#10

Date:
Monday Nov 18th, 2019

Topics

- Course Content Overview
- Cloud Security – Shared Responsibility Model
- VPC
- Hands-on Lab#17: VPC
- Encryption
- IAM
- Hands-on Lab#18: IAM
- Infrastructure-as-a-code
- CloudTrail
- CloudWatch
- IaC
- Hands-on Lab#19: KMS
- Next Steps

Course Content Overview

Course Content Overview

Week	Date	Topics, Readings, Assignments, Deadlines	Due Date
1	6:00 PM PT, 8/26	Course Logistics & projects Introduction to Cloud Technologies	
2	6:00 PM PT, 09/02	Labor Day Campus Closed – No Lecture	
3	6:00 PM PT, 09/09	Fundamentals	Homework #1 Due
4	6:00 PM PT, 09/16	Storage Content Delivery Network	Team Formation Due
5	6:00 PM PT, 09/23	Compute, Serverless	
6	6:00 PM PT, 09/30	Databases, Migrating Data to Cloud	Homework #2 Due
7	6:00 PM PT, 10/07	Big Data, Data Streaming	Quiz #1 Due
8	6:00 PM PT, 10/14	MIDTERM EXAM (Close book, Close notes). Bring student ID	
9	6:00 PM PT, 10/21	Artificial Intelligence I	Project #1 Due:
10	6:00 PM PT, 10/28	Artificial Intelligence II	
11	6:00 PM PT, 11/04	Internet of Things (IoT)	Project #2: Design Due
12	6:00 PM PT, 11/11	Veterans Day Campus Closed – No Lecture	
13	6:00 PM PT, 11/18	Cloud Security	Project #2: Component I Due
14	6:00 PM PT, 11/25	Cloud Management	Project #2: Component II Due
15	6:00 PM PT, 12/02	Project presentation & discussion I	
16	6:00 PM PT, 12/09	Project presentation & discussion II	Quiz #2 Due
17	6:00 PM PT, 12/16	FINAL EXAM Thu, Dec 16 (close book, close notes). Bring student ID	

We are here!

The security paradigm shifted

“Imagine the ability to create or destroy an entire datacenter with just the proper credentials, or a short script.”

Legacy Datacenters

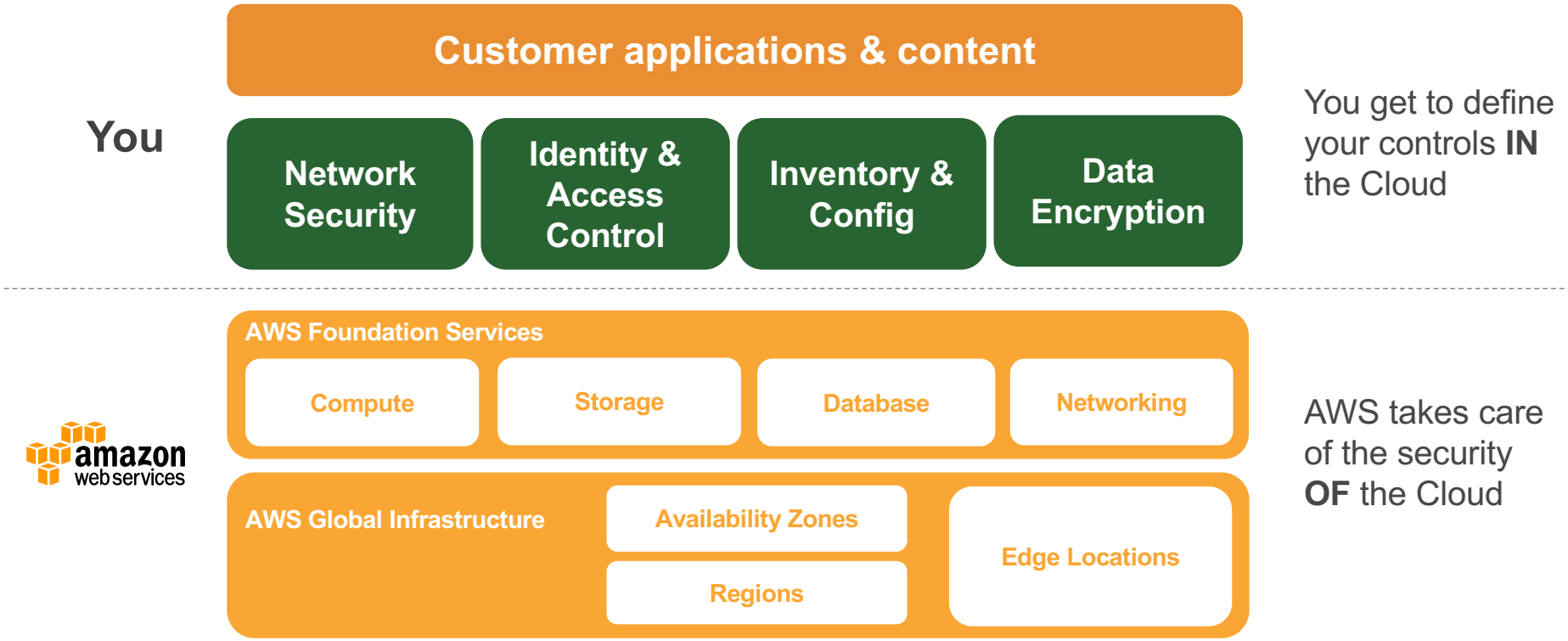
- Big Perimeter
- End-to-End Ownership
- Build it all yourself
- Server-centric approach
- Self-managed Services
- Static Architecture
- De-centralized Administration

Cloud

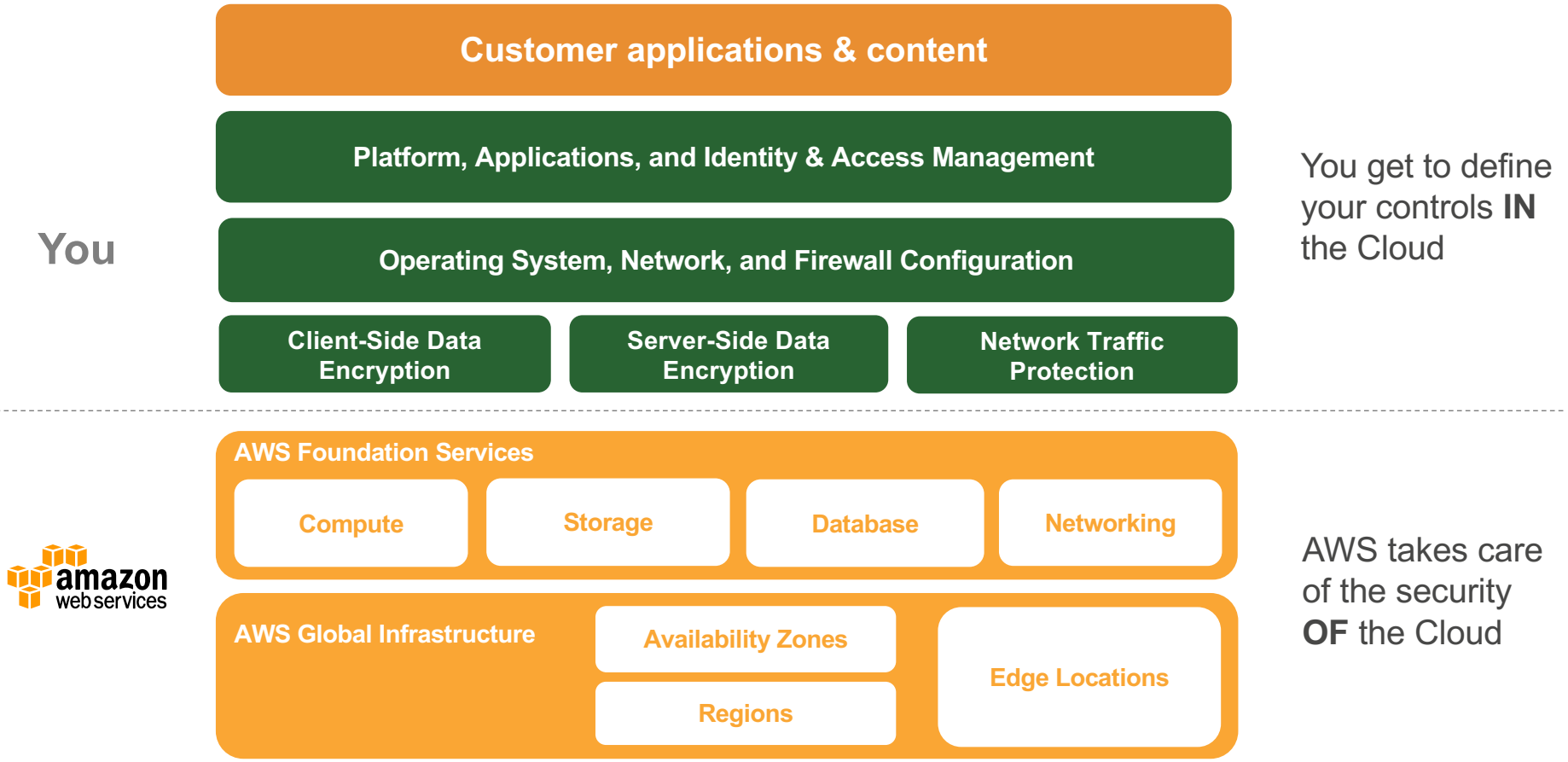
- Micro-Perimeters
- Own just enough
- Focus on your core value
- Service-Centric
- Platform Services
- Continuously Evolving
- Central Control Plane (API)

Shared Responsibility Model

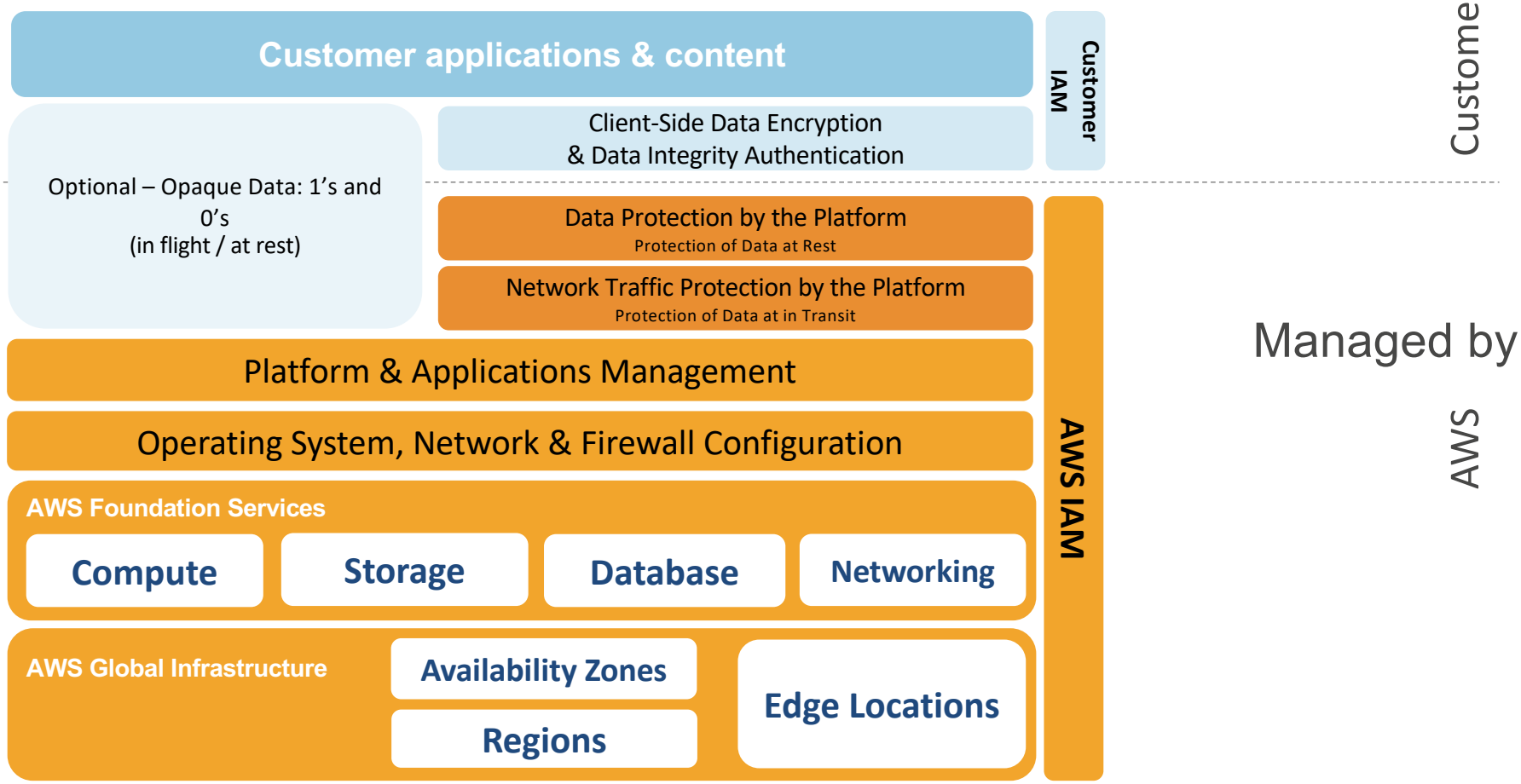
AWS and you share responsibility for security



AWS and you share responsibility for security



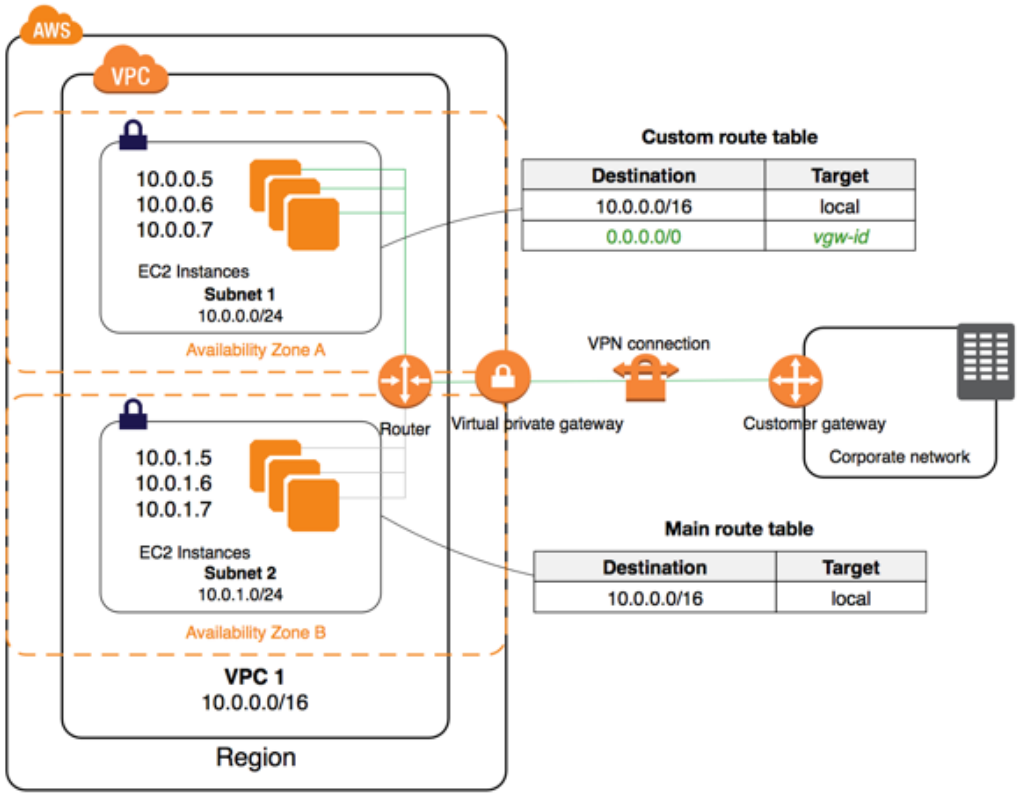
Shared Security Model: E.g. S3 Services



You are in control of privacy

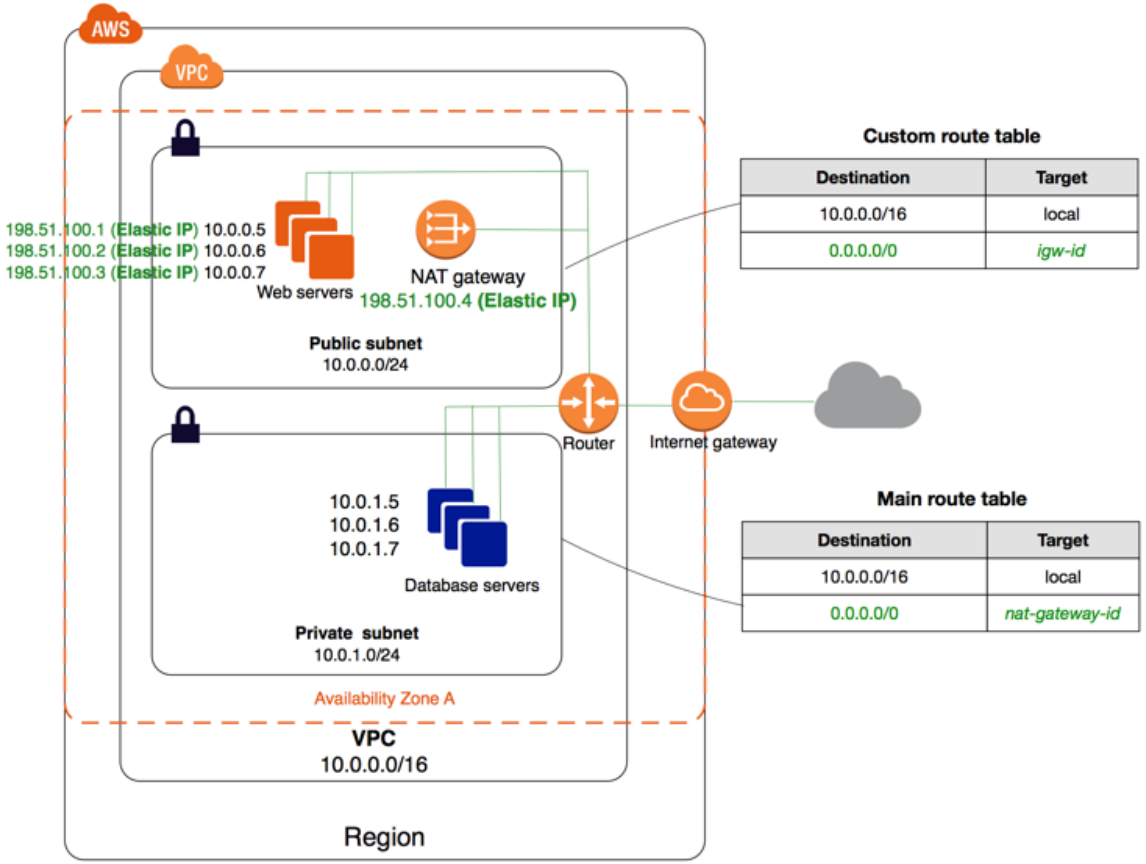
- Customers retain full ownership and control of your content
- Control format, accuracy, and encryption any way that you choose.
- Control who can access content.
- Control content lifecycle and disposal.

Amazon Virtual Private Cloud (Amazon VPC)



- Amazon VPC lets you provision a logically isolated section of the Amazon Web Services (AWS) cloud where you can launch AWS resources in a virtual network that you define.
- You have complete control over your virtual networking environment, including selection of your own IP address ranges, creation of subnets, and configuration of route tables and network gateways.

Amazon Virtual Private Cloud (Amazon VPC)

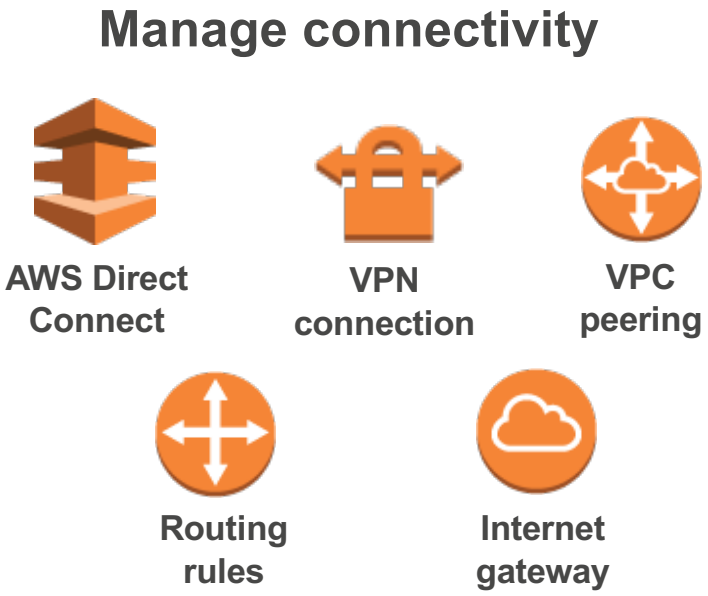


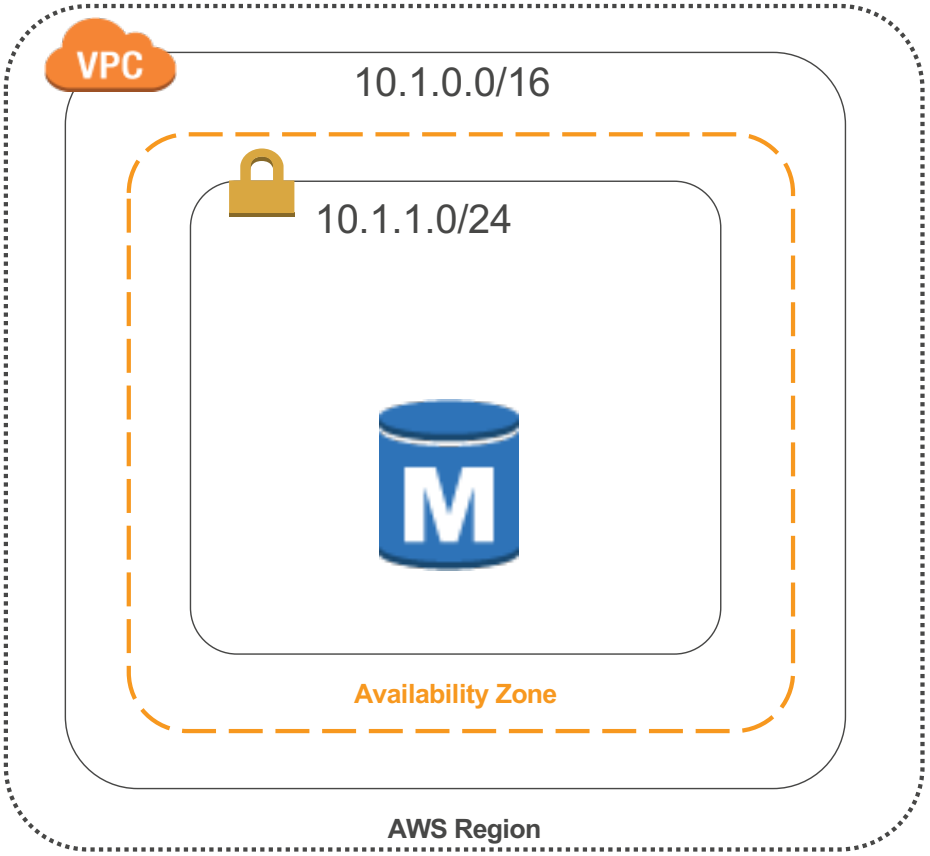
Amazon Virtual Private Cloud (Amazon VPC)

- **A Virtual Private Cloud (VPC)**: A logically isolated virtual network in the AWS cloud. You define a VPC's IP address space from ranges you select.
- **Subnet**: A segment of a VPC's IP address range where you can place groups of isolated resources.
- **Internet Gateway**: The Amazon VPC side of a connection to the public Internet.
- **NAT Gateway**: A highly available, managed Network Address Translation (NAT) service for your resources in a private subnet to access the Internet.
- **Router**: Routers interconnect subnets and direct traffic between Internet gateways, virtual private gateways, NAT gateways, and subnets.
- **Hardware VPN Connection**: A hardware-based VPN connection between your Amazon VPC and your datacenter, home network, or co-location facility.
- **Virtual Private Gateway**: The Amazon VPC side of a VPN connection.
- **Customer Gateway**: Your side of a VPN connection.
- **Peering Connection**: A peering connection enables you to route traffic via private IP addresses between two peered VPCs.

Amazon Virtual Private Cloud (Amazon VPC)

- Securely control network configuration

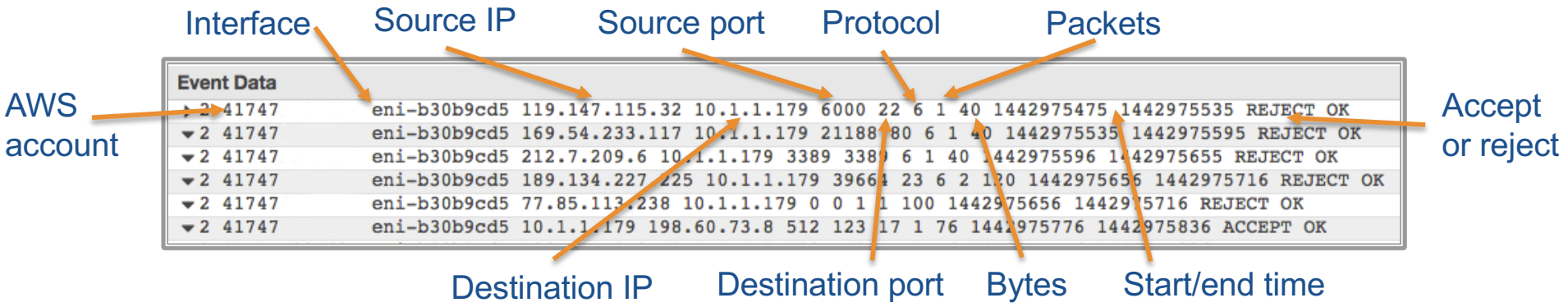




VPC Flow Logs

VPC Flow Logs – Monitor your traffic

- Agentless
- Enable per ENI, per subnet, or per VPC
- Logged to AWS CloudWatch Logs
- Create CloudWatch metrics from log data
- Alarm on those metrics



Hands-on Lab #17: VPC

Hands-on Lab#17: VPC

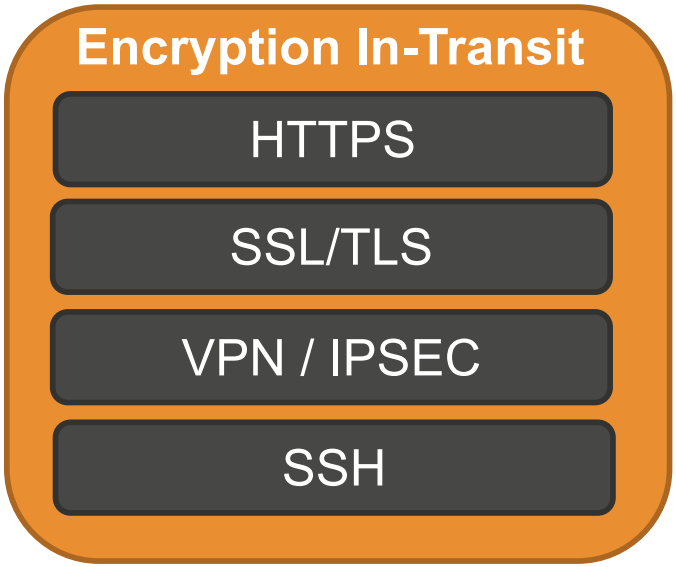
- The lab will cover following topics:
  - Create an Amazon VPC using the VPC Wizard
  - Explore the basic components of VPC
    - Public and private subnets
    - Route tables and routes
    - NAT Gateways
    - Network ACLs
    - Elastic IPs

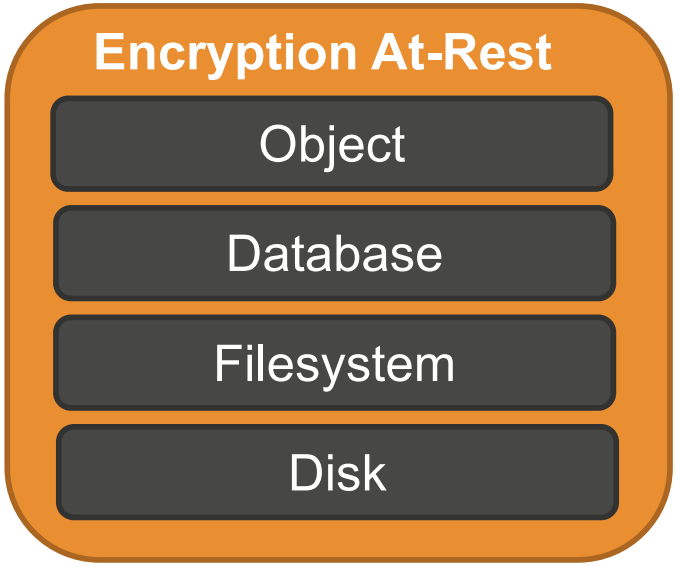
- Please follow https://docs.aws.amazon.com/vpc/latest/userguide/getting-started-ipv4.html

Encryption

Encryption.

Protecting data in-transit and at-rest.

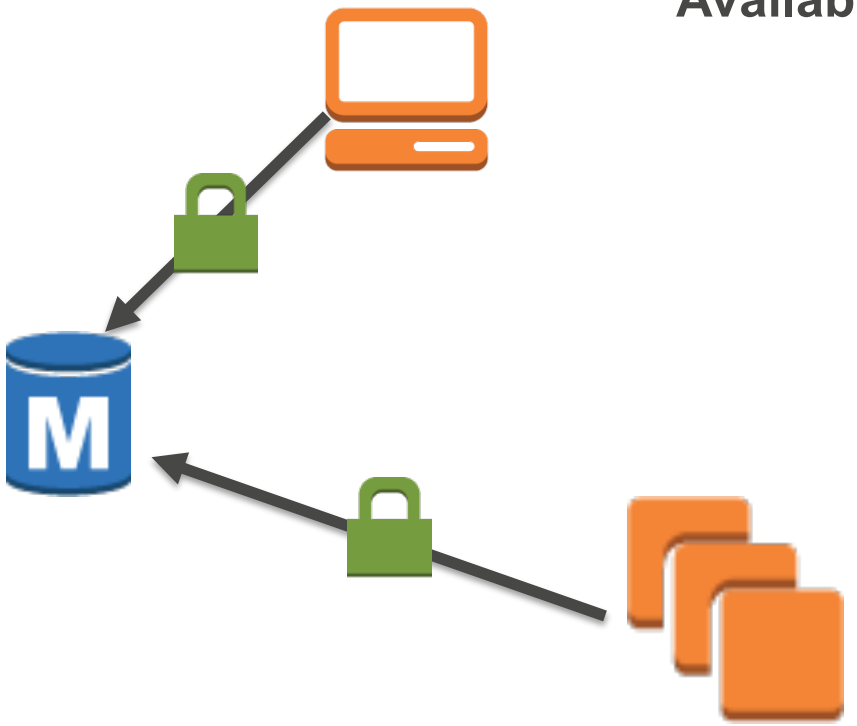




Details about encryption can be found in the AWS Whitepaper, "Securing Data at Rest with Encryption".

SSL

E.g. Database traffic encryption



Available for all six DB engines

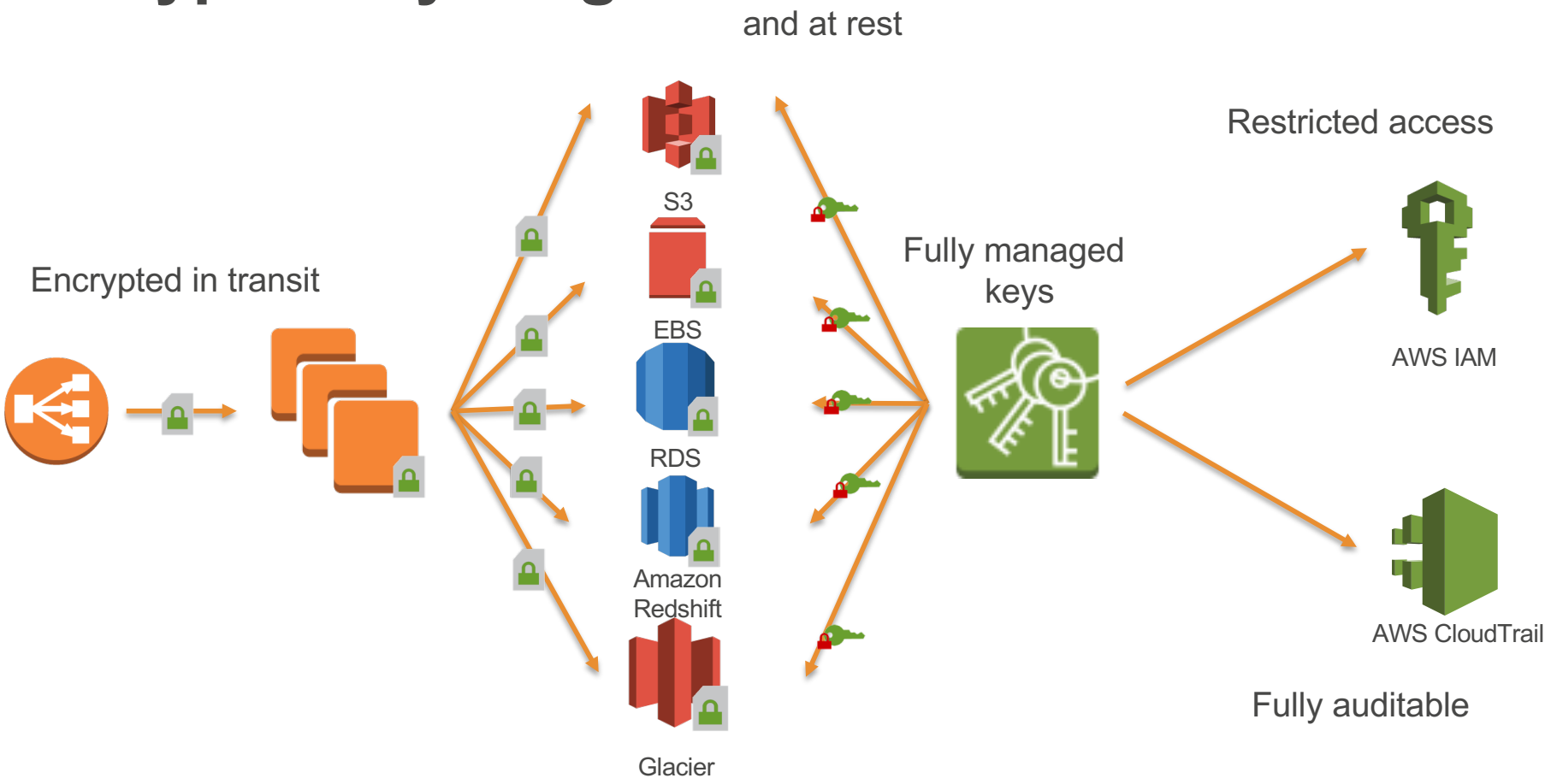
At-rest encryption

- DB instance storage
- Automated backups
- Read Replicas
- Snapshots

- Available for all six engines
- No additional cost
- Support compliance requirements

For additional details: http://docs.aws.amazon.com/AmazonRDS/latest/UserGuide/Overview.Encryption.html

Encrypt everything!



Encrypt your sensitive information

Native encryption across services for free

- Amazon S3, Amazon EBS, Amazon RDS, Amazon Redshift
- End-to-end SSL/TLS

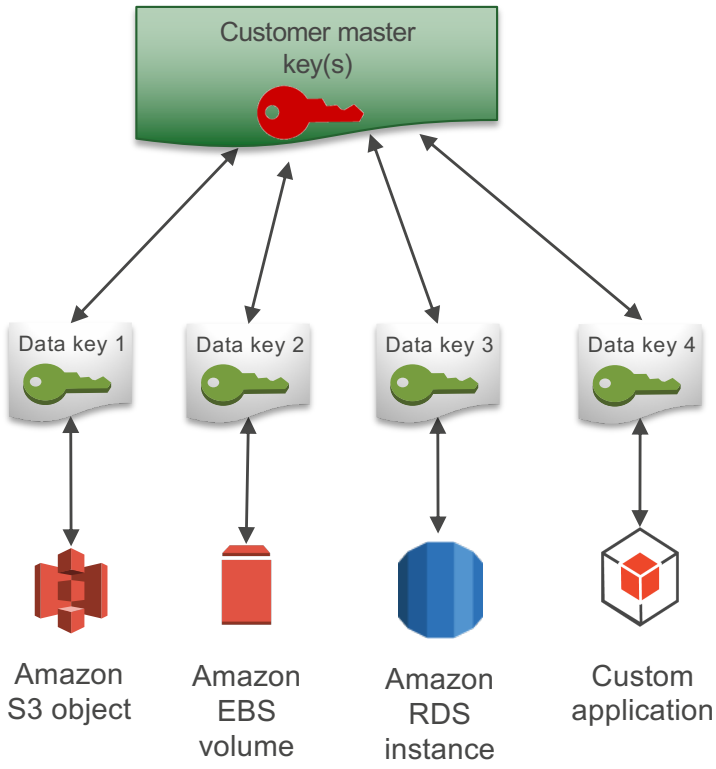
Scalable key management

- AWS Key Management Service (KMS) provides scalable, low-cost key management
- AWS CloudHSM provides hardware-based, high-assurance key generation, storage, and management

AWS KMS—RDS standard encryption

- Two-tiered key hierarchy using envelope encryption:
  - Unique data key encrypts customer data
  - AWS KMS master keys encrypt data keys

- Benefits:
  - Limits risk of compromised data key
  - Better performance for encrypting large data
  - Easier to manage small number of master keys than millions of data keys
  - Centralized access and audit of key activity
  - For the data encryption KMS uses industry standard AES-256 encryption to protect the data



Enabling encryption

AWS Command Line Interface (AWS CLI)

```
aws rds create-db-instance --region us-west-2 --db-instance-identifier sg-cli-test \
--allocated-storage 20 --storage-encrypted \
--db-instance-class db.m4.large --engine mysql \
--master-username myawsuser --master-user-password myawsuser
```

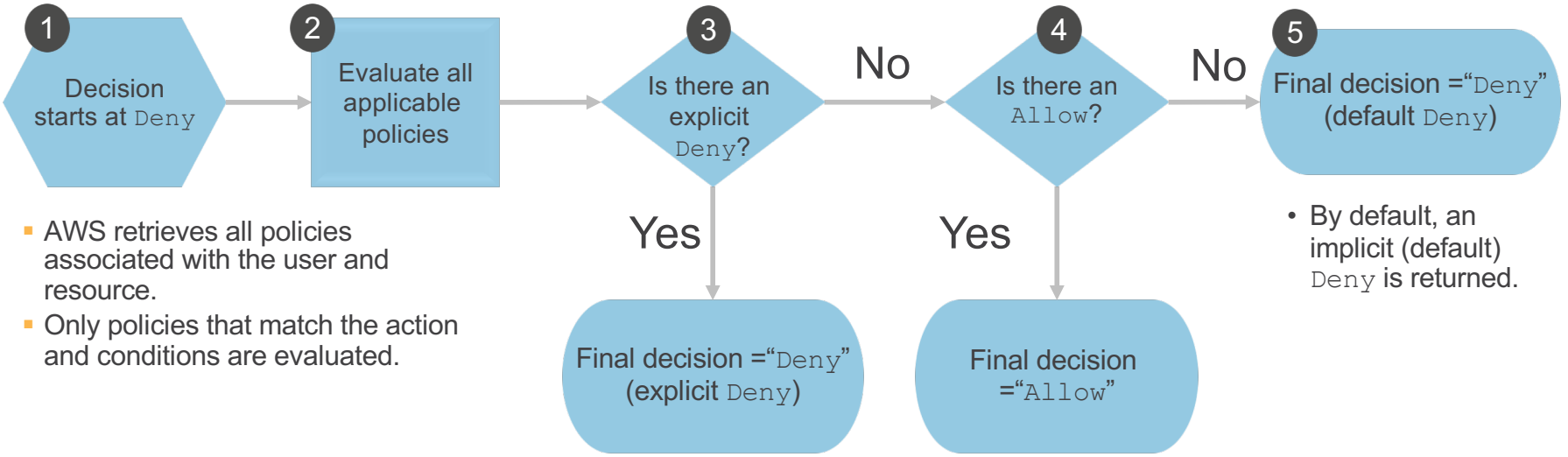
```
aws rds create-db-instance --region us-west-2 --db-instance-identifier sg-cli-test1 \
--allocated-storage 20 --storage-encrypted  --kms-key-id xxxxxxxxxxxxxxxxxx \
--db-instance-class db.m4.large --engine mysql \ --master-username myawsuser
--master-user-password myawsuser
```

Identity management

AWS Identity and Access Management (IAM)

- Enables you to control who can do what in your AWS account
- Splits into *users*, *groups*, *roles*, and *permissions*
- Control
  - Centralized
  - Fine-grained - APIs, resources, and AWS Management Console
- Security
  - Secure (deny) by default

Policy enforcement



- AWS retrieves all policies associated with the user and resource.
- Only policies that match the action and conditions are evaluated.

- If a policy statement has a `Deny`, it trumps all other policy statements.

- Access is granted if there is an explicit `Allow` and no `Deny`.

- By default, an implicit (default) `Deny` is returned.

Default Deny All.

IAM anatomy

- JSON-formatted documents
- Statement (permissions) specifies:
  - **P**rincipal
  - **A**ction
  - **R**esource
  - **C**ondition

```
{
 "Statement":[{
   "Effect":"effect",
   "Principal":"principal",
   "Action":"action",
   "Resource":"arn",
   "Condition":{
     "condition":{
       "key":"value" }
     }
   }
  ]
 }
```

Principal – Examples

- An entity that is allowed or denied access to a resource
- Indicated by an Amazon Resource Name (ARN)
- With IAM policies, the principal element is implicit (i.e., the user, group, or role

```
<!-- Everyone (anonymous users) -->
"Principal":"AWS":"*.*"

<!-- Specific account or accounts -->
"Principal":{"AWS":"arn:aws:iam::123456789012:root" }
"Principal":{"AWS":"123456789012"}

<!-- Individual IAM user -->
"Principal":"AWS":"arn:aws:iam::123456789012:user/username"

<!-- Federated user (using web identity federation) -->
"Principal":{"Federated":"www.amazon.com"}
"Principal":{"Federated":"graph.facebook.com"}
"Principal":{"Federated":"accounts.google.com"}

<!-- Specific role -->
"Principal":{"AWS":"arn:aws:iam::123456789012:role/rolename"}

<!-- Specific service -->
 "Principal":{"Service":"ec2.amazonaws.com"}
```

Replace with your account number

Action – Examples

- Describes the type of access that should be allowed or denied
- You can find these in the docs or use the policy editor to get a drop-down list
- Statements must include either an `Action` or `NotAction` element

```
<!-- EC2 action -->
"Action":"ec2:StartInstances"

<!-- IAM action -->
"Action":"iam:ChangePassword"

<!-- S3 action -->
"Action":"s3:GetObject"

<!-- Specify multiple values for the Action element-->
"Action":["sqs:SendMessage","sqs:ReceiveMessage"]

<--Use wildcards (* or ?) as part of the action name.  This would cover
Create/Delete/List/Update-->
"Action":"iam:*AccessKey*"
```

Resource – Examples

- The object or objects that are being requested
- Statements must include either a `Resource` or a `NotResource` element

```
<-- S3 Bucket -->
"Resource":"arn:aws:s3:::my_bucket/*"

<-- SQS queue-->
"Resource":"arn:aws:sqs:us-west-2:123456789012:queue1"

<-- Multiple DynamoDB tables -->
"Resource":["arn:aws:dynamodb:us-west-
2:123456789012:table/person_table",
          "arn:aws:dynamodb:us-west-
2:123456789012:table/employees_table"]

<-- All EC2 instances for an account in a region -->
 "Resource": "arn:aws:ec2:us-east-1:123456789012:instance/*"
```

Condition example

What if you wanted to restrict access to a time frame and IP address range?

```
"Condition" :  {
   "DateGreaterThan" : {"aws:CurrentTime" : "2015-10-08T12:00:00Z"},
   "DateLessThan": {"aws:CurrentTime" : "2015-10-08T15:00:00Z"},
   "IpAddress" : {"aws:SourceIp" : ["192.0.2.0/24", "203.0.113.0/24"]}
}
```

AND

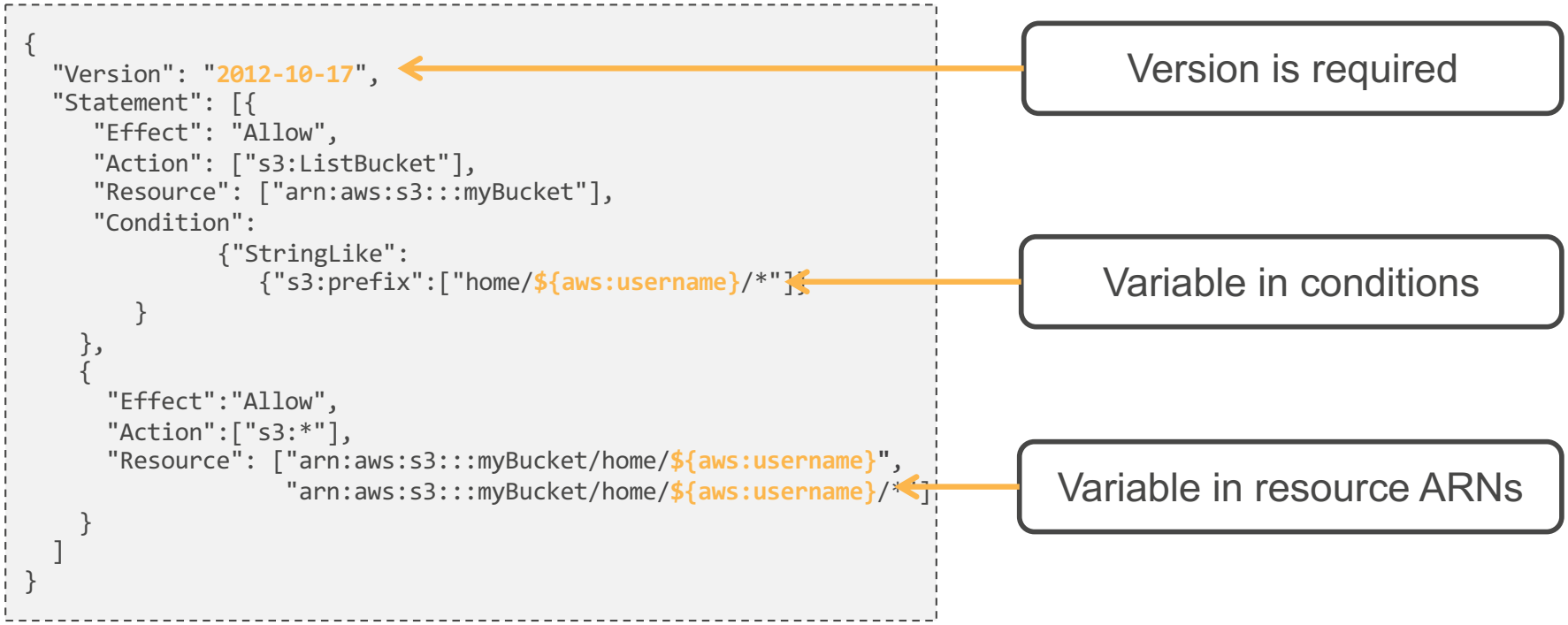
OR

Allows a user to access a resource under the following conditions:

- The time is after 12:00 P.M. on 10/8/2015 AND
- The time is before 3:00 P.M. on 10/8/2015 AND
- The request comes from an IP address in the 192.0.2.0 /24 OR 203.0.113.0 /24 range

All of these conditions must be met in order for the statement to evaluate to TRUE.

The anatomy of a policy with variables



Grants a user access to a home directory in Amazon S3 that can be accessed programmatically

IAM best practices

Top 11 IAM best practices

1. **Users** – Create individual users.
2. **Permissions** – Grant least privilege.
3. **Groups** – Manage permissions with groups.
4. **Conditions** – Restrict privileged access further with conditions.
5. **Auditing** – Enable AWS CloudTrail to get logs of API calls.
6. **Password** – Configure a strong password policy.
7. **Rotate** – Rotate security credentials regularly.
8. **MFA** – Enable MFA for privileged users.
9. **Sharing** – Use IAM roles to share access.
10. **Roles** – Use IAM roles for Amazon EC2 instances.
11. **Root** – Reduce or remove use of root.

AWS access keys vs. passwords

Depends on how your users will access AWS

- Console → Password
- API, CLI, SDK → Access keys

Make sure to rotate credentials regularly

- Use credential reports to audit credential rotation.
- Configure password policy.
- Configure policy to allow access key rotation.

Enabling credential rotation for IAM users

Enable access key rotation sample policy

- **Access keys**

```
{
 "Version":"2012-10-17",
  "Statement": [{
  "Effect": "Allow",
  "Action": [
      "iam:CreateAccessKey",
      "iam:DeleteAccessKey",
      "iam:ListAccessKeys",
      "iam:UpdateAccessKey"],
  "Resource":
     "arn:aws:iam::123456789012:
user/${aws:username}"
}]}
```

Steps to rotate access keys

1. While the first set of credentials is still active, create a second set of credentials, which will also be active by default.
2. Update all applications to use the new credentials.
3. Change the state of the first set of credentials to `Inactive`.
4. Using only the new credentials, confirm that your applications are working well.
5. Delete the first set of credentials.

One AWS account vs. multiple AWS accounts?

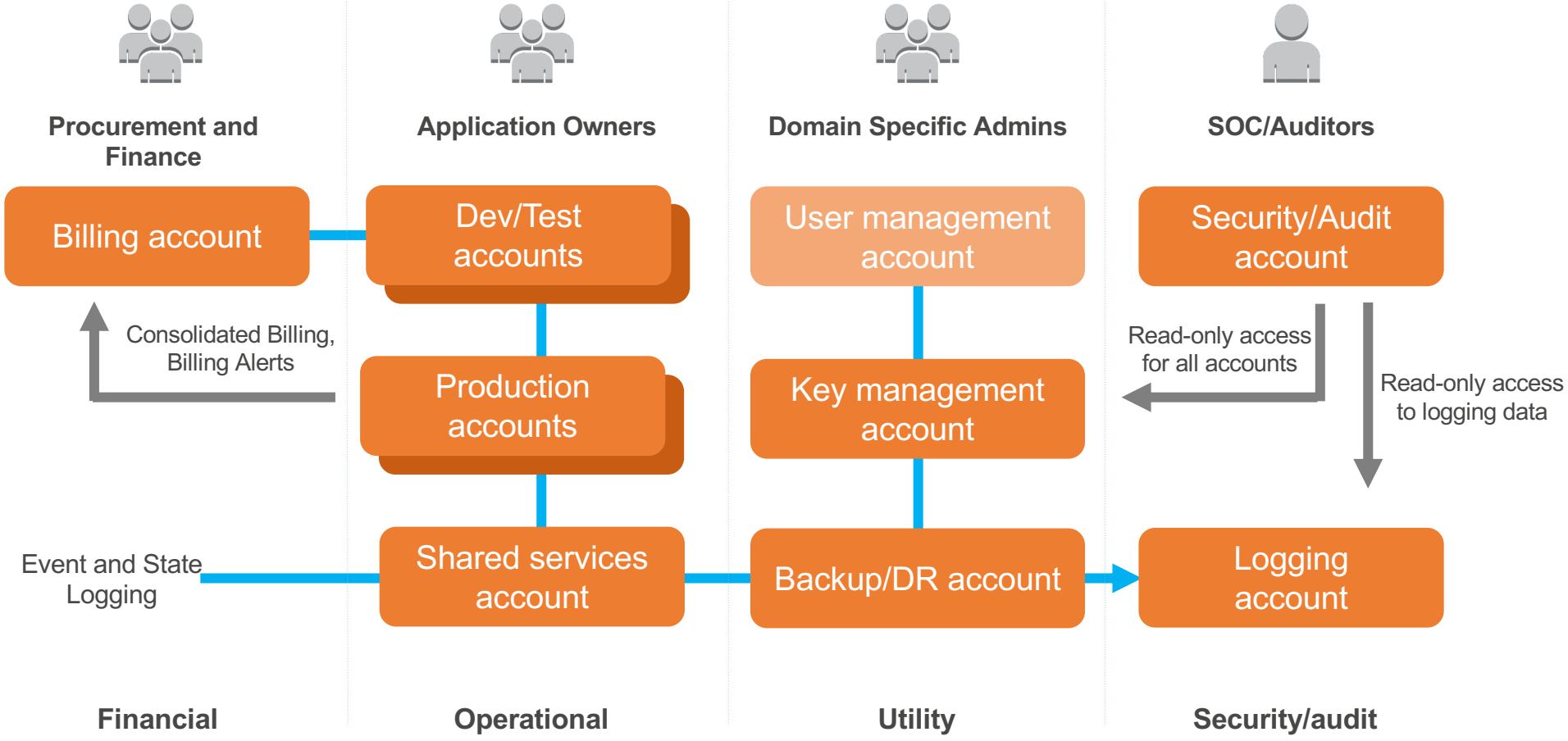
Use a single AWS account when you:

- Want simpler control of who does what in your AWS environment.
- Have no need to isolate projects/products/teams.
- Have no need for breaking up the cost.

Use multiple AWS accounts when you:

- Need full isolation between projects/teams/environments.
- Want to isolate recovery data and/or auditing data (e.g., writing your CloudTrail logs to a different account).
- Need a single bill, but want to break out the cost and usage.

Segmented AWS account structure



https://aws.amazon.com/answers/account-management/aws-multi-account-billing-strategy/

Hands-on Lab #18: IAM

Hands-on Lab#18: IAM

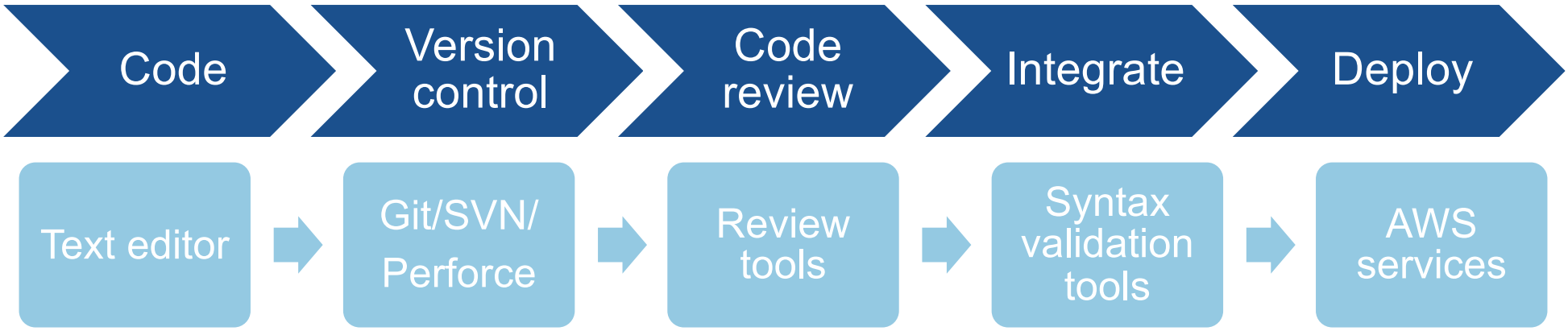
- The lab will cover following topics:
  - Exploring pre-created IAM users and groups
  - Inspecting IAM policies as applied to pre-created groups
  - Adding users to groups with specific capabilities enabled
  - Locating and using the IAM sign-in URL
  - Experimenting the effects of policies on service access

- If you do not have existing qwiklabs account, please create a free one at https://qwiklabs.com/
- Once you have the account created, complete following lab:
- https://www.qwiklabs.com/focuses/7782?catalog_rank=%7B%22rank%22%3A1%2C%22num_filters%22%3A0%2C%22has_search%22%3Atrue%7D&parent=catalog&search_id=3493843

Infrastructure as code

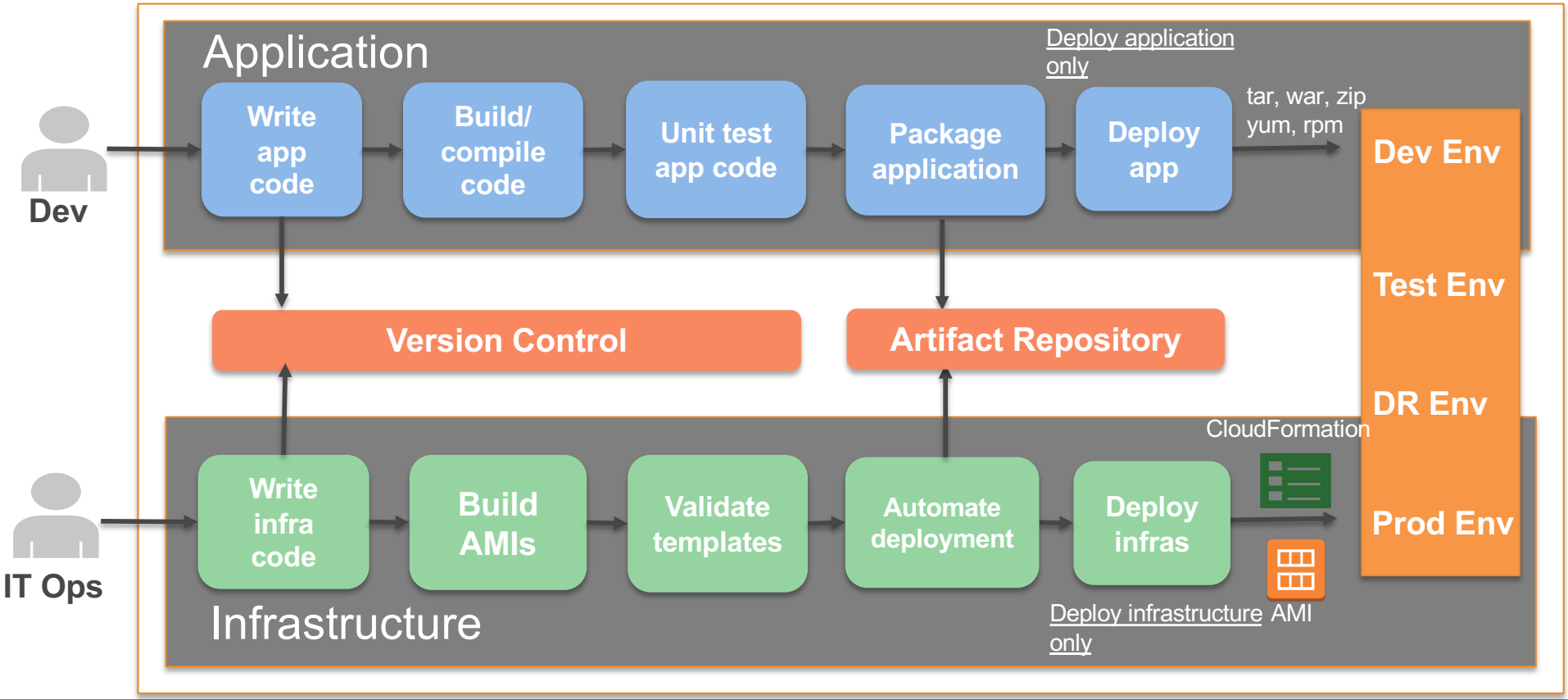
- **Infrastructure as code** is a practice whereby traditional infrastructure management techniques are supplemented and often replaced by using code-based tools and software development techniques.

Infrastructure as code workflow



“It’s all software”

Continuous integration/deployment and automation for security infrastructure



CloudFormation – Components & technology

Template

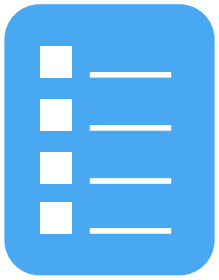
JSON formatted file

Parameter definition

Resource creation

Configuration actions

CloudFormation



Framework

Stack creation

Stack updates

Error detection and rollback

Stack

Configured AWS resources

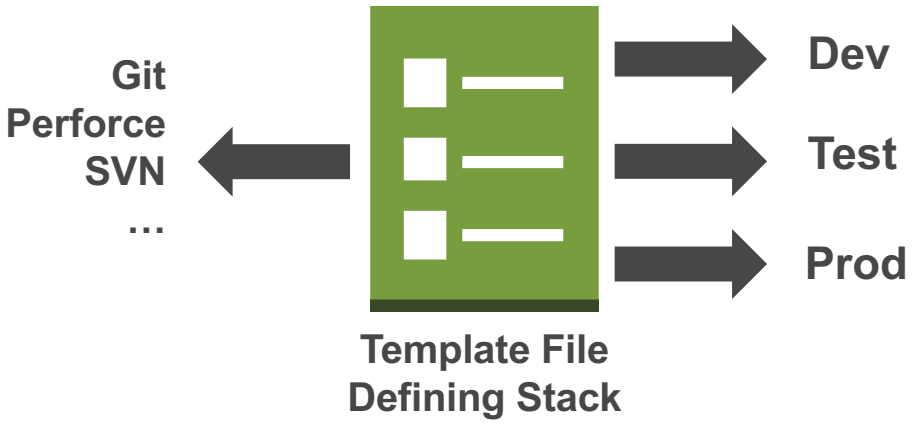
Comprehensive service support

Service event aware

Customizable

Many stacks & environments from one template

Use the version control system of your choice to store and track changes to this template



Build out multiple environments using the same template, such as for Development, Test, Production, and even DR

The entire infrastructure can be represented in an AWS CloudFormation template.

What security benefits does this give

Ability to perform "Code Audit" on your infrastructure

- Look for unauthorized network configurations
- Verify security groups
- Verify operating system
- Use with AWS CodeCommit trigger or GitHub hooks

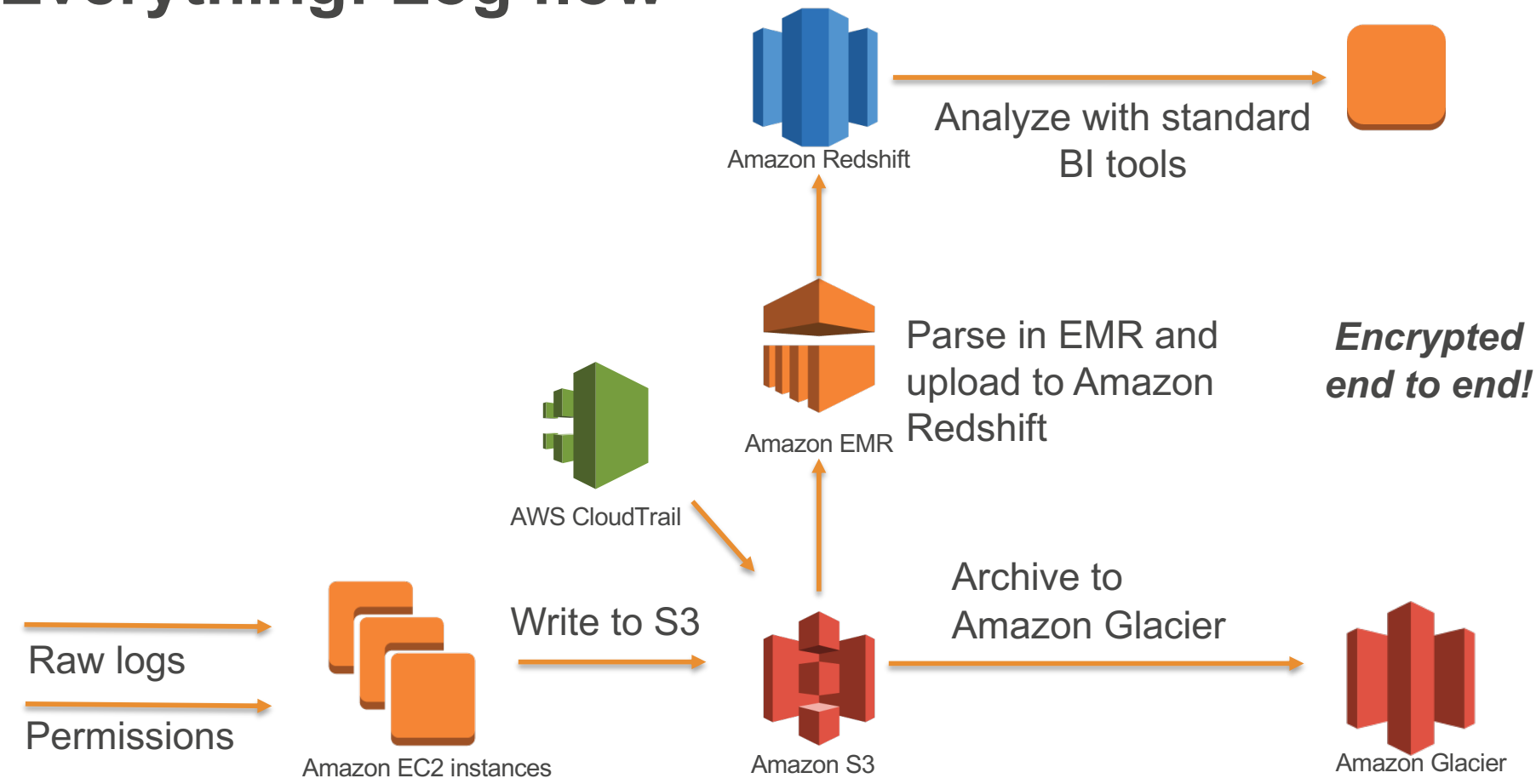
Split ownership (single file or merge)

- App team owns main section
- Network team owns VPC/subnets
- Security team owns security groups

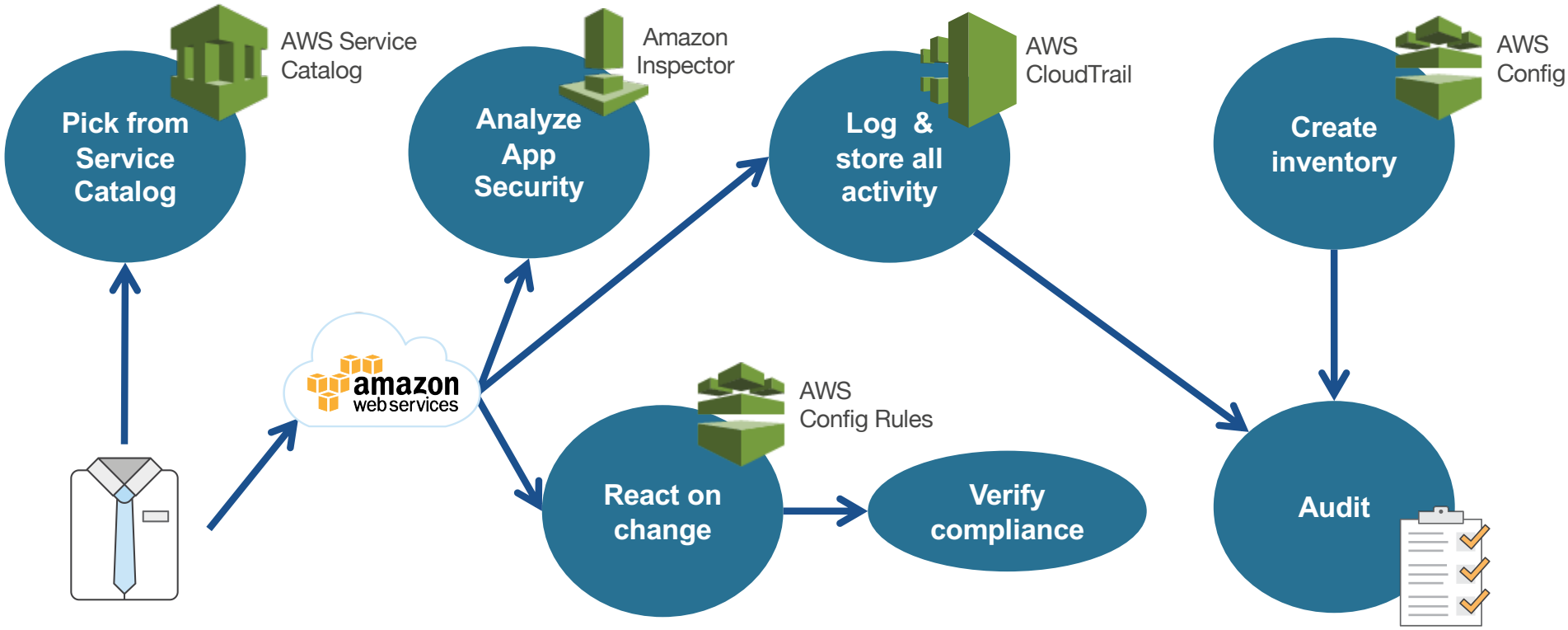
Automate upon check-in!

Audit and log your AWS service usage

Log Everything: Log flow



Example security pattern using AWS services



Security and compliance becomes visible and automated!

Why cloud logging/monitoring is different

- Distributed servers coming and going (e.g., Auto Scaling, micro services)
- More visibility (e.g., AWS CloudTrail)
- In the cloud, we have more log types than in the data center. More different kinds of data. Many distinct log sources not monitored by same systems on premises.
- Networking (Amazon VPC Flow Logs)
- System/application
- Configuration (very difficult on-premises)
- Large amount of information(e.g., Amazon VPC Flow Logs)

Different log categories

AWS infrastructure logs

- AWS CloudTrail
- Amazon VPC Flow Logs

AWS service logs

- Amazon S3
- Elastic Load Balancing
- Amazon CloudFront
- AWS Lambda
- AWS Elastic Beanstalk
- …

Host-based logs

- Messages
- Security
- NGINX/Apache/IIS
- Windows Event Logs
- Windows Performance Counters
- …

Amazon CloudWatch Logs

Ubiquitous logging and monitoring

Amazon CloudWatch Logs lets you grab everything and monitor activity

- Storage is cheap - collect and keep your logs
- Agent based (Linux and Windows)
- Export data
  - To Amazon S3
  - Stream to Amazon Elasticsearch Service or AWS Lambda
- Integration with metrics and alarms means you can continually scan for events you know might be suspicious
- Combine/use third-party products

  IF (detect web attack> 10 in a 1-minute period)

  ALARM == INCIDENT IN PROGRESS!

Amazon CloudWatch Events

Tools - Amazon CloudWatch Events

Trigger on event

- Amazon EC2 instance state change notification
- AWS API call *(*very specific*)*
- AWS Management Console sign-in
- Auto Scaling *(*no lifecycle hooks*)*

Or schedule (used by AWS Lambda)

- Cron is in the cloud!
- No more “unreliable town clock”
- Min 5 minutes

Single event can have multiple targets

AWS CloudTrail

What can you answer using a CloudTrail event?

- **Who** made the API call?
- **When** was the API call made?
- **What** was the API call?
- **Which** resources were acted upon in the API call?
- **Where** was the API call made from and made to?

Supported services:
http://docs.aws.amazon.com/awscloudtrail/latest/userguide/cloudtrail-supported-services.html

What does an event look like?


```
{
        "eventVersion": "1.01",
        "userIdentity": {
                "type": "IAMUser", // Who?
                "principalId": "AIDAJDPLRKLG7UEXAMPLE",
                "arn": "arn:aws:iam::123456789012:user/Alice", //Who?
                "accountId": "123456789012",
                "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
                "userName": "Alice",
                "sessionContext": {
                        "attributes": {
                                "mfaAuthenticated": "false",
                                "creationDate": "2014-03-18T14:29:23Z"
                        }
                }
        },
        "eventTime": "2014-03-18T14:30:07Z", //When?
        "eventSource": "cloudtrail.amazonaws.com",
        "eventName": "StartLogging", //What?
        "awsRegion": "us-west-2",//Where to?
        "sourceIPAddress": "72.21.198.64", // Where from?
        "userAgent": "AWSConsole, aws-sdk-java/1.4.5 Linux/x.xx.fleetxen Java_HotSpot(TM)_64-Bit_Server_VM/xx",
        "requestParameters": {
                "name": "Default“ // Which resource?
        },
        // more event details

}
```

AWS CloudTrail best practices

AWS CloudTrail best practices

1. Enable in all regions
2. Enable log file validation
3. Encrypted logs
4. Integrate with Amazon CloudWatch Logs
5. Centralize logs from all accounts

Benefits

- Also tracks unused regions
- Can be done in single configuration step
- Configure all accounts to send logs to a central security account
- Reduce risk for log tampering
- Can be combined with S3 CRR
- Configure alerting on events
- AWS CloudTrail will start delivering digest files on an hourly basis
- Include dev/stage accounts!

Log management and analytics

- ELK (Elasticsearch Service + Logstash + Kibana)
- Elasticsearch Service + Kibana + Amazon CloudWatch Logs
- Third-party solution

Automating your compliance checks

Multiple levels of automation

Self managed

- AWS CloudTrail -> Amazon CloudWatch Logs -> Amazon CloudWatch Alerts
- AWS CloudTrail -> Amazon SNS -> AWS Lambda

Compliance validation

- AWS Config Rules

Host-based compliance validation

- Amazon Inspector

Active change remediation

- Amazon CloudWatch Events

AWS Config Rules

Tools - AWS Config Rules

Time based

- When configuration snapshot is delivered
- Choose between 1, 3, 6, 12 or 24 hours

Change based

- EC2, IAM, CloudTrail, or tags

AWS managed or custom checks using Lambda

- Control compliance status using Lambda
- Encrypted volumes, CloudTrail, EIP attached, SSH access, Amazon EC2 in Amazon VPC, restricted common ports, and require tags

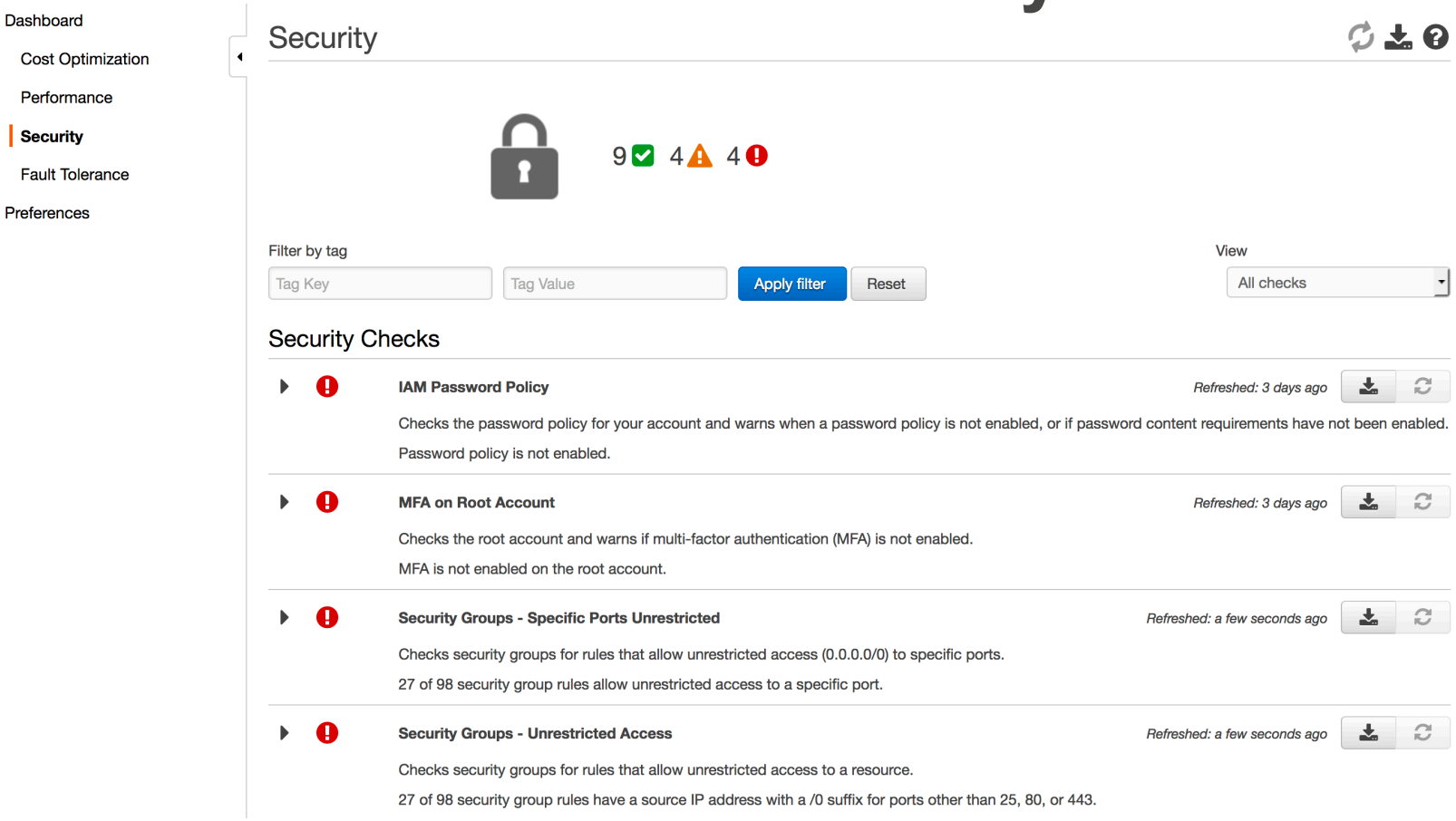
Amazon Inspector

What is Amazon Inspector?

- Enables you to analyze the behavior of your AWS resources and helps identify potential security issues
- **Application security assessment**
  - Agent based
  - 15 minutes–24 hours
- **Selectable built-in rules (rule packages)**
  - Common vulnerabilities and exposures
  - CIS Operating System Security Configuration Benchmarks
  - Security best practices
  - Run-time behavior analysis
- **Security findings – guidance and management**
- **Automatable via APIs**

AWS Trusted Advisor

AWS Trusted Advisor checks your account



Key Security Best Practices

- Enforce separation of duties and least privilege accounts
- MFA on users; enforce using IAM policies
- Know what is security vs. troubleshooting logs
- Storage is cheap, not knowing can be very expensive – log if possible
- Alerting is good, automating your security response is better
- Use managed services and built-in reporting to offload and automate
- See the big picture: what info do you want and what tool can give it to you

Hands-on Lab #19: KMS

Hands-on Lab#19: KMS

- The lab will cover following topics:
  - Creating Keys
  - Viewing Keys
  - Editing Keys
  - Tagging Keys
  - Enabling & Disabling Keys

- Please follow instructions from:
  https://docs.aws.amazon.com/kms/latest/developerguide/getting-started.html

Next Steps

Next Steps

- **Read/ Complete**:
  - Review the lecture #10 slides
  - Complete lab#17, lab# 18 and Lab# 19
  - Read VPC, IAM, CloudTrail, CloudWatch and KMS FAQ from AWS site