HLC307

AWS re:INVENT

Building a Healthcare-Compliant Platform for Adopting a Cloud-First Strategy Using AWS

Torsten Kablitz – VP of IT, Cloud Engineering, Change Healthcare
Akhtar Hossain – Enterprise Solutions Architect, AWS

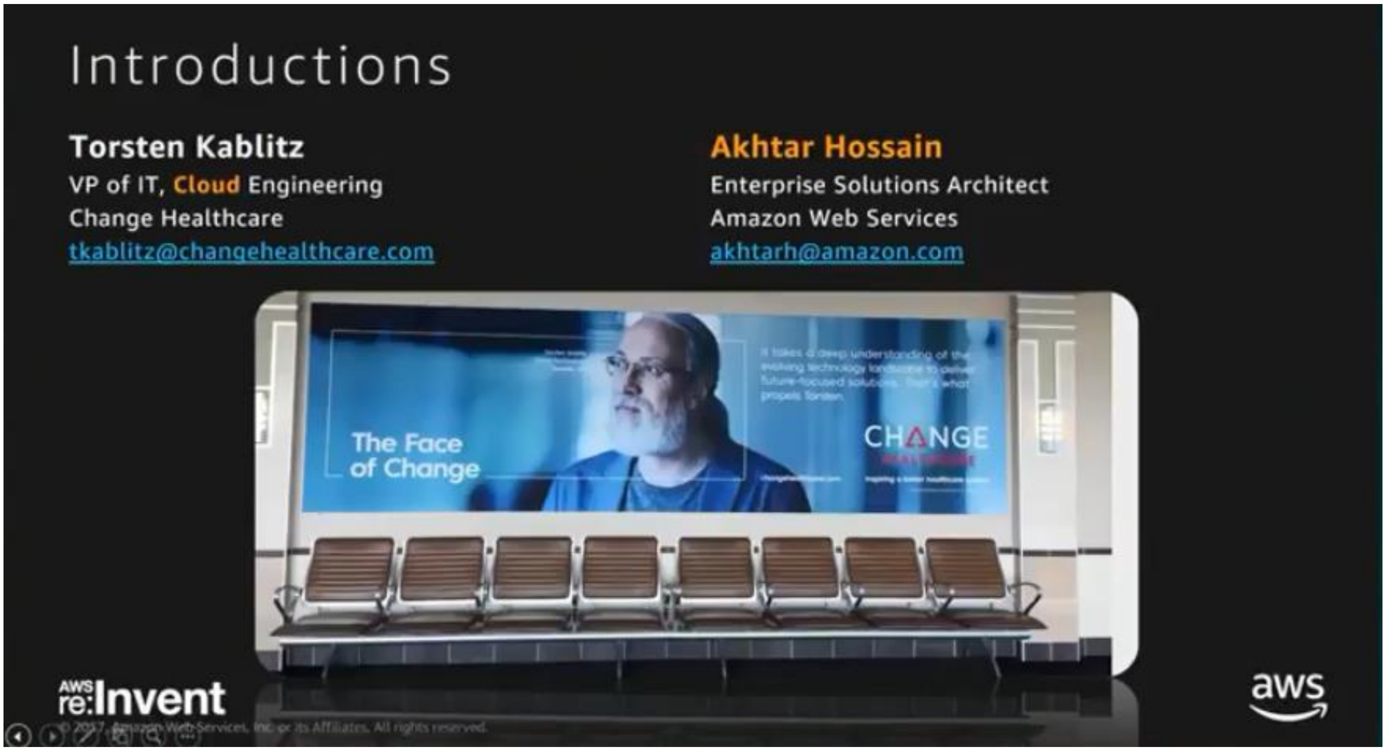
November 27, 2017

This session provides an overview of how Change Healthcare invested in people, process, and an automation platform to adopt a cloud-first strategy. Starting from building a Cloud Center of Excellence team, they identified the compliance, security, and cost optimization requirements and process required to build a framework. They also embedded healthcare compliance, security, architecture best practices, and customer-specific rules and standards for a managed adoption of the cloud. Change Healthcare is leveraging their Cloud 2.0 framework to rapidly deploy their mission applications into AWS. Come learn how Change Healthcare built a serverless architecture using Amazon ECS, AWS Lambda, AWS CodeDeploy, AWS CodeCommit, AWS CloudFormation, AWS Service Catalog, AWS OpsWorks, AWS Elastic Beanstalk, and other managed services.

Building a secure healthcare-compliant framework to accelerate the adoption of our Cloud-First strategy on AWS

Agenda

- Introductions
- A brief history of our journey to the Cloud
- Establishing a Cloud Center of Excellence
- Heuristics: Cloud-First and Security by Design
- Automation in all things
- Building a Gold Base AMI
- Scanning for compliance
- Cost Management



Introductions

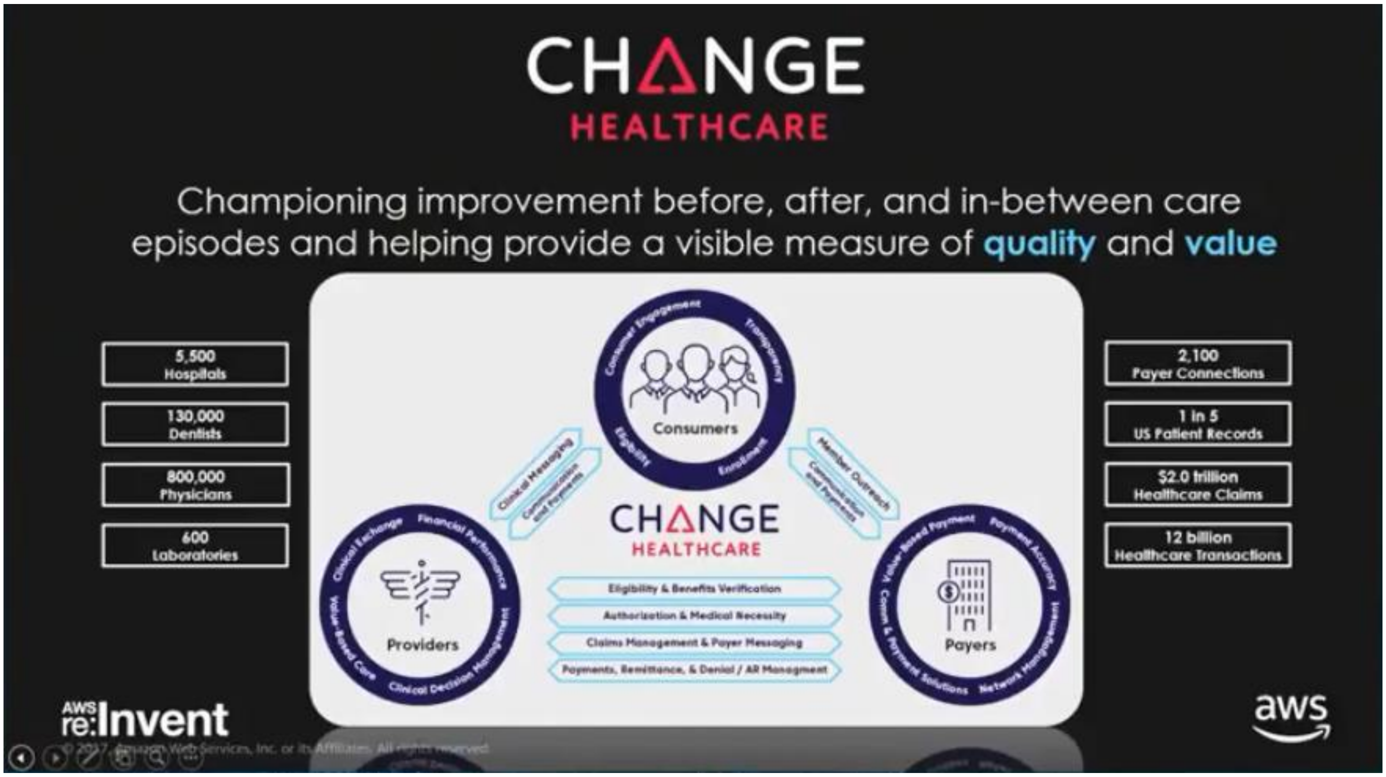
Torsten Kablitz
VP of IT, Cloud Engineering
Change Healthcare
tkablitz@changehealthcare.com

Akhtar Hossain
Enterprise Solutions Architect
Amazon Web Services
akhtarh@amazon.com

The Face
of Change
It takes a deep understanding of the evolving technology landscape to deliver future-focused solutions. That's what propels Torsten.
CHANGE

AWS re:Invent

aws



CHANGE
HEALTHCARE

Championing improvement before, after, and in-between care episodes and helping provide a visible measure of **quality** and **value**

5,500
Hospitals

130,000
Dentists

800,000
Physicians

600
Laboratories

2,100
Payer Connections

1 in 5
US Patient Records

$2.0 trillion
Healthcare Claims

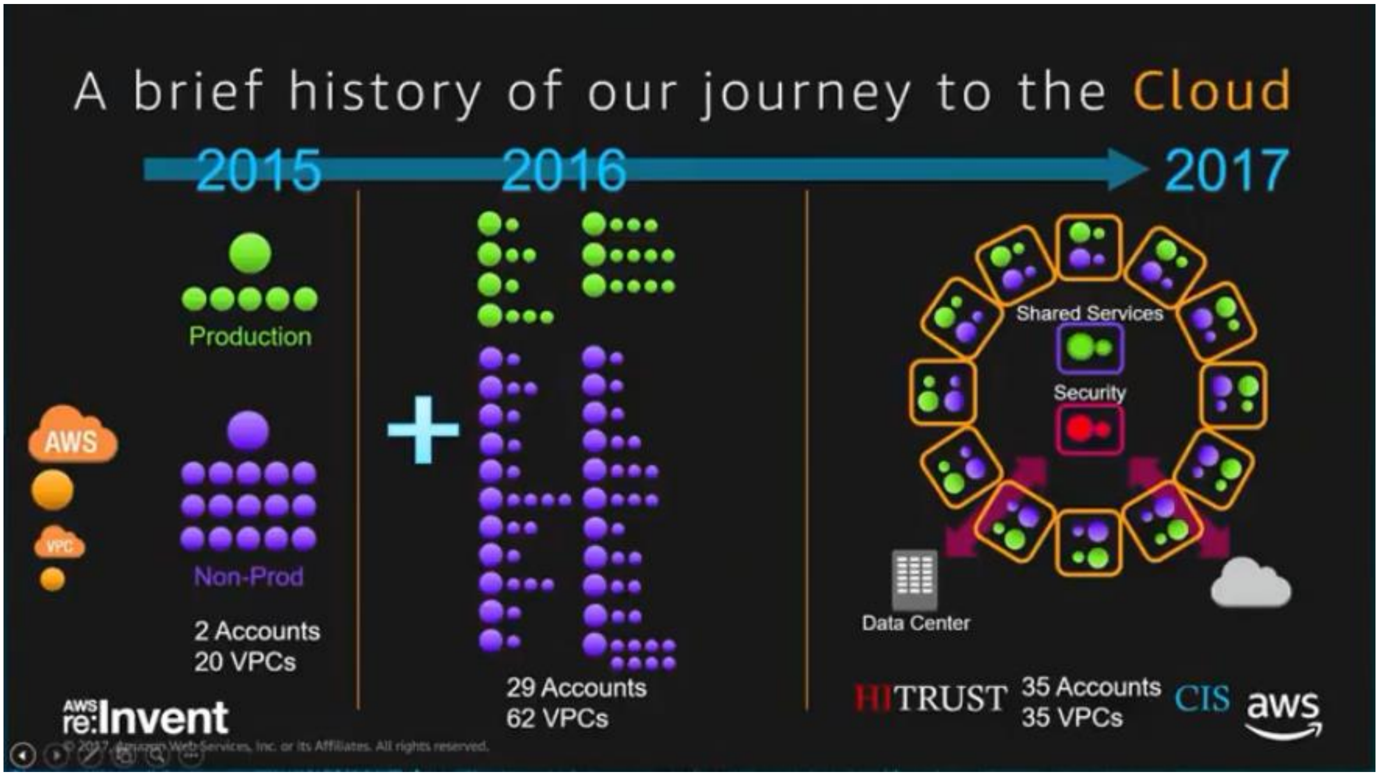
12 billion
Healthcare Transactions

Eligibility & Benefits Verification
Authorization & Medical Necessity
Claims Management & Payer Messaging
Payments, Remittance, & Denial / AR Managment

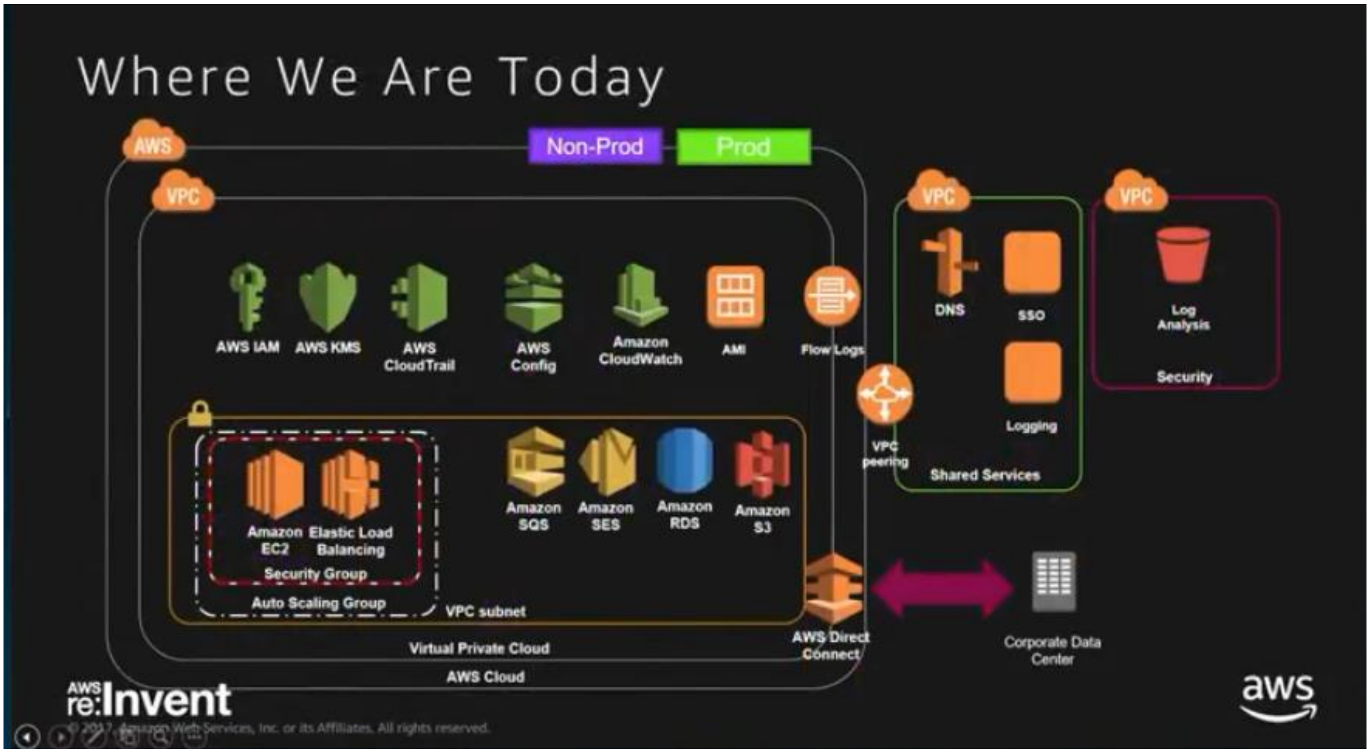
AWS re:Invent

aws

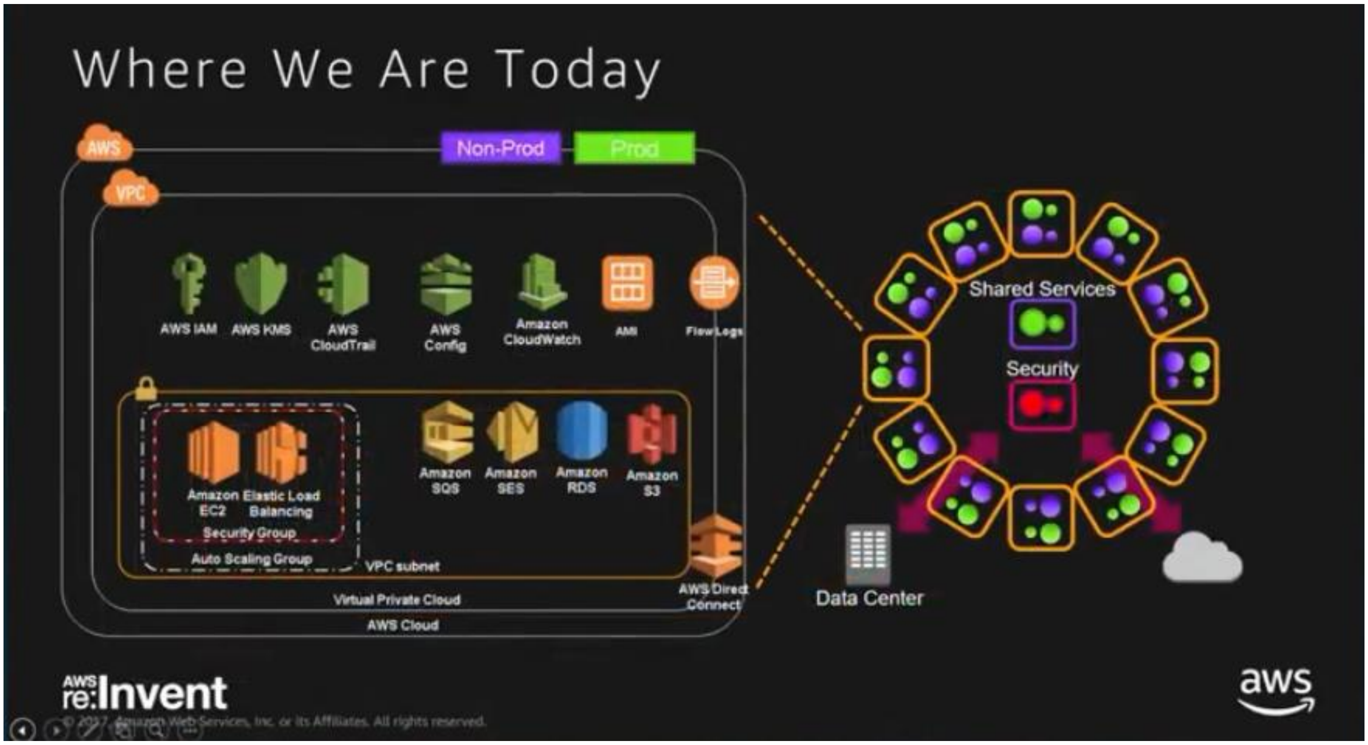
Change Healthcare is an Independent healthcare and IT services company



We finally went to an automated services approach where the teams own certain things and the cloud team own and managed some things through automation. Our major business units have accounts that they manage and deploy their applications into

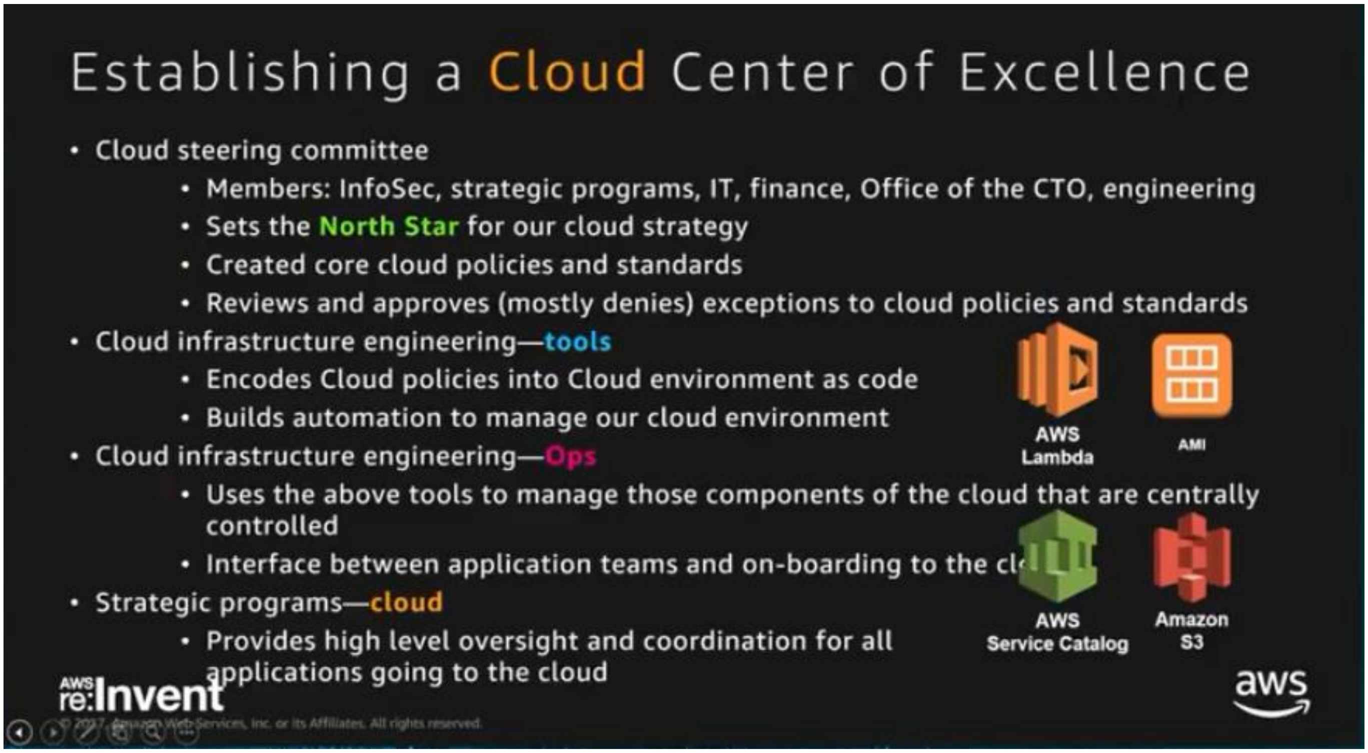


The same code creates all these environments automatically and for reproduceability. We create a new VPC and then a subnet for any new business units or just a subnet for a new business application if a VPC already exists



Where We Are Today

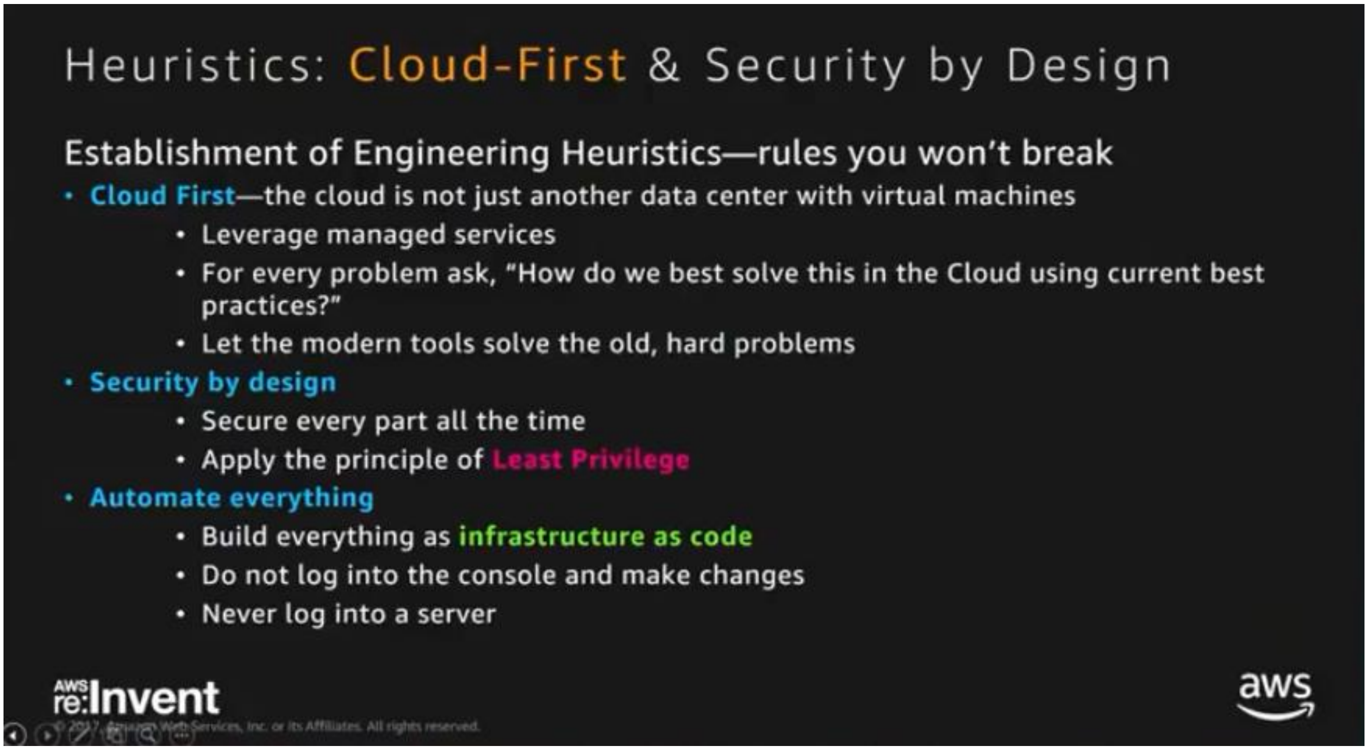
re:Invent



Establishing a Cloud Center of Excellence

- Cloud steering committee
  - Members: InfoSec, strategic programs, IT, finance, Office of the CTO, engineering
  - Sets the North Star for our cloud strategy
  - Created core cloud policies and standards
  - Reviews and approves (mostly denies) exceptions to cloud policies and standards
- Cloud infrastructure engineering—tools
  - Encodes Cloud policies into Cloud environment as code
  - Builds automation to manage our cloud environment
- Cloud infrastructure engineering—Ops
  - Uses the above tools to manage those components of the cloud that are centrally controlled
  - Interface between application teams and on-boarding to the cl
- Strategic programs—cloud
  - Provides high level oversight and coordination for all applications going to the cloud

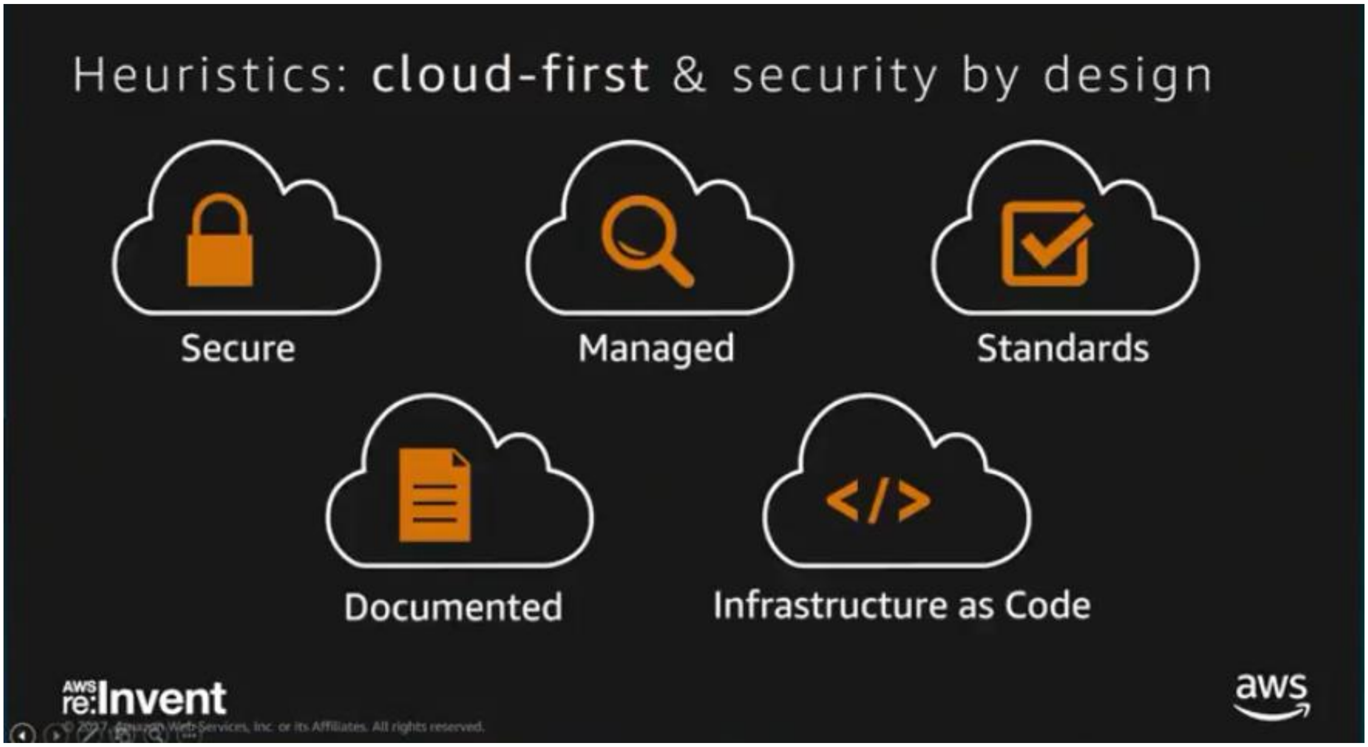
re:Invent



Heuristics: Cloud-First & Security by Design

Establishment of Engineering Heuristics—rules you won't break

- Cloud First—the cloud is not just another data center with virtual machines
  - Leverage managed services
  - For every problem ask, "How do we best solve this in the Cloud using current best practices?"
  - Let the modern tools solve the old, hard problems
- Security by design
  - Secure every part all the time
  - Apply the principle of Least Privilege
- Automate everything
  - Build everything as infrastructure as code
  - Do not log into the console and make changes
  - Never log into a server



Heuristics: cloud-first & security by design

Secure

Managed

Standards

Documented

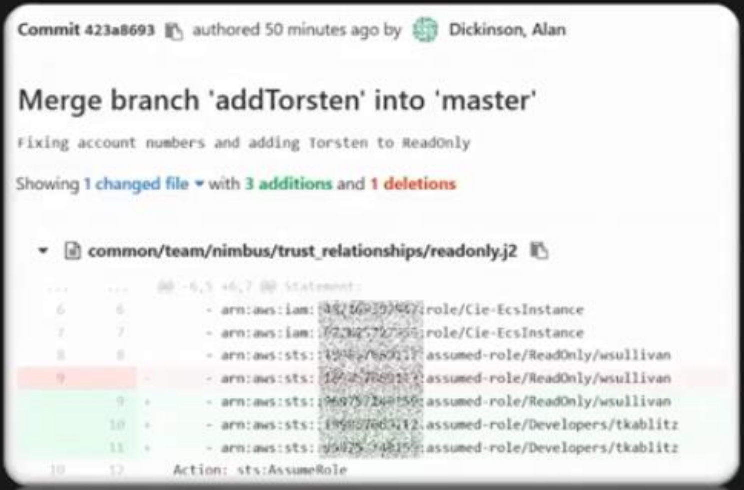
Infrastructure as Code

Automate Everything!

Automate Everything: Example

For this presentation, I asked my **Cloud Operations** team for access to our environment so I could see code and create screenshots.

I logged into **Git** and saw this…



Commit 423a8693 authored 50 minutes ago by Dickinson, Alan

Merge branch 'addTorsten' into 'master'

```
Fixing account numbers and adding Torsten to ReadOnly
```

Showing 1 changed file with 3 additions and 1 deletions

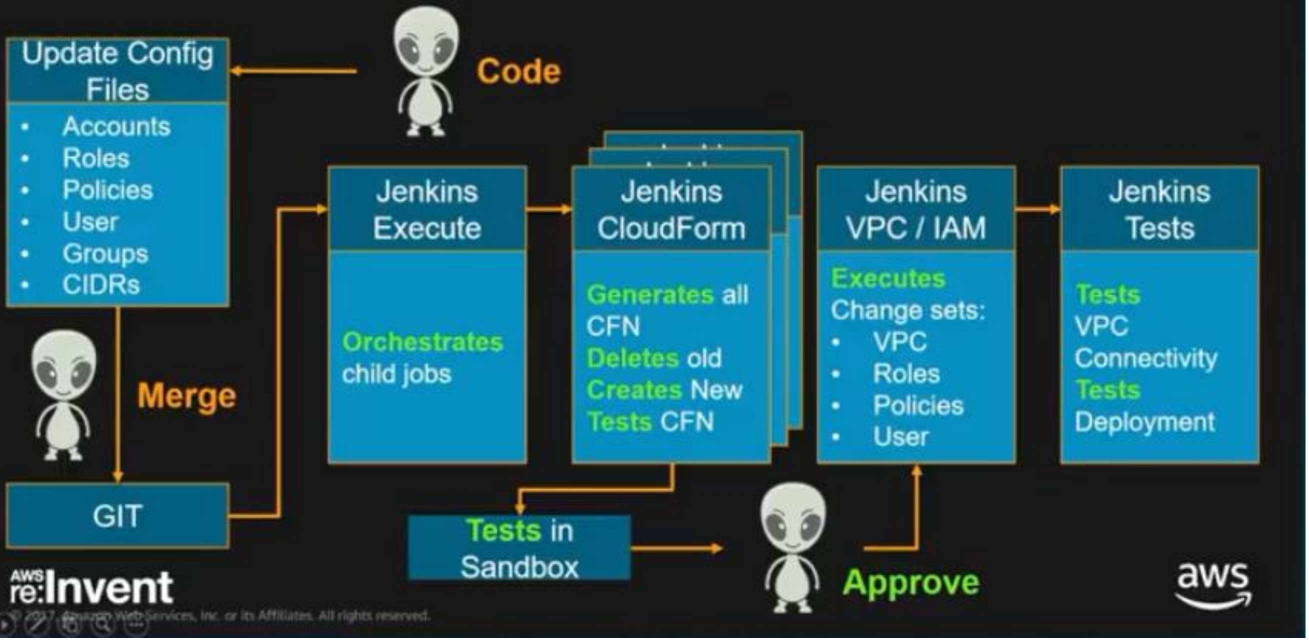
common/team/nimbus/trust_relationships/readonly.j2

```
@@ -6,5 +6,7 @@ Statement:
          - arn:aws:iam::[redacted]:role/Cie-EcsInstance
          - arn:aws:iam::[redacted]:role/Cie-EcsInstance
          - arn:aws:sts::[redacted]:assumed-role/ReadOnly/wsullivan
-         - arn:aws:sts::[redacted]:assumed-role/ReadOnly/wsullivan
+         - arn:aws:sts::[redacted]:assumed-role/ReadOnly/wsullivan
+         - arn:aws:sts::[redacted]:assumed-role/Developers/tkablitz
+         - arn:aws:sts::[redacted]:assumed-role/Developers/tkablitz
        Action: sts:AssumeRole
```

AWS re:Invent

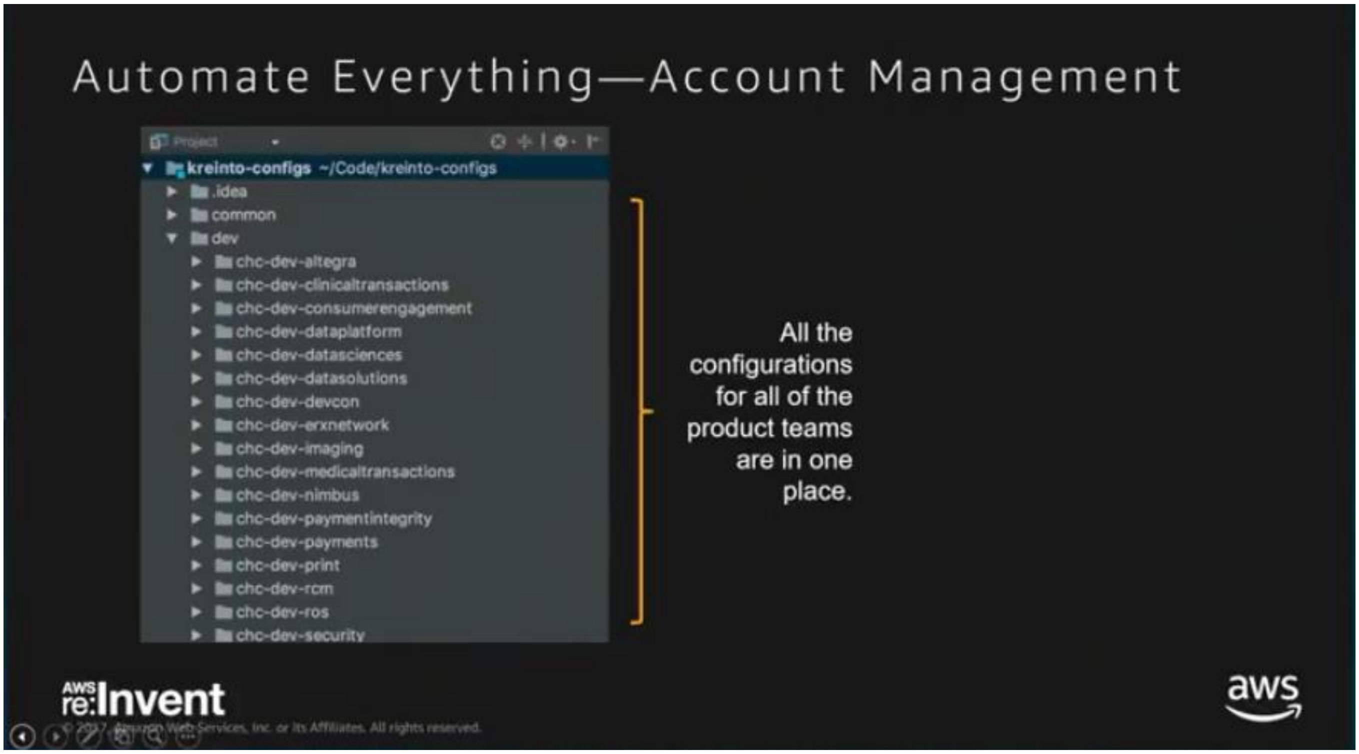
aws

Automate Everything—Account Management



AWS re:Invent

aws

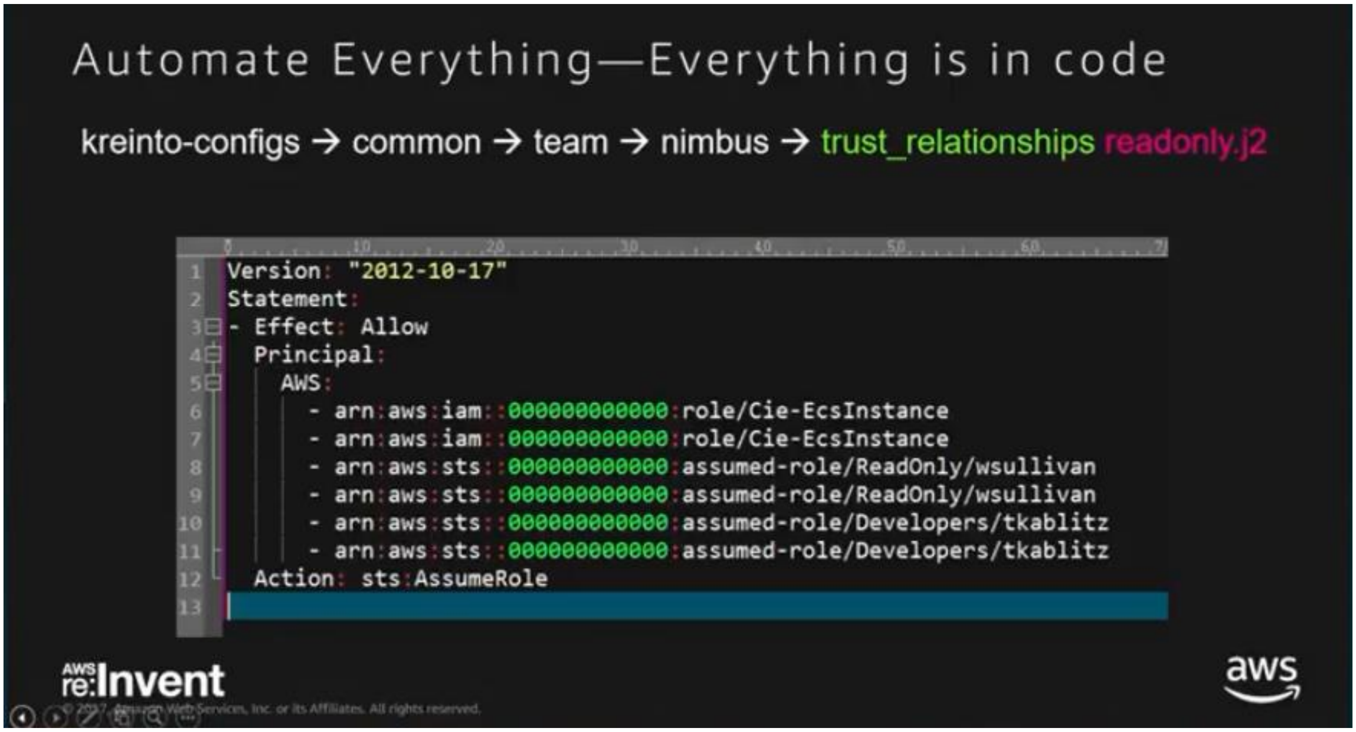


Automate Everything—Account Management

```
▼ kreinto-configs ~/Code/kreinto-configs
  ► .idea
  ► common
  ▼ dev
    ► chc-dev-altegra
    ► chc-dev-clinicaltransactions
    ► chc-dev-consumerengagement
    ► chc-dev-dataplatform
    ► chc-dev-datasciences
    ► chc-dev-datasolutions
    ► chc-dev-devcon
    ► chc-dev-erxnetwork
    ► chc-dev-imaging
    ► chc-dev-medicaltransactions
    ► chc-dev-nimbus
    ► chc-dev-paymentintegrity
    ► chc-dev-payments
    ► chc-dev-print
    ► chc-dev-rcm
    ► chc-dev-ros
    ► chc-dev-security
```

All the configurations for all of the product teams are in one place.

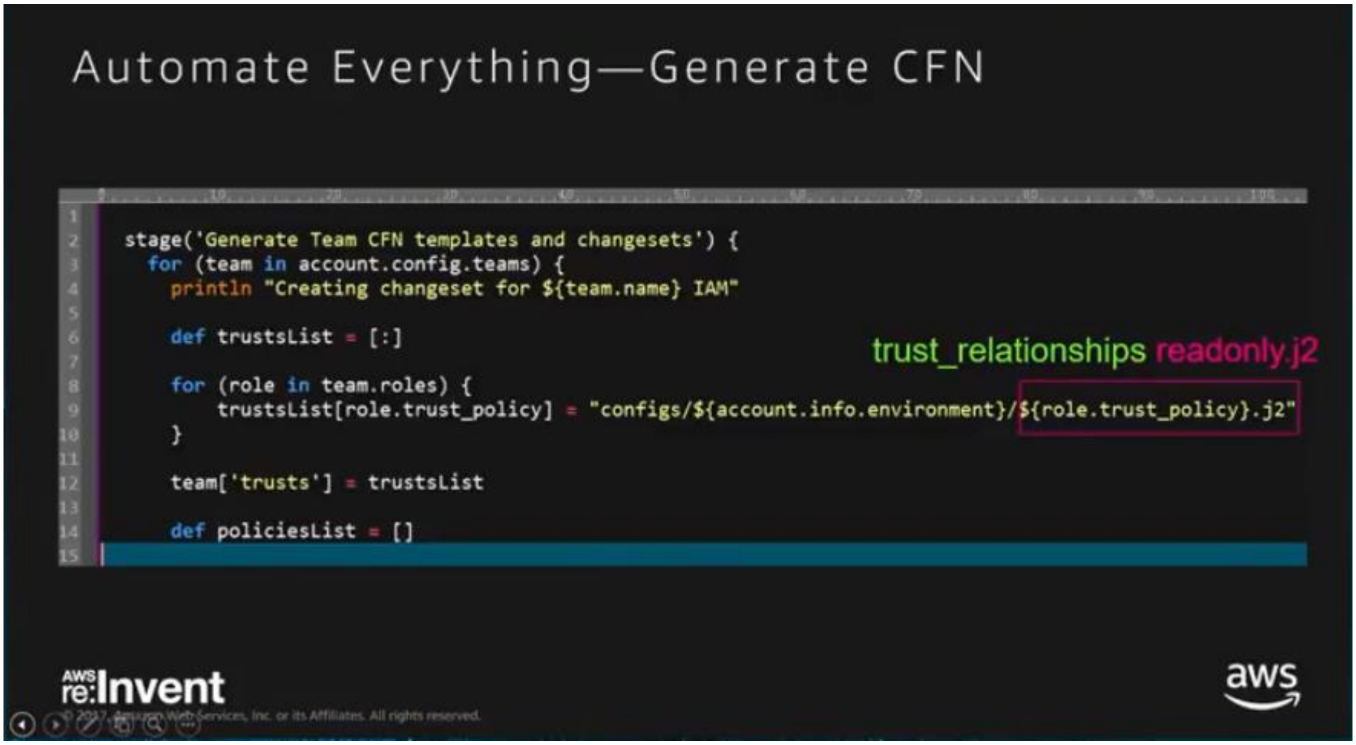
All of our environments are in code, with all our configurations



Automate Everything—Everything is in code

kreinto-configs → common → team → nimbus → trust_relationships readonly.j2

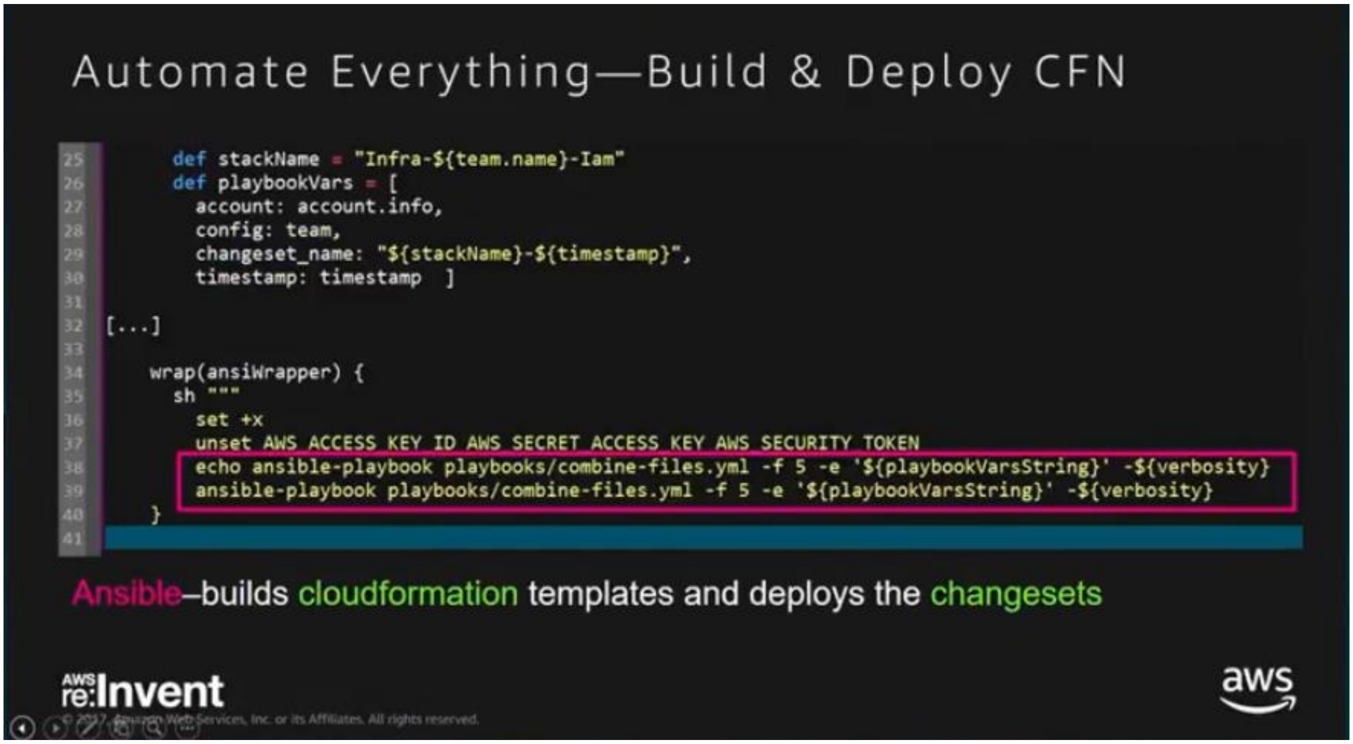
```
Version: "2012-10-17"
Statement:
- Effect: Allow
  Principal:
    AWS:
      - arn:aws:iam::000000000000:role/Cie-EcsInstance
      - arn:aws:iam::000000000000:role/Cie-EcsInstance
      - arn:aws:sts::000000000000:assumed-role/ReadOnly/wsullivan
      - arn:aws:sts::000000000000:assumed-role/ReadOnly/wsullivan
      - arn:aws:sts::000000000000:assumed-role/Developers/tkablitz
      - arn:aws:sts::000000000000:assumed-role/Developers/tkablitz
  Action: sts:AssumeRole
```



Automate Everything—Generate CFN

```

stage('Generate Team CFN templates and changesets') {
  for (team in account.config.teams) {
    println "Creating changeset for ${team.name} IAM"

    def trustsList = [:]

    for (role in team.roles) {
        trustsList[role.trust_policy] = "configs/${account.info.environment}/${role.trust_policy}.j2"
    }

    team['trusts'] = trustsList

    def policiesList = []

```

trust_relationships readonly.j2



Automate Everything—Build & Deploy CFN

```
        def stackName = "Infra-${team.name}-Iam"
        def playbookVars = [
          account: account.info,
          config: team,
          changeset_name: "${stackName}-${timestamp}",
          timestamp: timestamp  ]

[...]

    wrap(ansiWrapper) {
      sh """
        set +x
        unset AWS_ACCESS_KEY_ID AWS_SECRET_ACCESS_KEY AWS_SECURITY_TOKEN
        echo ansible-playbook playbooks/combine-files.yml -f 5 -e '${playbookVarsString}' -${verbosity}
        ansible-playbook playbooks/combine-files.yml -f 5 -e '${playbookVarsString}' -${verbosity}
    }

```

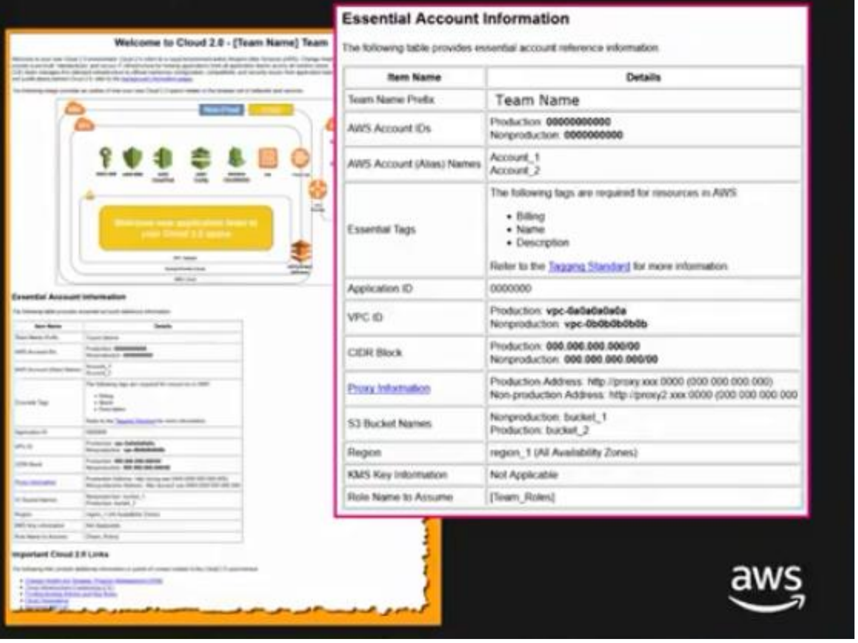
Ansible–builds cloudformation templates and deploys the changesets

Welcome to Cloud 2.0

At the end of the build process, the job generates a **welcome** page and drops it into the Amazon S3 bucket for that team.

All the application team's AWS resources are **tagged** with:

- Application ID
- Billing
- Name
- Description
- Environment



Welcome to Cloud 2.0 - [Team Name] Team

Essential Account Information

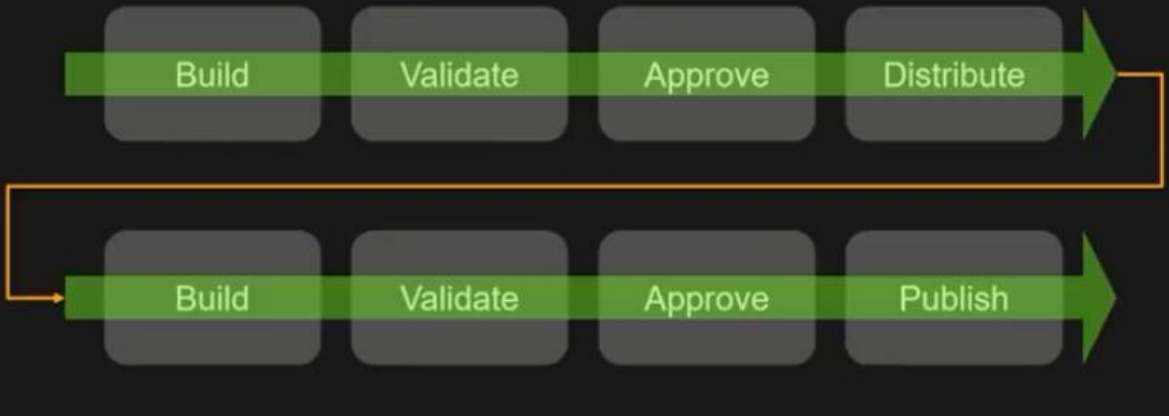
The following table provides essential account reference information

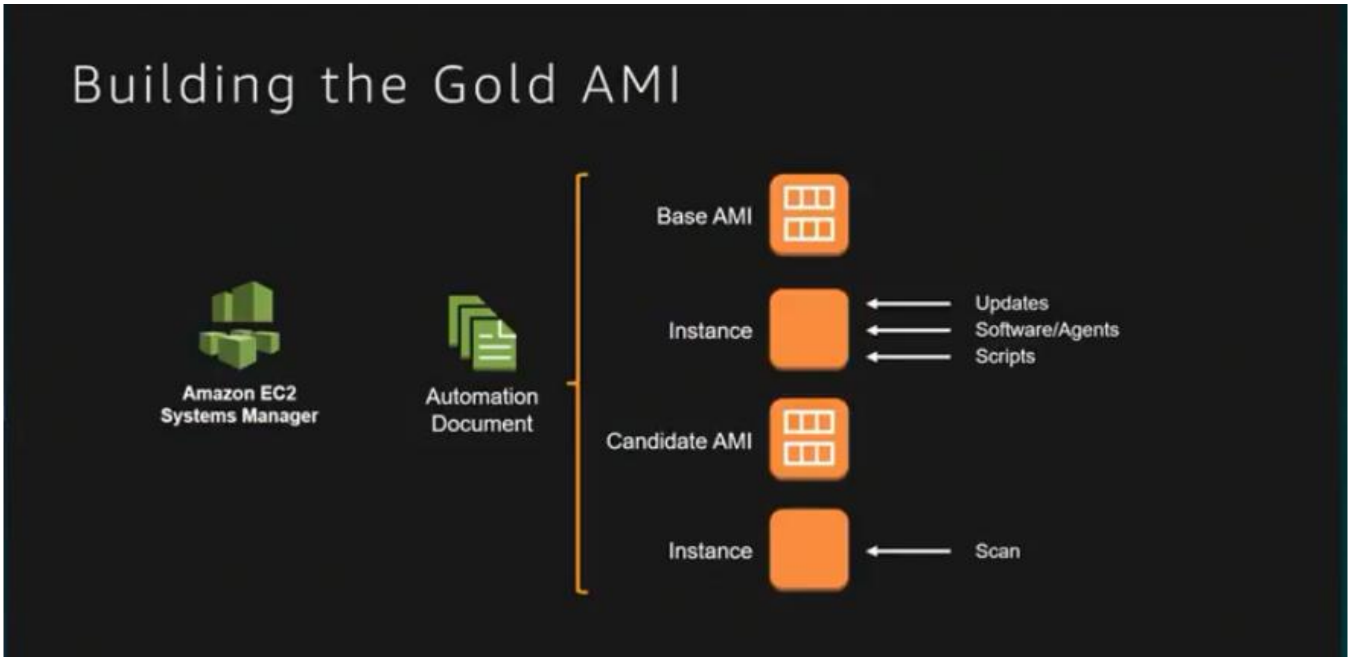
Item Name	Details
Team Name Prefix	Team Name
AWS Account IDs	Production: **00000000000** Nonproduction: **000000000**
AWS Account (Alias) Names	Account_1 Account_2
Essential Tags	The following tags are required for resources in AWS • Billing • Name • Description Refer to the Tagging Standard for more information
Application ID	0000000
VPC ID	Production: **vpc-0a0a0a0a0a** Nonproduction: **vpc-0b0b0b0b0b**
CIDR Block	Production: **000.000.000.000/00** Nonproduction: **000.000.000.000/00**
Proxy Information	Production Address: http://proxy.xxx:0000 (000.000.000.000) Non-production Address: http://proxy2.xxx:0000 (000.000.000.000
S3 Bucket Names	Nonproduction: bucket_1 Production: bucket_2
Region	region_1 (All Availability Zones)
KMS Key Information	Not Applicable
Role Name to Assume	[Team_Roles]

Important Cloud 2.0 Links

Automate Everything—AMI Gold Image

Gold Base AMI—Overall Process





Building the Gold AMI

Amazon EC2 Systems Manager

Automation Document

Base AMI

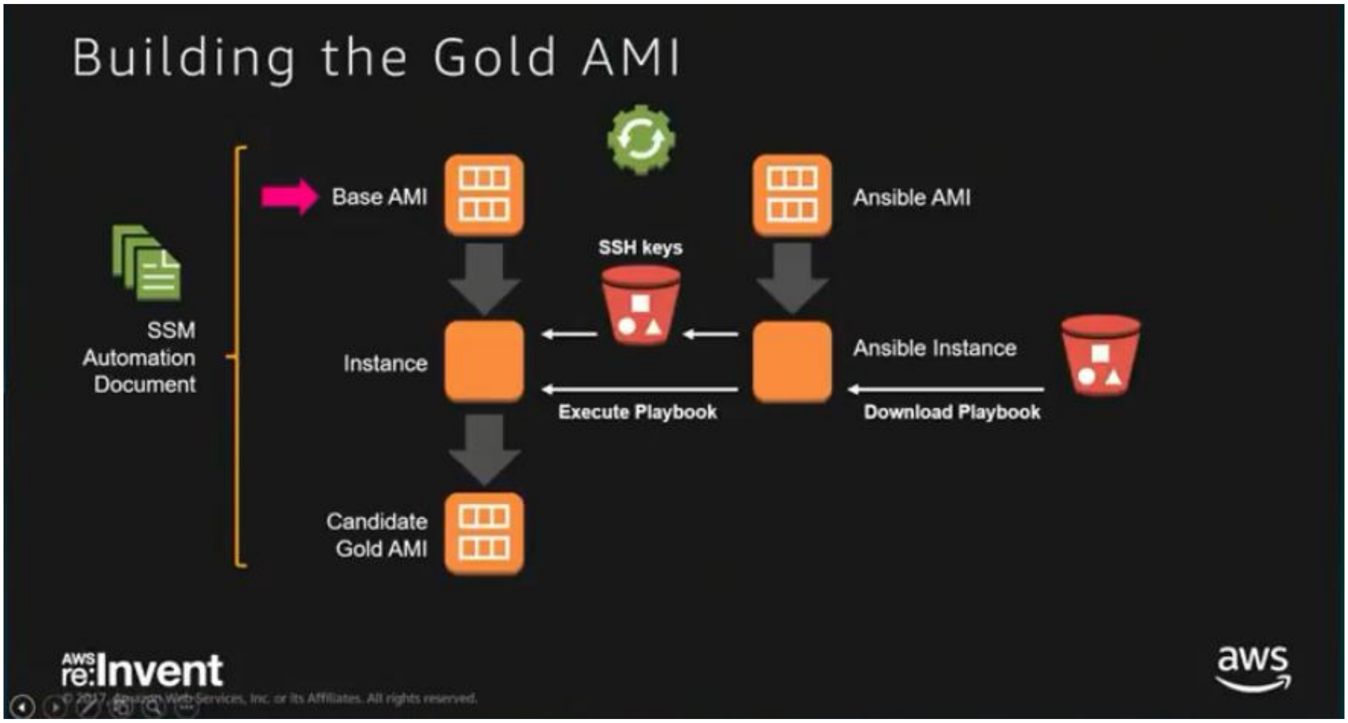
Instance

Updates
Software/Agents
Scripts

Candidate AMI

Instance

Scan



Building the Gold AMI

SSM Automation Document

Base AMI

Instance

Candidate Gold AMI

SSH keys

Ansible AMI

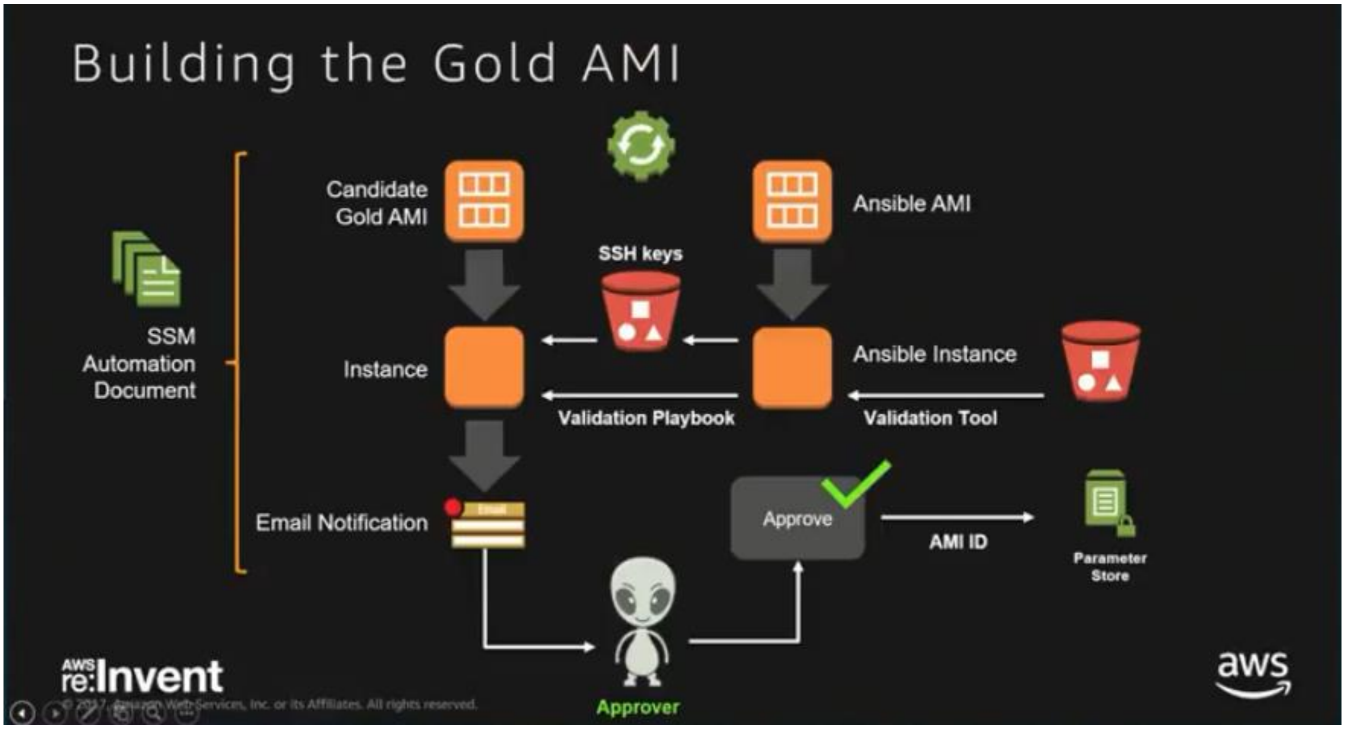
Ansible Instance

Execute Playbook

Download Playbook

AWS re:Invent

aws



Building the Gold AMI

SSM Automation Document

Candidate Gold AMI

Instance

Email Notification

SSH keys

Validation Playbook

Ansible AMI

Ansible Instance

Validation Tool

Approve

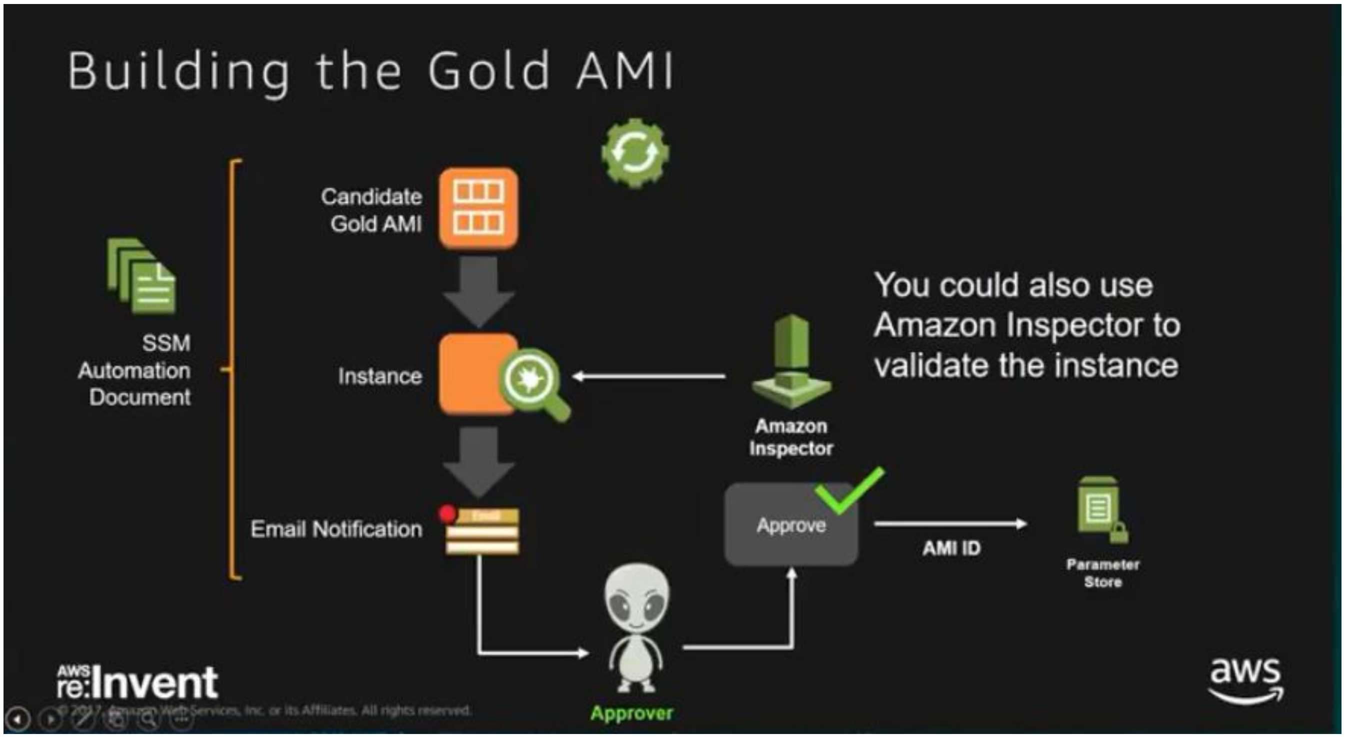
AMI ID

Parameter Store

Approver

AWS re:Invent

aws



Building the Gold AMI

SSM Automation Document

Candidate Gold AMI

Instance

Email Notification

Amazon Inspector

You could also use Amazon Inspector to validate the instance

Approve

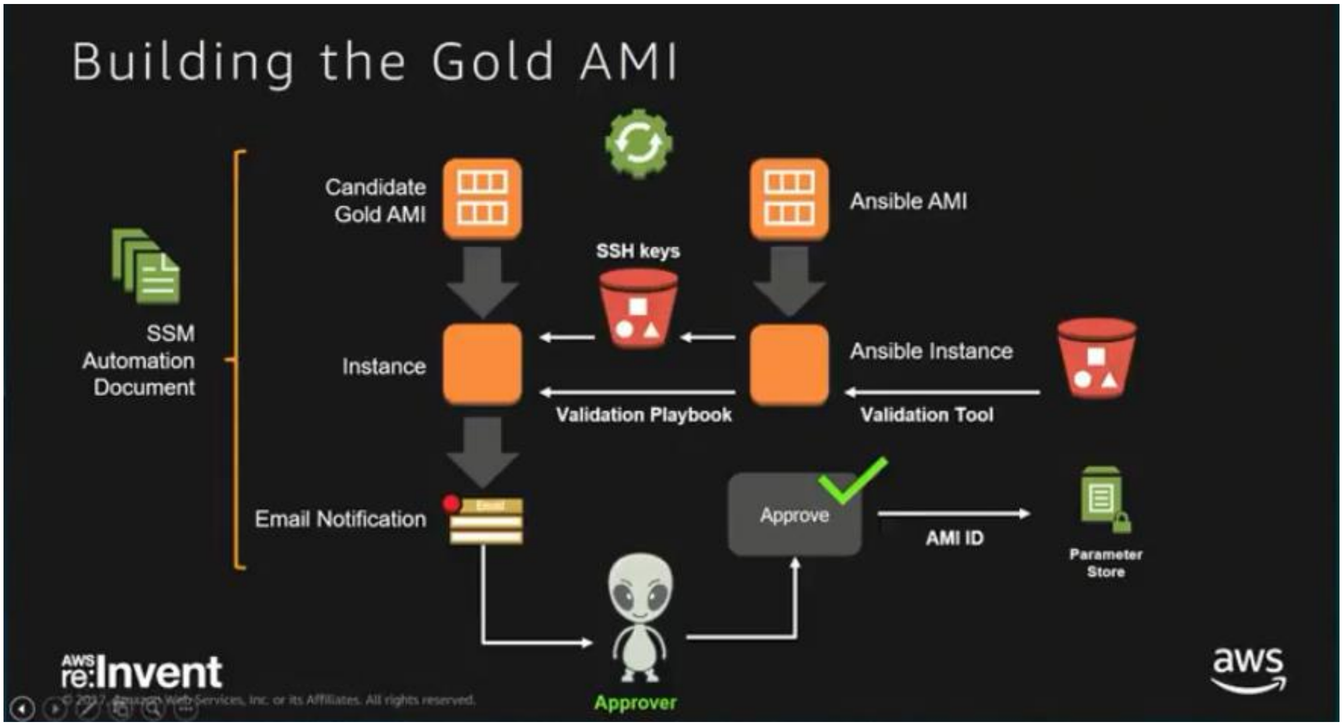
AMI ID

Parameter Store

Approver

AWS re:Invent

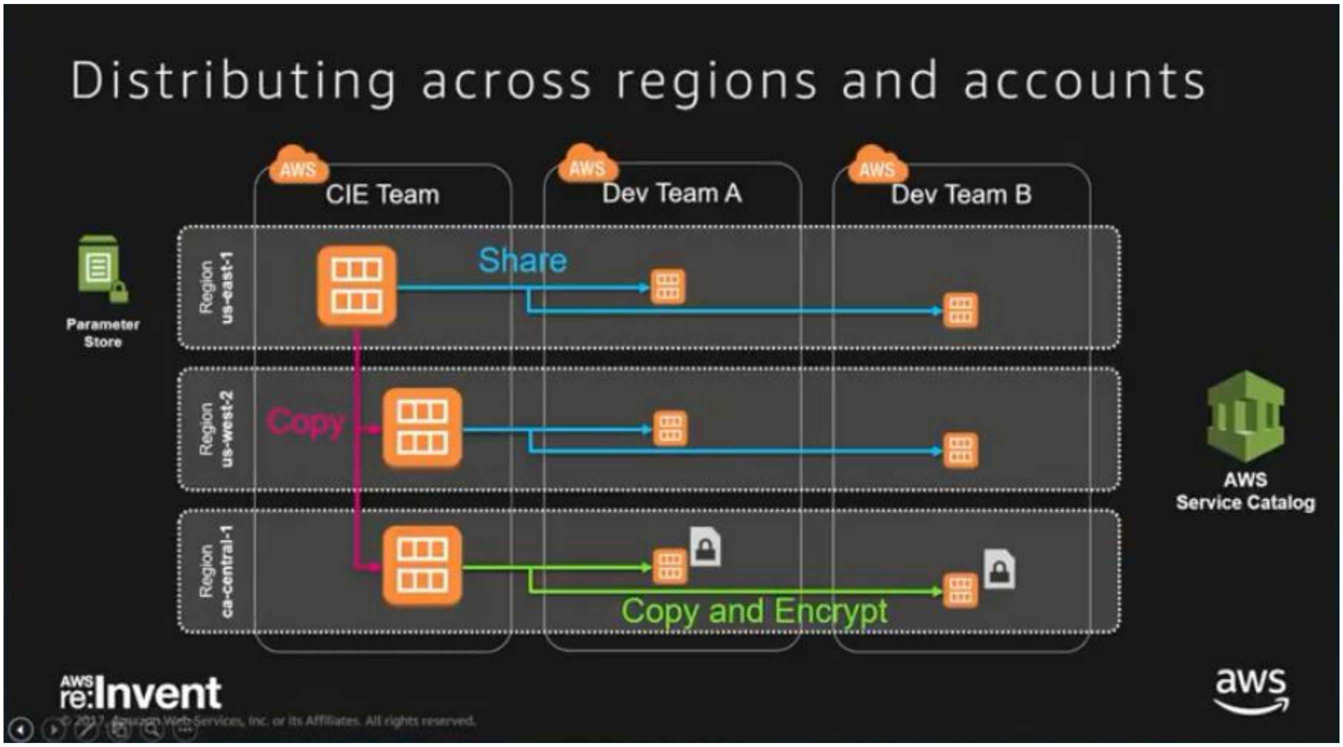
aws



Building the Gold AMI

AWS re:Invent

aws



Distributing across regions and accounts

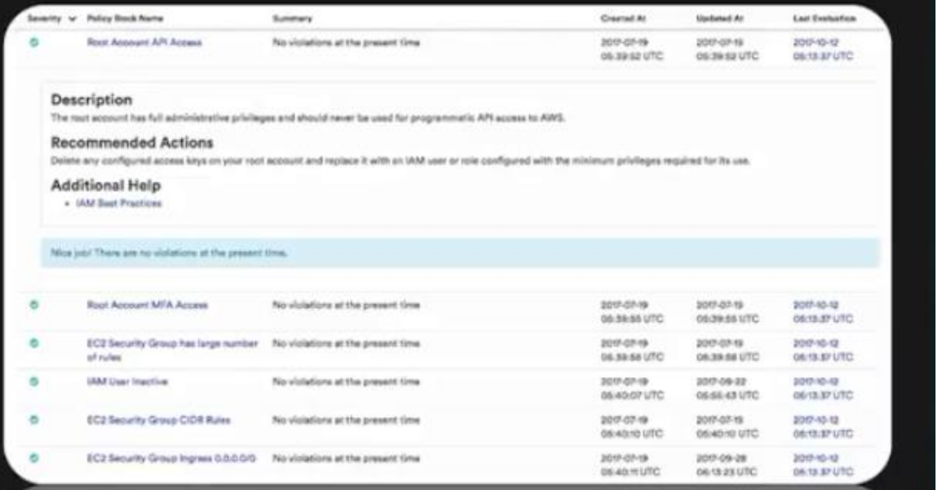
AWS re:Invent

aws

Scanning

Scanning for Compliance

We use **CloudHealth** to scan for **CIS** security benchmarks and **AWS** best practices compliance



Severity	Policy Block Name	Summary	Created At	Updated At	Last Evaluation
●	Root Account API Access	No violations at the present time	2017-07-19 05:39:52 UTC	2017-07-19 05:39:52 UTC	2017-10-12 06:13:37 UTC

Description

The root account has full administrative privileges and should never be used for programmatic API access to AWS.

Recommended Actions

Delete any configured access keys on your root account and replace it with an IAM user or role configured with the minimum privileges required for its use.

Additional Help

- IAM Best Practices

Nice job! There are no violations at the present time.

Severity	Policy Block Name	Summary	Created At	Updated At	Last Evaluation
●	Root Account MFA Access	No violations at the present time	2017-07-19 05:39:55 UTC	2017-07-19 05:39:55 UTC	2017-10-12 06:13:37 UTC
●	EC2 Security Group has large number of rules	No violations at the present time	2017-07-19 05:39:58 UTC	2017-07-19 05:39:58 UTC	2017-10-12 06:13:37 UTC
●	IAM User Inactive	No violations at the present time	2017-07-19 05:40:07 UTC	2017-09-22 05:55:43 UTC	2017-10-12 06:13:37 UTC
●	EC2 Security Group CIDR Rules	No violations at the present time	2017-07-19 05:40:10 UTC	2017-07-19 05:40:10 UTC	2017-10-12 06:13:37 UTC
●	EC2 Security Group Ingress 0.0.0.0/0	No violations at the present time	2017-07-19 05:40:11 UTC	2017-09-28 06:13:23 UTC	2017-10-12 06:13:37 UTC

AWS re:Invent

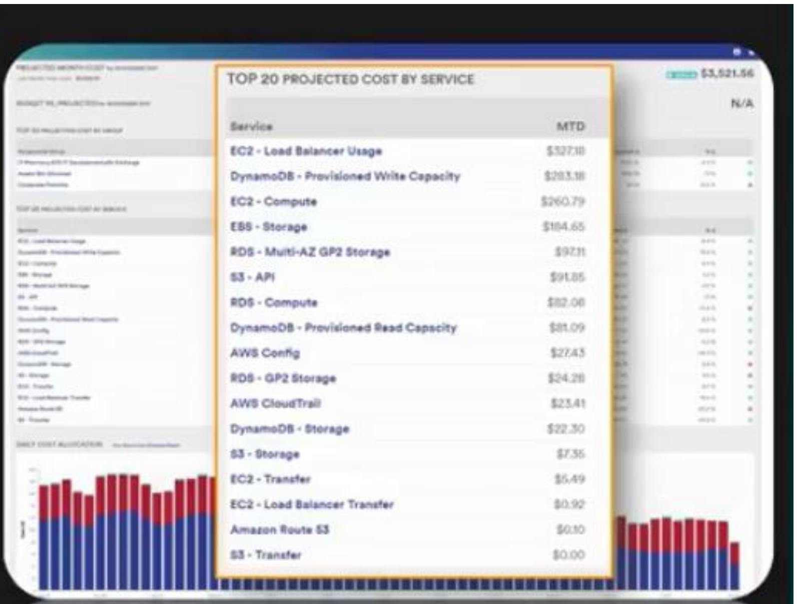
aws

Cost Management

Cost management

AWS provides very detailed billing information, but this can be difficult to organize at times.

The **billing tag** allows us to filter on what team owns and are using what services on AWS.

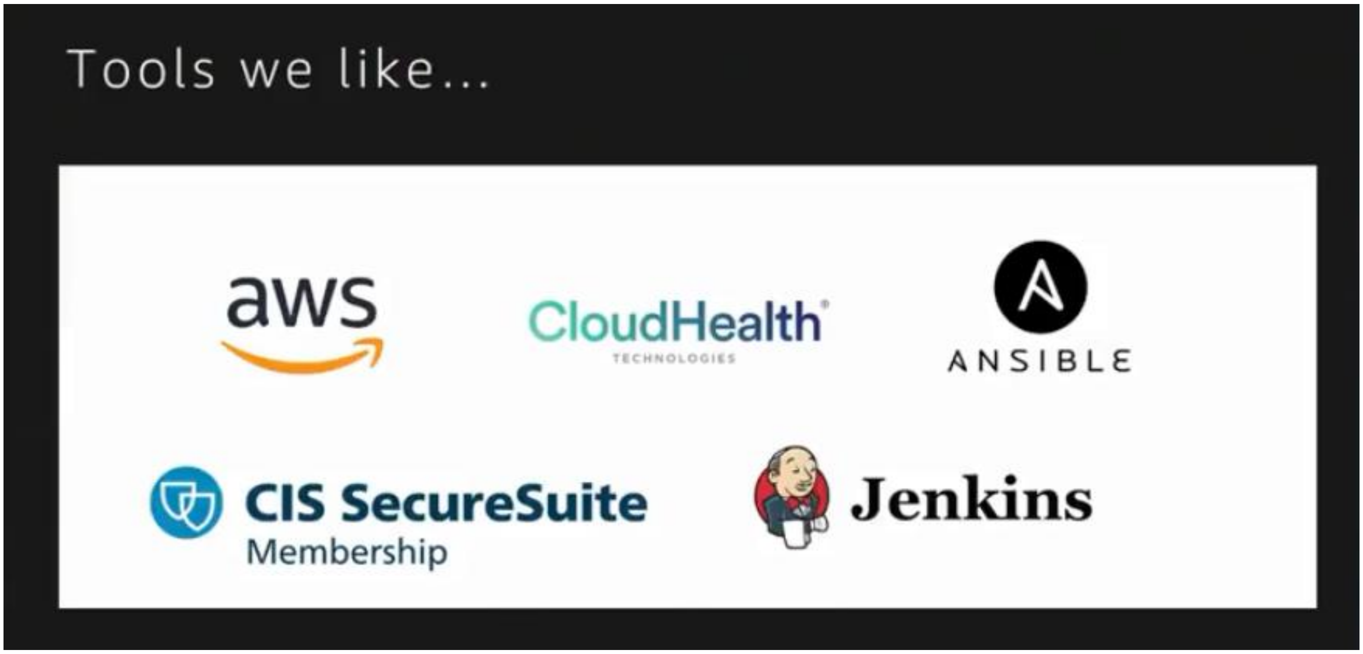


TOP 20 PROJECTED COST BY SERVICE

Service	MTD
EC2 - Load Balancer Usage	$327.18
DynamoDB - Provisioned Write Capacity	$283.18
EC2 - Compute	$260.79
EBS - Storage	$184.65
RDS - Multi-AZ GP2 Storage	$97.11
S3 - API	$91.85
RDS - Compute	$82.06
DynamoDB - Provisioned Read Capacity	$81.09
AWS Config	$27.43
RDS - GP2 Storage	$24.28
AWS CloudTrail	$23.41
DynamoDB - Storage	$22.30
S3 - Storage	$7.35
EC2 - Transfer	$5.49
EC2 - Load Balancer Transfer	$0.92
Amazon Route 53	$0.10
S3 - Transfer	$0.00

AWS re:Invent

aws



Tools we like...

aws

CloudHealth TECHNOLOGIES

ANSIBLE

CIS SecureSuite Membership

Jenkins

Torsten Kablitz
VP of IT, Cloud Engineering
Change Healthcare
tkablitz@changehealthcare.com

Akhtar Hossain
Enterprise Solutions Architect
Amazon Web Services
akhtarh@amazon.com