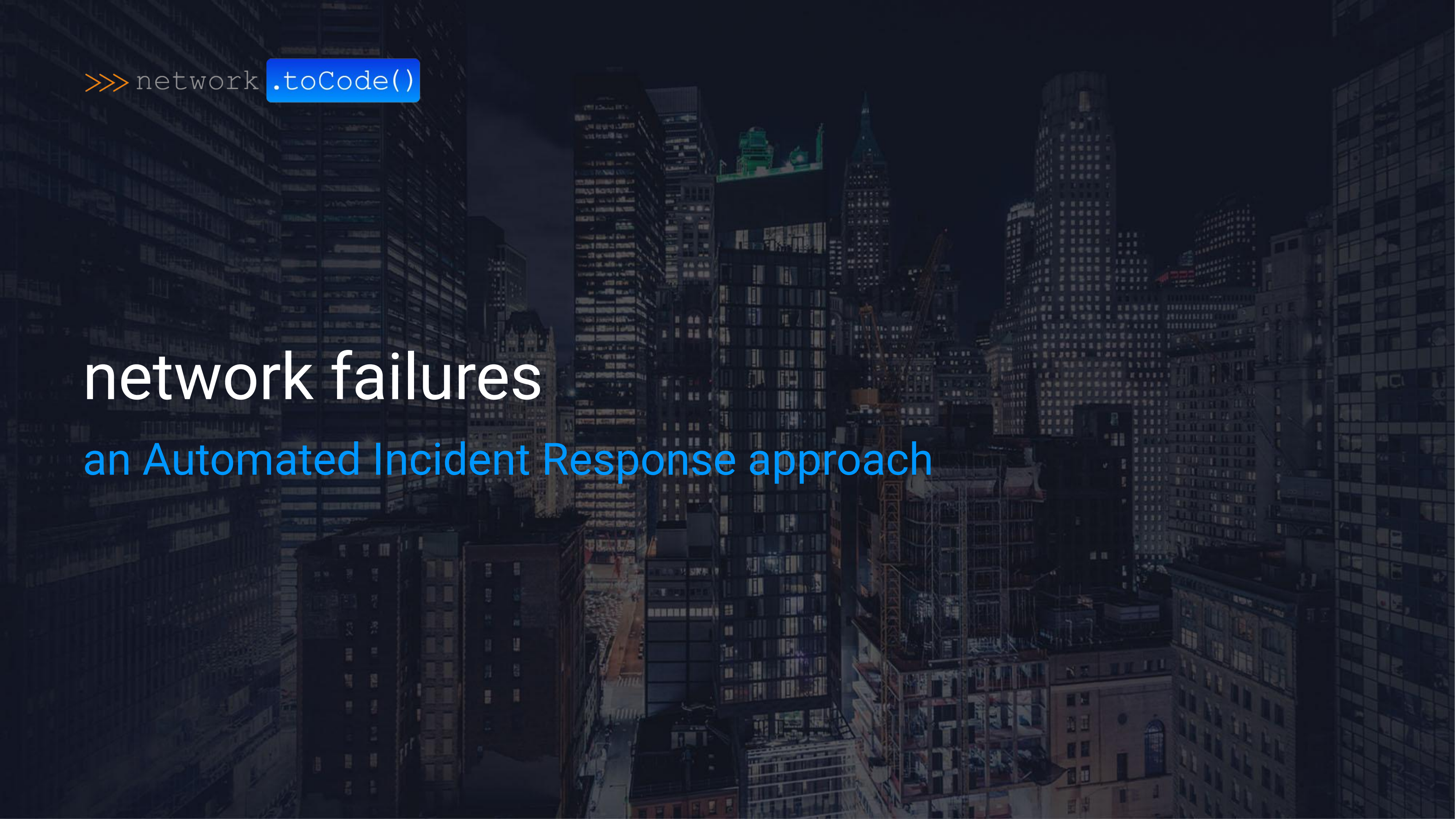


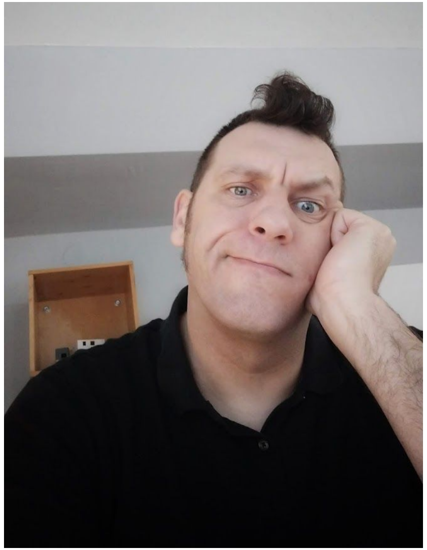
>>> network.toCode()

network failures

an Automated Incident Response approach

>>> introduction

- with NTC since November 2021
- jack of all trades, master of none
- great sandwich maker

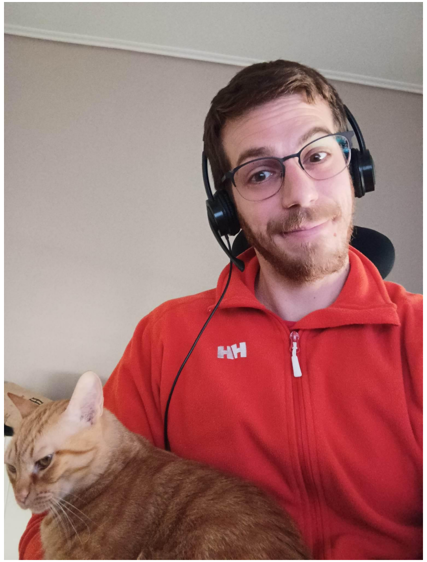


in @nkallergis

@nkallergis

Introduction

- Network Automation Engineer at NTC
- Experience in Networking & Network Security
- Enjoy working with systems in general
- Live in Athens, Greece
- Cat lover (the ginger beauty is called Yuppie)



in @gtzakis

@gertzakis

Agenda

- current state of incident response
- automated alerting is nice :)
- AIR is a cool acronym
- deeper into AIR
- demo

a promising Friday night

into the current state of incident response

on a Friday afternoon…

Out for a Guinness?

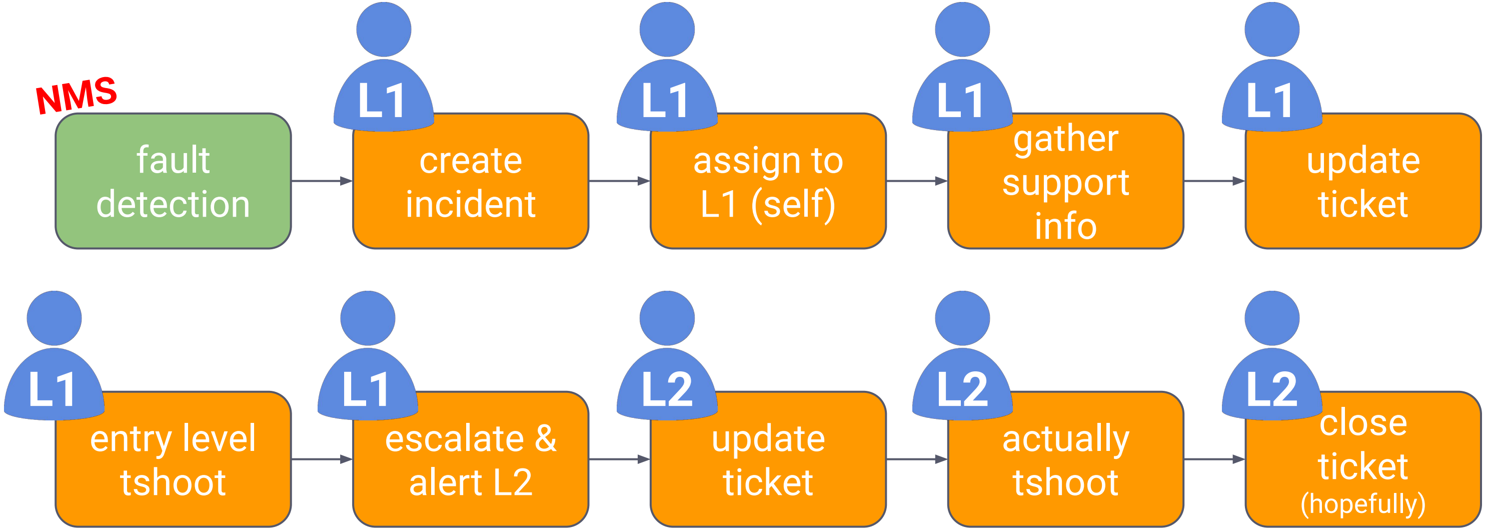
Sure, fancy a couple but not more…

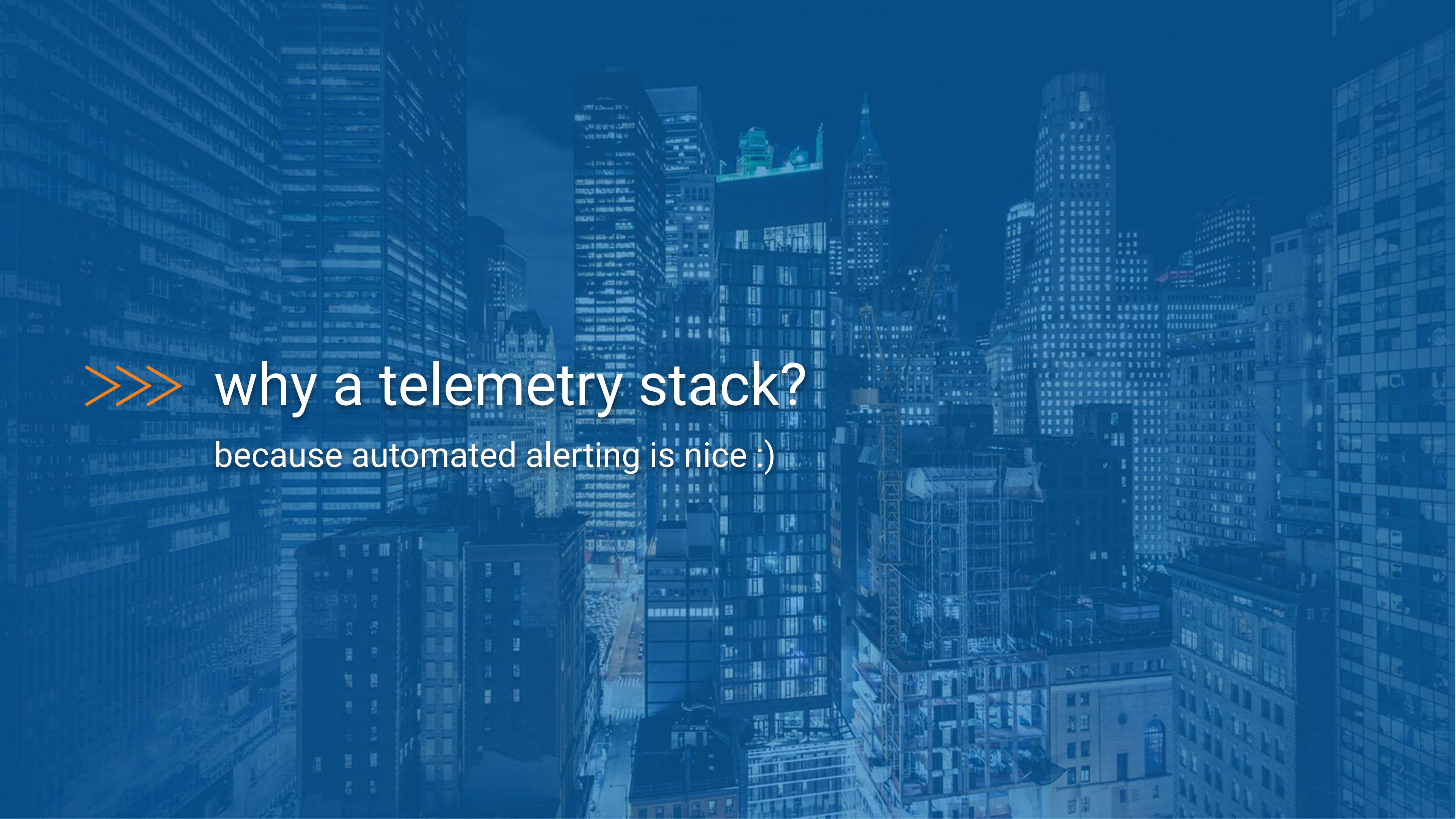
I’m on call today.

Sure, no worries :)

a bit later...

>>> the process

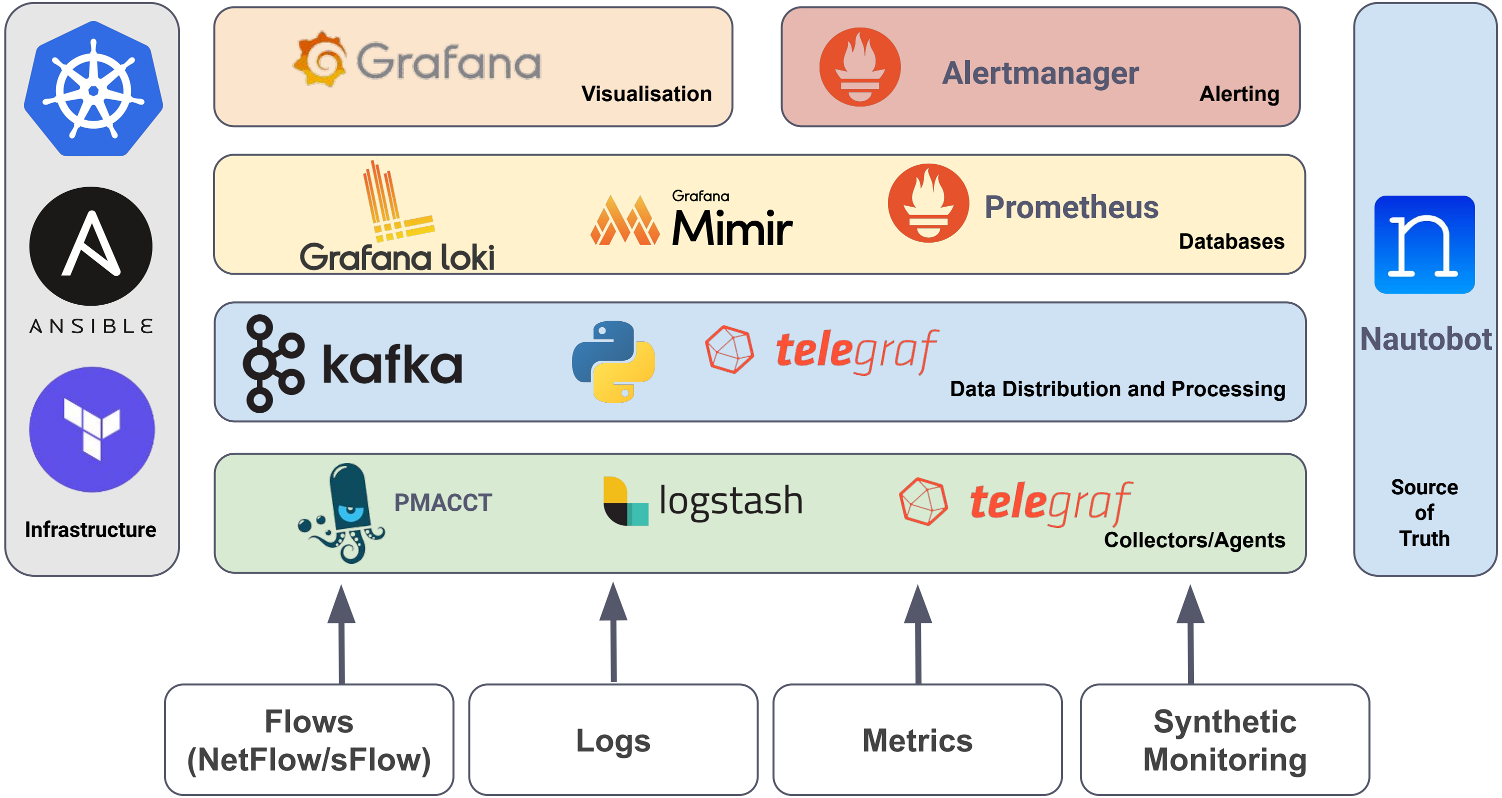

NMS
fault detection
L1
create incident
L1
assign to L1 (self)
L1
gather support info
L1
update ticket
L1
entry level tshoot
L1
escalate & alert L2
L2
update ticket
L2
actually tshoot
L2
close ticket
(hopefully)



why a telemetry stack?

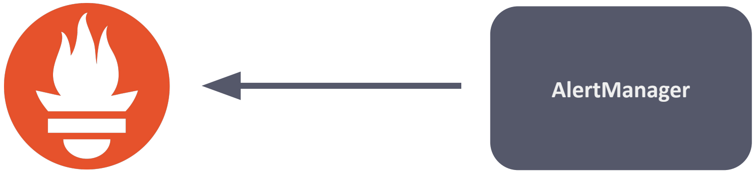
because automated alerting is nice :)

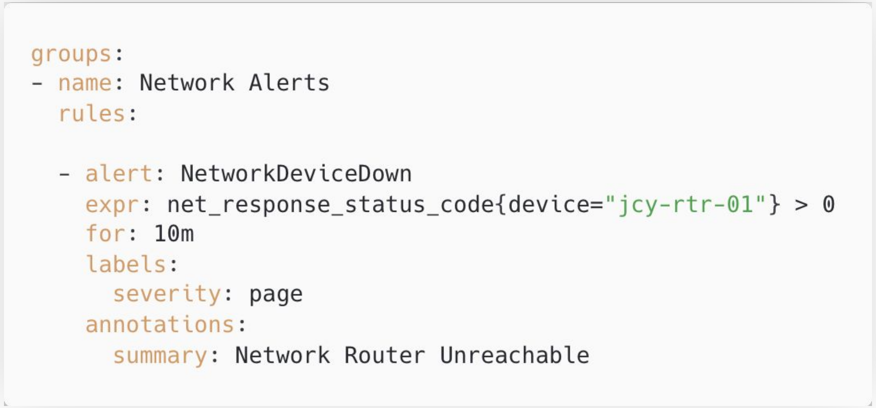
example of full telemetry stack deployment



alerting

- Alert component leverages PromQL for creating rules
- Rich integration with multiple platforms and systems
  - Slack, Telegram
  - PagerDuty
  - Opsgenie
  - Alerta

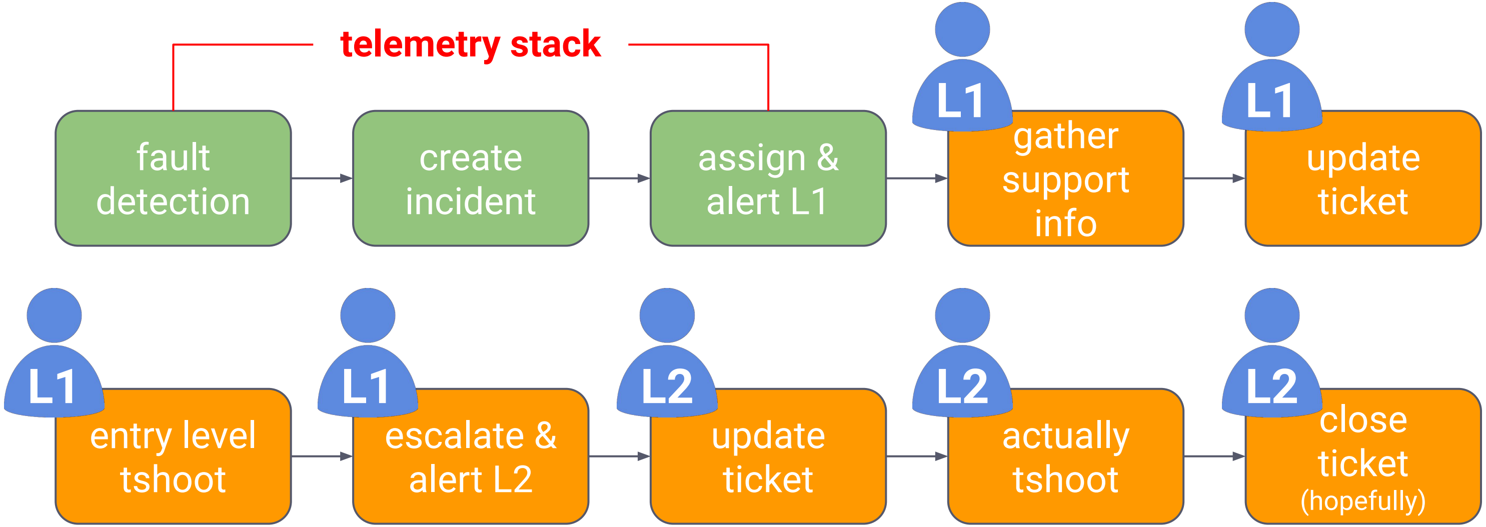




```yaml
groups:
- name: Network Alerts
  rules:

  - alert: NetworkDeviceDown
    expr: net_response_status_code{device="jcy-rtr-01"} > 0
    for: 10m
    labels:
      severity: page
    annotations:
      summary: Network Router Unreachable
```

the process



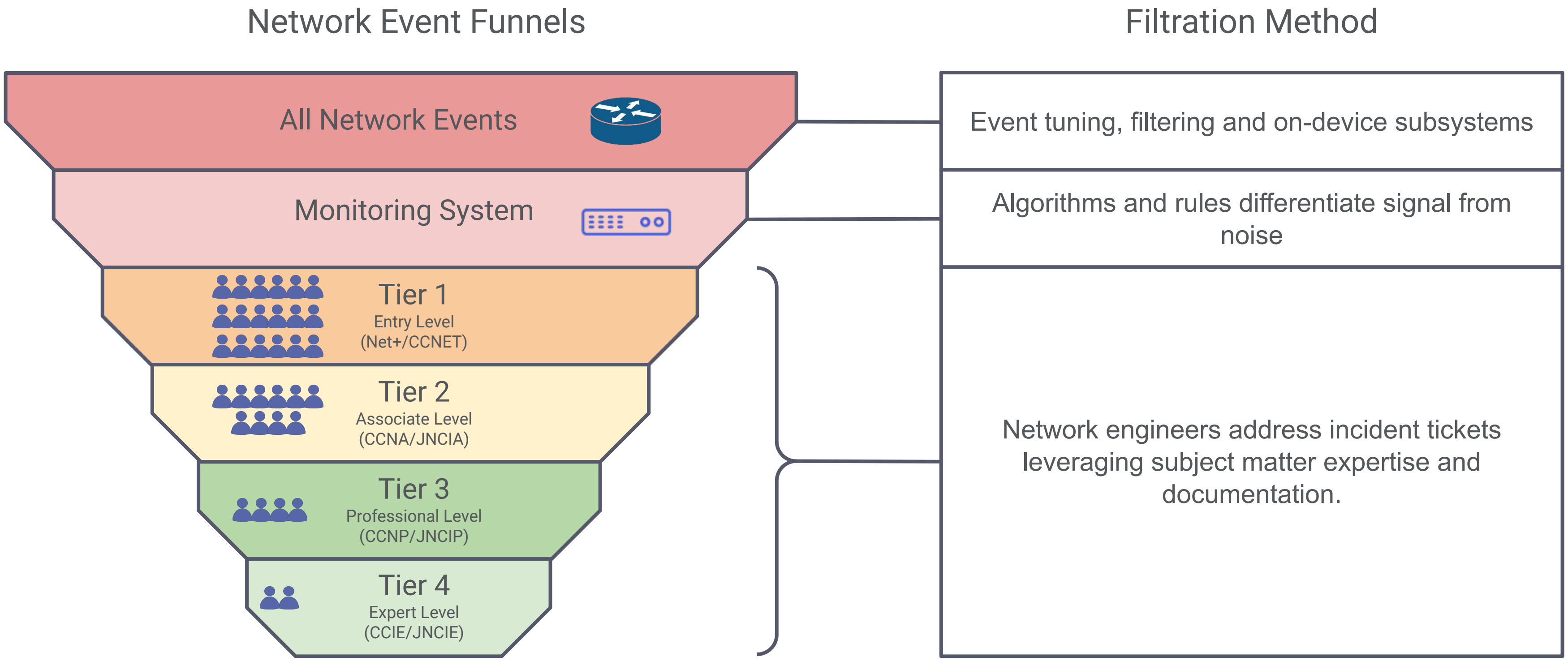
telemetry stack
fault detection
create incident
assign & alert L1
L1
gather support info
L1
update ticket
L1
entry level tshoot
L1
escalate & alert L2
L2
update ticket
L2
actually tshoot
L2
close ticket (hopefully)

introducing Automated Incident Response

because AIR is a cool acronym

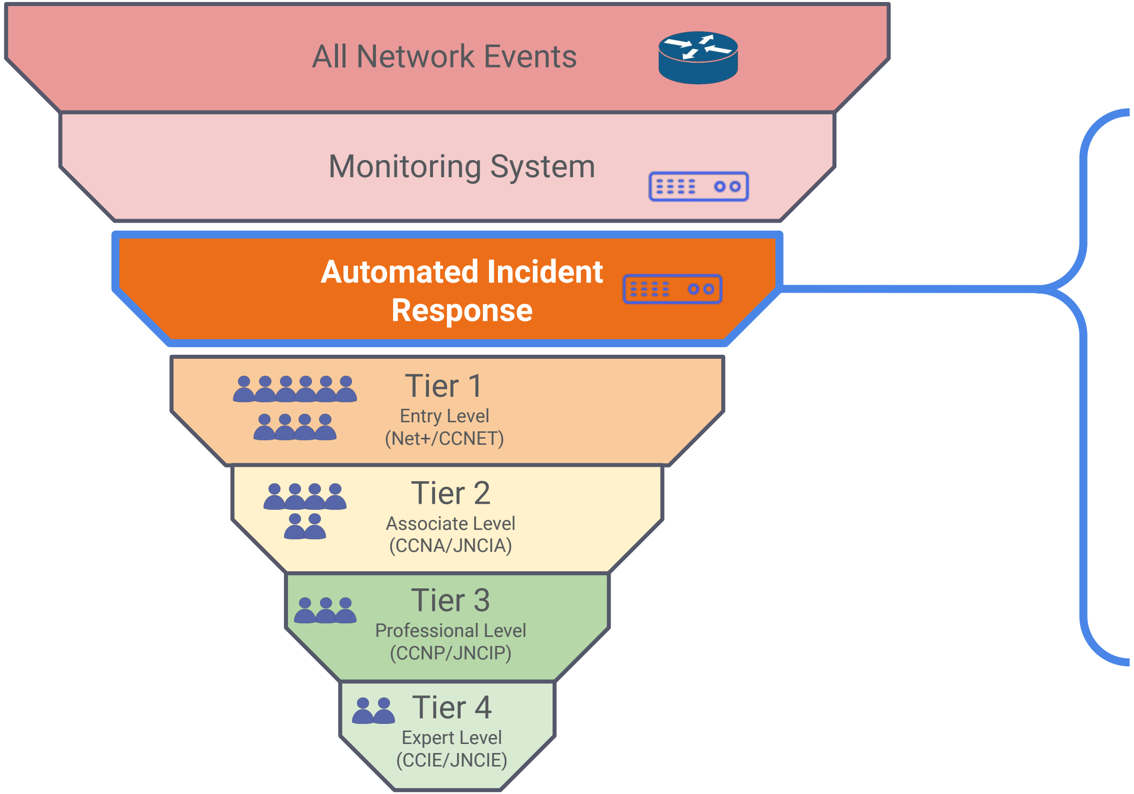
>>> legacy network event funnel

Incident management can be visualized as a series of funnels designed analyze and address network events



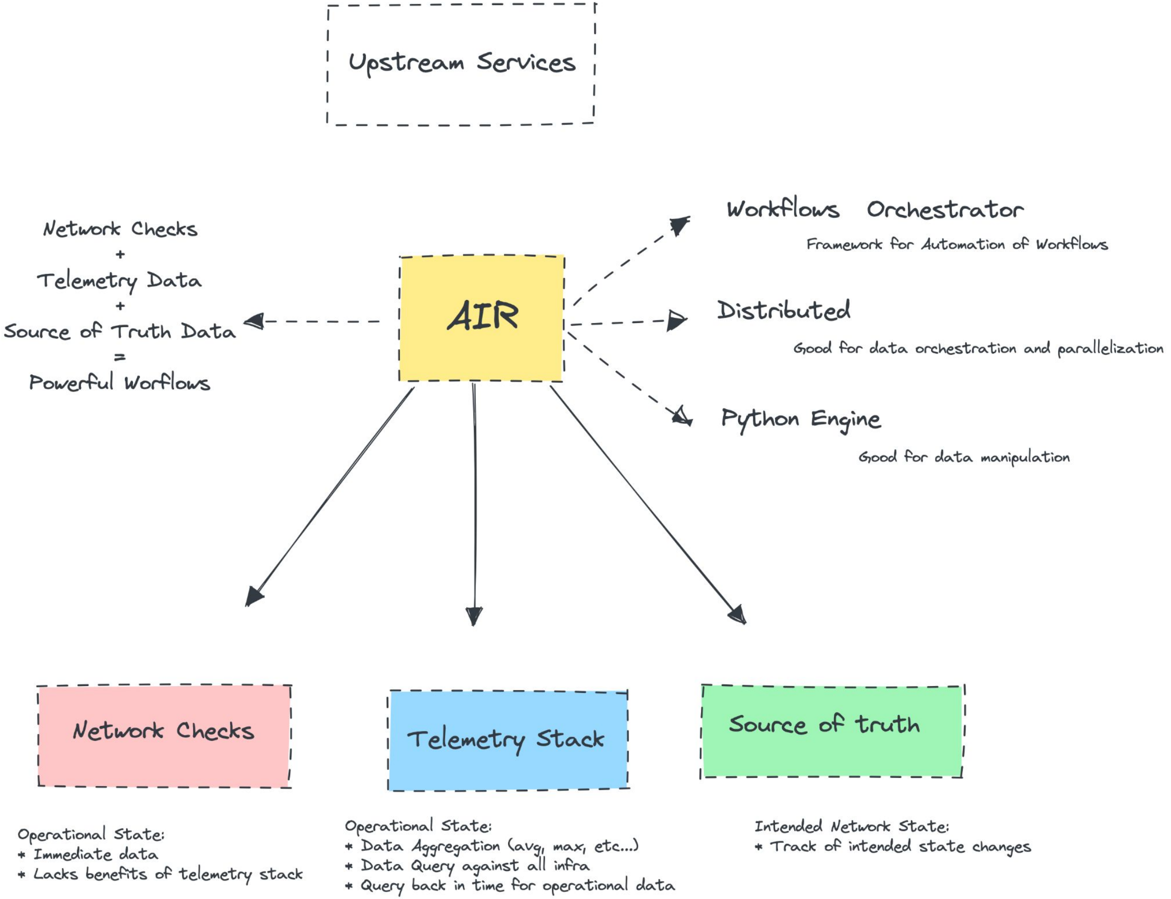
>>> Automated Incident Response engine - a new funnel

AIR inserts a funnel between the monitoring system and the first tier of support. Resolving tickets before they're handled by staff.



- Resolves tickets before they're seen by Tier 1
- Reduces ticket volume by ~50%
- Executes immediately upon ticket creation.
- Follows well-established troubleshooting procedures particular to the type of failure.
- Adds rich diagnostic information to every ticket
- Ensures consistent closure codes and stable troubleshooting.

AIR high level architecture



Upstream Services

Network Checks
+
Telemetry Data
+
Source of Truth Data
=
Powerful Worflows

AIR

Workflows Orchestrator

Framework for Automation of Workflows

Distributed

Good for data orchestration and parallelization

Python Engine

Good for data manipulation

Network Checks

Telemetry Stack

Source of truth

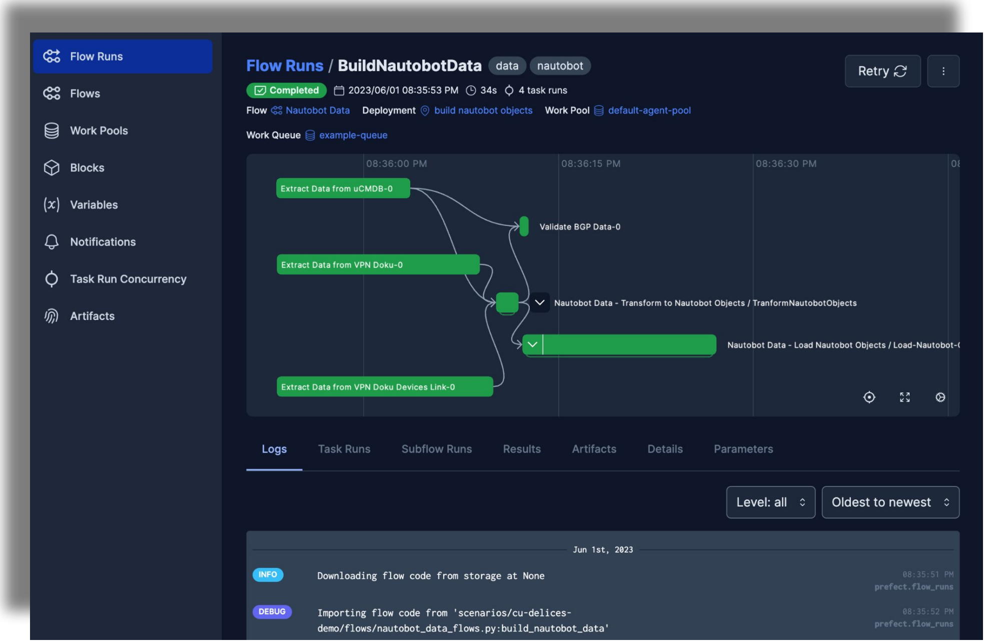
Operational State:
- Immediate data
- Lacks benefits of telemetry stack

Operational State:
- Data Aggregation (avg, max, etc...)
- Data Query against all infra
- Query back in time for operational data

Intended Network State:
- Track of intended state changes

AIR engine

PREFECT


Flow Runs
Flows
Work Pools
Blocks
Variables
Notifications
Task Run Concurrency
Artifacts
Retry
08:36:00 PM
08:36:15 PM
08:36:30 PM
Extract Data from uCMDB-0
Validate BGP Data-0
Extract Data from VPN Doku-0
Nautobot Data - Transform to Nautobot Objects / TranformNautobotObjects
Nautobot Data - Load Nautobot Objects / Load-Nautobot-(
Extract Data from VPN Doku Devices Link-0
Logs
Task Runs
Subflow Runs
Results
Artifacts
Details
Parameters

Flow Runs / BuildNautobotData data nautobot

Completed 2023/06/01 08:35:53 PM 34s 4 task runs

Flow Nautobot Data Deployment build nautobot objects Work Pool default-agent-pool

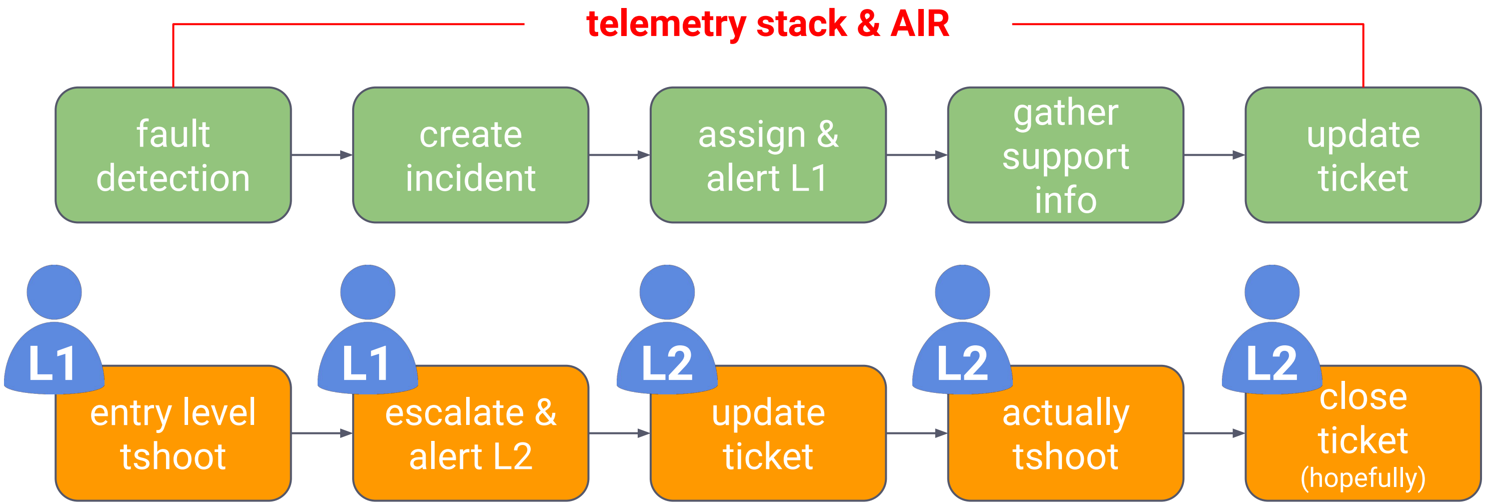
Work Queue example-queue

Level: all Oldest to newest

```
Jun 1st, 2023
INFO   Downloading flow code from storage at None                       08:35:51 PM
                                                                   prefect.flow_runs
DEBUG  Importing flow code from 'scenarios/cu-delices-                  08:35:52 PM
       demo/flows/nautobot_data_flows.py:build_nautobot_data'      prefect.flow_runs
```

- Workflow Orchestration Engine
  - **Scheduling**
  - **Tasks chaining** and isolation
  - **Asynchronous** tasks capabilities
  - **UI** and **API** workflows executions
- Focused on data workflows and thus provides data validation and **Pydantic** compatibility
- It turns a **Python function** into a unit of work that can be **observed** and **orchestrated**
- Reliability and observability capabilities out of the box
  - Retries
  - Logging
  - Caching
- Simplified **testing**
- Ability to scale in a Docker or Kubernetes environment

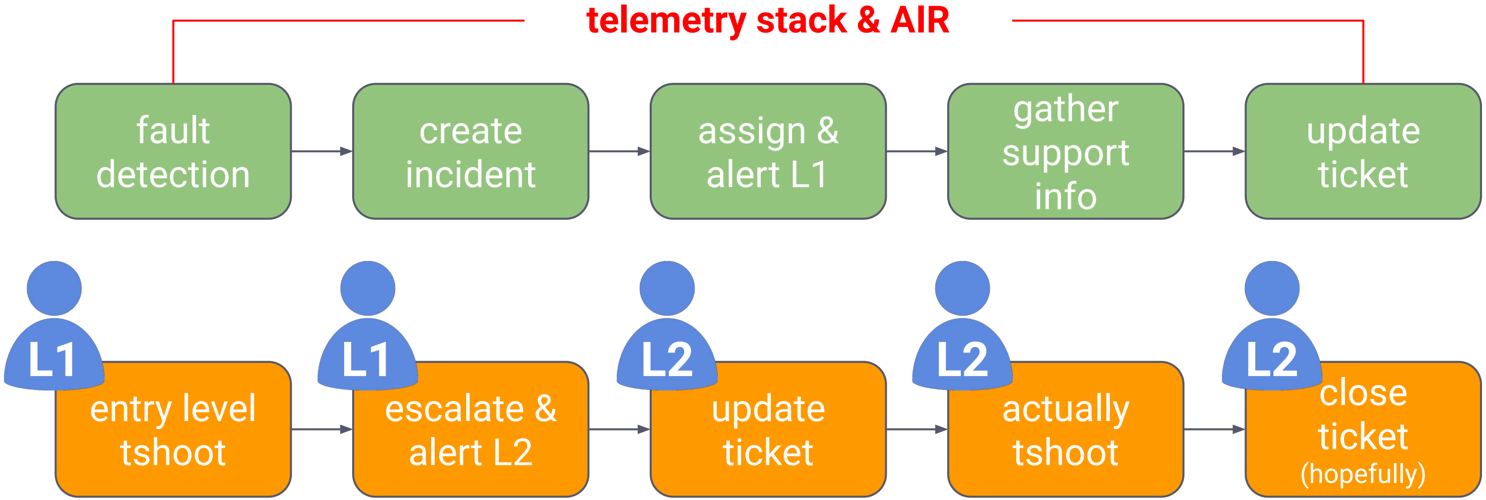
>>> the process


telemetry stack & AIR
fault detection
create incident
assign & alert L1
gather support info
update ticket
L1
entry level tshoot
L1
escalate & alert L2
L2
update ticket
L2
actually tshoot
L2
close ticket
(hopefully)

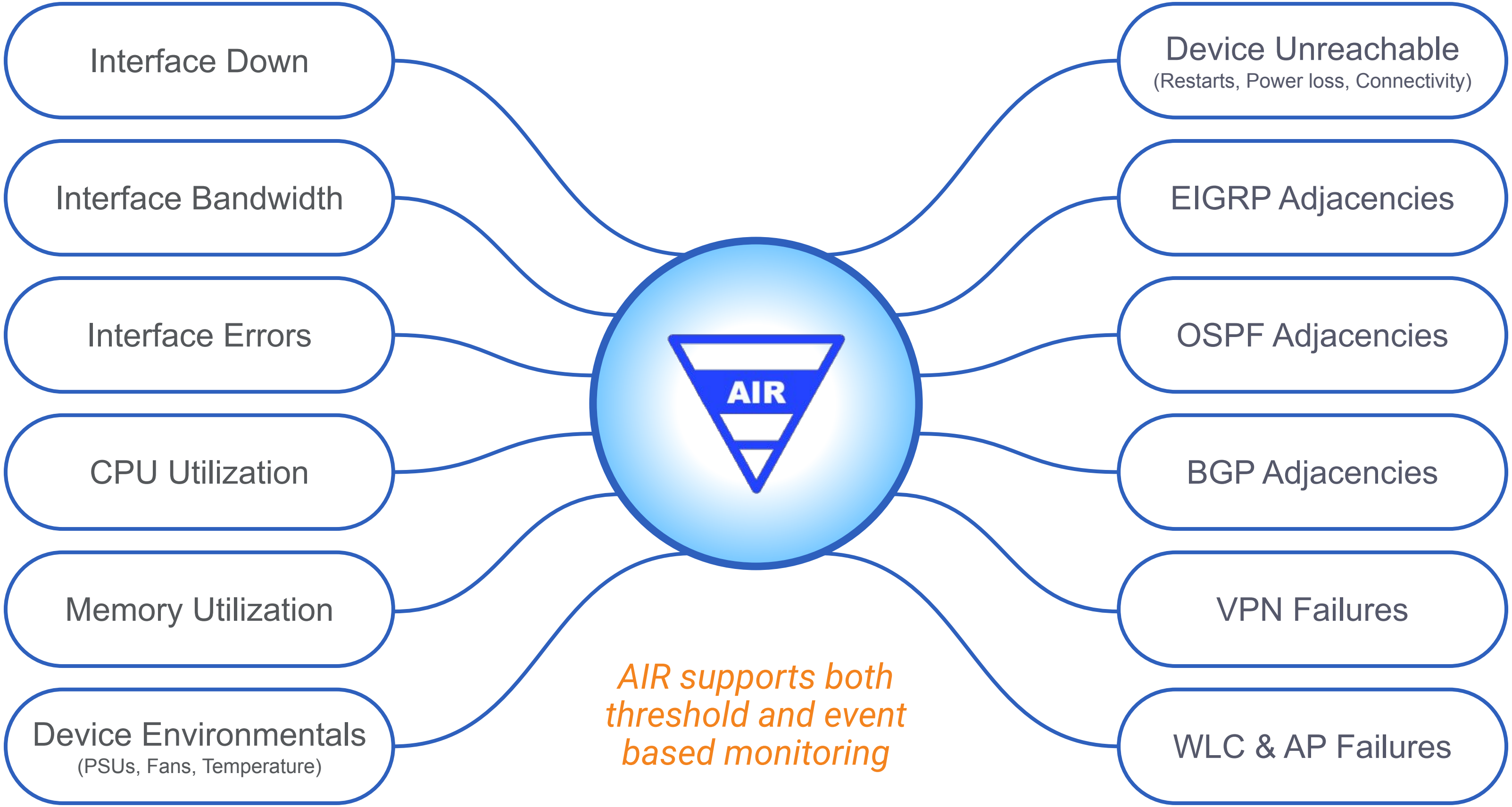
automating the resolution

deeper into AIR

the process


telemetry stack & AIR
fault detection
create incident
assign & alert L1
gather support info
update ticket
L1
entry level tshoot
L1
escalate & alert L2
L2
update ticket
L2
actually tshoot
L2
close ticket (hopefully)

AIR capabilities



Interface Down

Interface Bandwidth

Interface Errors

CPU Utilization

Memory Utilization

Device Environmentals
(PSUs, Fans, Temperature)

Device Unreachable
(Restarts, Power loss, Connectivity)

EIGRP Adjacencies

OSPF Adjacencies

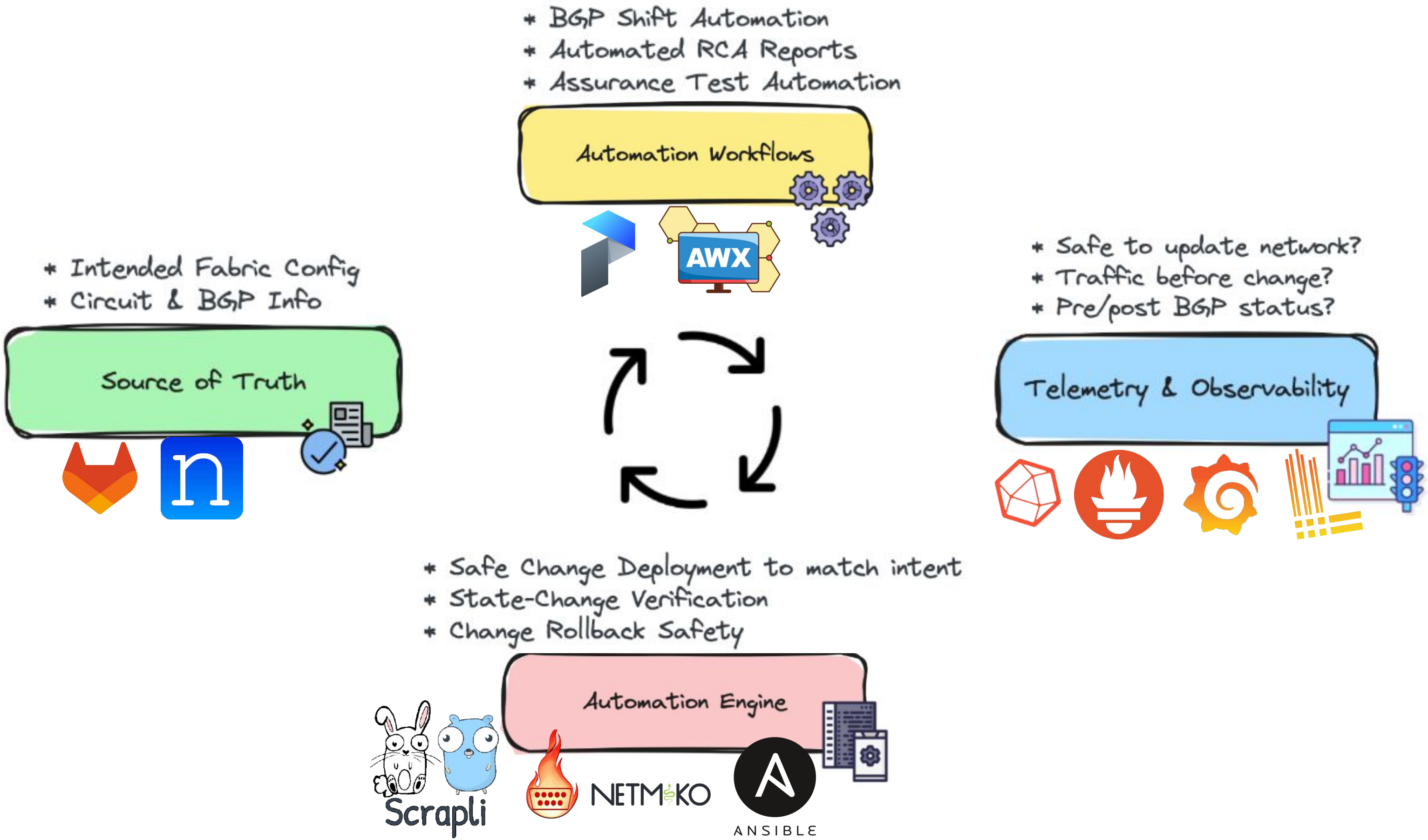
BGP Adjacencies

VPN Failures

WLC & AP Failures

AIR supports both threshold and event based monitoring

AIR platform architecture



- BGP Shift Automation
- Automated RCA Reports
- Assurance Test Automation

Automation Workflows

- Intended Fabric Config
- Circuit & BGP Info

Source of Truth

- Safe to update network?
- Traffic before change?
- Pre/post BGP status?

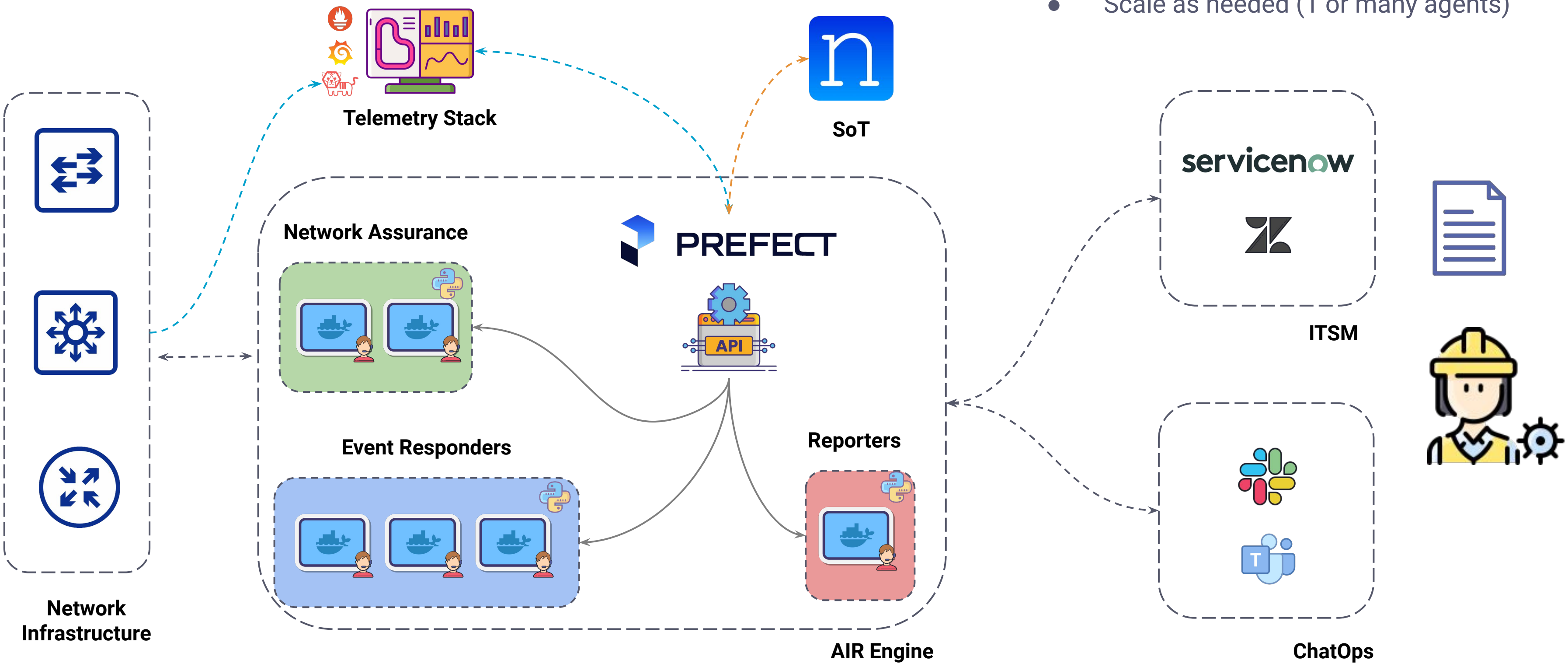
Telemetry & Observability

- Safe Change Deployment to match intent
- State-Change Verification
- Change Rollback Safety

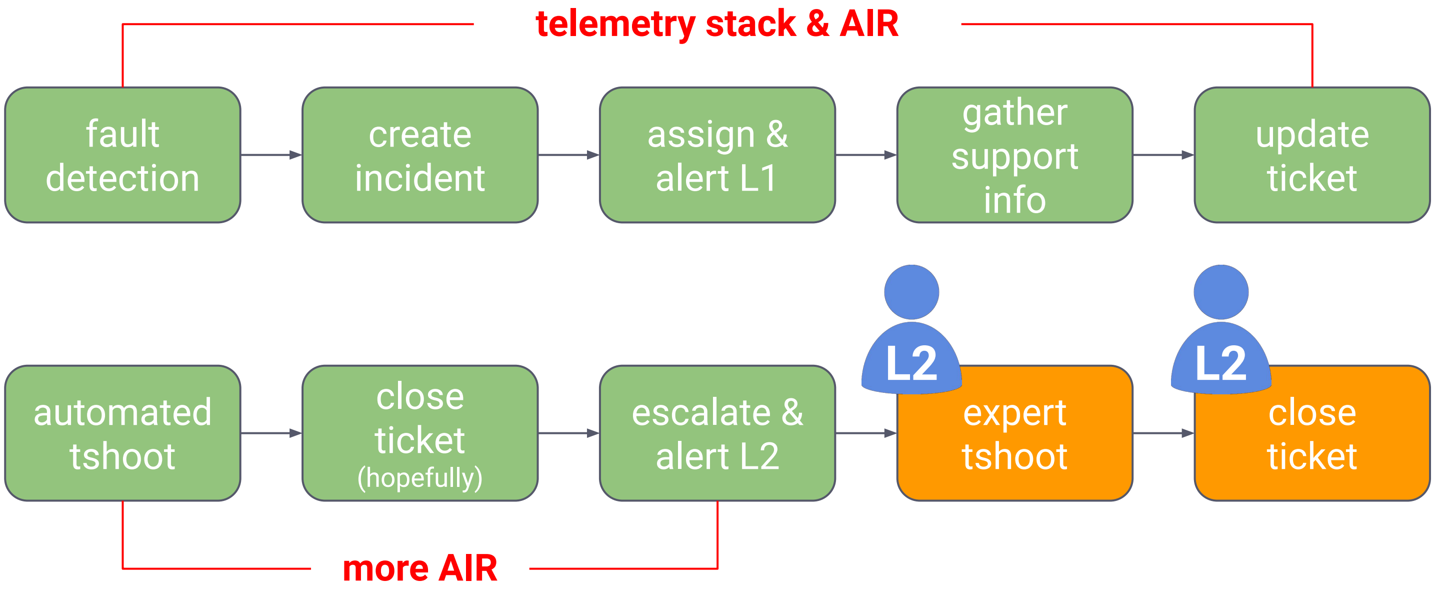
Automation Engine

>>> AIR dataflow

- Example of overall solution and components
- Scale as needed (1 or many agents)



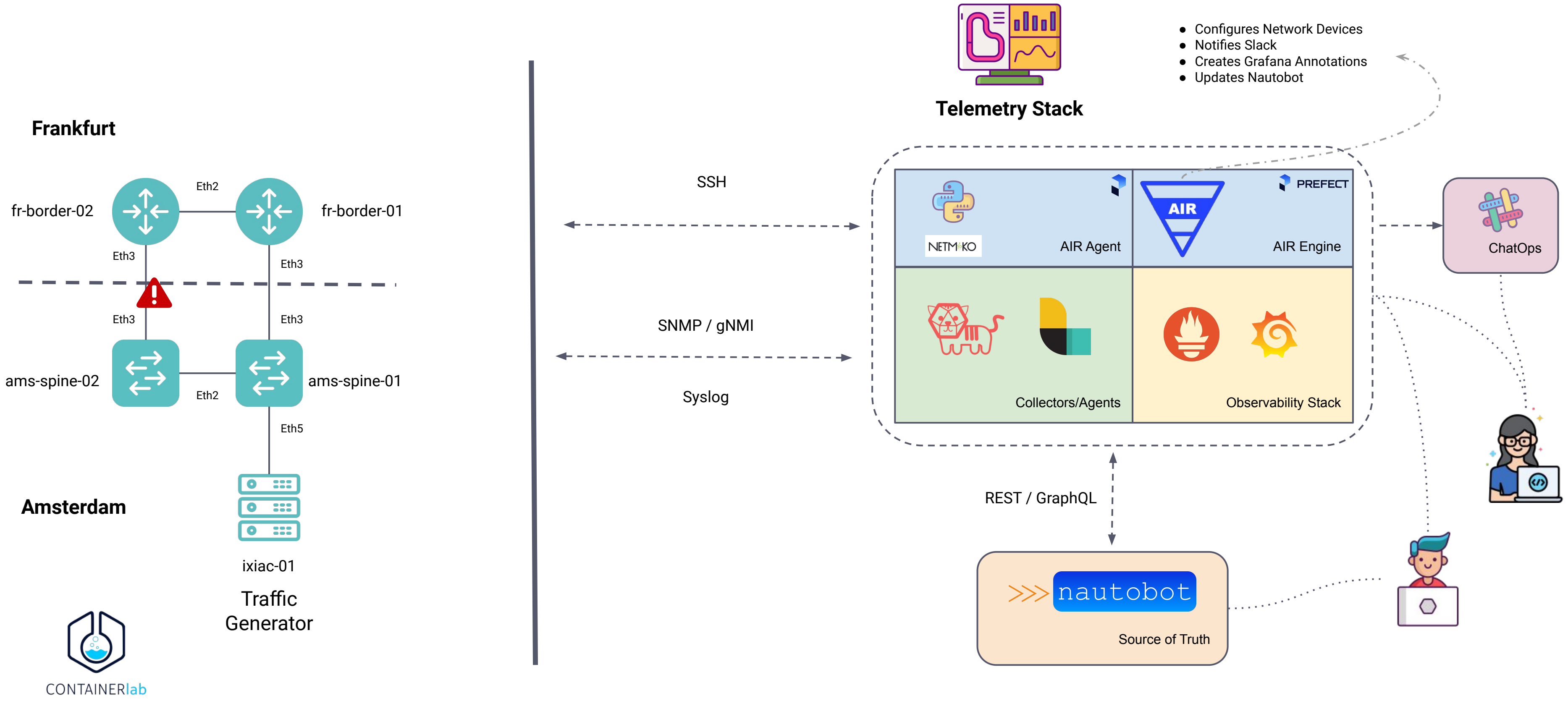
the process


telemetry stack & AIR
fault detection
create incident
assign & alert L1
gather support info
update ticket
automated tshoot
close ticket (hopefully)
escalate & alert L2
L2
expert tshoot
L2
close ticket
more AIR

the process

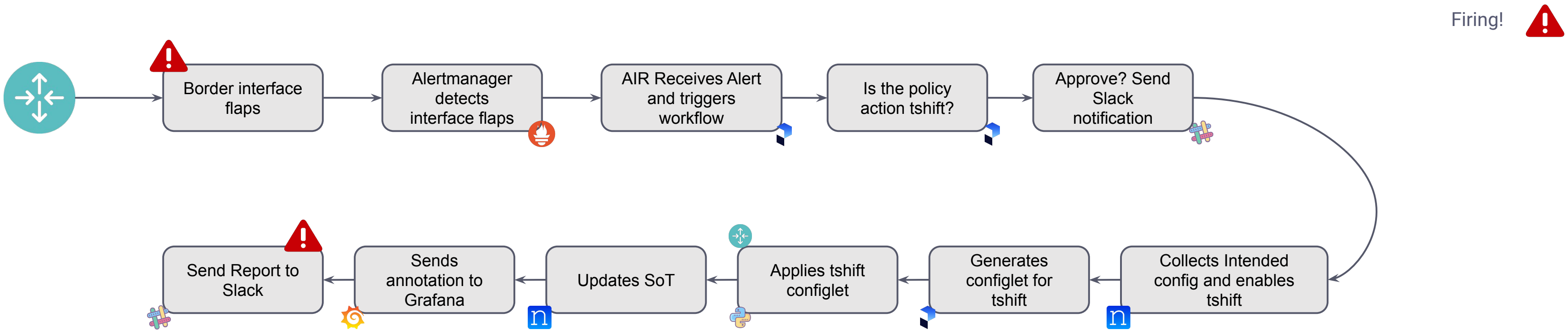
>>> demo

Use Case - Uplink Disruption



- Configures Network Devices
- Notifies Slack
- Creates Grafana Annotations
- Updates Nautobot

Uplink Disruption Resolution Workflow



Firing!

Border interface flaps

Alertmanager detects interface flaps

AIR Receives Alert and triggers workflow

Is the policy action tshift?

Approve? Send Slack notification

Collects Intended config and enables tshift

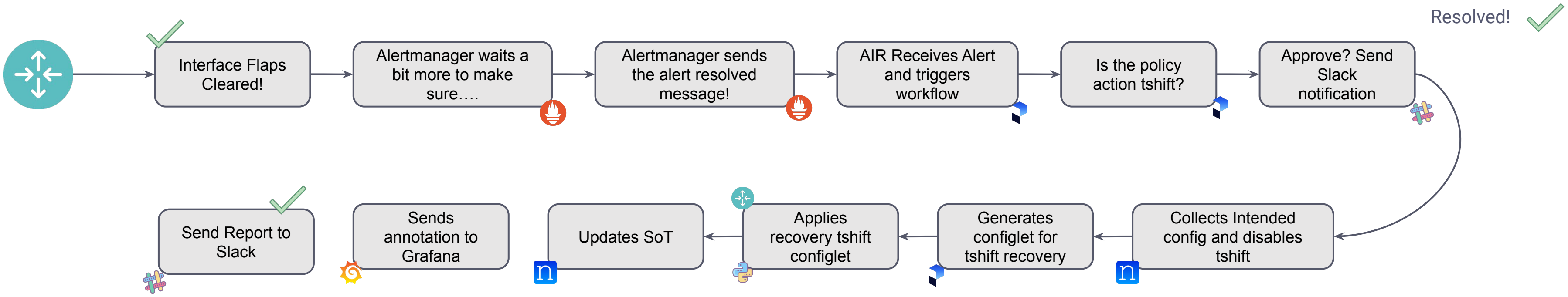
Generates configlet for tshift

Applies tshift configlet

Updates SoT

Sends annotation to Grafana

Send Report to Slack



Resolved!

Interface Flaps Cleared!

Alertmanager waits a bit more to make sure….

Alertmanager sends the alert resolved message!

AIR Receives Alert and triggers workflow

Is the policy action tshift?

Approve? Send Slack notification

Collects Intended config and disables tshift

Generates configlet for tshift recovery

Applies recovery tshift configlet

Updates SoT

Sends annotation to Grafana

Send Report to Slack

Thank you!