AWS re:Invent

CMP208-S

Achieving unparalleled elasticity on AWS at the lowest cost

Andy Walton
VP of Technical Advisors
Densify

Alvin Huang
Senior Release Engineer
HashiCorp

AWS re:Invent

© 2019, Amazon Web Services, Inc. or its affiliates. All rights reserved.

aws

Overview

The Lucky 7 Recommendations

HashiCorp Terraform and Densify –
Effective Cloud Infrastructure Management

Demo

1 – Align Key Stakeholders

© 2019, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Different Roles & Priorities – Day 1

Line of Business

LOB Head

LOB IT Finance

Application Owner

Container Platform Service Owner

LOB Cloud Engineering

- Often, leaders in cloud innovation
- Deploying quickly and micro-purchasing
- No time to “manage resources”

Different Roles & Priorities – Day 2

Line of Business

LOB Head

LOB IT Finance

Application Owner

Container Platform Service Owner

LOB Cloud Engineering

Central IT

Standardize, Govern, and Manage Costs

Platform Engineering

IT Finance

Central Container Service Owner

Central CloudOps

Different Roles & Priorities – Day 2

Line of Business

LOB Head

My app *needs* to perform! Save money at what risk?

LOB IT Finance

Application Owner

What about node scaling?

Container Platform Service Owner

That breaks our DevOps and IaC processes!

LOB Cloud Engineering

Central IT

Standardize, Govern, and Ma[illegible]ge Costs

We can save by purchasing *more* RIs.

Platform Engineering

IT Finance

We can save *more* if we downsize instances.

Central Container Service Owner

Central CloudOps

2 – Gain Cost Visibility

© 2019, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Gain Cost Visibility

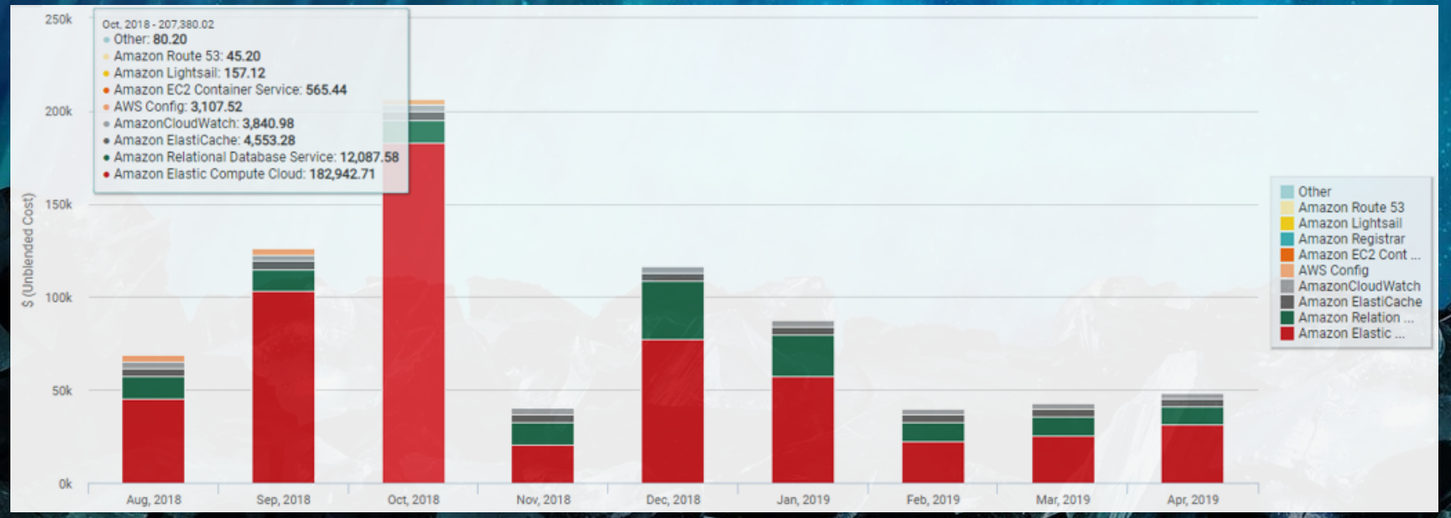
The Challenge

- Wasted spend
- Complex invoices
- Understanding chargeback

Gain Cost Visibility

The Solution

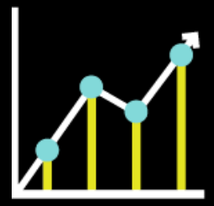
- Gain complete visibility
  - Daily, weekly, monthly, quarterly, yearly
- LOB chargebacks
- Detailed budget management
- Create custom reports



3 – Ensure Application Performance

© 2019, Amazon Web Services, Inc. or its affiliates. All rights reserved.

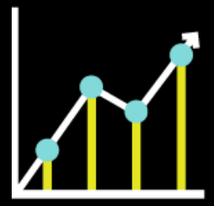
Ensure Application Performance



The Challenge

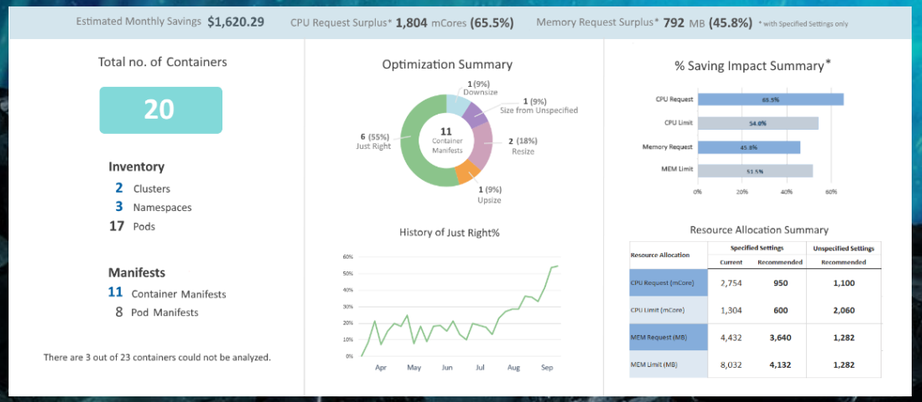
- Understanding provider SLA
- 2 key questions
  - Are you using the right instance types?
  - Is your application designed correctly for the cloud?

Ensure Application Performance



The Solution

- Optimization tool requirements
  - Average, predictive, peak for benchmark
  - Minimum amount of data to be analyzed
  - Cross-platform recommendations
  - Standard or custom performance thresholds
  - CPU, RAM, storage, network



4 – Achieve Governance & Compliance

© 2019, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Enforce Governance & Compliance

The Challenge

- Assigning the appropriate tags to ensure…
  - That the right people have the right access
  - That you are meeting regulatory compliance

Enforce Governance & Compliance

The Solution

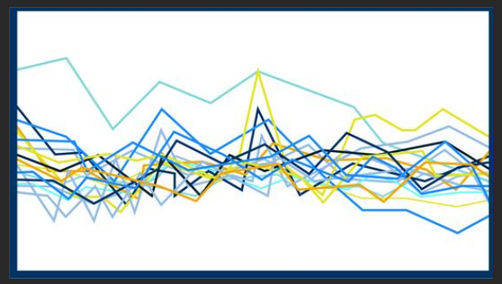
- Implement or design a solution that adheres to regulatory compliance (including tags)
- Identify and fix
  - Missing or non-compliant tags
  - Non-compliant systems (shut down)

5 – Select the Right Resources

© 2019, Amazon Web Services, Inc. or its affiliates. All rights reserved.

The Challenge – Millions of Permutations

Application



Complex resource requirements

Public Cloud

Numerous service offerings

Considerations

Regions?

Instance Type & Size?

Application

RI, Savings Plan, or On-Demand?

Storage? Networking?

It can be difficult to be sure you are buying the right resources

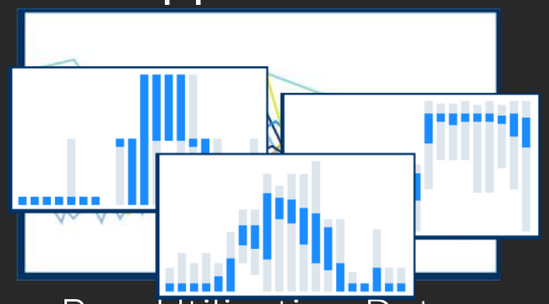
Framework to Get It Right

3. Optimize Supply & Demand

1. Establish Application Demand Patterns

2. Create Normalized Models

Application



Raw Utilization Data

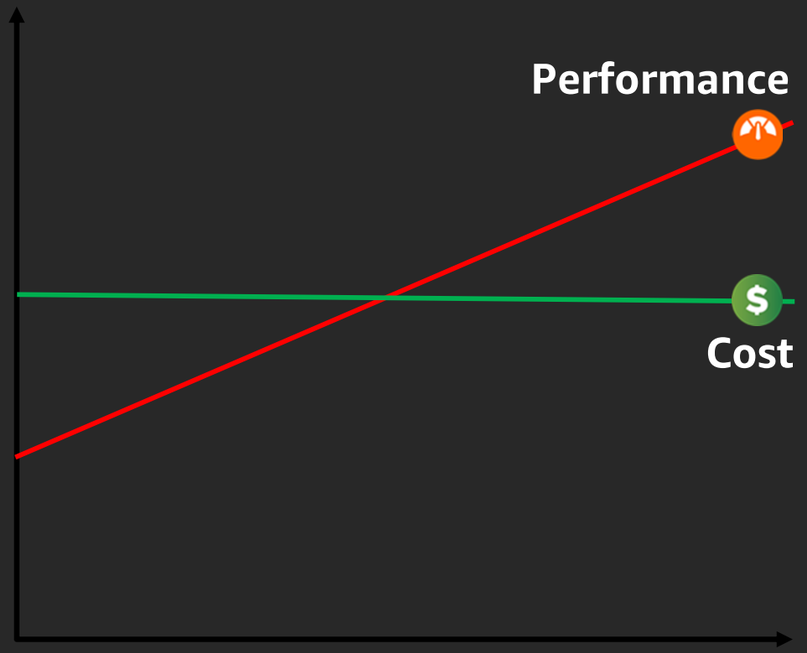
Public Cloud

Numerous Service Offerings

Balancing Priorities

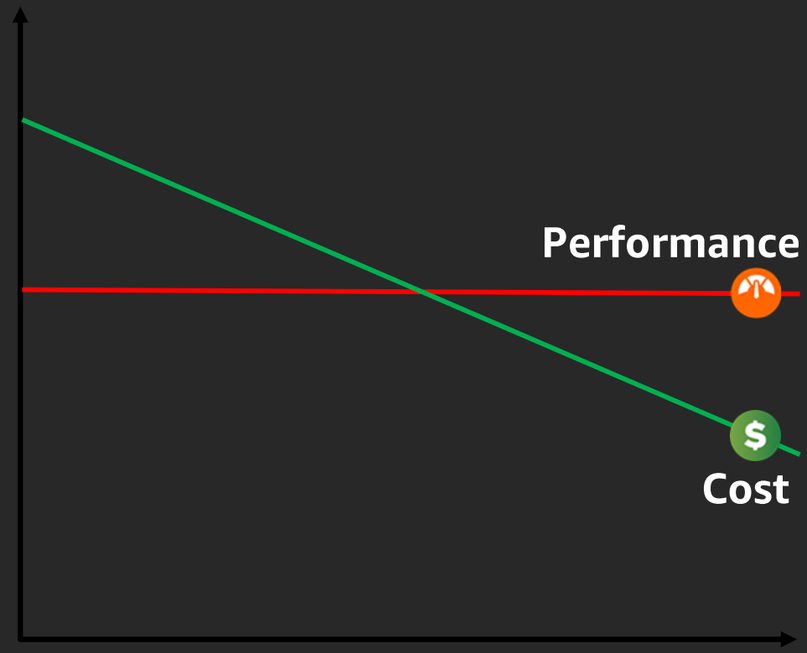
Cloud Engineering & Operations

PERFORMANCE FIRST



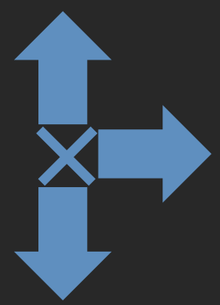
IT Finance

COST FIRST



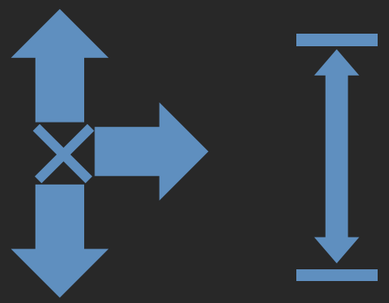
Tactics to Get It Right

Amazon EC2 & Amazon RDS Optimization



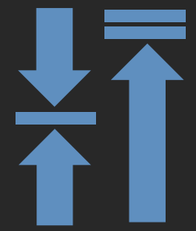
- Upsize
- Modernize
- Cross-Family
- Reclaim
- Terminate

Auto Scaling Group Optimization



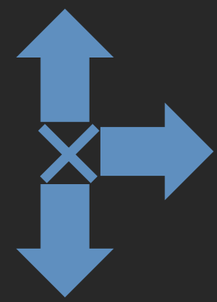
- Node Upsize
- Node Downsize
- Modernize
- Cross-Family
- Scaling Min/Max

Container Resource Optimization



- Requests
- Limits
- Initial Values

Reserved Instance Optimization



- Buy New
- Renew
- Expire
- Convert

Getting Approval from Application Owners

Effort to Optimize Instance Type

Effort: Moderate

Rule Description	Impact on Effort	Property
ENA Support is NOT Enabled and Going to Gen 5 (Nitro Hypervisor)	Medium (10%)	ENA Support
Changing processor architecture from Intel to AMD	Small (5%)	Catalog Model - Processor Features
Changing into Different Instance Family and Newer Generation	Small (3%)	Catalog Model - Instance Generation
Going to Gen 5 (Nitro Hypervisor) from Gen 4 Instance	Small (3%)	Catalog Model - Instance Generation

Cost Impact

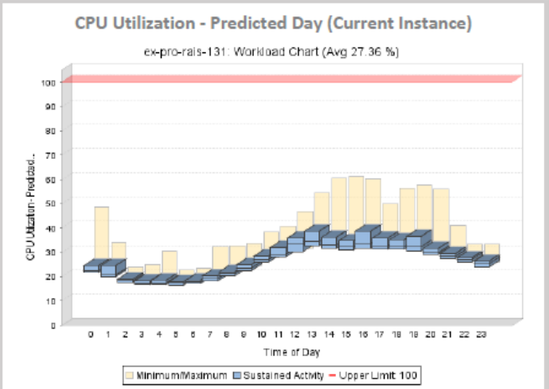
Current Monthly Cost	$109.50 Cost is based on on-demand price and predict
Recommended Cost	$61.87 Cost is based on on-demand price and predict
Monthly Savings	$47.63
Savings %	43.5%
Predicted Uptime %	75%

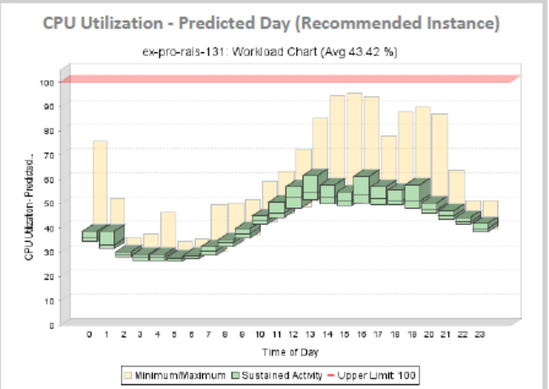
	m4.xlarge (Current)
On-Demand ($/Month)	146.00
Reserved ($/Month)†	84.42
Purchasing Strategy	On-Demand
RI Coverage %	0%

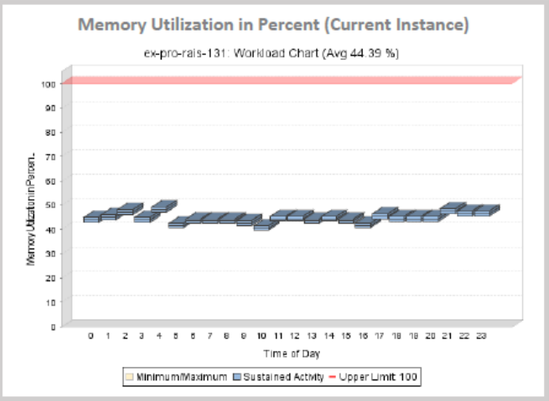
†Reserved Instance (RI) Price is based on 'Reserved Pricing Option' in the Policy.

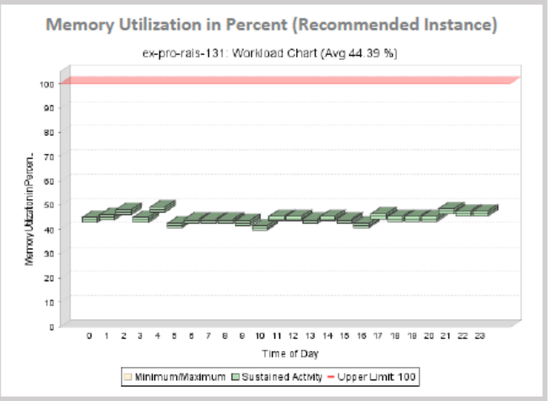
Predicted Utilization (CPU & Memory)

Utilization Metrics	m4.xlarge (Current)	r5a.large (Recommended)	Change
CPU Utilization (%)	38.8	61.5	↑ 22.8
Memory Utilization (%)	49.3	49.3	—









	ex-pro-rais-131
	i-036ddcaa03393433b
	AWS
	Alliances
	EC2 instance
	m4.xlarge
	Linux
	On-Demand
	Change the instance type from m4.xlarge (4x17.1) to r5a.large (2x17.1) [Downsize-Optimal Family].
	On-Demand
	75%
	The system has been running for 15 days (360 hours) over a period of 20 days (480 hours).
	AWS Assessement (Alliances)

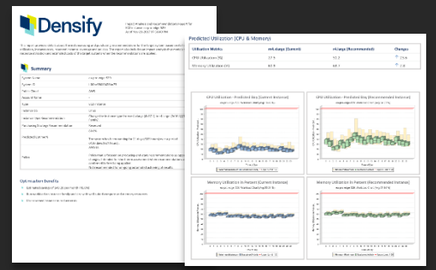
Application Owner

Making It Happen

Cloe

Cloud Learning Optimization Engine

Human-readable application owner reports



Application Owners

"I'll approve this change because I am comfortable with the proof you provided"

Machine-readable actions

```
provider "aws" {
  region = "${var.aws_region}"
}

resource "aws_instance" "web" {
    name = "Web Server"

    instance_type = "m4.large"
    instance_type =["${aws_instance.tags:Densify-optimal-instance-type}"]

    ami = "${lookup(var.aws_amis, var.aws_region)}"
}
```

Cloud Ops/Engineering Teams

"Quality answers and integration means we can automate"

6 – Optimize Containers

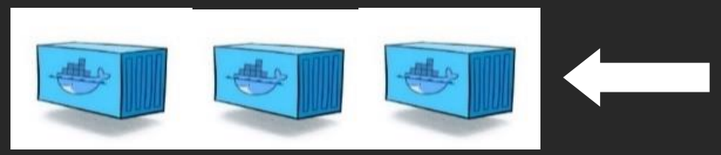
© 2019, Amazon Web Services, Inc. or its affiliates. All rights reserved.

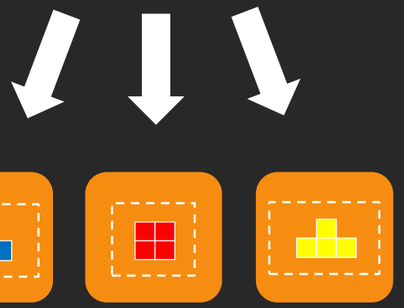
Optimize Containers

The Challenge

- Resource Optimization
- Human errors in resource specification
- Elasticity management

Hard Coding Human Generated Estimates…





Nodes (EC2 Instances) running in Scale Groups

```
resources:
  resources:
    cpu: 100m
    memory: 100Mi
  limits:
    cpu: 200m
    memory: 200Mi
```

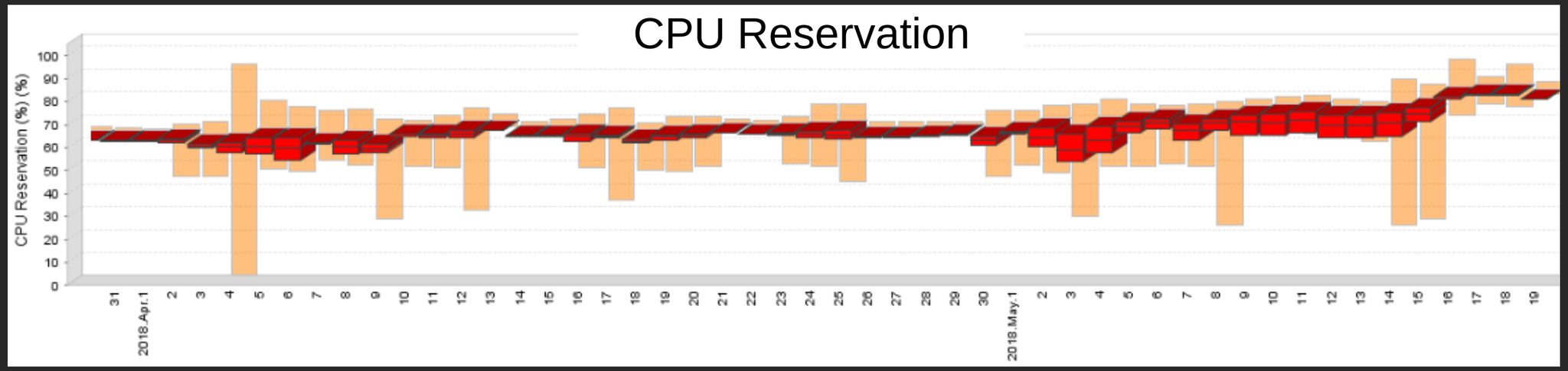
Dev & App Owners

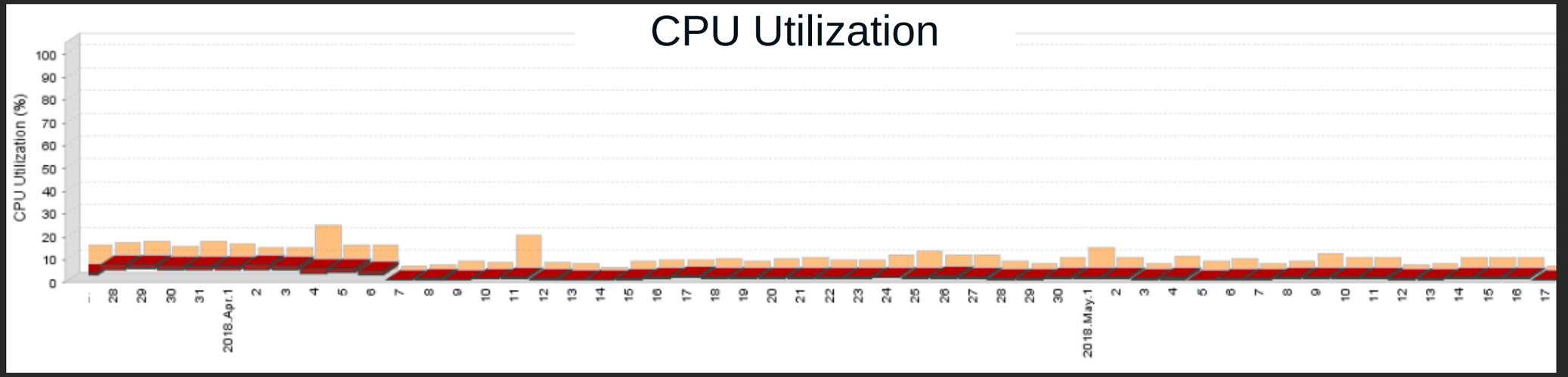
AWS CloudFormation

Terraform

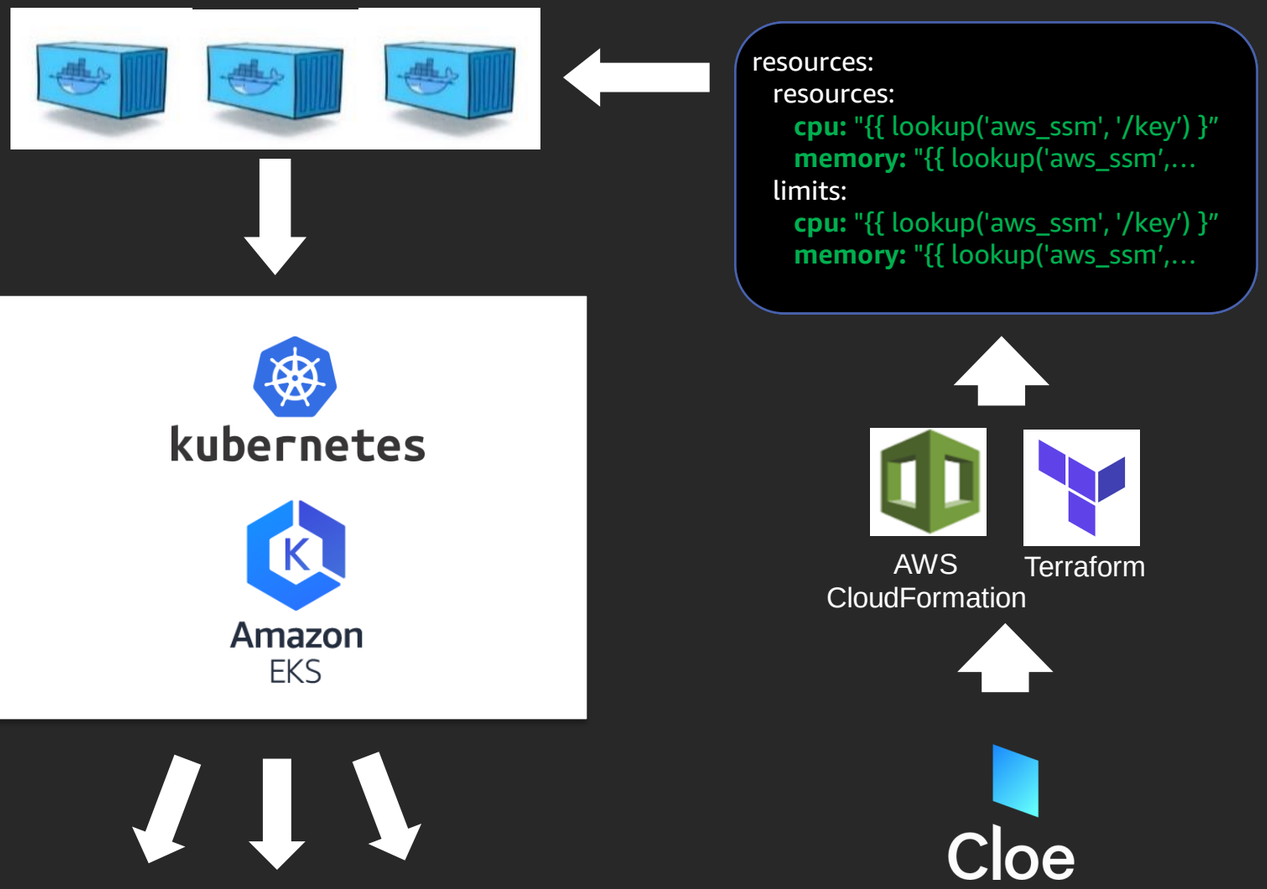
The problem this creates…

Node resources are allocated but are not actually used





Solving this using Predictive Analytics…



2 **Node Resource Optimization**

1 **Container Resource Optimization**

This is great…

But how do you manage it at scale?

7 – Automate at Scale

© 2019, Amazon Web Services, Inc. or its affiliates. All rights reserved.

HashiCorp Terraform and Densify: Effective Cloud Infrastructure Management

Alvin Huang
Sr. Release Engineer
HashiCorp

AWS re:Invent

© 2019, Amazon Web Services, Inc. or its affiliates. All rights reserved.

aws

Overview

Managing traditional vs. modern data centers

Is there a better way?

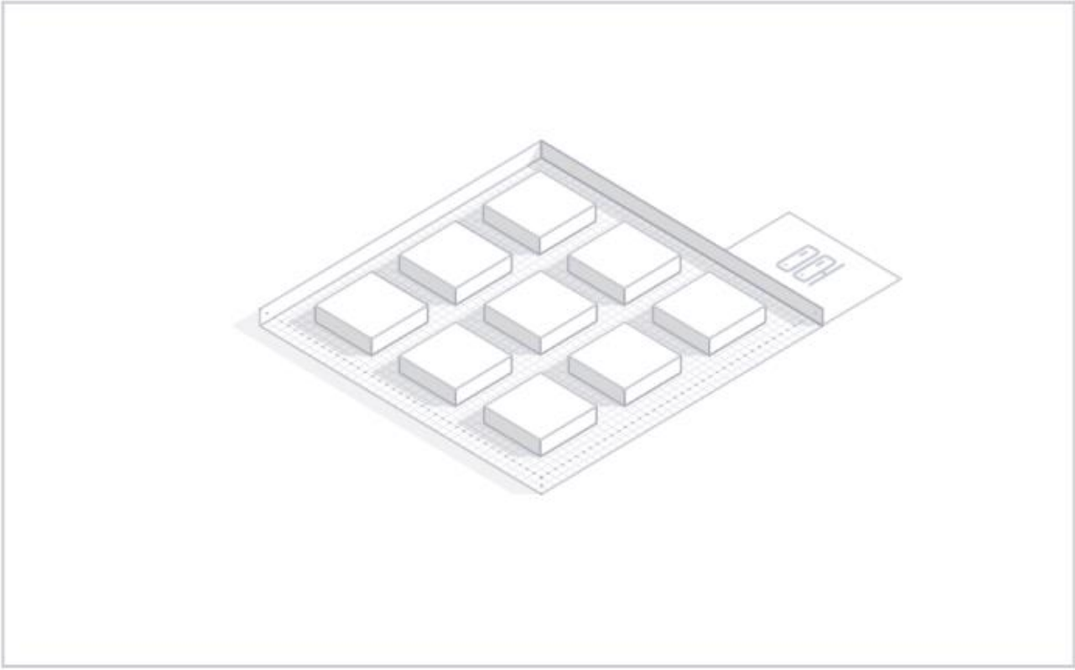
Overcoming organizational challenges

Managing traditional vs. modern data centers

© 2019, Amazon Web Services, Inc. or its affiliates. All rights reserved.

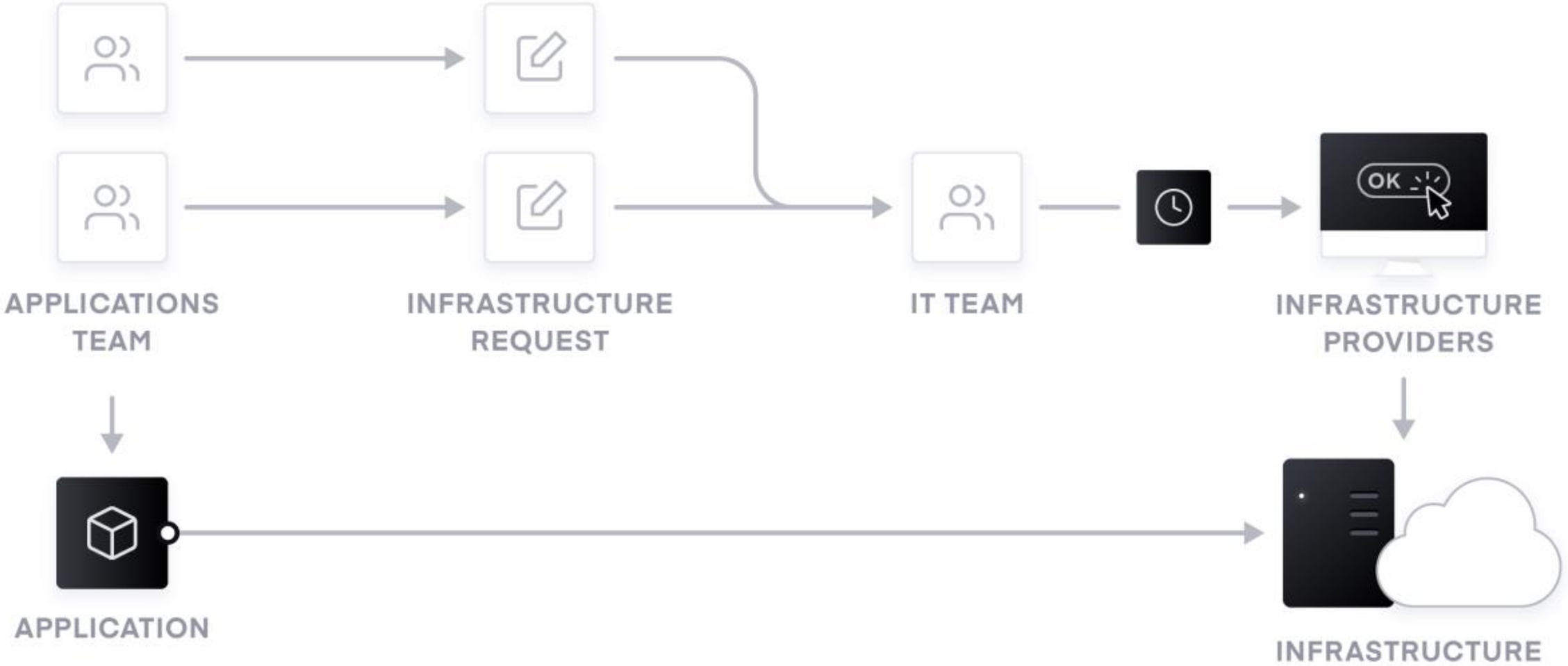
Provisioning Infrastructure Was Easy…

- Data centers had fixed sets of resources that lived for long periods of time
- IT Ops was the central gatekeeper to procure, validate, and provision infrastructure

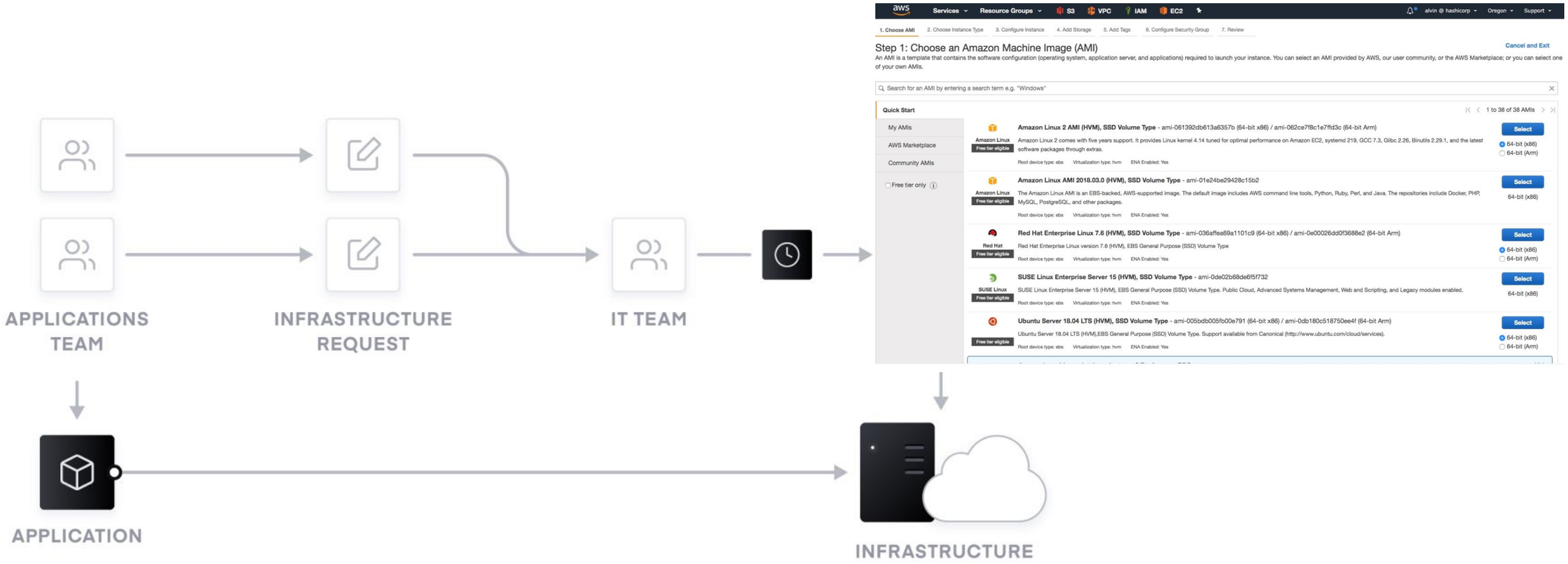


But what happens when your applications and infrastructure extend to multiple data centers, to the cloud, or all of the above?

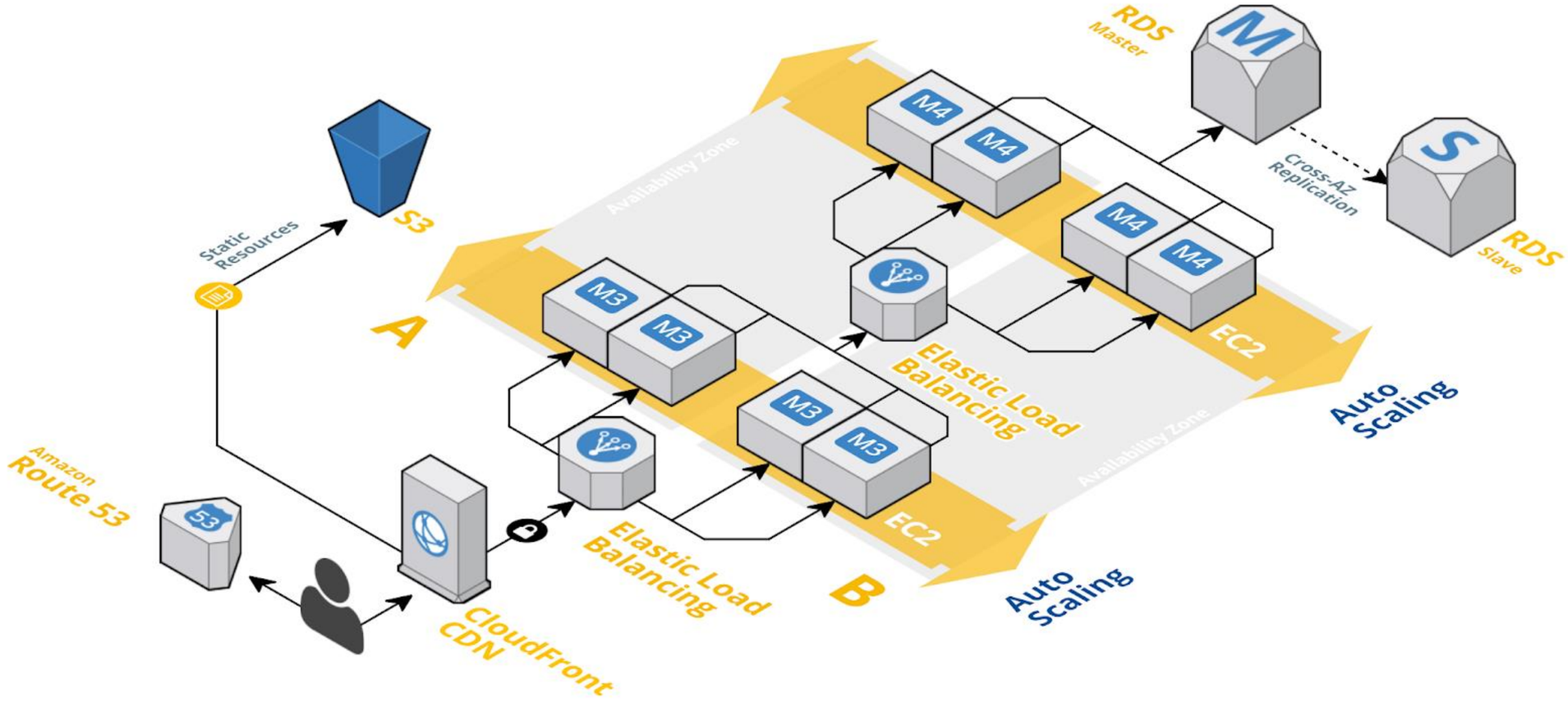
Traditional Approach



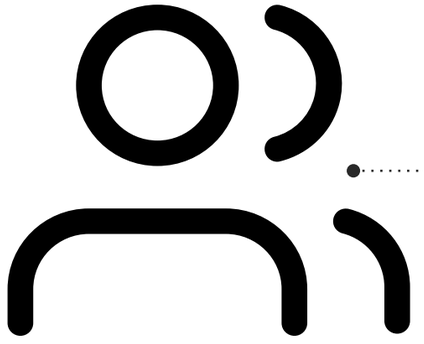
Cloud Provisioning & Management

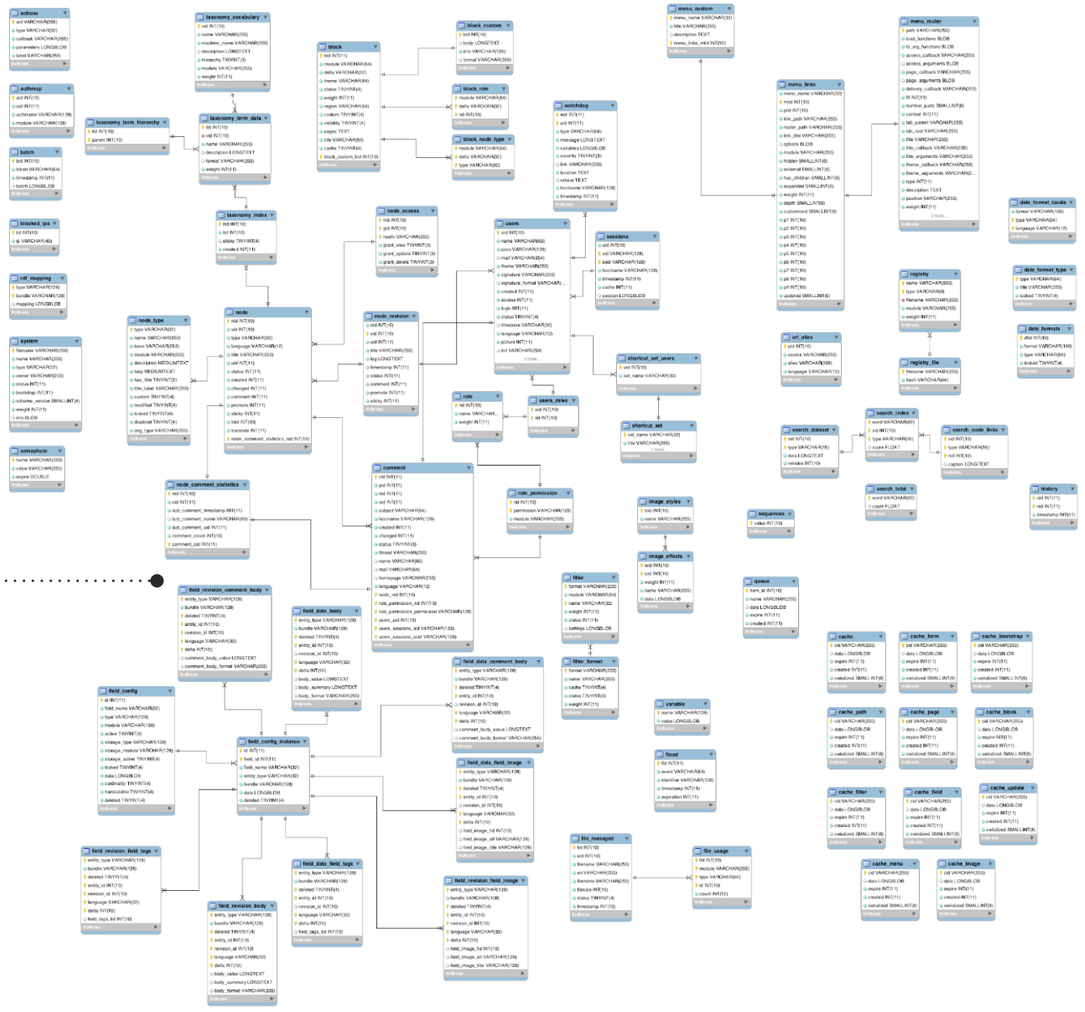


Modern Web Application



Modern Data Center Complexity





How do we capture the things we point and click, in a codified way?

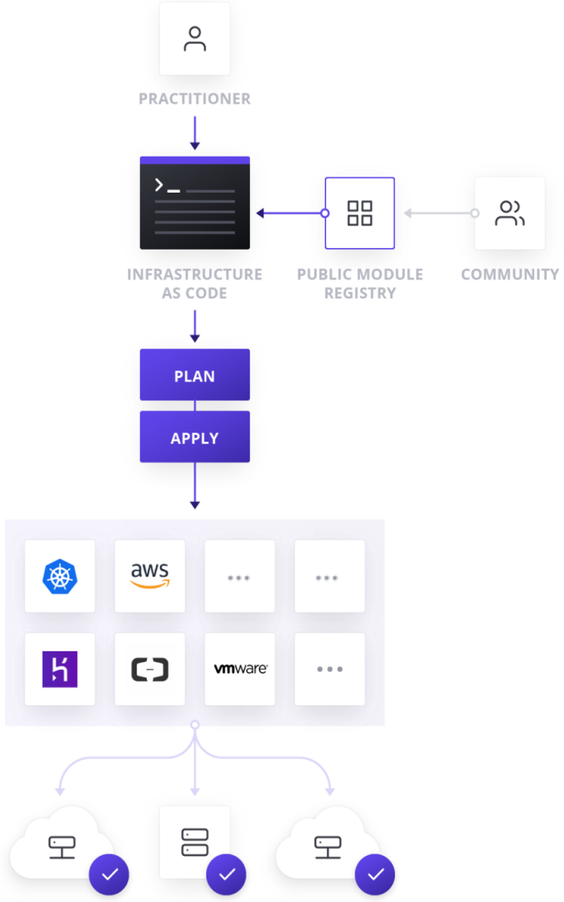
Is There a Better Way?

© 2019, Amazon Web Services, Inc. or its affiliates. All rights reserved.

HashiCorp Terraform

Provides the foundation for cloud infrastructure automation using infrastructure as code for provisioning and compliance in the cloud operating model

- Using version control and automation to provision in minutes and reduce human error and failed builds
- Terraform infrastructure as code and policy as code to automate everything
- Open-source providers allow rapid creation and support for any infrastructure



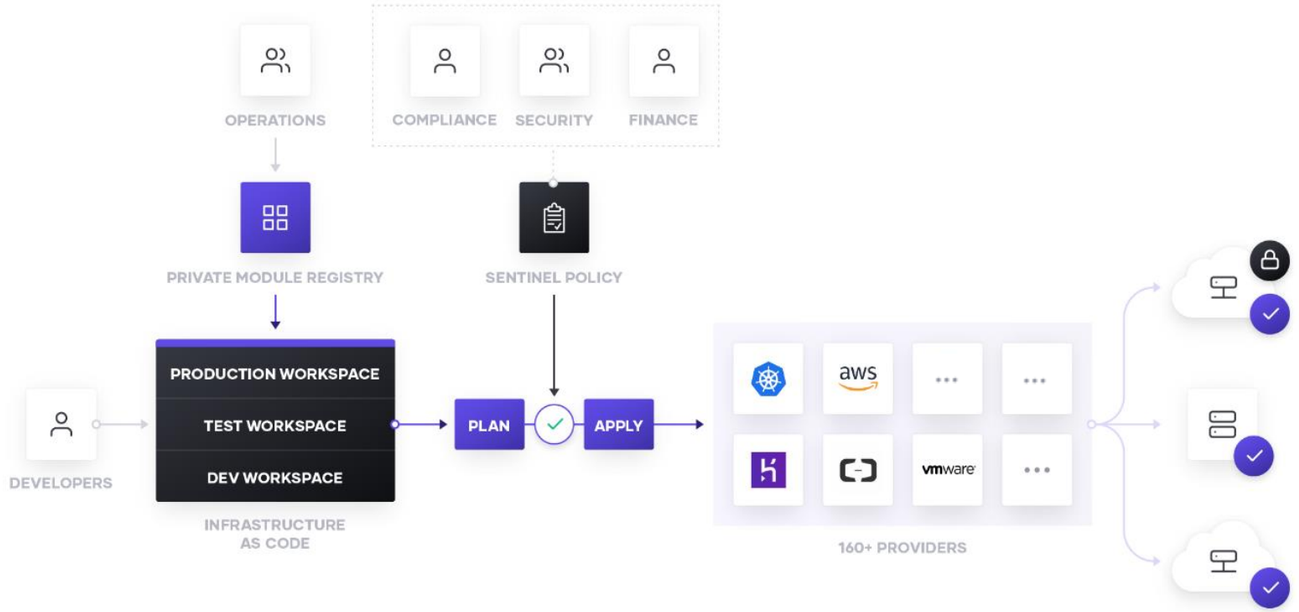
CODE EDITOR

```
provider "aws" {
  region     = "us-east-1"
}

resource "aws_instance" "example" {
  ami           = "ami-2757f631"
  instance_type = "t2.micro"
}
```

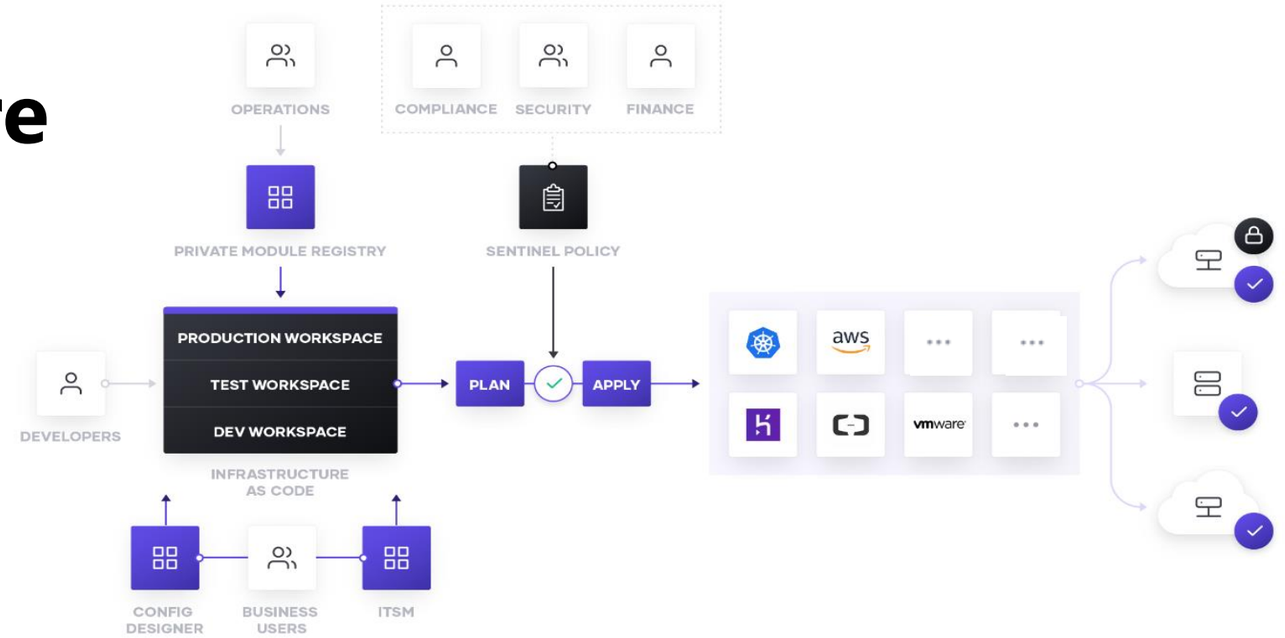
Use Case:
Cloud Compliance and Management

- **Reduce risk** with a central point for compliance & management
- **Reduce costs** by eliminating overspending on idle, orphaned, and oversized resources
- **Increase productivity** with a single workflow with 200+ unique infrastructure providers



Use Case:
Self-Service Infrastructure

- **Increase productivity** using a library of infrastructure modules for technical and non-technical end users
- **Reduce risk** with Sentinel Policies automatically enforced against every request

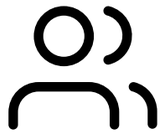


Overcoming Organizational Challenges

© 2019, Amazon Web Services, Inc. or its affiliates. All rights reserved.

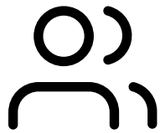
Challenges of Managing Cloud Environments

Many groups, different objectives



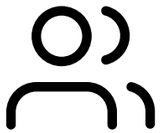
Developers/
Operators

- Efficient Deployments
- Application Downtime
- Controlled Environments



Security

- Access Control
- Policy Enforcement
- Risk and Accountability



Finance/
Corporate

- Cost Control
- Productivity
- Scalability

Developers & Operators

Solving the Challenges

Challenge		Solution
Efficient Deployments How can I do this repeatedly?	→	**Infrastructure as Code** Reusable configurations shared across operators
Application Downtime Is my infrastructure up when I need it?	→	**Self-Service Infrastructure** Enable operators to deploy infrastructure on demand
Controlled Environments Are the right applications in the right places?	→	**Cloud Management** Consistent workflow for infrastructure provisioning in all environments

Security

Solving the Challenges

Challenge	Solution
Access Control How do I control who can provision?	**Cloud Management (Organizational Structure)** Develop groups for dedicated workspaces and restrict access
Policy Enforcement Are deployments in compliance?	**Cloud Governance (Policy as Code)** Implement compliance policies that are enforced at the time of provisioning
Risk & Accountability How do I track all of this?	**Cloud Governance (Monitoring)** Leverage logs and version control to monitor who provisions what infrastructure

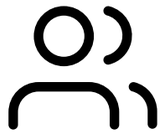
Finance and Corporate

Solving the Challenges

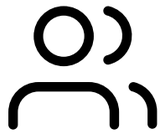
Challenge	Solution
Cost Control How do I keep spending down?	**Cloud Governance (Cost Estimation)** Understand the impact of changes to infrastructure before provisioning
Productivity Are my operators working efficiently?	**Infrastructure as Code** Enable operators to deploy infrastructure more quickly and efficiently
Scalability Are we meeting end-user demand?	**Self-Service Infrastructure** Operators provision on demand, ensuring that infrastructure is ready for end users

Solving the Management Challenge

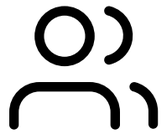
Methods for solving across the organizations



Developers/
Operators



Security



Finance/
Corporate

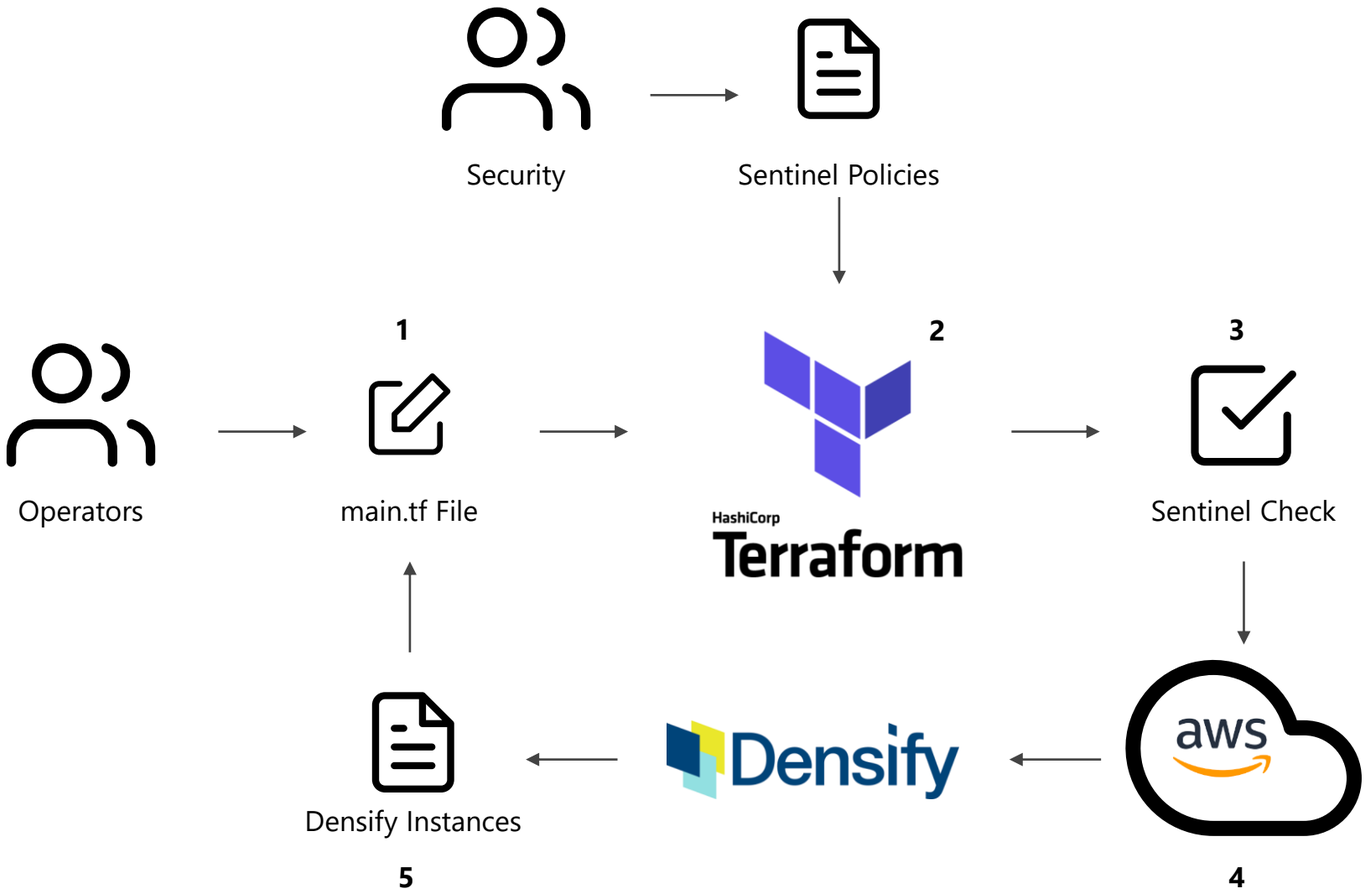
Infrastructure as Code ✓

Self-Service Infrastructure ✓

Cloud Management & Compliance ✓

Implementing Terraform & Densify

Using the products to manage the AWS environment

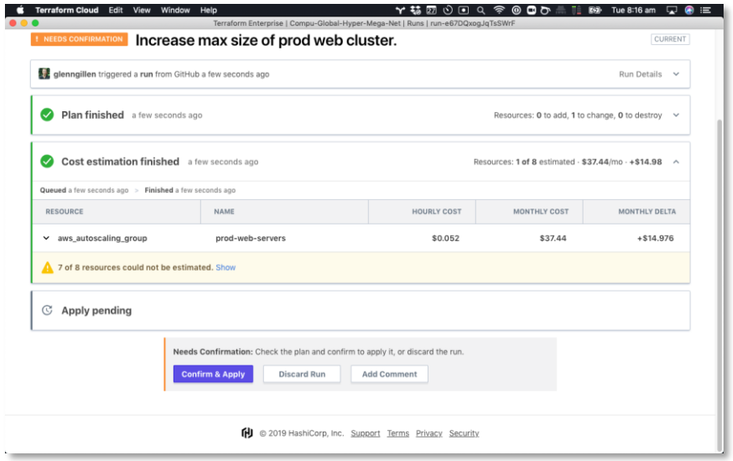


1. Operators write Terraform configuration file with Densify module
2. Terraform constructs plan based on config file
3. Terraform checks infrastructure against Sentinel Policies
4. Terraform builds desired AWS environment
5. Densify pulls data from infrastructure for future optimizations

Cost Control with Terraform & Densify

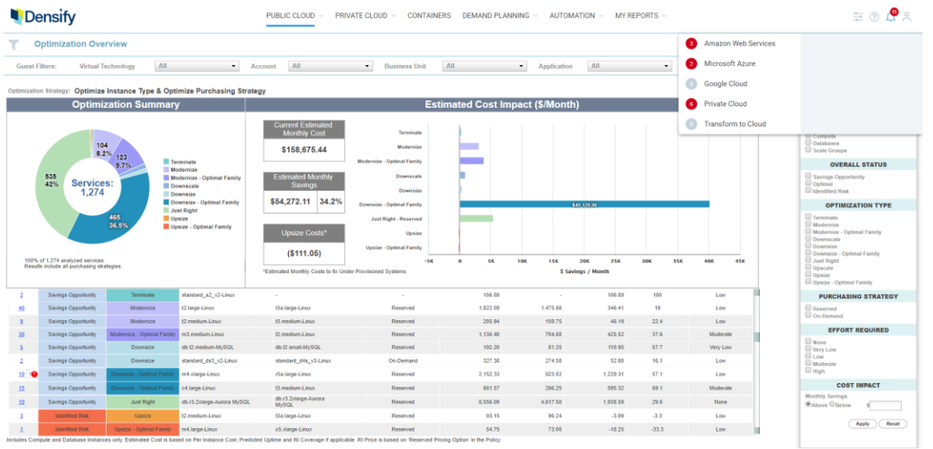
Pre-Provisioning

Build Sentinel Policies that require Densify instance types and enforce budget restrictions using Terraform's cost estimation capabilities



Post-Provisioning

Use Densify's machine learning platform to develop optimized instance types to help control organizational cost



Enabling Automation at the Infrastructure as Code Level

?

LOB Cloud Engineering

```
1  provider "aws" {
2      region = "${var.aws_region}"
3  }
4
5  resource "aws_instance" "web" {
6      name = "Web Server"
7
8      # instance_type = "m4.large"
9      instance_type = ["${aws_instance.tags:Densify-optimal-instance-type}"]
10
11     ami = "${lookup(var.aws_amis, var.aws_region)}"
12 }
```

Application *optimizes itself* based on learned behavior

CI/CD/CO (Continuous Optimization)

ML Repo

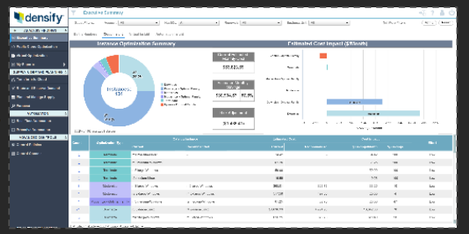
Human-Readable

Machine-Readable

slack

Subscribe to notifications
push policies to Densify

Densify

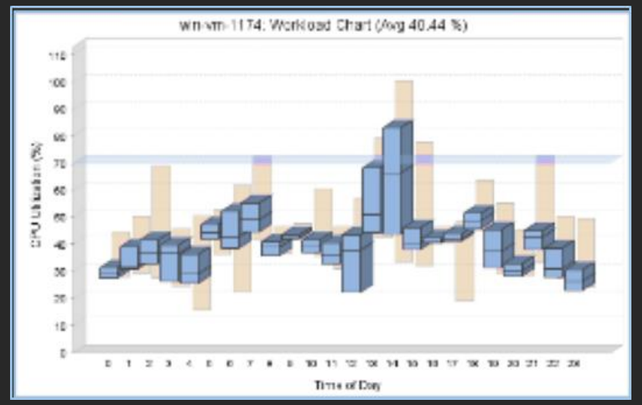


Recommendations

2

Learn & analyze

1



3

Build trigger

DevOps Pipeline

Update instance
(“self-optimizing”)

Check for approved actions

4

Application Owner

Application Source Code

LOB Cloud Engineering

Merge/Build

Deploy

Demo

© 2019, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Related breakouts

Monday

DOP205-S – Amplifying CI/CD helped Wag! reduce release process by 600%

GPSTEC410 – GitOps compliance on AWS: Enforcing immutable infrastructure

CON420-R – Infrastructure as code for containers

MOB406-R – Expanding the AWS Amplify CLI: Adding custom features

OPN309-R – CI/CD using Jenkins and Spinnaker on Amazon EKS

Tuesday

GPSTEC344 – HashiCorp Enterprise on AWS best practices

NET211-S – Service mesh across hybrid infrastructure

Thank you!

Andy Walton

awalton@densify.com
Booth 1412

Alvin Huang

alvin@hashicorp.com
Booth 2434

AWS re:Invent

© 2019, Amazon Web Services, Inc. or its affiliates. All rights reserved.

aws

Please complete the session survey in the mobile app.

© 2019, Amazon Web Services, Inc. or its affiliates. All rights reserved.