DEPLOYING AND MAINTAINING CDN ENVIRONMENTS WITH ANSIBLE

Oct, 27th, 2020

COMCAST

BIO

JONATHAN GRAY

Comcast Content Delivery Network Team

Site Reliability Engineer 4

Apache TrafficControl Committer

Twitter : @jhg03a

GitHub : @jhg03a

traffic-control-cdn.slack.com : @jhg03a

The-asf.slack.com : @jhg03a

jhg03a@apache.org

https://about.twitter.com/en_us/company/brand-resources.html
https://github.com/logos
https://slack.com/media-kit
https://commons.wikimedia.org/wiki/File:Antu_mail-folder-sent.svg

COMCAST

MISSION

1. Replicate and automate the creation of a number of production clone environments

2. Facilitate the maintenance of a minimally viable, but useful, test dataset

3. Solve in such a manner that others can modularly adopt and integrate the components they desire in their implementations

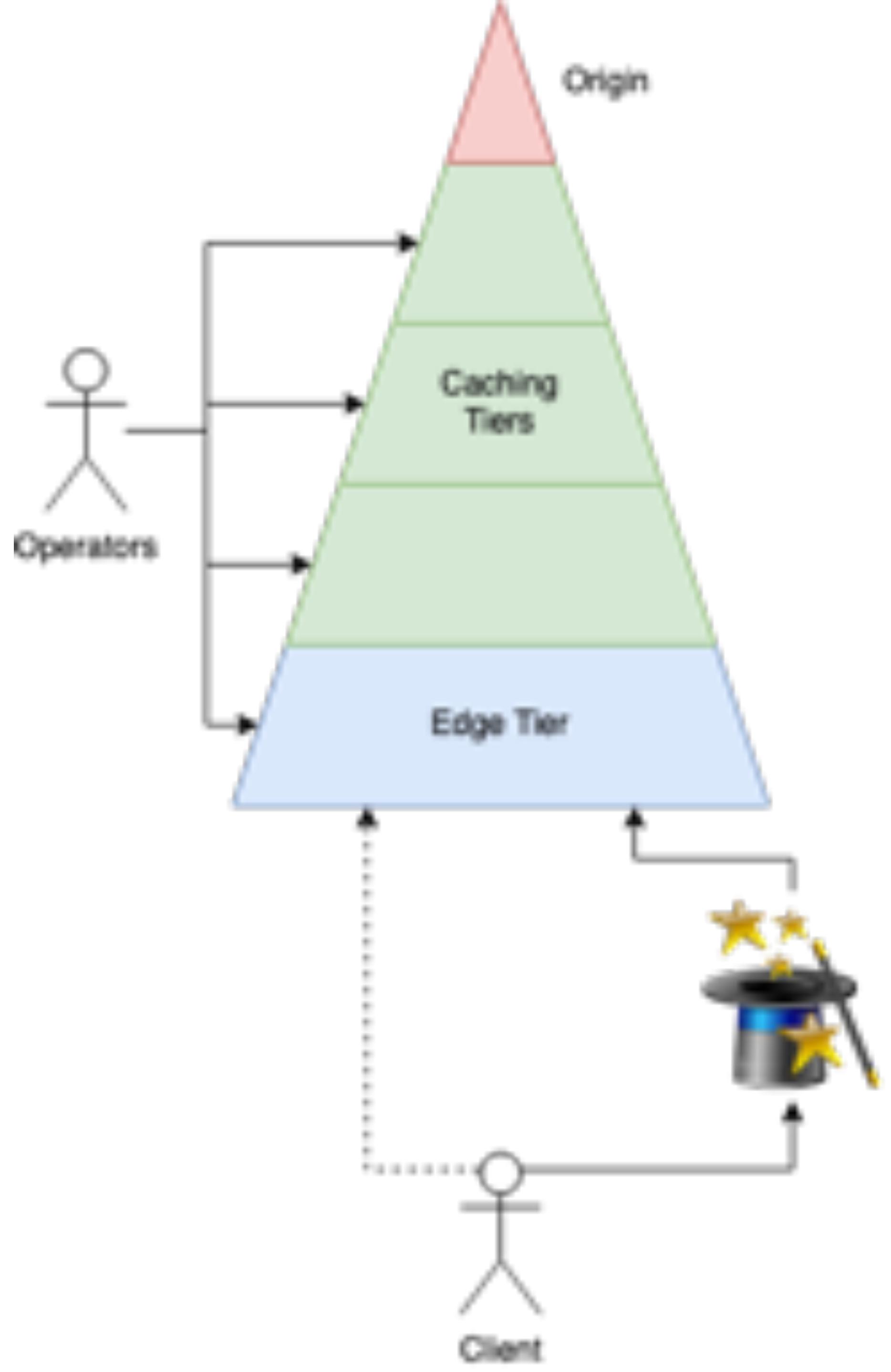
COMCAST

CDN ANATOMY

ATS BASED CDN

BENEFITS

- One application knowledgebase/skillset (ATS)
- Deeper insight into the power of ATS
- Greater Flexibility in CDN Design



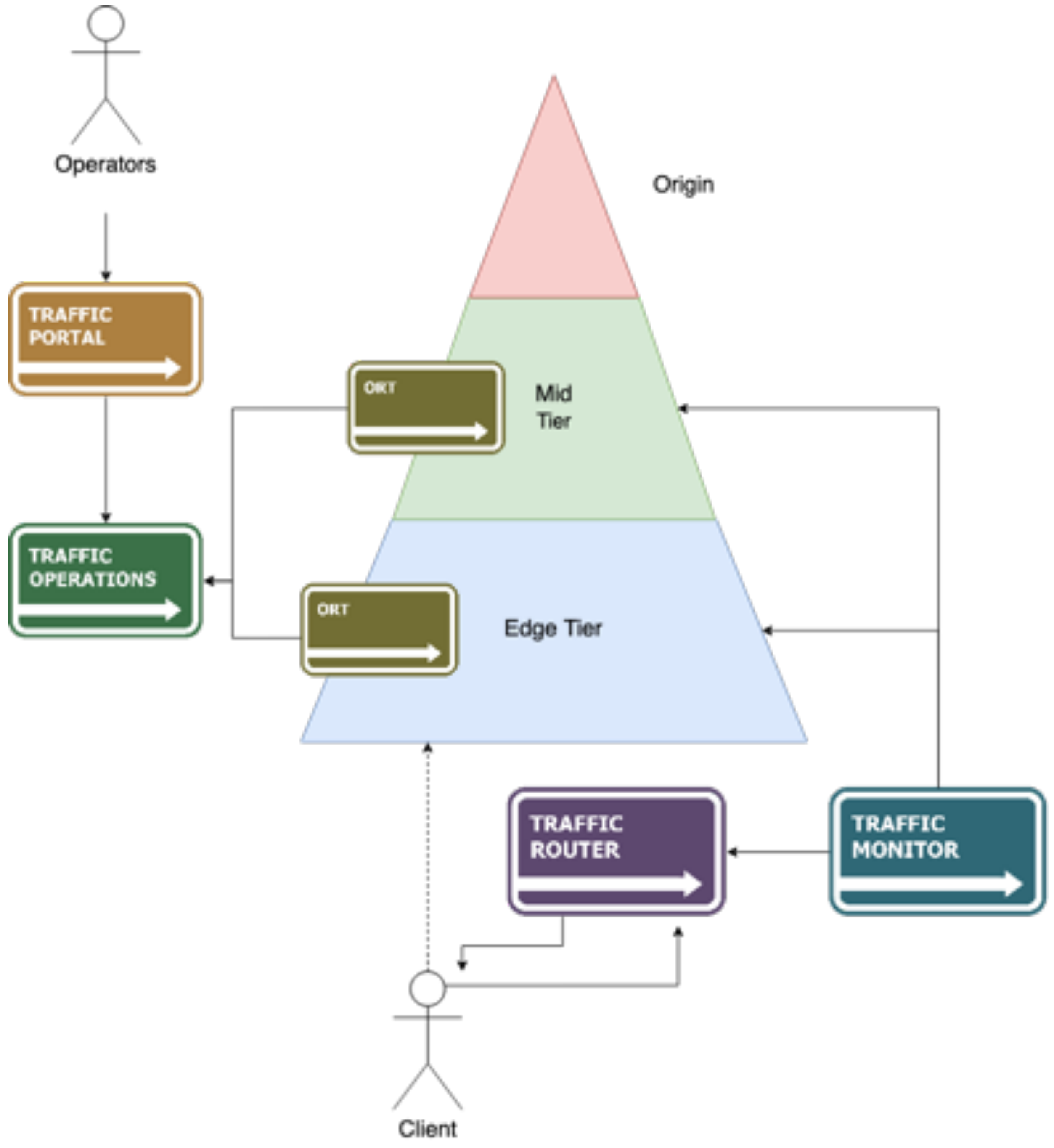
APACHE TRAFFIC CONTROL

TRAFFIC PORTAL	Web-based User Interface for CDN Operations
TRAFFIC ROUTER	Intelligent Routing of CDN Client Requests
TRAFFIC OPERATIONS	Business logic API Layer for CDN Operations
TRAFFIC MONITOR	Health Evaluation of CDN Caches
TRAFFIC STATS	Collection, Aggregation, Transformation for CDN Metrics
GROVE	Experimental Caching Proxy Server
ORT	Apache Traffic Server Configuration Management

ATS + ATC BASED CDN

BENEFITS

- Reducing learning curve of ATS for common delivery service config
- Greater implementation consistency among peers
- Expand user audience beyond ATS Engineers



ANSIBLE OVERVIEW

ORGANIZATION

LARGELY ALL YAML

- Tasks
  - Simple modules
  - Include/import of tasks/playbooks
  - Roles & Collections
- Plays
- Playbooks

SCOPES

- Application
- Operating System
- Hardware
- Network

https://pxhere.com/en/photo/497946

PLUGINS

PUBLISHED OR CUSTOM PLUGINS

- Action – Does something
- Become – Privilege Escalation
- Cache – Fact caching
- Callback – Output
- Cliconf – Network device CLI Interfacing
- Connection – How ansible connects to places
- Httpapi – Network device HTTP Interfacing
- Inventory – Defines the scope of devices to consider
- Lookup – Runtime evaluation of data from external
- Netconf – Network device Netconf Interfacing
- Shell – Low-level execution CLI type
- Strategy – Parallelization extensibility

https://pixabay.com/photos/alternative-energy-biofuel-1042411/

INVENTORY

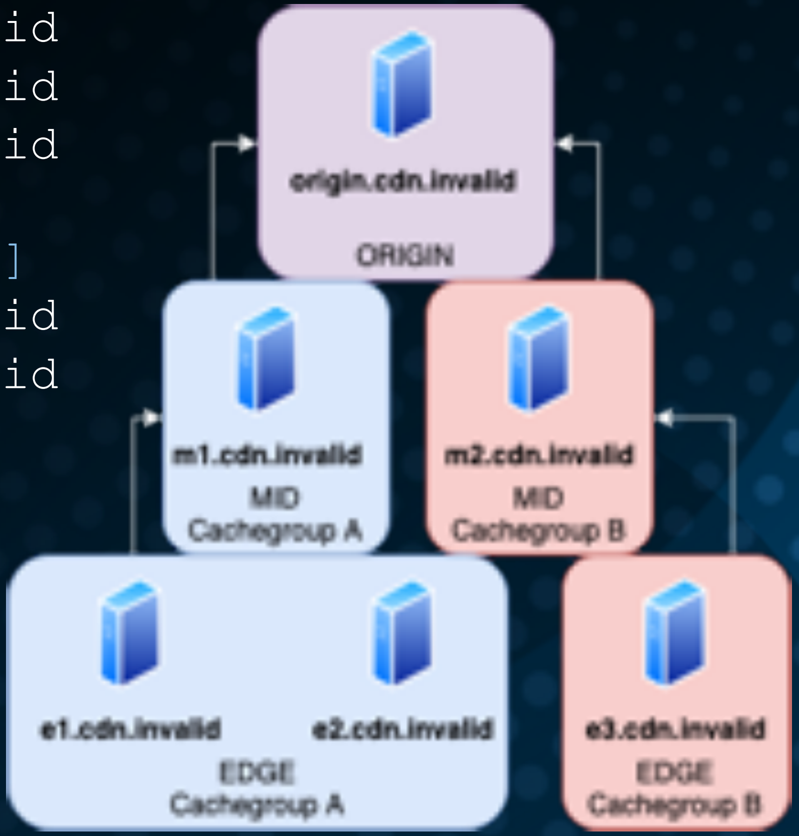
```
[EDGE]
e1.cdn.invalid target_cachegroup=A
e2.cdn.invalid target_cachegroup=A
e3.cdn.invalid target_cachegroup=B
[EDGE:vars]
primary_component=edge

[MID]
m1.cdn.invalid target_cachegroup=A
m2.cdn.invalid target_cachegroup=B
[MID:vars]
primary_component=mid

[Origin]
origin.cdn.invalid ansible_host=192.168.1.70
[Origin:vars]
primary_component=origin
```

```
[CachegroupA]
e1.cdn.invalid
e2.cdn.invalid
m1.cdn.invalid

[CachegroupB]
e3.cdn.invalid
m2.cdn.invalid
```



Example Ansible Pattern: **`EDGE:&CachegroupA:!~.*2.*`**

VARIABLE PRECEDENCE

From most to least important

1. extra vars (always win precedence)
2. set_facts / registered vars
3. include_vars
4. include params
5. role (and include_role) params
6. task vars (only for the task)
7. block vars (only for tasks in block)
8. role vars (defined in role/vars/main.yml)
9. play vars_files
10. play vars_prompt
11. play vars
12. host facts
13. playbook host_vars/*
14. inventory host_vars/*
15. inventory file or script host vars
16. playbook group_vars/*
17. inventory group_vars/*
18. playbook group_vars/all
19. inventory group_vars/all
20. inventory file or script group vars
21. role defaults

🔶 **Used in lab**

https://gist.github.com/ekreutz/301c3d38a50abbaad38e638d8361a89e

COMCAST

VARIABLE HIERARCHY

A RICHER HIERARCHICAL VARIABLE PRECEDENCE ORDERING

- Leverages the `include_vars` precedence order level
- Functionally similar to the common Puppet companion project Hiera
- Significant addition to ansible-pull variable definitions

```
/opt/ansible
├── vars
│   ├── default.yml
│   ├── environment
│   │   ├── nightly.lab.yml
│   │   └── production.yml
│   ├── manufacturer
│   │   ├── Dell.yml
│   │   └── HP.yml
│   ├── productName
│   │   └── R740.yml
│   └── host
│       └── edge1.cdn.invalid.yml
└── mainplaybook.yml
```

Example

```
- name: Load fqdn-based values in variable hierarchy
  include_vars:
    file: "{{ lookup('first_found', possible_files, errors='ignore') }}"
  failed_when: false
  vars:
    possible_files:
      - "vars/host/{{ ansible_fqdn }}.json"
      - "vars/host/{{ ansible_fqdn }}.yml"
```

COMCAST

IDEMPOTENCY

WHILE ASSERTING TRUTH DID YOUR TASK…

- Execute, but change nothing
- Execute and change something
- Fail
- Not try to execute at all

https://pixabay.com/photos/hammer-nails-wood-board-tool-work-1629587/

VERSIONING

WHERE TO LOOK FOR REUSABLE CODE

ANSIBLE GALAXY

Officially endorsed marketplace for reusable Ansible roles.

https://galaxy.ansible.com/

ANSIBLE COMMUNITY COLLECTIONS

Community supported modules and plugins

https://github.com/ansible-collections/

APACHE TRAFFIC CONTROL

Roles, samples, and support utilities specifically for ATC components

https://github.com/apache/trafficcontrol/tree/master/infrastructure/ansible

https://commons.wikimedia.org/wiki/File:Ansible_logo.svg
https://github.com/apache/trafficcontrol/blob/master/misc/logos/ATC-SVG.svg

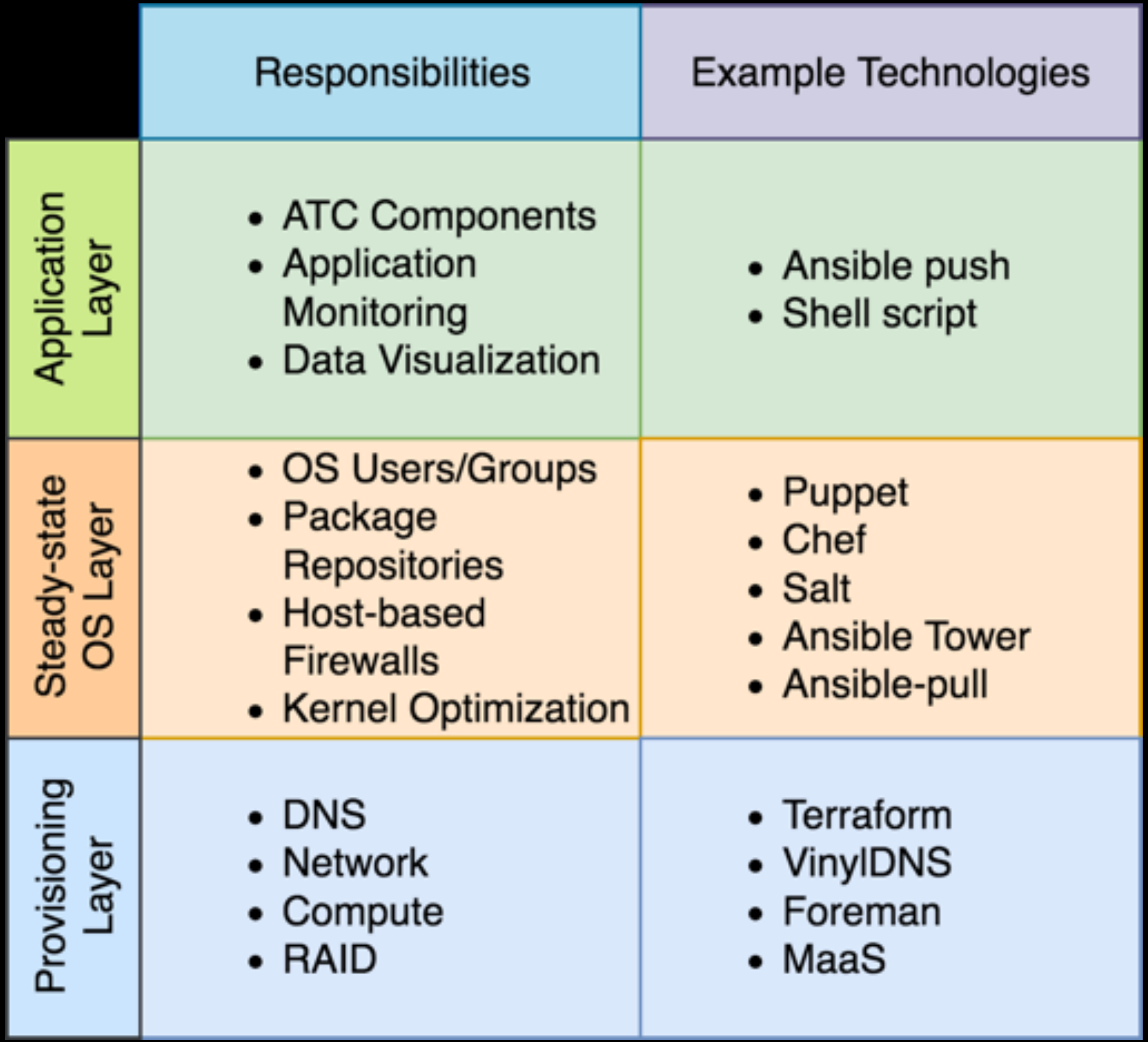
CDN ENVIRONMENTS

ENVIRONMENT ABSTRACTION LAYERS

NOT CDN-OUT-OF-THE-BOX

Complexity breeds greater complexity

Every abstraction layer comes at a price; some are more expensive than others. Lower costs through reuse of existing tools/skillsets.



	Responsibilities	Example Technologies
Application Layer	• ATC Components • Application Monitoring • Data Visualization	• Ansible push • Shell script
Steady-state OS Layer	• OS Users/Groups • Package Repositories • Host-based Firewalls • Kernel Optimization	• Puppet • Chef • Salt • Ansible Tower • Ansible-pull
Provisioning Layer	• DNS • Network • Compute • RAID	• Terraform • VinylDNS • Foreman • MaaS

https://github.com/apache/trafficcontrol/pull/3585

COMCAST

PROVISIONING

PHYSICAL DEPLOYMENT

UNIVERSAL ISO WITH TC_NETCONFIG

PRO

- One ISO for all hosts
- Continuous network identity maintenance via TrafficOps
- ISO Creation process is separate from TrafficOps

CON

- Requires IPv6 Autoconf RA

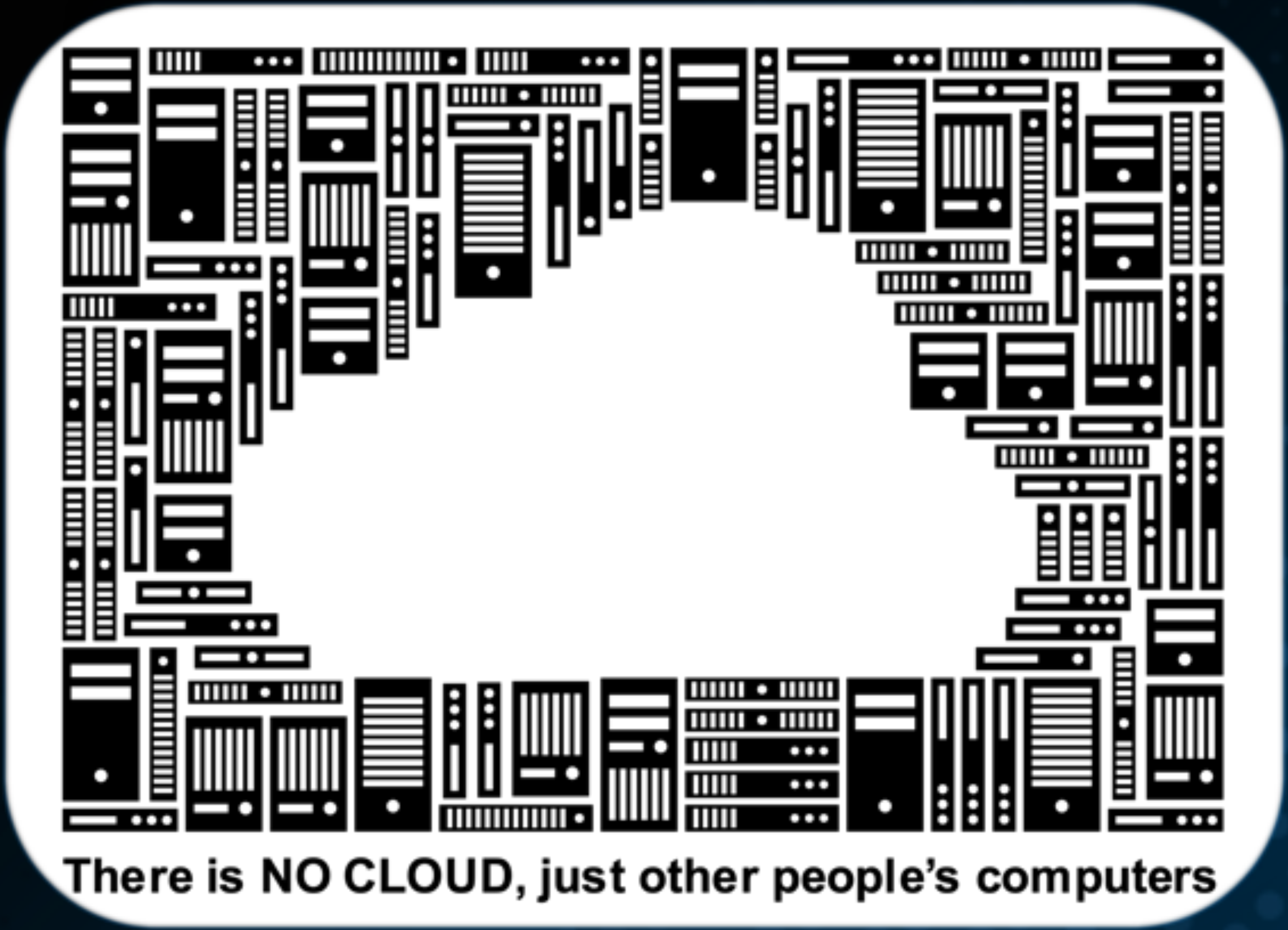
RESOURCES

- GitHub: https://github.com/Comcast/tc-netconfig
- ApacheCon 2019 Presentation: https://tinyurl.com/tcnetconfig-video
- ApacheCon 2019 Slides: https://tinyurl.com/tcnetconfig-slides

CLOUD

TOOLING

- HashiCorp Terraform
- VinylDNS
- OpenStack
- Cloud-Init

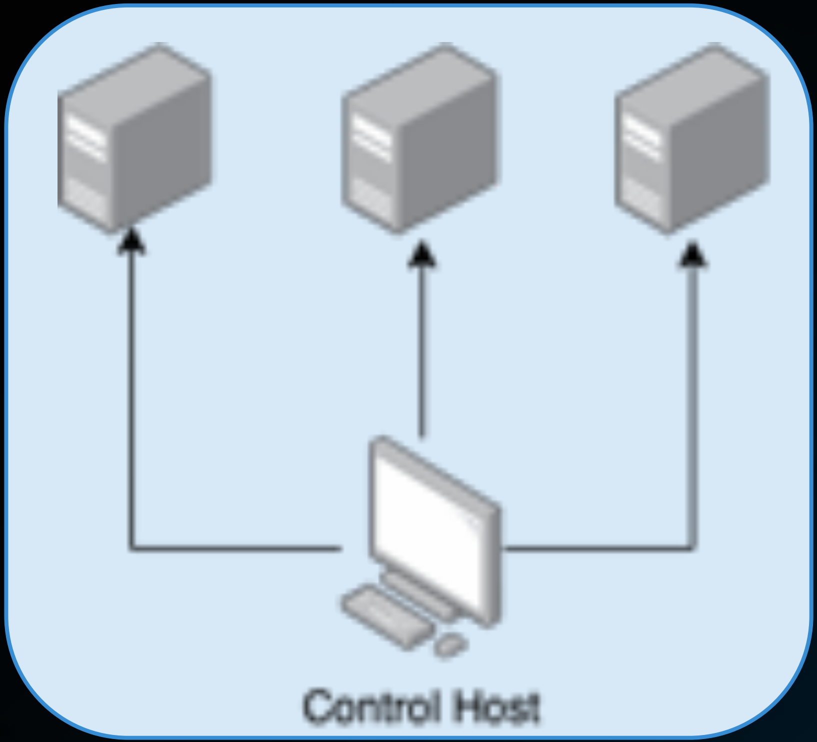


https://commons.wikimedia.org/wiki/File:FSFE_There_is_no_cloud_postcard_en.svg

STEADY STATE

ANSIBLE WORKFLOWS

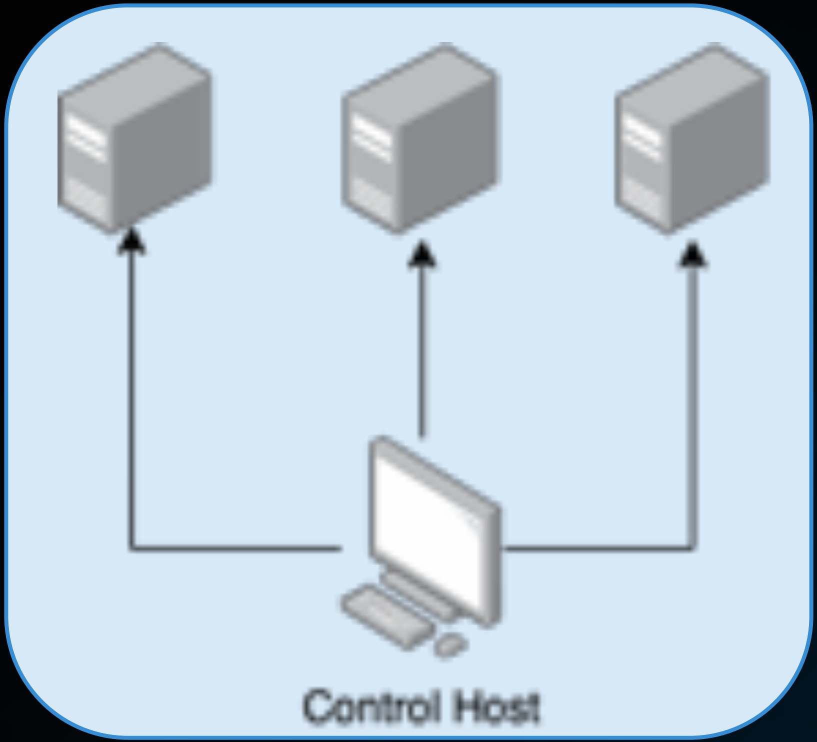
ANSIBLE (PUSH)



"Do this"

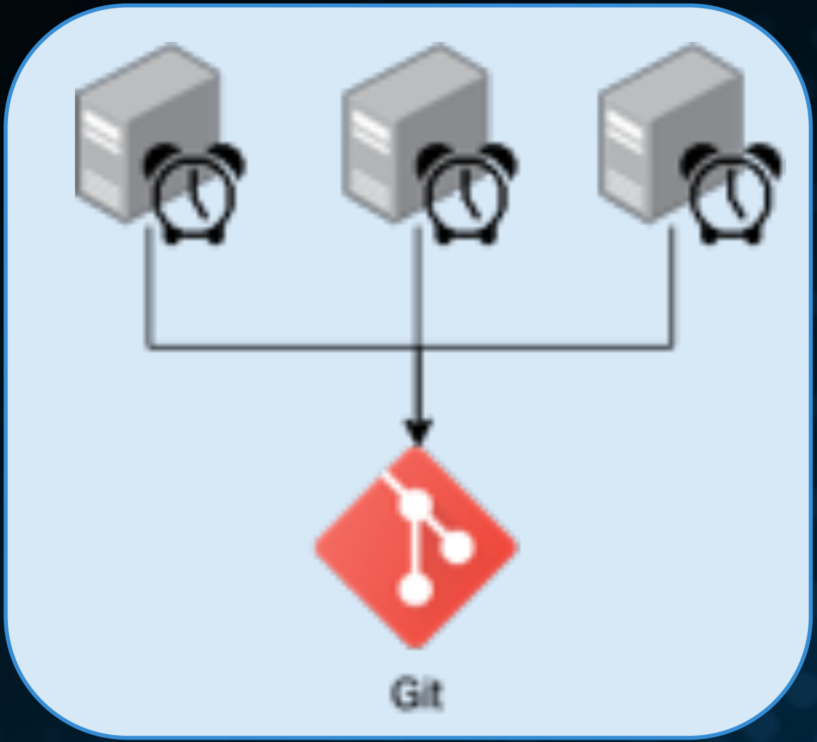
ANSIBLE WORKFLOWS

ANSIBLE (PUSH)



"Do this"

ANSIBLE-PULL

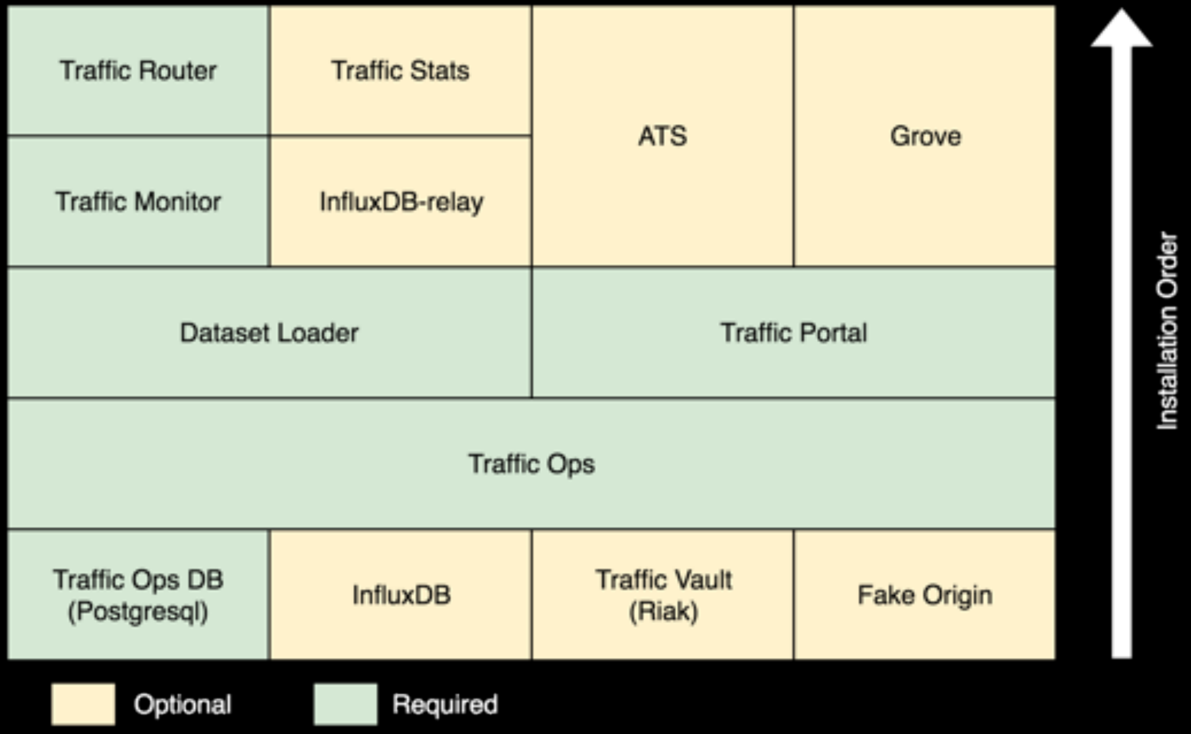


"Do what applies"

COMCAST

APPLICATION

CDN LAB COMPONENTS



ATC COMPONENT INSTALLATION ORDER

Not all ATC Components are strictly required however are important at some scales or for some functionality.

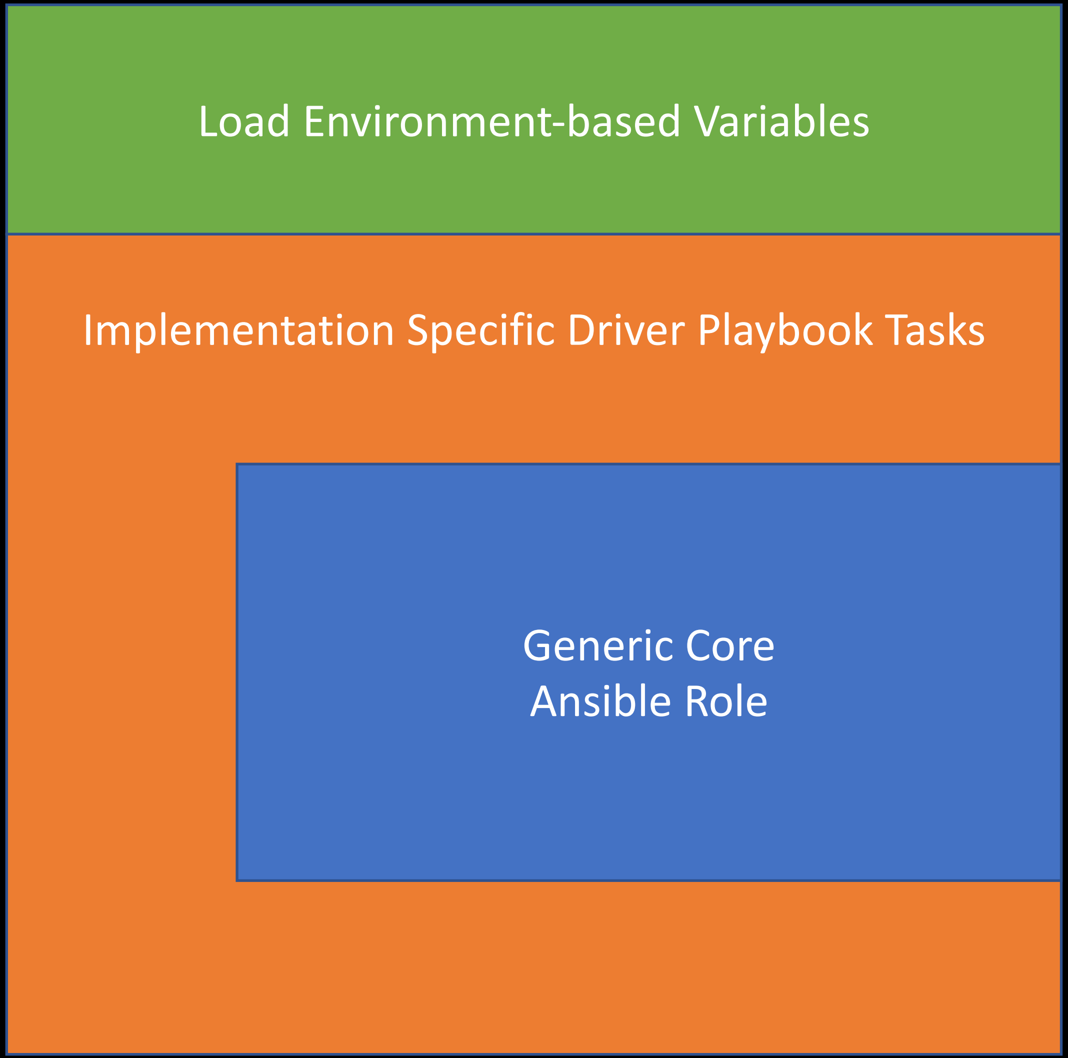
Due to application stack dependencies, care should be taken regarding order and parallelization of installation.

https://github.com/apache/trafficcontrol/pull/3585

COMCAST

ATC COMPONENT ANSIBLE PLAYBOOK PATTERN

1. Load environment-based variables
2. Implementation-specific Pre-tasks
3. Generic Core role
4. Implementation-specific Post-Tasks


Load Environment-based Variables
Implementation Specific Driver Playbook Tasks
Generic Core Ansible Role

COMCAST

BONUS DYNAMIC INVENTORY SCRIPT

SAMPLE ANSIBLE GROUPS FOR PATTERNS:

- Simple Hostname: `atsedge*`
- Status: `server_status|OFFLINE`
- Type: `server_type|EDGE`
- CDN Name: `server_cdnName|Kabletown2.0`
- Profile: `server_profile|ATS_EDGE_7`
- Cachegroup: `cachegroup|edge_east`
- Parent Cachegroup: `parentCachegroup|mid_east`
- Secondary Parent Cachegroup: `secondaryParentCachegroup|mid_west`

COMCAST

Photo by Júnior Ferreira on Unsplash

LAB MANAGER

GOALS

- Simple
- Focus on Data Relationships and Integrity
- Reliable System of Record
- Resolve inherent Chicken/Egg problem with ATC TrafficOps

CONCEPTS

- Environment definition & lifecycle
- Resource Pools
- Jobs
- Logs
- Fact Inventory

Photo by Mr Cup / Fabien Barral on Unsplash

GRAPHQL API PROTOCOL

OPEN SOURCE PROTOCOL

Originally created by Facebook and donated to the Linux Foundation in 2017 where now it resides under the GraphQL Foundation.

Designed around flexibility of the client request. "Get what you want, only what you want, and nothing more." Traditionally viewed as an upcoming alternative to REST.

https://foundation.graphql.org

Current adopters include:

- Facebook
- GitHub
- PayPal
- The New York Times
- Twitter

https://github.com/graphql/graphql-spec/blob/master/resources/GraphQL%20Logo.png , **OWFa1.0**

COMCAST

POSTGRESQL DATABASE

RELATIONAL DATABASE BACKEND

Originally created by engineers at UC Berkley with version 1 released in 1989, PostgreSQL continues to be a major force in Open-Source RDBMS.

https://www.postgresql.org

Current adopters include:

- Apache Traffic Control
- Uber
- Netflix
- Reddit
- Spotify

POSTGRAPHILE API

OPEN SOURCE GRAPHQL IMPLEMENTATION

Started in 2016, Postgraphile is an easy-to-use API library for GraphQL. The robust open-source NodeJS library is MIT licensed, however additional enterprise features are available for a small license fee.

Postgraphile is low to no-code required for a functional API as it leverages data from PostgresQL to correctly build out the GraphQL Schema automatically with documentation that's available.

https://www.graphile.org/postgraphile/

While Postgraphile can be leveraged standalone or as a NodeJS library, I mix-in several other NodeJS libraries and frameworks for the Lab Manager:

- ExpressJS
- Grant
- Winston
- JsonWebToken
- GraphQL-Voyager

https://github.com/graphile/postgraphile/blob/v4/assets/favicon.ico, MIT

COMCAST

POSTGRAPHILE PRIMER

SECURITY

AUTHENTICATION

The Lab Manager leverages OAuth2.0 flows to obtain a valid JWT

ADAPTATION

The Lab Manager verifies the JWT and extracts the user, role, and capabilities to pass along through Postgraphile to PostgresQL

AUTHORIZATION

Authorization is handled via native PostgreSQL security mechanisms built into the database.

COMCAST

SECURITY

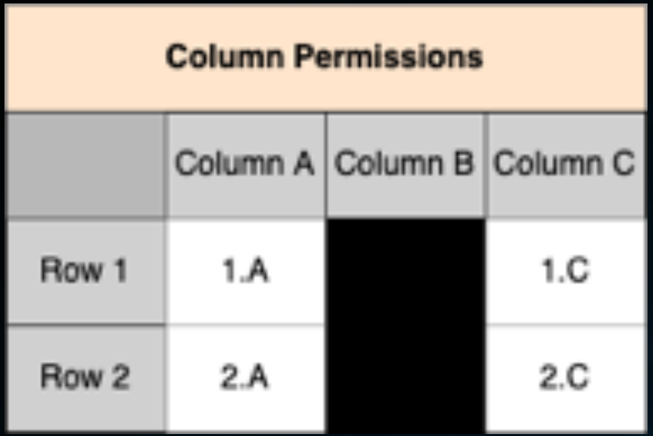
NATIVE POSTGRESQL AUTHORIZATION

- Column
- Table
- Row Policies

ADDITIONAL INTEGRITY VALIDATION

- Usage of Check Constraints & Defaults to enforce JWT values

With the use of security definers, it is possible to override the security settings of a request and user



Column Permissions

	Column A	Column B	Column C
Row 1	1.A	■	1.C
Row 2	2.A	■	2.C

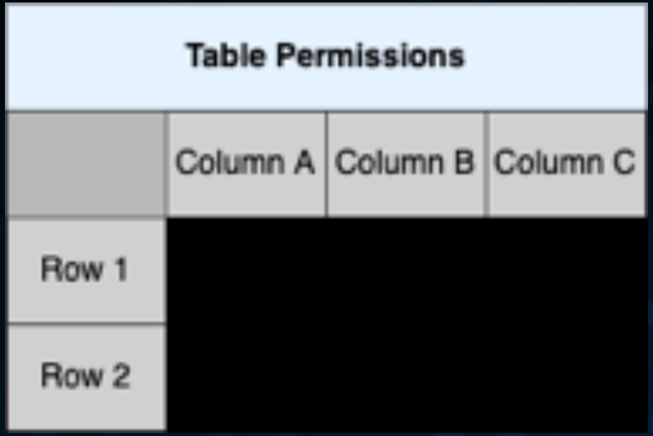
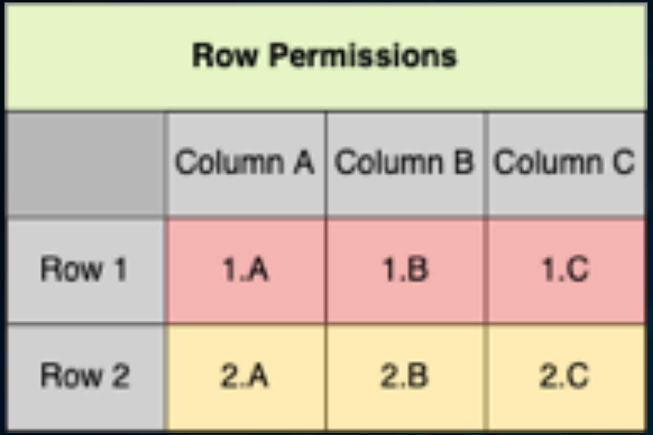


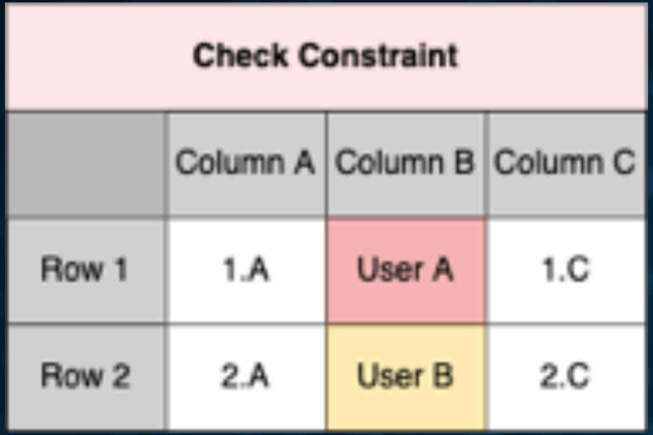
Table Permissions

	Column A	Column B	Column C
Row 1	■	■	■
Row 2	■	■	■



Row Permissions

	Column A	Column B	Column C
Row 1	1.A	1.B	1.C
Row 2	2.A	2.B	2.C



Check Constraint

	Column A	Column B	Column C
Row 1	1.A	User A	1.C
Row 2	2.A	User B	2.C

COMCAST

BUSINESS LOGIC

GRAPHQL ISN'T JUST CRUD

Mutations in GraphQL vernacular encompass all potentially modifying operations.

```
mutation CreateMyDivision {
  createDivision(input:
    {division:
      {name: "MyDivision"}
    }
  ) {division {
      name
      nodeId
      regionsByDivision {
        nodes {
          name
        }}
    }}
}
```

COMCAST

BUSINESS LOGIC

GRAPHQL ISN'T JUST CRUD

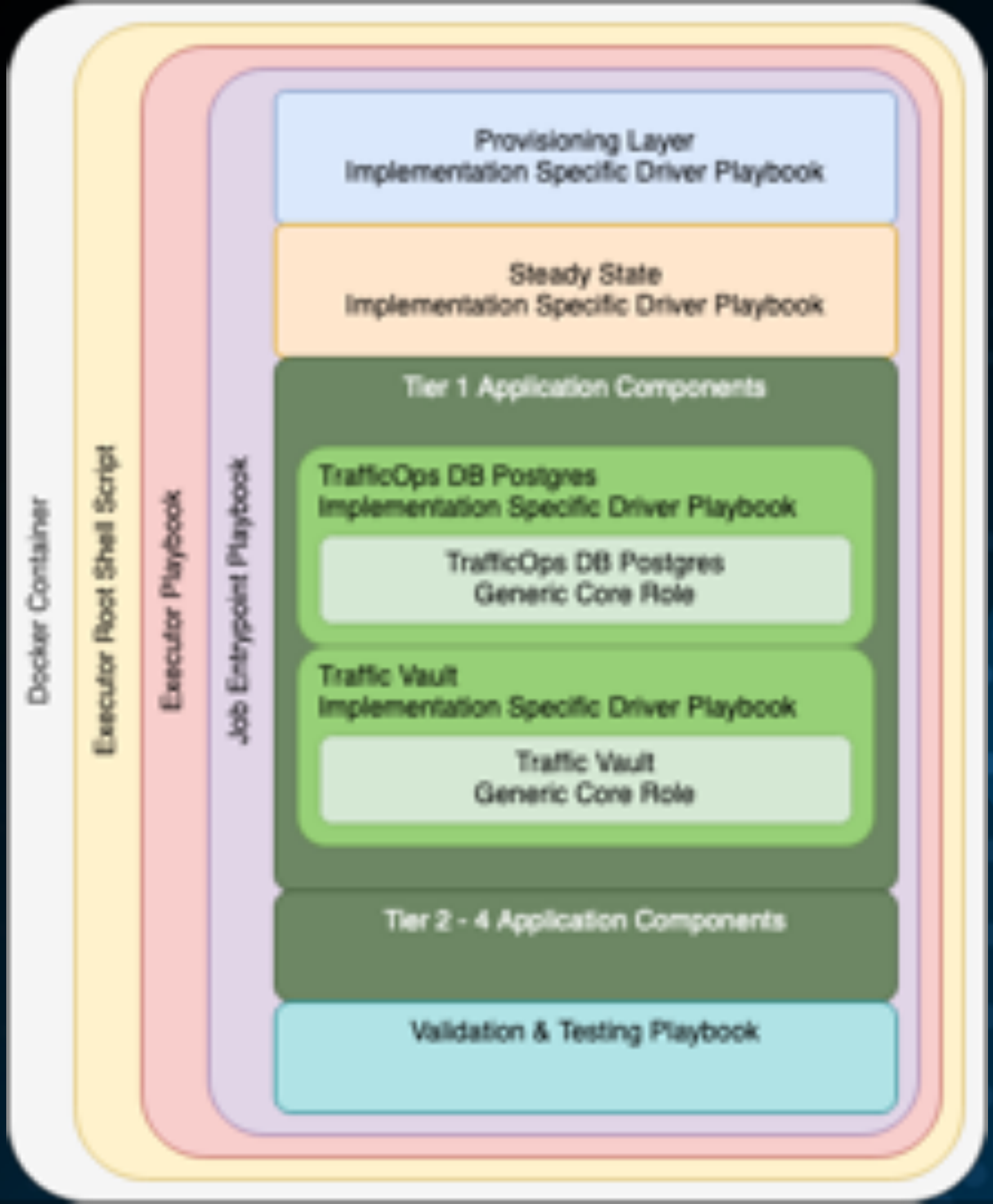
Mutations in GraphQL vernacular encompass all potentially modifying operations.

```
mutation CreateMyDivision {
  createDivision(input:
    {division:
      {name: "MyDivision"}
    }
  ) {division {
      name
      nodeId
      regionsByDivision {
        nodes {
          name
        }}
    }}
}
```

```
mutation DeepDivisionCreation {
  deepDivisionCreation(input:
    {division:
      {name: "MyDivision"}
    },
    {region:[
       {name: "MyRegion1"},{name: "MyRegion2"}
    ]}
  ) {division {
      name
      nodeId
      regionsByDivision {
        nodes {
          name
      }}
    }}
}
```

LAB EXECUTOR

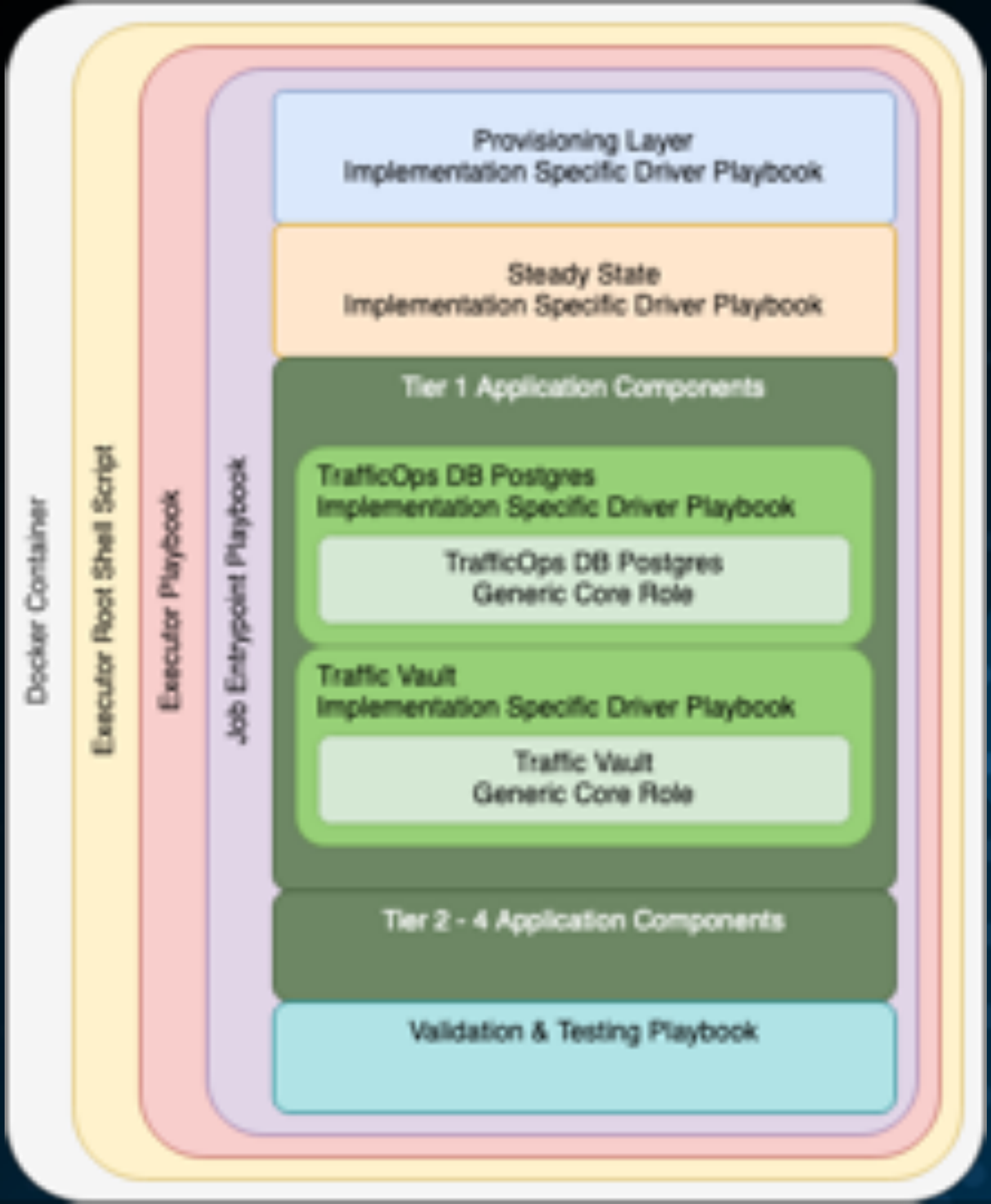
ABSTRACTIONS


Docker Container
Executor Root Shell Script
Executor Playbook
Job Entrypoint Playbook
Provisioning Layer
Implementation Specific Driver Playbook
Steady State
Implementation Specific Driver Playbook
Tier 1 Application Components
TrafficOps DB Postgres
Implementation Specific Driver Playbook
TrafficOps DB Postgres
Generic Core Role
Traffic Vault
Implementation Specific Driver Playbook
Traffic Vault
Generic Core Role
Tier 2 - 4 Application Components
Validation & Testing Playbook

ABSTRACTIONS

DOCKER CONTAINER

- Insulate Dependencies
- Improve Portability


Docker Container
Executor Root Shell Script
Executor Playbook
Job Entrypoint Playbook
Provisioning Layer
Implementation Specific Driver Playbook
Steady State
Implementation Specific Driver Playbook
Tier 1 Application Components
TrafficOps DB Postgres
Implementation Specific Driver Playbook
TrafficOps DB Postgres
Generic Core Role
Traffic Vault
Implementation Specific Driver Playbook
Traffic Vault
Generic Core Role
Tier 2 - 4 Application Components
Validation & Testing Playbook

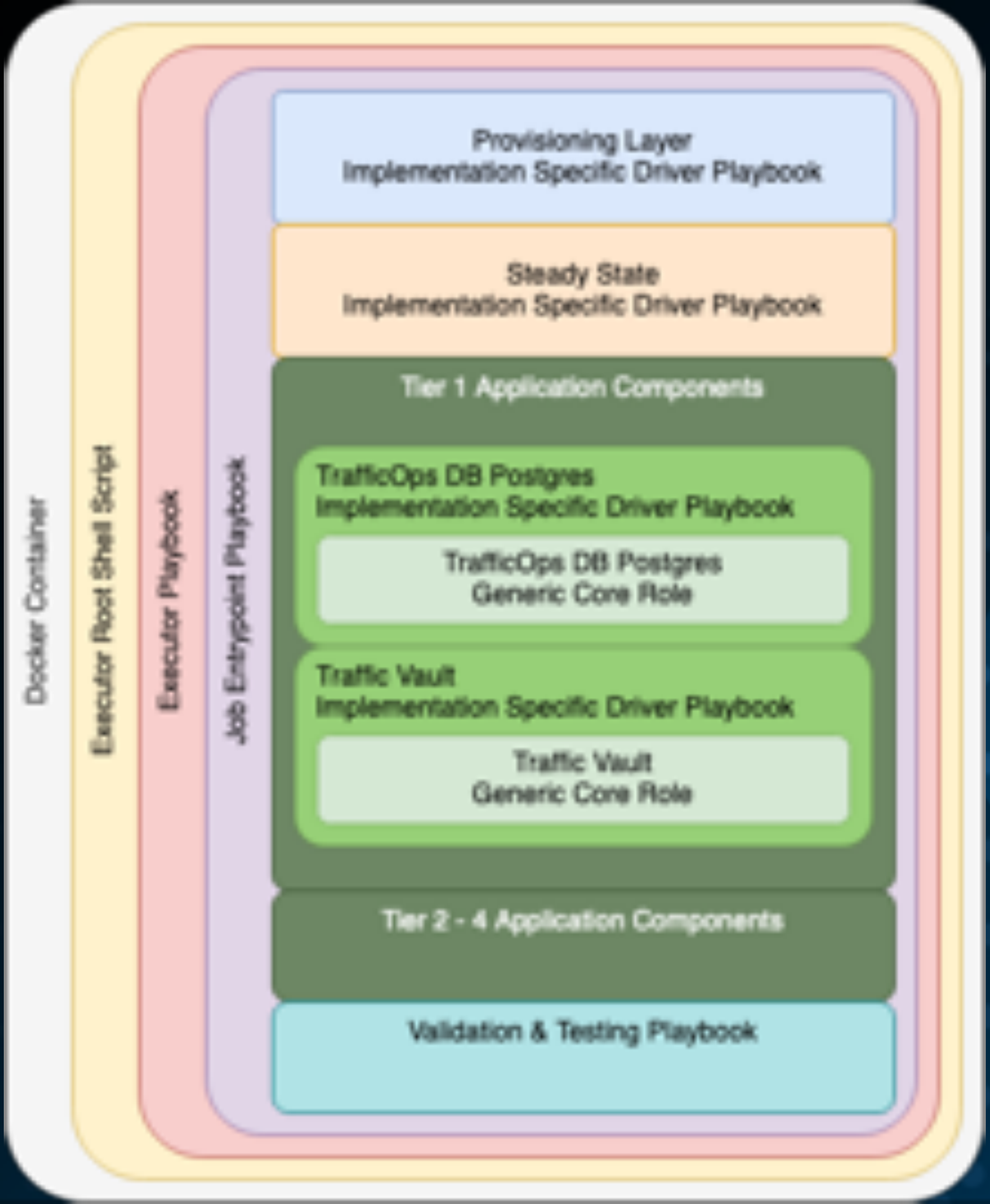
ABSTRACTIONS

DOCKER CONTAINER

- Insulate Dependencies
- Improve Portability

EXECUTOR ROOT SHELL SCRIPT

- Redirect its own output to itself
- Scrub & Submit Logs
- Update Job State



Docker Container
Executor Root Shell Script
Executor Playbook
Job Entrypoint Playbook
Provisioning Layer
Implementation Specific Driver Playbook
Steady State
Implementation Specific Driver Playbook
Tier 1 Application Components
TrafficOps DB Postgres
Implementation Specific Driver Playbook
TrafficOps DB Postgres
Generic Core Role
Traffic Vault
Implementation Specific Driver Playbook
Traffic Vault
Generic Core Role
Tier 2 - 4 Application Components
Validation & Testing Playbook

ABSTRACTIONS

DOCKER CONTAINER

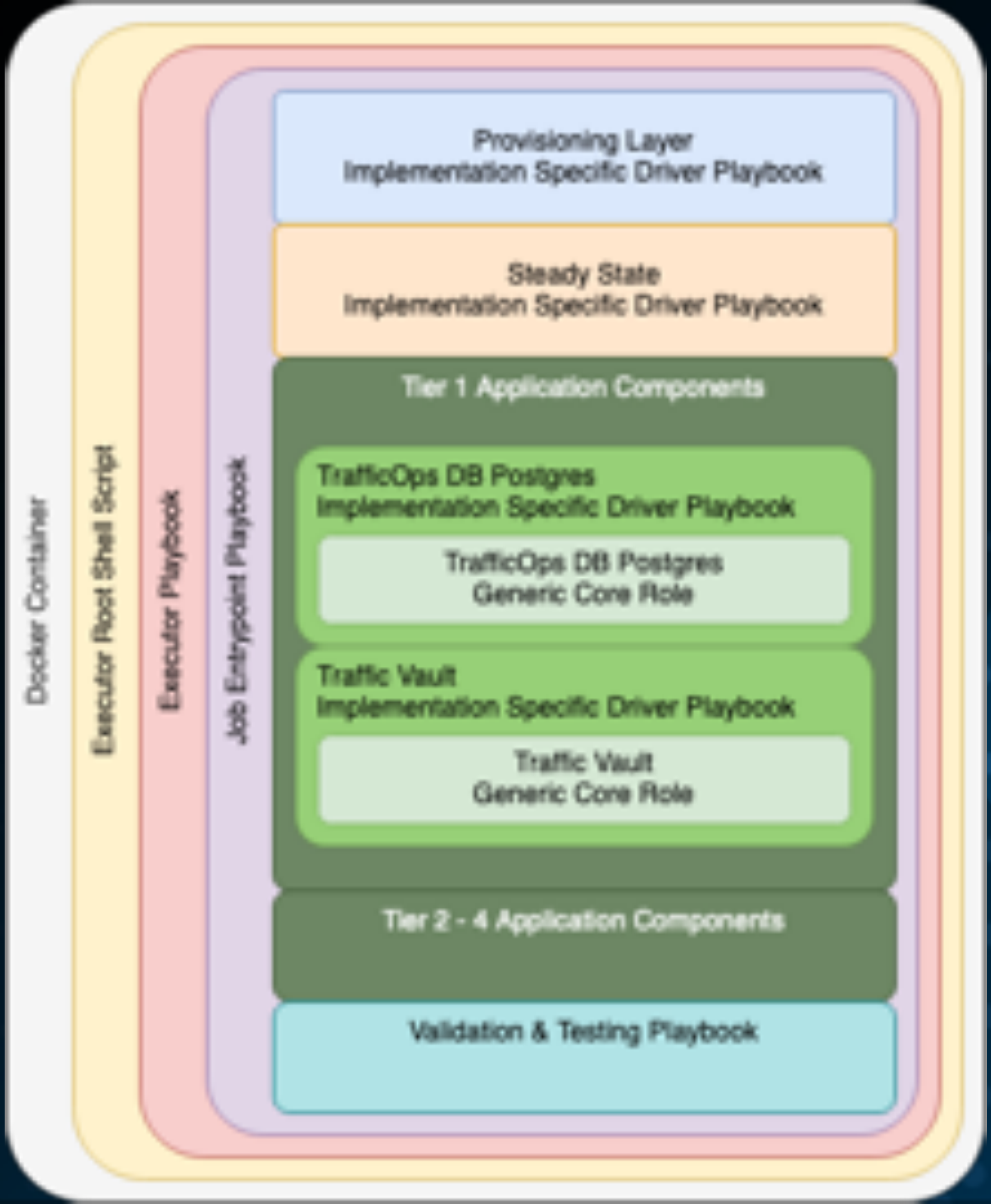
- Insulate Dependencies
- Improve Portability

EXECUTOR ROOT SHELL SCRIPT

- Redirect its own output to itself
- Scrub & Submit Logs
- Update Job State

EXECUTOR PLAYBOOK

- Obtain available Job
- Weave execution directory code
- Dump all job information


Docker Container
Executor Root Shell Script
Executor Playbook
Job Entrypoint Playbook
Provisioning Layer
Implementation Specific Driver Playbook
Steady State
Implementation Specific Driver Playbook
Tier 1 Application Components
TrafficOps DB Postgres
Implementation Specific Driver Playbook
TrafficOps DB Postgres
Generic Core Role
Traffic Vault
Implementation Specific Driver Playbook
Traffic Vault
Generic Core Role
Tier 2 - 4 Application Components
Validation & Testing Playbook

ABSTRACTIONS

DOCKER CONTAINER

- Insulate Dependencies
- Improve Portability

EXECUTOR ROOT SHELL SCRIPT

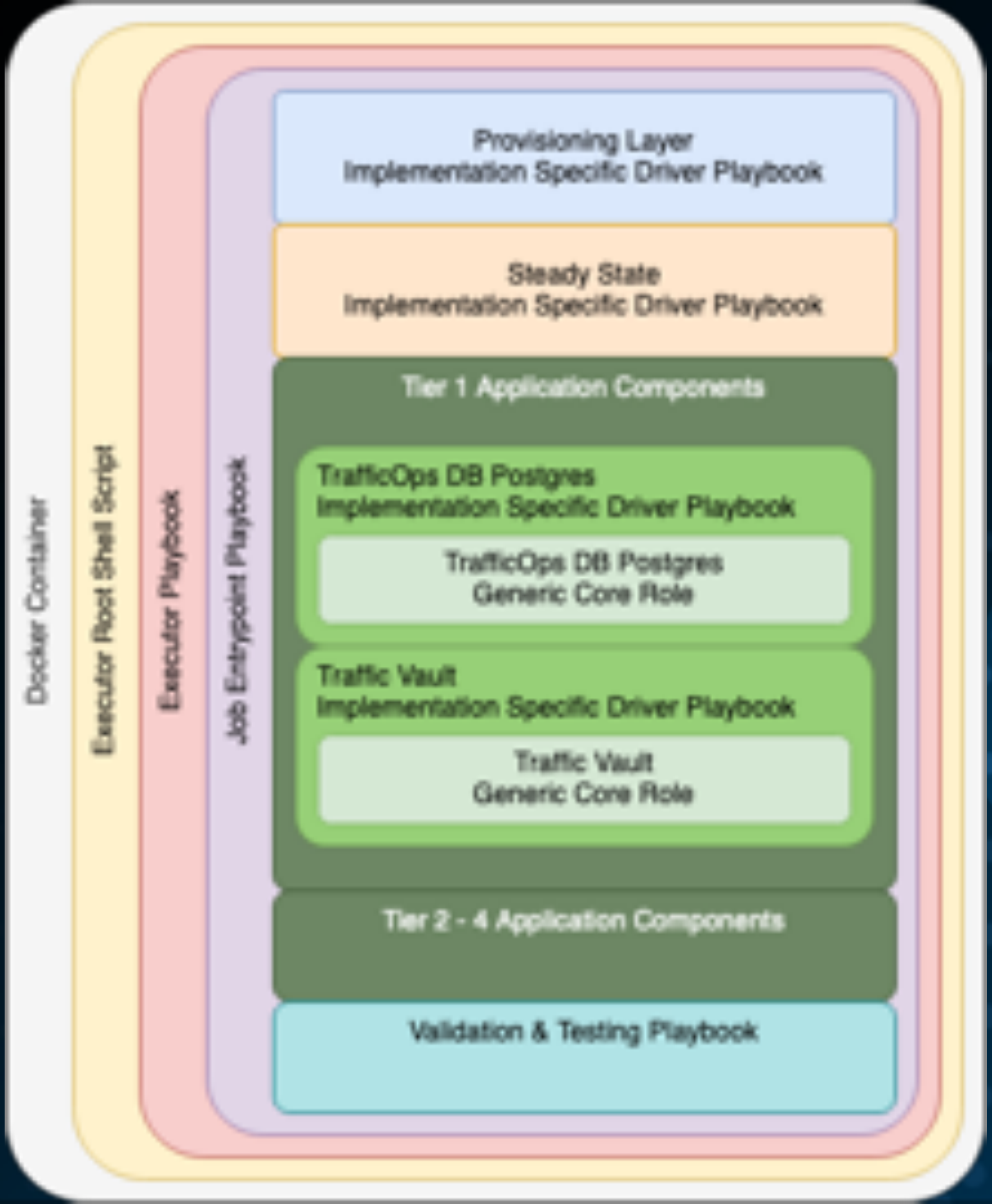
- Redirect its own output to itself
- Scrub & Submit Logs
- Update Job State

EXECUTOR PLAYBOOK

- Obtain available Job
- Weave execution directory code
- Dump all job information

JOB ENTRYPOINT PLAYBOOK

- Considered Main Execution for Job


Docker Container
Executor Root Shell Script
Executor Playbook
Job Entrypoint Playbook
Provisioning Layer
Implementation Specific Driver Playbook
Steady State
Implementation Specific Driver Playbook
Tier 1 Application Components
TrafficOps DB Postgres
Implementation Specific Driver Playbook
TrafficOps DB Postgres
Generic Core Role
Traffic Vault
Implementation Specific Driver Playbook
Traffic Vault
Generic Core Role
Tier 2 - 4 Application Components
Validation & Testing Playbook

EXECUTION LOGGING & SECURITY

INTERESTED?

APACHECON 2019

- https://tinyurl.com/AutomatingATCSlides
- https://tinyurl.com/AutomatingATCVideo

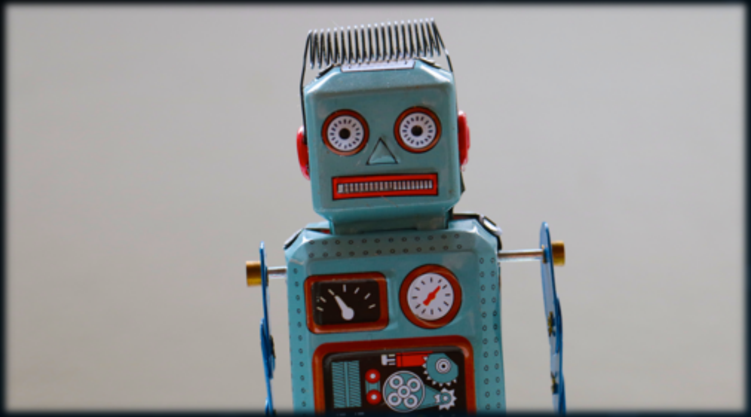
APACHECON 2020

- https://tinyurl.com/SelfServiceCDNSlides
- https://tinyurl.com/SelfServiceCDNVideo

Photo by Branden Tate on Unsplash

TAKEAWAYS

1. Obtain a basic understanding of Ansible
2. See how Comcast has leveraged the Open-Source Ansible roles for ATC.
3. Learn more about technology stack choices we've made.
4. Gain a better understanding of how deep the rabbit hole goes with modeling complex systems.



Jonathan Gray

Twitter : @jhg03a
GitHub : @jhg03a
traffic-control-cdn.slack.com : @jhg03a
The-asf.slack.com : @jhg03a
jhg03a@apache.org

https://unsplash.com/photos/R4WCbazrD1g
https://about.twitter.com/en_us/company/brand-resources.html
https://github.com/logos
https://slack.com/media-kit
https://commons.wikimedia.org/wiki/File:Antu_mail-folder-sent.svg

COMCAST