources look like?
- => every resource is described in the cluster OpenAPI specificaiton
- cluster.local/openapi/v2

AVOLENS

K8S SPECIFIC CHALLENGES

TROOPERS

INPUT GENERATION - OPENAPI



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.  W

        "type": "string",
        "format": "int-or-string"
      },
      "scheme": {
        "description": "Scheme to use for connecting to the host. Defaults to HTTP.\
        "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

AVOLENS

TROOPERS

K8S ADMISSIONREVIEW

```
{
   "apiVersion":"admission.k8s.io/v1",
   "kind":"AdmissionReview",
   "response":{

      "uid":"763d425931716b6f5568635751...",
      "allowed":false,
      "status":{
         "code":403,
         "message":"You cannot do this because it is Tuesday and your name starts wi
      }
   }
}
```

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"
```

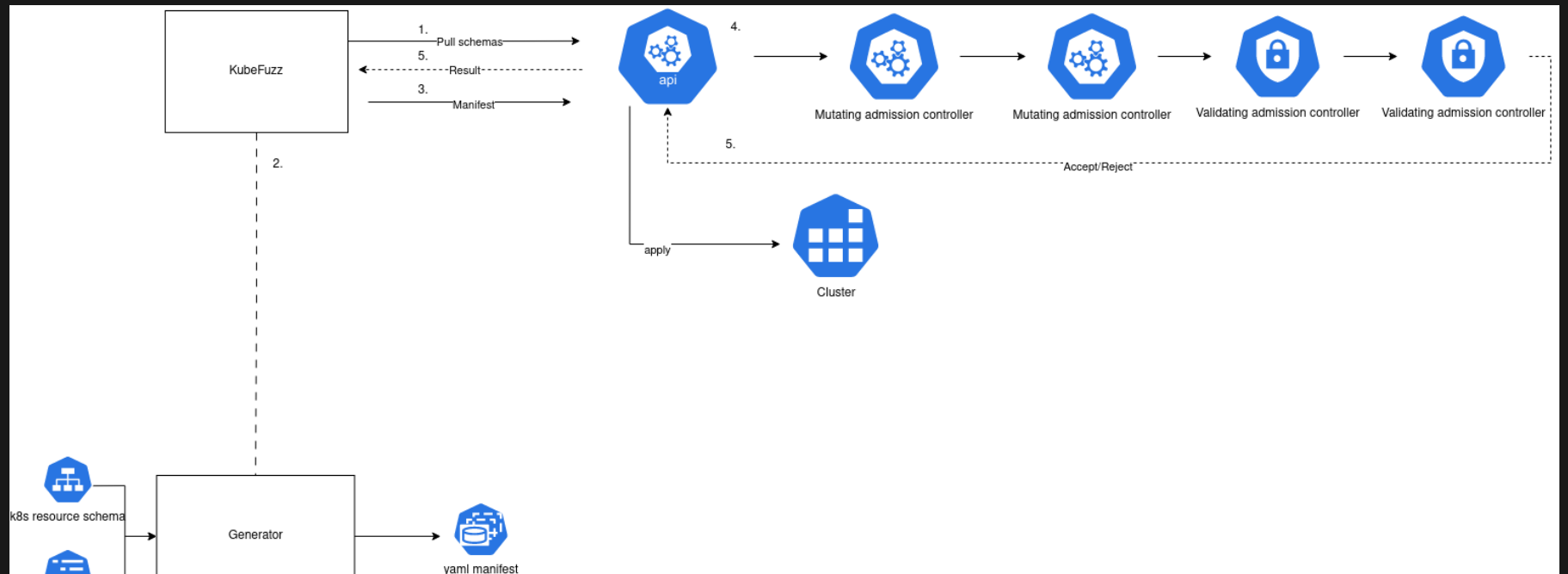
WHAT ARE WE SEARCHING FOR?

- unwanted manifests being allowed (logic bugs)
- runtime crashes (DOS or AC Bypasses via Failopen policy)

AVOLENS

TROOPERS

KUBEFUZZ



AVOLENS

KUBEFUZZ

TROOPERS

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

GENERATE WHOLE RESOURCE RANDOMLY

```
fields:
    - path: "$."
      required: true

group: ""
version: "v1"
kind: "Pod"
```

- KubeFuzz will obey formats, types and enums and try to guess formats based on names (e.g port/username)

AVOLENS

TROOPERS

KUBEFUZZ

CLI

```
maxi@lnx ~> kubefuzz fuzz --help
fuzz admission controller chain with constraints

Usage: kubefuzz fuzz [OPTIONS] --constraints <CONSTRAINTS> --schemadir <SCHEMADIR> --

Options:
  -k, --kubeconfig <KUBECONFIG>    optional custom path to kubeconfig
  -c, --constraints <CONSTRAINTS>  comma seperated list of constraint files to apply
  -s, --schemadir <SCHEMADIR>      directory containing k8s json resource schemas
  -n, --namespaces <NAMESPACES>    comma seperated list of namespaces to fuzz [defaul
  -f, --fuzzdir <FUZZDIR>          directory to save and update fuzzing results
  -t, --timeout <TIMEOUT>          time in seconds until an api request is considered
  -h, --help
      Print help

maxi@lnx ~/kubefuzz>
```

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

HURDLES DURING DEVELOPMENT

- K8s resource specs aren't always fully descriptive
  - some CRD's are not typed
  - qunataties rely on description to be identified as such
  - formats such as int32 do not have a min/max (but are checked at runtime for k8s native resources)
  - some required fields are not marked as such
  - some semantic poroperties are not described (e.g field a can only exist if field b = x)
- Admission controller side effects and coverage problems

KUBEFUZZ

FUTURE IMPROVEMENTS

- coverage:
  - process optional warnings emitted
  - process resource differentials (mutating AC)
- plugin system for post processing (crcs/custom formats)

KUBEFUZZ

DEMO

AVOLENS

TROOPERS

THANK YOU!

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