

Intro to Sensu



2018-10-13
Ohio LinuxFest
Columbus, OH

Garrett Honeycutt
gh@learnpuppet.com
learnpuppet.com
 **@learnpuppet**



Nagios®

Monitoring with Nagios and graphing with PerfParse



CINLUG
Central INdiana LINux Users Group
www.cinlug.org

Garrett Honeycutt
2005-10-05

Nagios[®]

0 - OK

1 - Warning

2 - Critical

3 - Unknown




/sensu/sensu-puppet



sensu
sensu

PARTNER

A module to install the Sensu monitoring framework
Version 2.55.1

Updated	10 days ago
Total downloads	2,488,433
Quality score	4.8 



@learnpuppet



/sensu/sensu-puppet



VAGRANT

What is Sensu?

Flexible monitoring framework

Router for your metrics and alerts

Status checks and telemetry at scale

Designed for dynamic environments

Why Sensu?

- Collect metrics and status checks with one system

Why Sensu?

- Collect metrics and status checks with one system
- Interface with other monitoring and metrics systems

Why Sensu?

- Collect metrics and status checks with one system
- Interface with other monitoring and metrics systems
- Reuse existing Nagios plugins

Why Sensu?

- Collect metrics and status checks with one system
- Interface with other monitoring and metrics systems
- Reuse existing Nagios plugins
- Plugins can be written in any language

Why Sensu?

- Collect metrics and status checks with one system
- Interface with other monitoring and metrics systems
- Reuse existing Nagios plugins
- Plugins can be written in any language
- Decoupled model for easy-to-scale architecture

Why Sensu?

- Collect metrics and status checks with one system
- Interface with other monitoring and metrics systems
- Reuse existing Nagios plugins
- Plugins can be written in any language
- Decoupled model for easy-to-scale architecture
- Great user community! (<http://sensucommunity.slack.com/>)



/sensu-plugins

- 5+ Years
- 500 Checks, handlers, mutators
- 400 Contributors
- 1100 Pull requests
- 3000 commits

Monitoring at Scale

1:N execution scheduling

One execution request yields results
from N clients

Dynamic Environments

Monitoring ephemeral systems/resources

Dynamic Environments

Driven by configuration management

Sensu Core

Open Source under MIT license

Sensu Core



Written in Ruby

Sensu Core

First commit July 10, 2011

Sensu Core

1.0 released July 7th, 2017

Sensu Core

2.0 alpha released February 27th, 2018

Sensu Enterprise

- Sensu Enterprise built on Sensu Core
- Drop-in replacement for sensu-server & sensu-api
- Proprietary license with commercial support
- Runs on JRuby for improved performance
- Adds enhanced integrations, contact routing and more

Architecture Overview

- Transport
- Data Store
- Monitoring Agent (client)
- Event Processor (server)
- RESTful API
- Dashboard

Transport

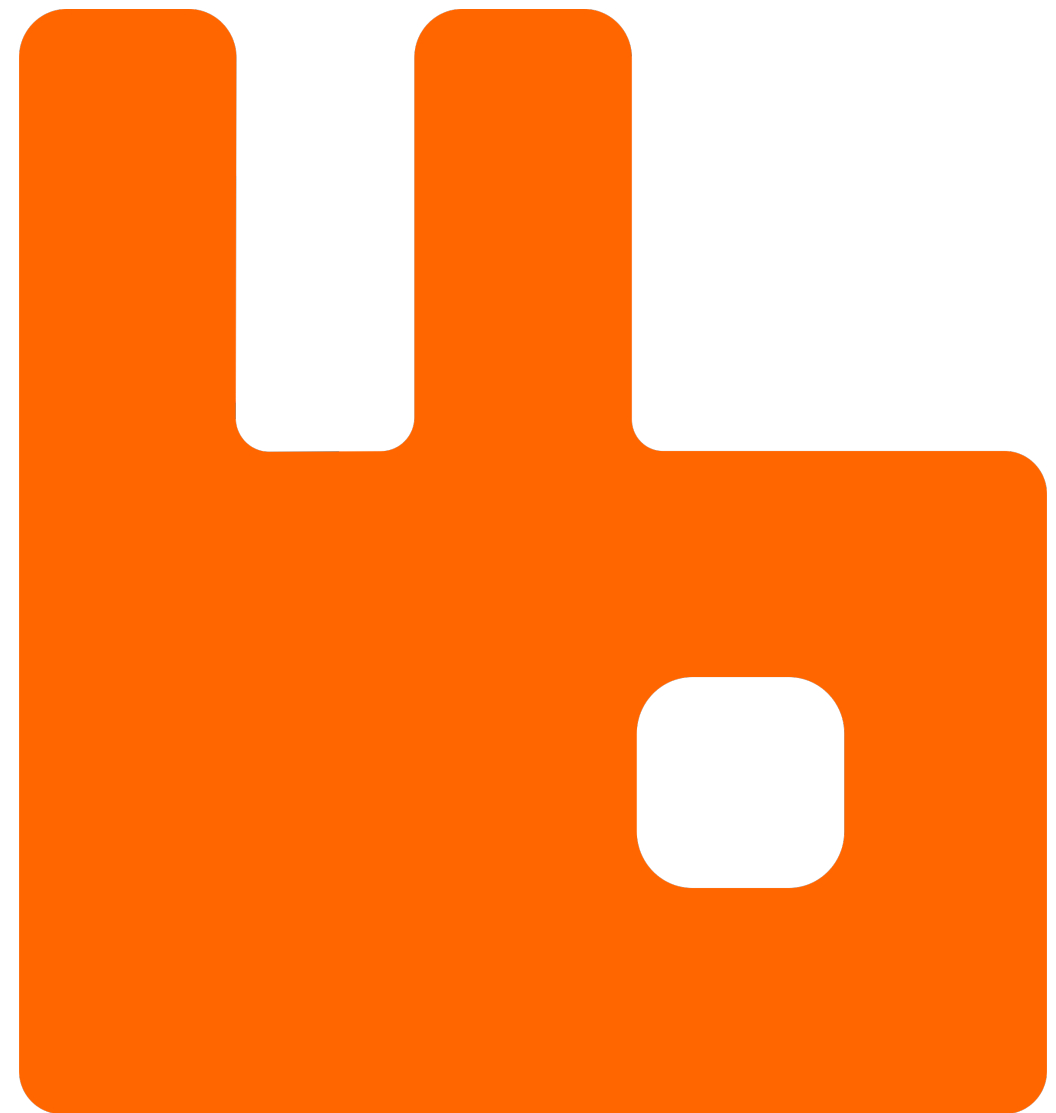
Message bus that enables
decoupling

Transport

Reduces need for service discovery

RabbitMQ as Transport

- TLS with peer verification
- Multi-user with ACLs
- Clustering and federation (WAN)



Data store

Sensu services are stateless

Data Store



Redis

Data Stored

Stored data is generally ephemeral

Data Stored



Data Stored

Some history for context

Monitoring Agent

Sensu Core package provides
sensu-client (agent)

Check execution scheduler

Subscription (server)

- Sensu server publishes execution requests
- Sensu agent executes check commands
- Check result messages are published to transport

Check execution scheduler

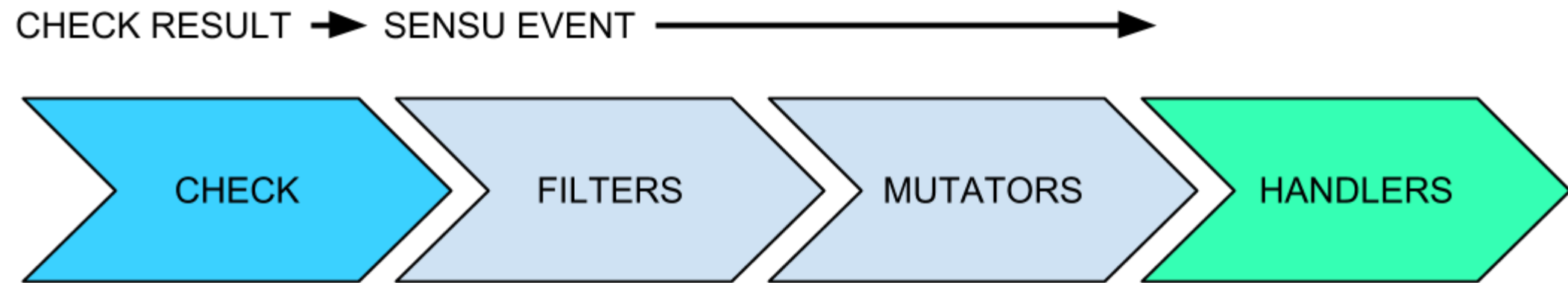
Stand alone (client)

- Sensu agent schedules check execution
- Sensu agent executes check commands
- Check result messages are published to transport

Event Processor

- Maintains client and incident registries
- Processes check result messages to create events
- Processes events according to configuration
 - Filters improve signal-to-noise ratio
 - Mutators transform event data format
 - Handlers act on event data

Check Result Life Cycle





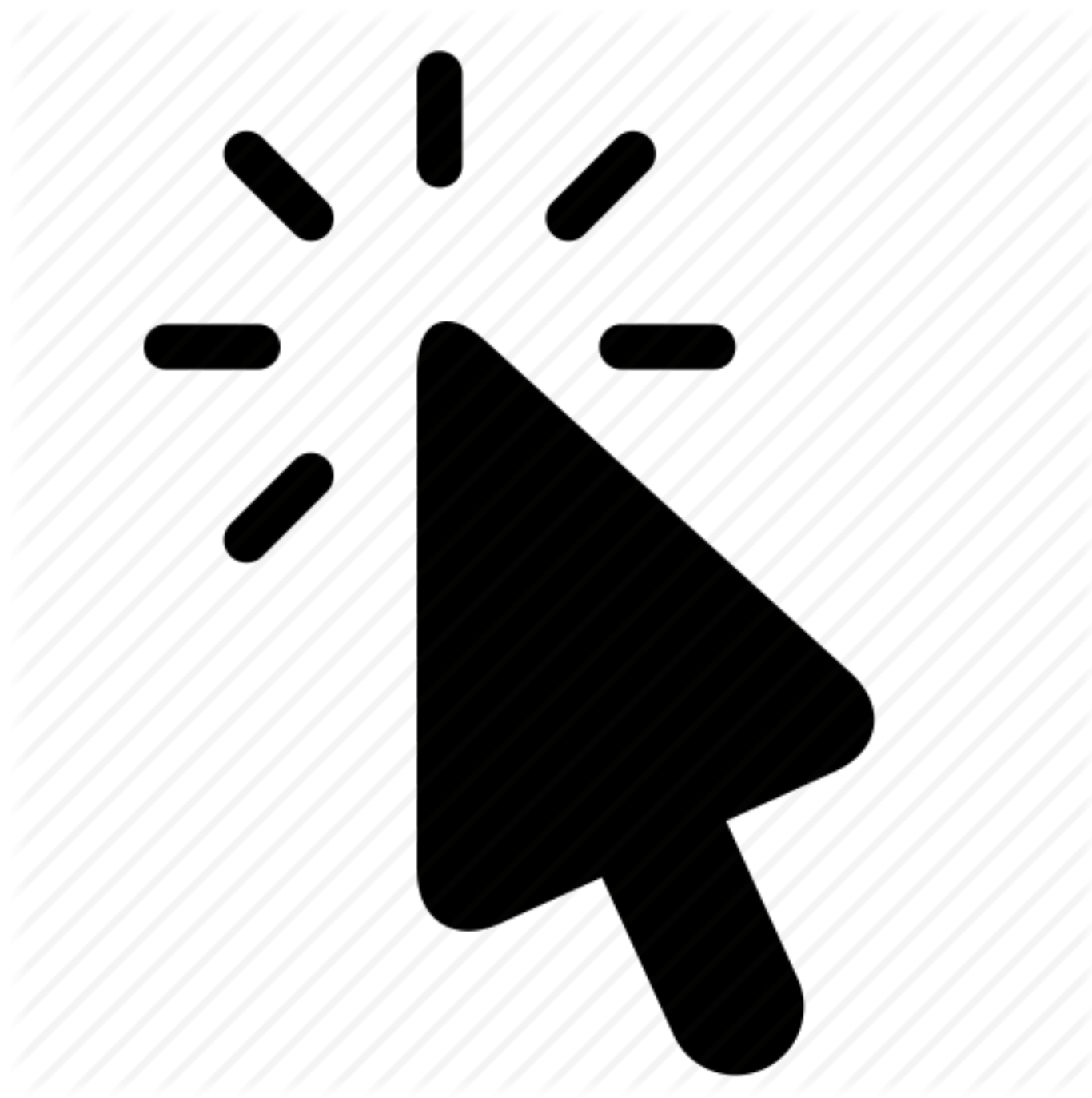
 librato.
pagerduty



influxdb



RESTful API



No Click Fail

RESTful API Endpoints

/clients

/checks

/incidents

/results

/request

/silenced

Dashboard

Uchiwa open source dashboard

Uchiwa

Demo time

Review: Sensus Flow

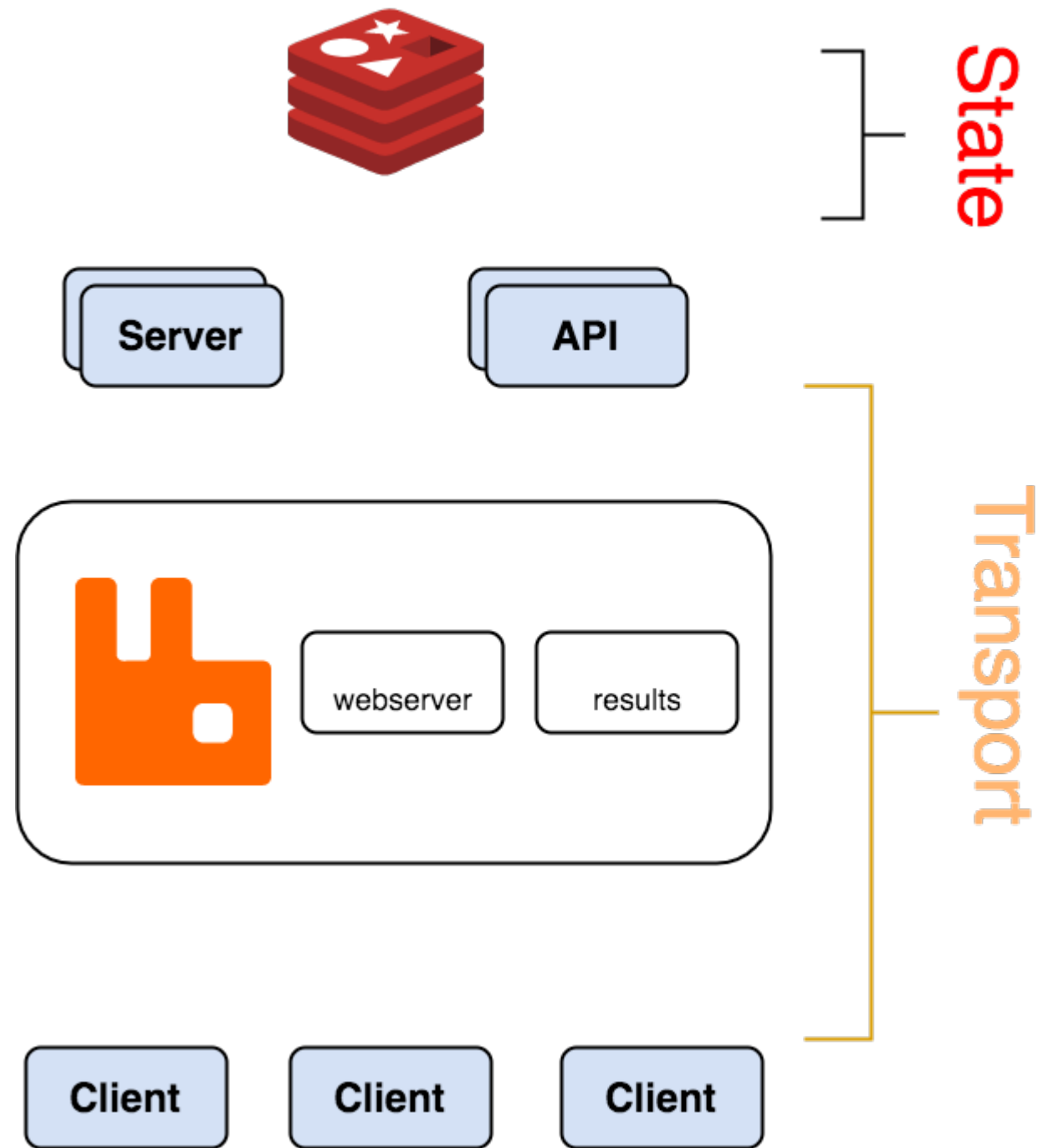
- Check result is published to Transport

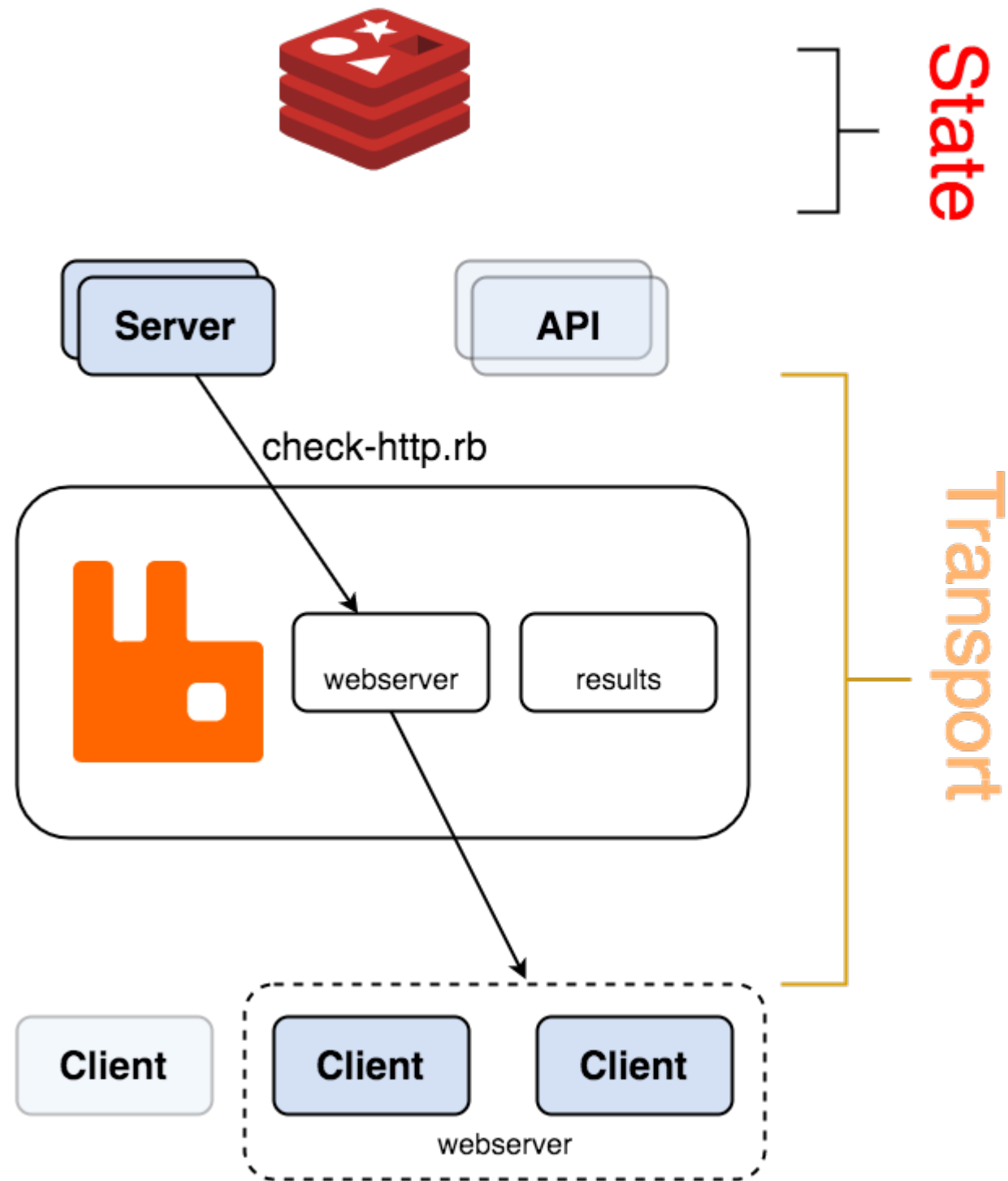
Review: Sensus Flow

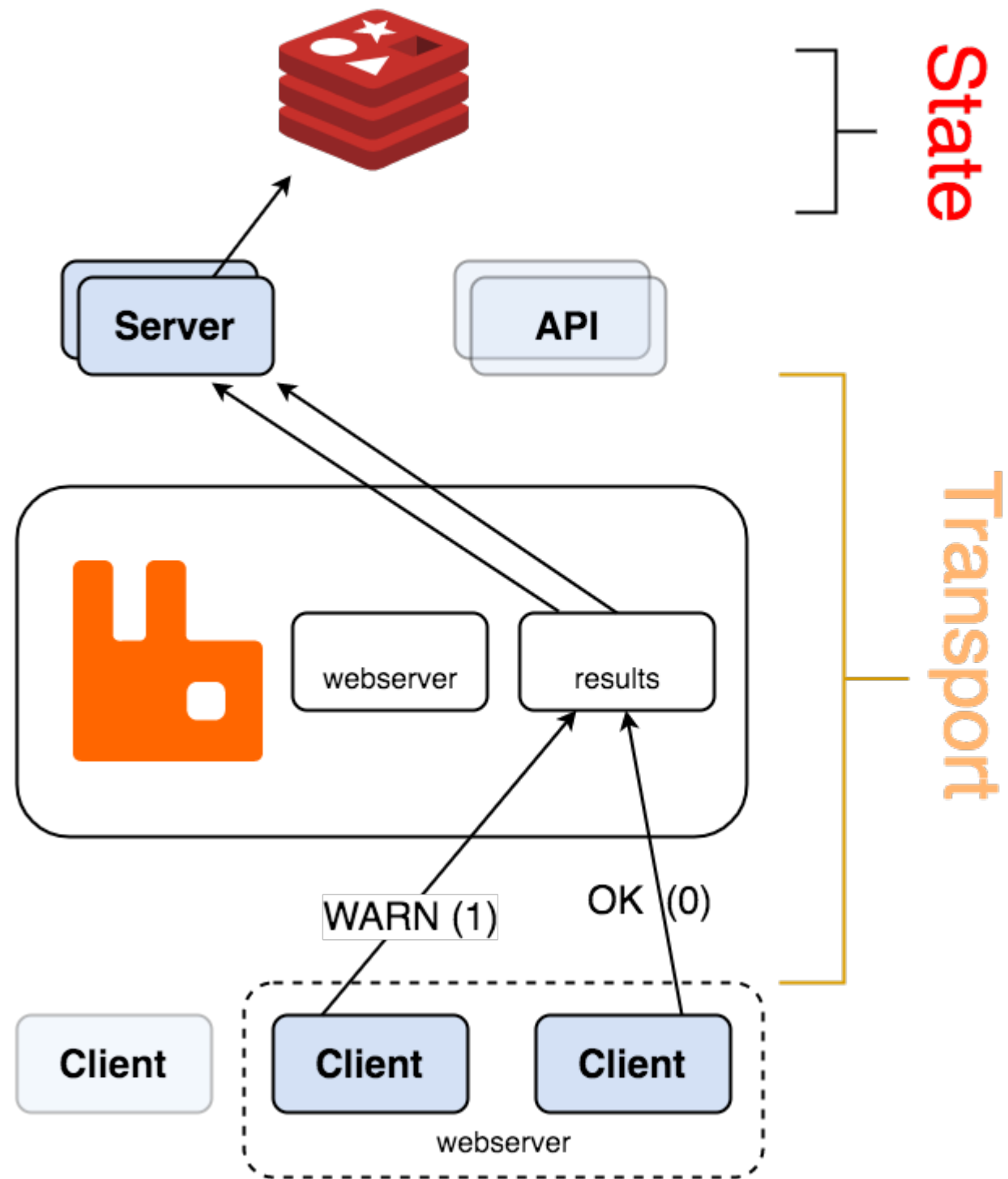
- Check result is published to Transport
- Event processor (server) reads result from Transport
 - Data Store is updated based on result
 - Result combined with additional context becomes an event
 - Filters, mutators and handlers evaluate and take action on event

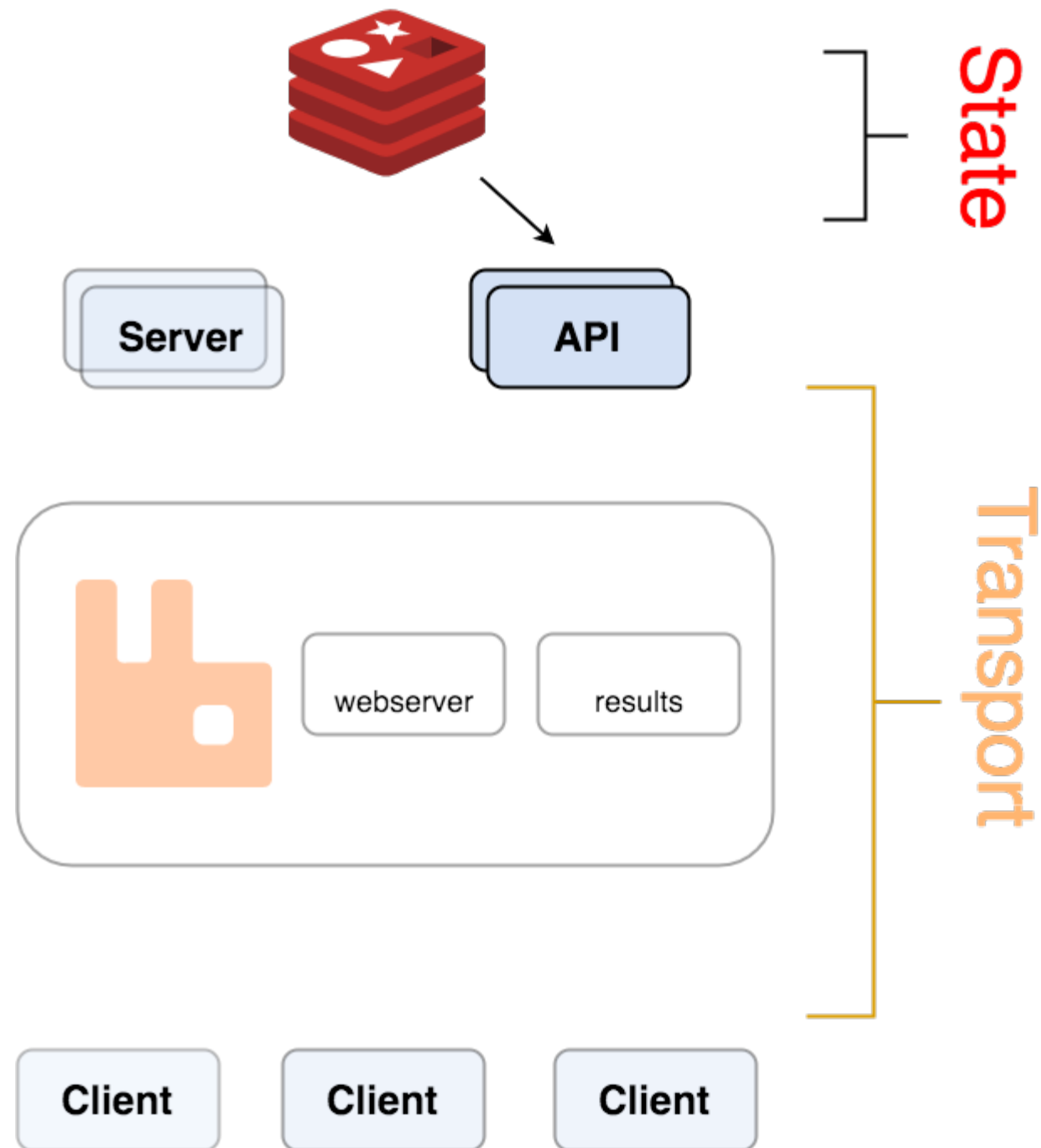
Review: Sensus Flow

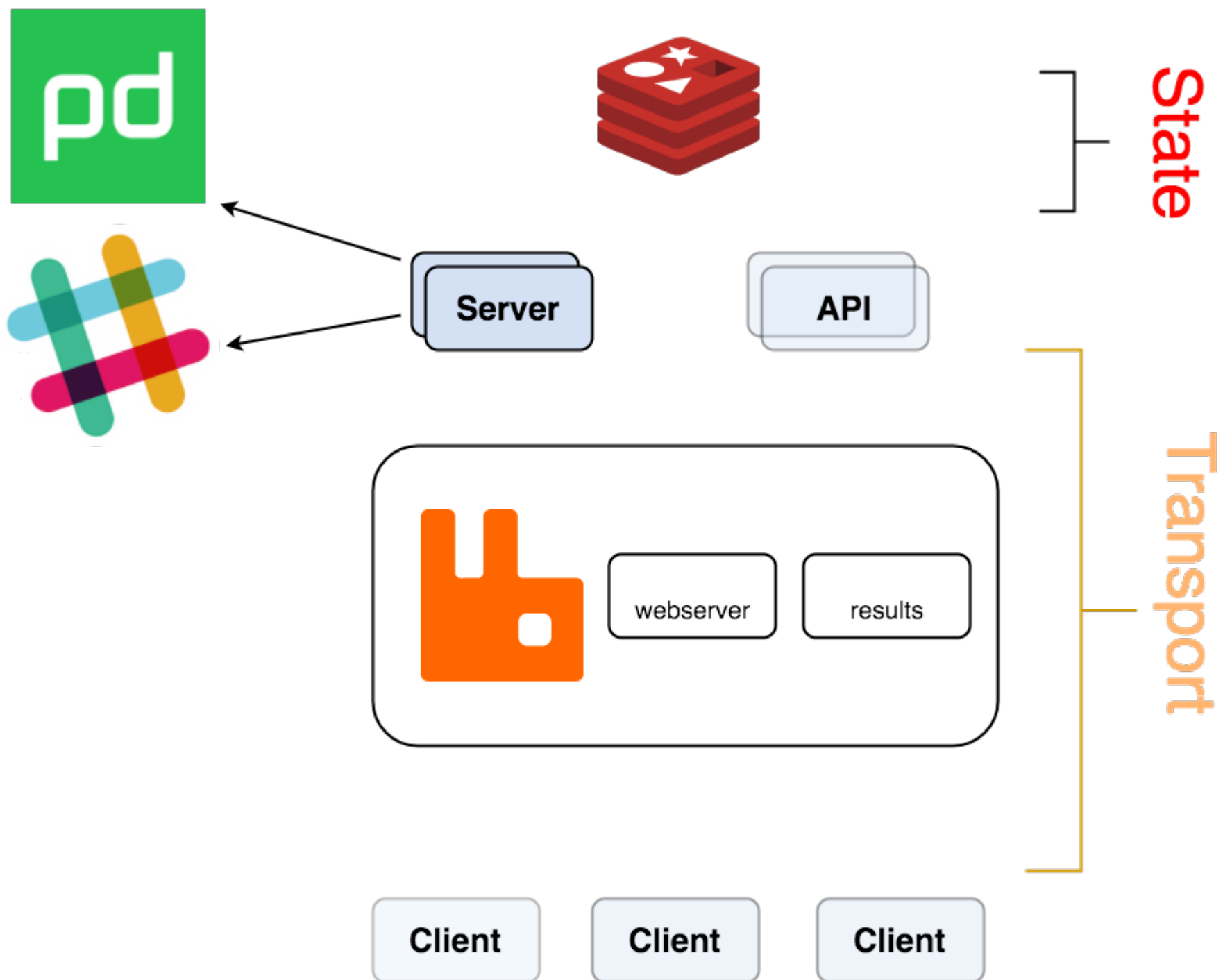
- Check result is published to Transport
- Event processor (server) reads result from Transport
 - Data Store is updated based on result
 - Result combined with additional context becomes an event
 - Filters, mutators and handlers evaluate and take action on event
- API and Dashboard expose state from Data Store











Configuration Philosophy

inspired by UNIX-style chaining of
tools

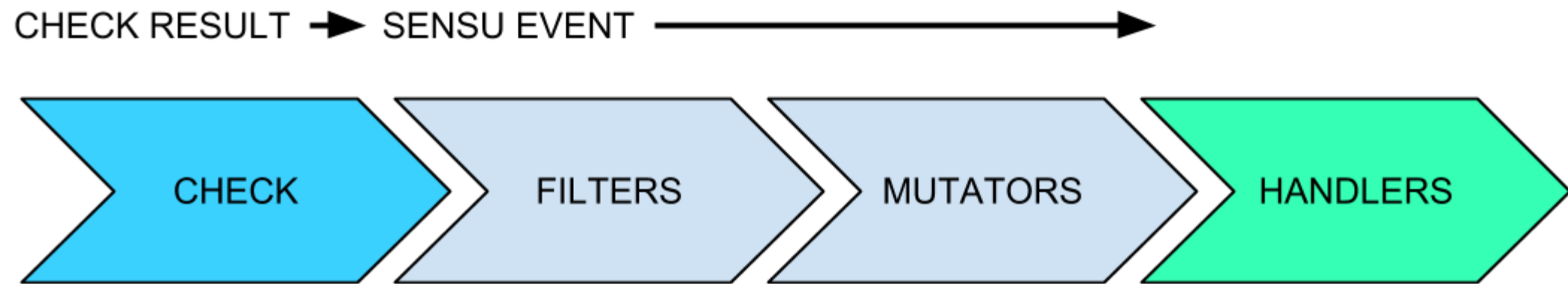
Configuration Conventions

JSON configuration

`/etc/sensu/config.json`

`/etc/sensu/conf.d/**/*.*.json`

Sensu Primitives



Checks

Monitor status or collect
measurement

Checks

Configured on agent or server

Checks

Configuration on disk can provide client-specific overrides and/or security

Example: Check Definition

```
$ cat /etc/sensu/conf.d/checks/check_haproxy.json
```

```
{  
  "name": "check_haproxy",  
  "command": "check_haproxy.rb -s app1 -w 40 -c 25",  
  "subscribers": ["load_balancer"],  
  "interval": 30  
}
```

Example: Check Result

```
{  
  "name": "check_haproxy",  
  "command": "check_haproxy.rb -s app01 -w 40 -c 25",  
  "subscribers": ["load_balancer"],  
  "interval": 30,  
  "timestamp": 1445569640,  
  "output": "UP - 85% of 10 /app01/ services",  
  "status": 0,  
  "duration": 0.87  
}
```

Metric Checks

Collect measurements

Metric Checks

Unlike status checks, OK events are sent to handlers

Events

Created for every check result

Events

- An event object is created for every check result
- Events with **ok** status update history in data store
- Events should be passed to handlers when they:
 - Indicate an unhealthy state
 - Contain formatted data (metrics)
 - Change from an unhealthy state to healthy (resolve)

Example: Event Data

```
{  
  "id": "42ccc91a-5449-4060-b7c3-c444db177fd7",  
  "client": {"...": "..."},  
  "check": {"...": "..."},  
  "occurrences": 1,  
  "action": "create",  
  "timestamp": 1445569640  
}
```


Check hooks

Auto remediation

```
{
  "checks": {
    "nginx_process": {
      "command": "check-process.rb -p nginx",
      "subscribers": ["proxy"],
      "interval": 30,
      "hooks": {
        "critical": {
          "command": "sudo /etc/init.d/nginx start",
          "timeout": 30
        },
        "non-zero": {
          "command": "ps aux",
          "timeout": 10
        }
      }
    }
  }
}
```

Filters

Improve signal to noise

Example: Filter Definition

```
$ cat /etc/sensu/conf.d/filters/office_hours.json
{
  "filters": {
    "nine_to_fiver": {
      "negate": false,
      "attributes": {
        "timestamp": "eval: [1,2,3,4,5].include?
(Time.at(value).wday) &&
Time.at(value).hour.between?(9,17) "
      }
    }
  }
}
```

Mutators

Transform event data

Example: Mutator Definition

```
$ cat /etc/sensu/conf.d/mutators/graphite_events.json
```

```
{  
  "mutators": {  
    "graphite_events": {  
      "command": "graphite_events.rb",  
      "timeout": 10  
    }  
  }  
}
```

Mutator Example: graphite-event.rb

Input:

```
{
  "id": "42ccc91a-5449-4060-b7c3-c444db177fd7",
  "client": { "name": "i-424242", ... },
  "check": { "name": "check_haproxy", "status":
1, ... },
  "occurrences": 1,
  "action": "create",
  "timestamp": 1445569640
}
```

Output:

```
sensu.i-424242.check_haproxy 1 1445569640
```

Handlers

Take action on events

Example: Handler Definition w/ Filter

```
$ cat /etc/sensu/conf.d/handlers/pushover.json
```

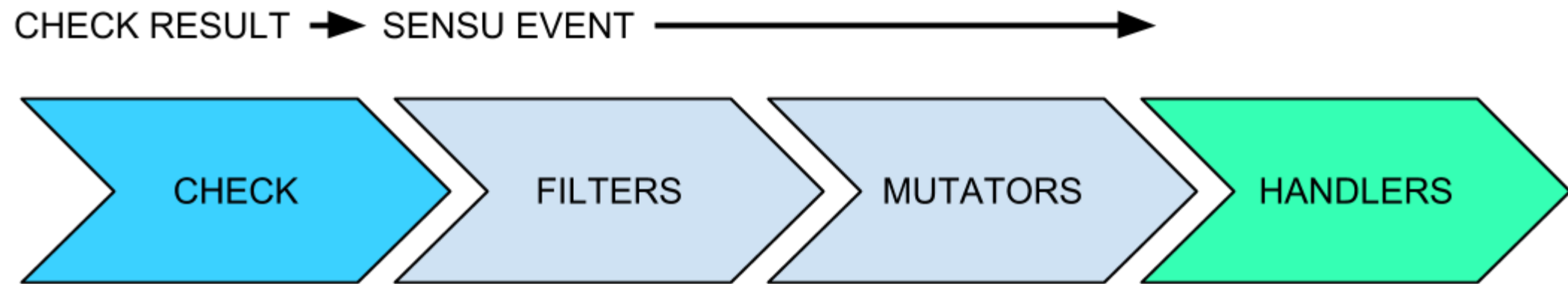
```
{  
  "handlers": {  
    "pushover": {  
      "type": "pipe",  
      "command": "handler-pushover.rb",  
      "filters": ["only_state_change"]  
    }  
  }  
}
```

Example: Handler Definition with Mutator

```
$ cat /etc/sensu/conf.d/handlers/graphite_event.json
```

```
{
  "handlers": {
    "sensu_events_to_graphite": {
      "type": "tcp",
      "socket": {
        "host": "graphite.example.com",
        "port": 2003
      },
      "mutator": "graphite_events"
    }
  }
}
```

Sensu Primitives



Sensu v2

Funded

Go



Self Contained

Configuration shared between backends




```
# resource.json
{
  "type": "CheckConfig",
  "spec": {
    "name": "marketing-site",
    "command": "check-http.rb",
    "subscriptions": ["demo"],
    "interval": 15,
    "organization": "default",
    "environment": "default"
  }
}

$ sensuctl create -f resource.json
```

Recap

Flexible monitoring framework



/sensu/sensu-puppet



VAGRANT

Intro to Sensu



2018-10-13
Ohio LinuxFest
Columbus, OH

Garrett Honeycutt
gh@learnpuppet.com
learnpuppet.com
 **@learnpuppet**