Notes Based on Real Failed Attempts

**Athena**- is an interactive query service that enables you to analyze and query data located in S3 using SQL.

* Serverless
* Noting to provision
* Pay per query and per TB scanned
* No need to set up complex/transform/load (ETL) processes
* Works directly with data stored in S3

Use cases

* Can be used to query log files stored in S3, ELB logs, S3 access logs, CloudTrail logs
* Generate business reports on data stored in S3
* Analyze AWS cost and usage reports
* Run queries on click-stream data

**Quicksights**- A business analytics service you can use to build visualizations, perform ad hoc analysis, and get business insights from your data

How to make sure you do not use AWS provided DNS

Set the EnableDNSHostnames=False – this indicates whether the instances launched in the VPC get public DNS names

Set EnableDNSSupport=False – disables the amazon provided DNS server

**Cognito Sync Trigger**- an AWS service and client library that enables cross-device syncing of application related user data

**SES Port number**- Open port 587 which is SMTP

**ElastiSearch**- a managed service that makes it easy to deploy, operate, and scale ElastiSearch clusters in the AWS cloud.

**Securing ECS**- EC2 instances use IAM roles to access ECS. ECS tasks use an IAM role to access services and resources. Also you secure them with SG and NACLs

Write once Read many Glacier policy

For this specific question it will give you policy documents that you have to read. Look for ones that only allow uploads to glacier, and they don’t allow archive deletes. There should be an action of Glacier:DeleteArchive with the condition of NumericLessThan Glacier:ArchiveInDays that abides by the amount of days mandatory before deletes are allowed to occur.

**How to recover a deleted CMK?** So you can’t, its lost forever, except the question did not have that as one of the answers…

I think that this was a CMK with imported key material and I missed that. If you have a CMK that is deleted with imported key material then you can reupload that key material again. This is the only logical explanation I got for that specific question.

AWS Security Specialty Notes Linux Academy Domain 1

**1.1 – Given an AWS Abuse Notice, Evaluate a Suspected Compromised Instance or Exposed Access Keys**

**AWS Abuse Notification**

Compromised Resources and Abuse

Abuse Activities- Externally observed behavior of AWS customer instances or resources that are malicious, offensive, illegal, or could harm other internet sites.

AWS will shut down malicious abusers, but many of the abuse complaints are about customers conducting legitimate business on AWS.

AWS have automated system monitoring in place and this is managed by the abuse team.

**Things the abuse team manages**

Monitors your account and all other customer accounts

* Monitors large changes in patterns
* Unusual or accuracies of connections
* Specific port usage for known exploits

The monitoring also looks internally

* Looks at AWS Hypervisors
* Or other internal virtual networking infrastructure

Your environments are also being accessed for any outgoing issues

* Are you attempting to do anything AWS deems unacceptable?
* Or have any of your resource been exploited and someone else is doing something unacceptable?

Also managed manual reports of abuse

* Could be abuse with you against others
* or it could be external people notifying AWS that you account credentials have been leaked

The AWS abuse team can identify issues that you are aware of or things you are unaware of. (Could be doing somethings on accident)

Incidents can originate from your account, or they can be targeting your account

Incidents will be breaches of the AUP (Acceptable Use Policy)- The AUP describes activities that are prohibited uses of the web services offered by AWS.

Penetration testing and other simulated load tests breach the acceptable use policy.

* AWS allows these testing’s but only if you get permissions from AWS before preforming the tests
* 3 main types of tests that need involvement from the AWS Abuse Team: Vulnerability tests, penetration tests, and simulated events.
* You need to fill out a web form as the root user of you account and submit it to perform pen tests. For simulated events you just need to email the simulation team a high-level detail of what you are going to simulate and how.
* Once submitted AWS will process the request and provide you with an authorization number. It is then when you know your request has been approved.
* The Request has a start and end date and the test can only be performed during that time.
* AWS only allow pen testing on specific services: EC2, RDS, Aurora, DNS Zone Walking, CloudFront, API Gateway, LightSail, and Lambda
* With RDS there is no pen testing on small or micro instance types
* With EC2 you cannot pen test on m1.small, t1.micro, and t2.nano. The reason behind this is because with the T instance types you are subscribing to those instance types and paying for a portion of those instances. By applying load testing you could impact other customers.

Sample Abuse Notification



Could be done by the owner who doesn’t know he is port scanning, or the instance might have been hacked and someone else is being malicious on his behalf.

Anatomy of notice

* Provides the instance ID which has been breaching the AUP
* Informs the action the abuse team has taken

What should you do if you receive this notice?

* Fix the issue
* Then respond to the abuse team
* If you don’t further action may be taken which could include termination of your account

Other reasons for receiving abuse notice (not your fault)

* Compromised resource- EC2 instance becomes a botnet then attacks other hosts on the internet. This traffic could be going to other AWS accounts as well. This could happen by leaking SSH keys, outdated software was compromised through a known exploit.
* Secondary Abuse- Infected file has been installed on a compromised instance. That file will call home to a malicious software endpoint.
* Application Function- Might be completely legit web traffic that may appear as a DDOS attack.
* False Complaints- Other AWS account users may complain about your traffic.

**Responding to AWS Abuse Notifications**

First and mandatory action is you need to respond to the abuse team. If you don’t the abuse team may assume you are doing something wrong on purpose and terminate your AWS account. The other thing you need to do is remove the immediate the threat.

* Account credentials have been leaked
* EC2 instance credentials have been leaked

Responding in terms of removing the immediate threat

**If you have been exploited by leaked credentials. This could be root account username and password, IAM username and password, or leaked Access Keys.**

* Change the root password and the passwords for all IAM users
* Rotate all Access keys
* Delete root Access Keys

How to change the root account password

1. Be logged in as root
2. Click my security credentials by the account drop down
3. There you can see the option to change your password and delete access keys for your root account

How to change the passwords of IAM users

1. Go to IAM and click on a user
2. Go to the security credentials tab
3. There you can manage your password and access keys

For a large number of IAM accounts you should look to script the process or changing all of your passwords or do it via the API.

Add MFA to all Admin Users and anyone who accessing the AWS Console.

How to

1. Select IAM user you want to add MFA too
2. Go to Security Credentials
3. Edit a sighed MFA Device
4. Click Virtual MFA device
5. Scan your QR code and activate

**Your EC2 has been hacked.**

1. Create a new key pair and update your instance with the new key pair. Also delete the compromised key pairs.
2. You can do 2 things when an instance has been hacked. You can either create an AMI and launch a new instance with the new key pair, or you can SSH in and edit the .ssh/authorized\_keys file and add the new key pair there.

Creating instance from scratch is the safer option. With the editing the existing file option you may be applying that new key to a still exploited instance.

**Access Key Leak.**

This is a very common scenario and generally happens for 1 or 2 reasons. Somebody commits Access Keys to a Git Repo or they just misunderstand how potentially threatening leaking Access Keys is.

There are bots that scan all public repos to capture any leaks and use them for harm.

Rotating keys steps

1. Go to an IAM users
2. Go to Security Credentials and locate the access keys
3. Make that access key inactive or delete it entirely
4. Create a new Access Key

You should do that for all IAM users within an exploited account and also double check that the root user does not have any active access keys. If he does delete them as it is best practice to have no Access Keys for the root user.

**Delete unrecognized resources.**

A fairly common tactic for hackers is too leave behind a future way to get into your account. This allows the attacker a way for future attacks to the account.

* May create a new EC2 instance
* Makes a new IAM user for him
* Puts in spot bids
* Check all regions

**Contact AWS Support.**

* Respond to the notification
* Tell them what happened and tell them the steps you took to remediate the incident

**Being Proactive to Avoid Being Compromised.**

* Vault root credentials and remove access keys from the root account
* Set up strong passwords using a strong password policy and MFA on all IAM accounts
* Use IAM Roles whenever possible
* Do not copy EC2 key pairs to instances and protect the key pairs on admin machines
* Rotate IAM Access Keys regularly
* Use Git Secrets to prevent committing Access Keys to public Git Repos

**1.2 - Verify That the Incident Response Plan Includes Relevant AWS Services.**

**What is Incident Response?**

**Incident-** An unplanned interruption or delegation of an IT service

The incident response framework

For the purpose of this course and the exam, we can think of incident response in the cloud as a progression of seven steps or phases.

Preparation, identification, containment, investigation, eradication, recovery, follow-up

From an IT standpoint, incidents stem from system failures. Can be caused by intentional actions from a 3rd party

**Incident Response-** Simply the way in which you respond to an incident

**Incident Handling-** How to effectively handle incidents end to end

**Information Spillage Response-** How to react when IAM credentials or Access Keys are leaked

7 Phases in the Incident Response Framework

**Preparation-** Taking proactive steps to prepare for incidents before they occur. Reduces the impact of incident before they occur.

**Identification-** Detection or being able to detect incidents

**Containment-** place temporary restrictions on things

* Close off security rules
* Preventing access that is otherwise allowed
* This allows us to prepare for the next phase

**Investigation-** Getting security professionals to deep dive into the state of systems

**Eradication-** Take the time to remove exploits from your systems

**Recovery-** Bring the system back to an operational state

* Replace damaged instance
* Restore any storage
* Remove any of the changes made during the containment phase

This is a recurring process. You go through the circle until you completely contain the exploit.

**Follow-Up-** Relax and look at the end to end process and look how successful it was

**Incident Response Framework Part 1**

Covers the first 3 stages of Preparation, Identification, and Containment in more detail.

**Preparation Phase-** This phase is all about doing everything we can to prevent breaches and failures. Eventually some type of security event will happen. We are building wall and barricades here.

4 parts

* Be proactive, limit the blast, log everything, encrypt it all

**Being Proactive-** to be proactive is to work ahead of the incident and anticipating outcomes. Best proactive practices are

* Risk management- Determine where the different levels of risk are
* Principle of least privilege- No unnecessary permissions
* Architect for failure- High availability and fault tolerance
* Train for the real thing- Test and simulate incident response practice
* Clear ownership and governance- Tag all resources so no time is wasted finding who or what group to contact
* Data classification- Tagging data stores with classification can identify spillage

AWS service involved- IAM, VPC, Route 53, EC2, EFS, and RDS

**Limit the blast-** Careful planning, segment off resources from each other.

**AWS Organizations-** Adding accounts under one main account

* If there is a breach, it will not affect multiple account
* Organizations allow for service Control Policies and can set up child account that can be limited
* Using multiple regions and VPC’s have similar affects

Best practices if to use separate account for each team because if keys get leaked it will only affect 1 team instead of all the teams

Region and VPC segregation only protect against network exploits, while organizations and multiple account segregate against account exploits as well as networking exploits.

**Log Everything-** The best way to get info about your environments is to log activity in services and accounts.

**Centralized Logging-** collecting all the logs from the org in one central place.

* Encrypt and protect logs
* It all starts with logs. Logs go to events, events go to alerts, and alerts go to response.

AWS Services involved with logging- CloudTrail, VPC Flow Logs, EC2 OS and app logs, S3, Config, CloudWatch logs, Lambda.

**Encrypt it all-** Encryption is the process of masking data by scrambling.

AWS services involved- KMS, S3, ACM, ELB, Route 53

**Identification Phase-** This phase is where we discover an incident is occurring.

The intention of this phase is to find compromised resources quickly

Blast radios- what resources were affected?

Data loss protection- A combination of encryption and access control.

Resources needing “Clean-Up” – What resources do we need to mitigate or isolate?

This phase is difficult and you should rely on automation to help with detection.

AWS Services involved- CloudWatch, Events, Alarms, S3 Events, 3rd party tools

**Containment Phase-** This phase is about removing the immediate threat.

This phase relies heavily on the identification phase.

* A SG that restrict egress traffic and only allows management ports in
* A separate subnet with restrictive NACL’s we can move resources too
* An s3 bucket policy that is designed to immediately stop spillage
* An explicit deny policy created in IAM
* A key policy that denies all decryption

Additional actions

* Snapshot volumes of compromise instances (allows offline investigation)
* Stop instances
* Disable encryption keys in KMS
* Change Route 53 Record sets

AWS services involved- VPC, S3, KMS, Route 53

**Incident Response Framework: Part 2**

**Investigation Phase-** Involves event correlation and forensics

* Building a timeline of the incident
* Get an idea of the exact nature of the incident

**Live Box-** Working on the incident that has been exploited

**Dead Box-** Using a Snapshot of an exploited instance to perform offline analysis. Put in an isolated account in a private subnet.

AWS Services- VPC, VPC Flow Logs, EC2, CloudTrail, CloudWatch

**Eradication Phase-** This phase you try to remove all the infections and compromises in your resources. Deleting resources.

* Deleting or disabling any KMS keys
* For EBS, delete spilled files, create a new encrypted volume, copy all good files
* For S3 with S3 managed encryption, delete the object
* For S3 with kms managed keys or customer keys, delete both the object and the CMK
* Secure wipe any affective files

AWS services- KMS, EBS, S3

**Recovery Phase-** Put everything back to normal

* Restore resources
* Use new encryption keys
* Restore network access
* Monitor
* Have the containment phase tools ready

AWS services- VPC, S3, KMS, Route 53

**Follow-Up Phase-** Review every single stage

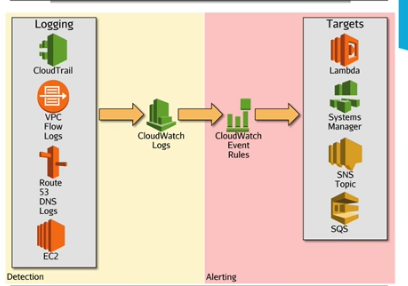
**1.3 – Evaluate the Config of Automated Alerting and Execute Possible Remediations**

**Automated Alerting**

**Automated Alerting-** The services we use in the cloud make scalability and reliability easy. These concepts should apply to your logging, monitoring, and alerting as well. Humans are much less reliable than cloud automation.

There are really 3 stages of the Automation of Security Incidents which are

* Detection- which is logging
* Alerting
* Response



**Detection-** The components that can evaluate incoming metrics and logs

**CloudWatch Logs-** Accepting data from other AWS services. Examples of AWS services that can send to CloudWatch logs

* **CloudTrail-** API calls
* **VPC Flow Logs-** Logs networking traffic in and out of your VPC. Metadata for IP traffic
* **Route 53 DNS Logs-** Logging of DNS queries and DNS alerts
* **EC2-** Configure EC2 via agents to logs OS and app logs

These are the main services, but almost every service has a way to send logs to CloudWatch Logs.

**Alerting-** Used to alert triggers

**CloudWatch Event Rules-** Gets triggered and can invoke a specific target. Example targets

* Lambda
* Systems Manager
* SNS Topic
* SQS Queues

You can use CloudWatch for Alerting in 2 ways.

CloudWatch Event Rules

CloudWatch Metric Filters

Differences

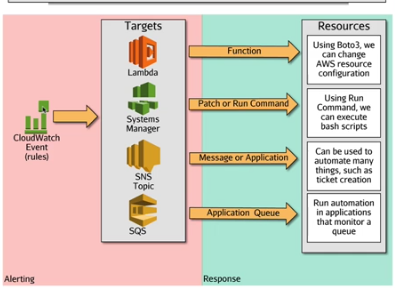
* CloudWatch Metric Filters are not Real Time. The delay is due to the fact that they are relying on the delivery of CloudTrail to CloudWatch logs.
* CloudWatch Event Rules can be real time depending on what the rules are looking for.4
* CloudWatch Metric Filters gives you an email with little information.
* The CloudWatch Event Rule will give you an email based on the API call that cause the Event to be triggered allowing for a better understanding of who, what, and when it happened.
* CloudWatch Events gives you a JSON formatted data that shows the exact event that triggered the event.

Uses

* Most of the time due to real time and the added function triggers like Lambda you should look to use CloudWatch Event rules. This is the newer version and Metric Filters are legacy.
* The exception is for when you need to monitor when a Root User logs in. Because this is a generic alert you do not need any more details besides when he logged in. Using CloudWatch Events rule would just be unnecessary.

**Automated Incident Response**

**Automated Response-** Once we get alerts generated in CloudWatch, there are a lot of target services we can trigger with those alerts. The most powerful and useful ones are Lambda and Systems Manager. We can configure these target services to automatically remediate our resources.



SNS can be used to integrate CloudWatch Events with external tools when you cannot perform the task using AWS products.

* Involving integration with 3rd party security products

Remember the 3 stages

* Detection- detects that a change or incident has occurred
* Alerting- CloudWatch Events or Metric filter
* Response- This is a target to perform some type of response

With CloudWatch Events you can completely automate all 3 of these stages. Which allows for incidents out of existence before they become a problem.

Legacy way is Metric Filters

New way is CloudWatch Event rules

AWS Security Specialty Notes Linux Academy Domain 2

**2.3 Design and Implement a Logging Solution**

**CloudTrail Logging**

**CloudTrail-** Logs API calls made to your account via the console, SDK’s, or CLI.

**Default Configuration:**

* CloudTrail stores events in your account using **event history** and by default it stores API calls for 90 days if you do not set up your own trail
* CloudTrail is turned on by default in your account

**Trail-** A configuration allowing for logs to be sent to an S3 bucket

* Single region or multi region (multi region applies to all current and future regions)
* Trails can make multi account logging possible

**Configuration Options:**

* **Management Events-** Enabling management events will log control plane events such as user login, configuration security, and even setting up future trails.
* **Data Events-** These are Object level events in S3 or Function level events in Lambda. You have the option of enabling per bucket/function or all buckets/functions.
* **Encryption-** The trail encrypts with SSE-S3 by default, but can be changed to SSE-KMS encryption when specified. Logs can be sent to an S3 bucket of choice or even to a prefixed folder of a bucket. Its always good to enable KMS because it separates the storage from the encryption. Especially good if you are using a CloudTrail Multi Account strategy.

CloudTrail is near real time.

All CloudTrail Events are JSON structured.

* Contains details about who performed the event
* Contains details about the event itself
* The time of the event
* Allows for event history auditing. Going back and looking at actions from a month ago or longer due to auditing requirements.

**CloudTrail Digest Folder: (Log file Validation)**

* This is an integrity system for logging to S3
* Useful for forensic investigation or other legal situations
* Good for governance, security, or auditing requirements

**Limitations:**

* Needs connectivity into AWS to verify the integrity of the logs
* Read access to the bucket that contains the log files
* The digest and log files cannot have been moved since AWS delivered them, renamed, changes to the file, or moving of the file will make the digest file invalid.
* There is a hash on every single file that is delivered. This makes it impossible to alter or adjust individual log files without CloudTrail being aware of it. If someone does something wrong like delete users and tries to cover up their tracks by changing the CloudTrail log files you can confirm that the log files were tampered with.
* Digest files allow you to demonstrate if any log files have been deleted
* Validation of log files can only be validated by the CLI

A trail that is applied to all regions will cover global events such as IAM. These events are always located in the Us-East-1 region which is Virginia.

**CloudWatch Logs: CloudTrail**

**CloudWatch Logs-** A central location which aggregates logs from many different services

* A central point for all types of logs within AWS and even your own on-premise servers

**CloudWatch Logs Components:**

* **Log Events-** A record of activity recorded by the monitored resource. (Single API call in a Log Stream)
* **Log Streams-** A sequence of log events from the same source. (Collects logs entries from the same host)
* **Log Groups-** A collection of logs streams with the same access control, monitoring, and retention settings. (A Log Group can have multiple Log Streams from different services)
* **Metric Filters-** Assigned to logs groups, and it extracts data from the group’s log streams and converts that data into a metric data point. (Metric Filters are near real time as it needs CloudTrail to inject logs into CloudWatch Logs)
* **Retention Settings-** A period of time logs is kept. Retention settings are assigned to groups, but it is applied to all streams of a group.
* **Subscriptions-** Streams logs from CloudWatch Logs into another system such as Kinesis or Lambda
  + Use for additional processing
  + Can stream logs to Lambda or ElastiSearch
  + You can also export logs to an S3 bucket

How can you have a central repository log for an on prem system? **CloudWatch Logs**

Individual expiry of Logs has to be configured at the Log Group level.

You can use CloudWatch logs to monitor, store, and access your log files from:

* CloudTrail
* VPC Flow Logs
* CloudWatch agent (EC2 instances, or On-Premise Servers)
* DNS Query Logs

**CloudWatch Logs: VPC Flow Logs**

**VPC Flow Logs-** Comprised of IP meta data traffic in and out of your VPC. These logs are sent to CloudWatch Logs by default.

* Flow Logs are useful for troubleshooting network conversations and can be assigned to a VPC, a single Subnet, or a single ENI. (Elastic Network Interface)
* It cannot inspect actually traffic. For that you need to look into the AWS marketplace or use a packet sniffer tool.
* What you can see: The source and destination IP, port, protocol, size, and more

**Capture Point-** The point that the IP traffic will start being logged: Either VPC, Subnet, or ENI.

**Capture Point VPC-** All traffic, from all instances, in all subnets inside a specific VPC the Flow Logs are attached too.

**Capture Point Subnet-** Only captures IP communications involving a single subnet and all the ENI’s inside that subnet.

**Capture Point ENI-** Only captures traffic where that specific ENI is involved in the communication.

A VPC Flow Logs created at the VPC level will automatically assign itself to every subnet and ENI in that VPC.

VPC Flow Logs Nomenclature:

1. Source IP Traffic
2. Destination IP Traffic
3. Source Port
4. Destination Port
5. The action allowed of that traffic. (Was it Denied or Allowed based on NACL’s and SG)

VPC Flow Logs are not real time

* Capture window can be 10-15 minutes long depending on the volume of traffic
* Do not try to use Flow Logs to automate with real time needs
* To do real time automation you should use a packet analysis tool running on EC2 Instances themselves
* Keep in mind that using a Packet Analysis tool may violate the Amazon Acceptable Use Policy so be careful

Flow Log Traffic Not Logged:

* Traffic to and from the Amazon DNS
* Window instance activation traffic
* Communications to and from the meta data IP (169.254.169.254)
* DHCP traffic
* Traffic directly to and from the VPC Router

VPC Flow Logs should be turned on during the preparation phase of the 7 phases previously discussed. They are often most important at the investigation phase in which you make a timeline of when the incident occurred.

**CloudWatch Logs Agent for EC2**

**CloudWatch Agent-** An installed agent on your Instances or On-Premise servers that sends log data up to CloudWatch Logs.

* Allows you to collect additional in-guest metrics and logs from EC2 (and on-prem) servers.
* This includes memory, disk-user percentages, and swap file usage.
* It can also collect logs from the applications
* These metrics and logs are sent up to CloudWatch logs

**System Level Metrics-** Metrics available about the overall system

**Application Logs and Operating System Logs-** Metrics of your application and operating system

What AWS service can you use to store secure pieces of information such as username and password or config files that may have secure elements? **System Store Manager Parameter Store**

You can use metric filters on the log groups to add some automation based on events on your Instances.

These log events are not JSON formatted.

**CloudWatch Logs: DNS Query Logs**

**DNS Query Logs-** These can be enabled on Route 53 Hosted Zones and sent to CloudWatch Logs. Route 53 uses common DNS return codes in the logs and includes the edge location in the logs. These logs can be used to determine when there is a DNS problem in an application.

* These logs are only available for hosted zones where Route 53 is the endpoint
* No outside hosting and no private hosted zones

2 Concepts to understand how query logging works:

* The entity of who you use to register a domain and who holds authority of that domain. The company who can alter that domain.
* The company who manages the DNS resolvers (Name Servers) for the domain.
* They can be the same company, but the DNS resolvers are different from the company that manages the domain.

**In order to run Query logging, you do not need AWS to manage the domain, but you do need them to be the resolvers of that domain.**

**If you register a domain through Route 53, both are handled by AWS.**

It will have 2 log streams: The regular group and the edge location used for the DNS.

* These logs are not real time
* Split based on edge location
* They contain an overview of the general state of your DNS infrastructure

**S3 Access Logs**

**S3 Role in Storage Logging:**

* S3 is the default storage for CloudTrail
* CloudWatch logs can be exported to S3. It is always best practice to have a secondary place to store logs just in case for backup. Could be a separate account in which you export CloudWatch Logs into a bucket located in a safe account with limited access.
* S3 can help cost saving while still assisting with compliance
  + Lifecycle policies to reduce storage costs
  + Archive older logs to glacier

**S3 Access Logs-** An access logging mechanism in S3

* Tracks access requests to buckets
* Each log event contains once access request
* Log events contain
  + Requester, Time of request, Request action, Bucket name, Response, and Error code.

**Important Features of S3 Access Logging:**

* The log delivery group must be granted write permissions on the target bucket
* It is not near real time logging
* Logs are delivered on a “Best Effort” basis
  + Newly enabled access logs might not be displayed in the target bucket for up to an hour
  + Changes to the target bucket might take up to an hour to propagate

With CloudWatch logs they are stored in the same account in which the resources are taken from. This raises a potential security risk so it is best practice to export logs to an S3 bucket in a separate account and region for better durability and security.

If you need real time automation from a website you should use apache logs logged from an EC2 instance or on-prem server with a CloudWatch agent and stream those logs up to CloudWatch Logs. This is a lot closer to real time as opposed to S3 Access Logs.

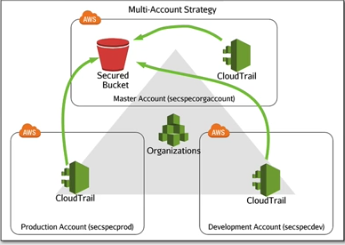
**Multi-Account: Centralized Logging**

**The Multi-Account Strategy:**

* Use AWS Organizations and set up accounts by environment or function
* Environment would be Prod, Test, Dev, Stage
* Function would be Security, Admin
* This will help reduce the blast radius of any incident

**Centralized Logging:**

* Logs should be contained in one location
* Logs should be read only for most job functions
* Logs should be encrypted by KMS preferably
* Roles can provide cross account access

****

You can use a more secure method of centralized logging in which you have 1 account for logging purposes only. This account may only allow 1-person access. You could also use log everything central and use Cross Region Replication on CloudTrail logs to that separate and secure account so if someone where to delete logs you would be able to see which logs where deleted. Best practice is to use KMS keys on your logs which will split up logging in encryption.

**2.4 Troubleshoot Logging Solutions**

**Troubleshoot Logging**

Why might CloudTrail not be logging to your S3 buckets?

* Check to make sure the Trail is enabled/on
* It may be configured incorrectly like wrong bucket name or check if you turned on data events
* There is a limit to 5 trails per region. An all-region trail counts as 1 per region.

Global service events come from Us-East-1 which is Virginia.

What if you are receiving duplicate global service events?

* You may have multiple trails for all regions which will duplicate global events
* Best practice is to only use 1 trail and apply it to all regions
* You can configure the CLI to include global service events or not include global service events

What if you are not receiving object level events from S3 or Lambda?

* Make sure you enable data events for S3 objects and Lambda functions
* If you have services that trigger off of these you need them enabled

**CloudWatch Logs Troubleshooting:**

**CloudTrail:**

* Check the last log delivery and make sure it is less than an hour
* Check the IAM Role and make sure it allows CloudTrail to CreateLogStream and PuLogEvents
* Check the CloudWatch Logs Console
* Check for the last event time is relatively recent
* Check the log group

**VPC Flow Logs:**

* Make sure Flow Logs are turned on
* Check the role and make sure that role allows CreateLogGroup, CaptureLogStream, DescribeLogGroup, DescribeLogStreams, and PutLogEvents
* Make sure the capture point is at an appropriate level
* Check the filter and make sure it is recording all traffic. The filter can record all traffic, only accepted traffic, or only rejected traffic

**Route 53 DNS Logging:**

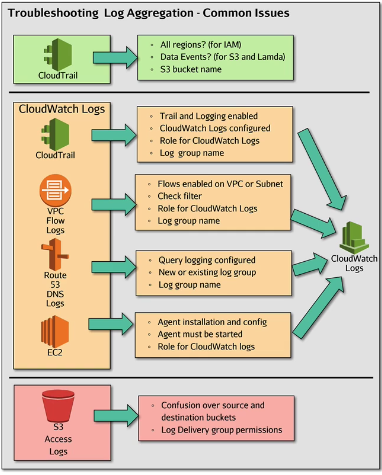
* Check role
* Check log group
* Make sure the domain is registered in Route 53. Can be registered from an outside domain as long as the name servers point to Route 53
* Name servers must be in Route 53 and make sure it is not apart of a private hosted zone

**EC2:**

* Make sure the agent is installed and configured correctly
* Make sure the agent is started
* Make sure the role on your EC2 instance is EC2CloudwatchRole

**S3 Access Logs:**

* Enable it on the bucket you want to be captured and point it to a target bucket that will store the logs
* The log delivery group needs to have write permissions
* Has to send to an S3 bucket
* First time and changes can take upwards of an hour to take effect
* Not real time



**Multi-Account: Troubleshoot Logging**

**CloudTrail Logging across multiple account:**

* Make sure the S3 bucket policy is set up correctly
* Double check the bucket names

**CloudWatch Logs across multi account:**

* CloudWatch does not send logs directly to another acc
* S3 access issues blocking exports
* Kinesis stream is not setup properly (only target for “Real Time” logs)

**Common issues with multi account logging:**

* Issues will mostly be around permissions to roles and resource policies
* Make sure all permissions only grant read only access to the account with the logs

**CloudTrail multi account:**

* S3 policy: To allow access for other accounts to log you need to allow that access in the resource section of the bucket policy
* KMS policy: To allow all account with the ability to generate a data key based on the same KMS key you need to allow GenerateDataKey by accounts in the Key policy.
* You may need to go into the sub account to determine what the problem is

**CloudWatch Logs multi account:**

* Implementations or multi account CloudWatch logs will usually use S3 exports or log subscriptions
* Bucket policy allowing the other account to write the logs
* Correct bucket naming
* Real time logging exports need subscriptions

**Common issues:**

* Start with checking permissions, roles, and resource policies
* Make sure all permissions only grant read only access to the account with the central logging

**2.1 Design and Implement Security, Monitoring, and Alerting**

**S3 Events**

**S3 Events-** A way to implement security, monitoring, and alerting on S3 in an automated way

* Allows for alerting on object actions in S3
* Can send notifications to 3 different services: SNS Topics, SQS Queues, and Lambda Functions
* Works at the object level
* Puts, copy, deletes generates an event and that event is delivered to a target

S3 events is an event driven system

* Near real time
* Can be part of a serverless architecture

2 types of events

* Object events (put/deletes to objects)
* System events (RRSObjectLost)

No cost for the events. The only cost is with the targets.

**Why might you use S3 events?**

* It is push not poll which makes it near real time
* Event driven push-based system
* You can use it to check permission on objects
* Based on specific S3 events like losing an object stored in RRS or OneZone storage

**CloudWatch Logs: Metric Filters and Custom Metrics**

**CloudWatch-** AWS monitoring, logging and alerting product

* A central hub for monitor logging

**CloudWatch Logs-** Configure any service to push to CloudWatch logs

**Components:**

**Metric Filters-** Used to create a custom metric from log data

* Assigned at the log group level
* Uses a filter in pattern syntax

**Metric Namespace-** The “Folder” or category the custom metric will appear in

**Metric Name-** The name given to the custom metric

**Alarms-** Assigned to the filter

* Alarms can trigger SNS topics, auto scaling groups, and EC2 actions if the metric relates

you can export log data to S3 or you can stream log data to Lambda and Elastisearch service

**Custom metric uses**

* Failed logins on instances
* Root user logins

**CloudWatch Events**

**CloudWatch Events-** Instead of configuring thresholds and alarms on metrics, CloudWatch events are monitoring event patterns.

* Near real time

**Consists of 3 parts:**

* **Event source-** An operational change in a service or a scheduled event
* **Rules-** Matching events to targets
* **Targets-** The service that will react to the event
  + There can be more than 1 target responding to an event
  + **Examples:** EC2, Lambda Functions, ECS Tasks, Kinesis data streams and Firehose, System manager run command, Code Projects, SNS, and SQS.
  + **Supports event driven security which is security that is proactive**

**Examples of uses:**

* Alerting object uploads in S3 that are public. (Can use Lambda to turn those object ACL’s back to private)
* Alerting on EC2 instance state changes. (Rules trigger actions on the instance)
* Alerting on user creation in IAM

**2 Types of Event Types:**

* Direct events made like EC2 state change
* Specific API calls with CloudTrail and camel case which takes longer than direct events

If you use the CloudTrail integrations there is a small delay.

If you use S3 and you want to use CloudWatch Events that trigger based on object level actions, you need CloudTrail turned on and logging data events for your S3 buckets.

CloudWatch Events is not limited to AWS.

You can also have fine grained policies for CloudWatch Rules. You can delegate a rule to a specific account or delegate a user to a specific rule.

**Limitations:**

* CloudWatch Events is a public service and needs internet access in order to work.
* If you want a private VPC using these Events you can allow access through VPC Endpoints.

**Multi-Account: CloudWatch Event Buses**

**CloudWatch Buses-** Allows different AWS accounts to share CloudWatch Events

* Newer feature
* Can collect events from all your accounts together into 1 account
* Must grant an account permission by adding and then sending the account number to the receiving configuration
* The sending account sends an event to an event bus target

**Centralized Events**

**Test Tips**

* Add explicit permissions by account IP
* If you do allow everybody you need to edit the rules in which the account responds too

Event bus needs to be in the same region as the rules sending the events.

You can send all rules to an account.

**AWS Config**

**Config-** A detailed view of the configuration of AWS resources like EC2, EBS, SG, VPC, ETC.

**You can:**

* Record what changed and who changed it
* Evaluate resource configurations for desired settings
* Get a snapshot of the current configurations associated with your account
* Retrieve configurations of resource in your account
* Retrieve historical configurations
* Receive notifications for creating, deletions, and modifications
* View relationships between resource (members of a security group)

**Uses of AWS Config:**

* Administrating resources: Notifications when a resource violates config rules
* Auditing and Compliance: Historical records of configurations are sometimes needed in auditing
* Configuration Management and Troubleshooting: Config changes on one resource might affect others. Can help you find issues quickly and restored last know configurations.
* Security Analysis: Allows for historical records of IAM policies. Allows for historical records of Security Groups.

**Configuration Recorder-** Records all info about resources and their configurations and it does this constantly when enabled.

* Tracks every change that occurred, to every resource in that region in that account

AWS config is enabled on a per region basis.

**Configuration Item-** A single record of the state of a particular resource at a particular time. Examples would be a state of a Security Group at 9AM. It stores the state and all details like what rules are allows, ports, and names.

**Configuration History-** A collection of config items. Every single config item for that resource since the recorder was enabled.

**Configuration Stream-** The SNS topic you can enable to send any changed to the infrastructure. You can use this to integrate with External tools.

**Config Rules-** A compliance feature of AWS config. You define a rule and this dictates how you want your account to be setup. Which ports are allowed to be opened?

You can define actions on no compliant rules such as CloudWatch Events rule triggered whenever a rule is marked as non-compliant, then CloudWatch Events can take some actions to auto remediate that non-compliant rule.

**Config Rule:**

* You can apply it to all resource
* Or by specific tags

**2 Ways it can be triggered:**

* It can be triggered on a periodic basis
* Or triggered by a change

You can force a reevaluate if you think your config recorder missed something. It can take some time to catch resource changes.

**AWS Inspector**

**Inspector-** A product that proactively identifies any suspicious activity within windows or Linux EC2 Instances.

**Allows for:**

* Analyzing behavior of your AWS resources
* Identifying potential security issues

**Target-** A collection of AWS resources

**Assessment Template-** Made up of security rules and produces a list of findings

**Assessment Run-** Applying the necessary template to a target

**Features:**

* Configuration scanning and activity monitoring engine
* Determines what a target looks like, it’s behavior, and any dependencies it may have
* Identifies security and compliance issues
* Built-in content library
* Rules and reports built in
* Best practices, compliance standards, vulnerability evaluations
* Detailed recommendations for resolving issues
* API automation
* Allows for security testing to be included in the Dev and Design stages

**Rule packages:**

Not every package is available for every instance type

**Common Vulnerabilities and Exposures-** Looks on instances and looks for unpatched vulnerabilities.

**Security Best Practices-** Determine whether your systems are configured securely.

**Runtime Behavior Analysis-** Looks for insecure protocols or unused TCP ports.

**CIS Security Config Benchmark-** Checks your instance based off the security config benchmark.

You can use inspector as an event source

**Config vs Inspector:**

* Config looks at everything at the product perspective
* Records changes to AWS resources overtime
* Point in time changes of products within AWS
* Inspector is a security tool that allows you toe valuate the configuration settings, packages, and security foot prints of AWS EC2 instances
* Inspect instances and generate repots of those instances and apps
* Can generate vulnerability assessment repots
* Recommendations

**2.2 Troubleshoot Security Monitoring and Alerting**

**Troubleshoot CloudWatch Events**

When troubleshooting CloudWatch Events you should start at the detection phase, then go to the alerting phase, and lastly troubleshoot the response phase. Essentially you should go through one phase at a time.

**Common Issues:**

* General configuration issues: Wrong resource name when “connecting” resources
* Typos: Check for typos with API calls, filter patterns, and Lambda functions
* You may be not waiting long enough after making changes or new configurations
* Roles do not have sufficient permissions. If this is wrong you will not get an error, it just won’t work.
* This includes targets and subscriptions to encrypted resources. You must provide permissions in the Key policy.

Remember that IAM API calls and global API calls come from and are supported by US-East-1 which is Virginia.

**Using Automation to Monitor Automation:**

* **CloudWatch Events:** You can create an alarm on event metrics based on the FailedInvocations API call which is based off of Lambda functions. This FailedInvocations means that Lambda was unable to perform its function.
* **Lambda Functions:** Lambda delivers logs to CloudWatch Logs and that will log errors. You can create and metric filter and sent it to an alarm that emails you on FailedInvocations or you can use CloudWatch Events and connect it to an SNS topic that will also notify you.

**Non-Triggering Rules:**

* **Scheduled Rules:** Can be Fixed time or based on a Cron expression. Check the syntax of the Cron expression.
* **Pattern Rules:** Check the JSON and make sure it is correct. Make sure the source and detail are correct.

**2 Types of Permissions:**

* IAM user permissions
* Rules need to invoke a target
  + Resource based policy
  + Make sure the edited policies are correct

**Multi-Account:**

* Diagnose permissions issues
* Event busses need appropriate permissions on the sending account

AWS Security Specialty Notes Linux Academy Domain 3

**Infrastructure Security**

**3.1 Design Edge Security on AWS**

**CloudFront**

**CloudFront-** AWS’s global CDN network.

* Global content delivery network operating from AWS edge locations
* It can use HTTP or HTTPS
* It removed many invalid HTTP request at the edge using a basic filter
* CloudFront supports SNI (Server Name Identifier)
* This allows edge location IP’s to be shared
* It also supports Dedicated IP SSL for all browsers, but cost extra
* For the most part using SNI is fine, the only time you need Dedicated IP is for older computers to access your websites.

**Viewer Protocol Policy-** How can someone view your CloudFront distributions. The policy between your viewer and the edge location. Can be configured 3 ways: HTTP and HTTPS (Default), HTTPS only, and HTTP redirect to HTTPS.

**Advanced Security Features:**

* Integrates with AWS WAF
* Supports full access control and signed URL’s and cookies
* Provides basic Whit/Backlists geo restrictions per distribution
* Integrates with 3rd party using signed URL’s and cookies
* Supports field level encryptions
* Supports Lambda at the edge

**Non-S3 Origin:**

* You can specify the ports
* You can use other ports if you have a bespoke configuration

**Origin SSL Protocols:**

* Restricts origin SSL protocols
* The default is TLS1.2, TLS1.1, and TLSv1
* you can optionally enable SSLv3

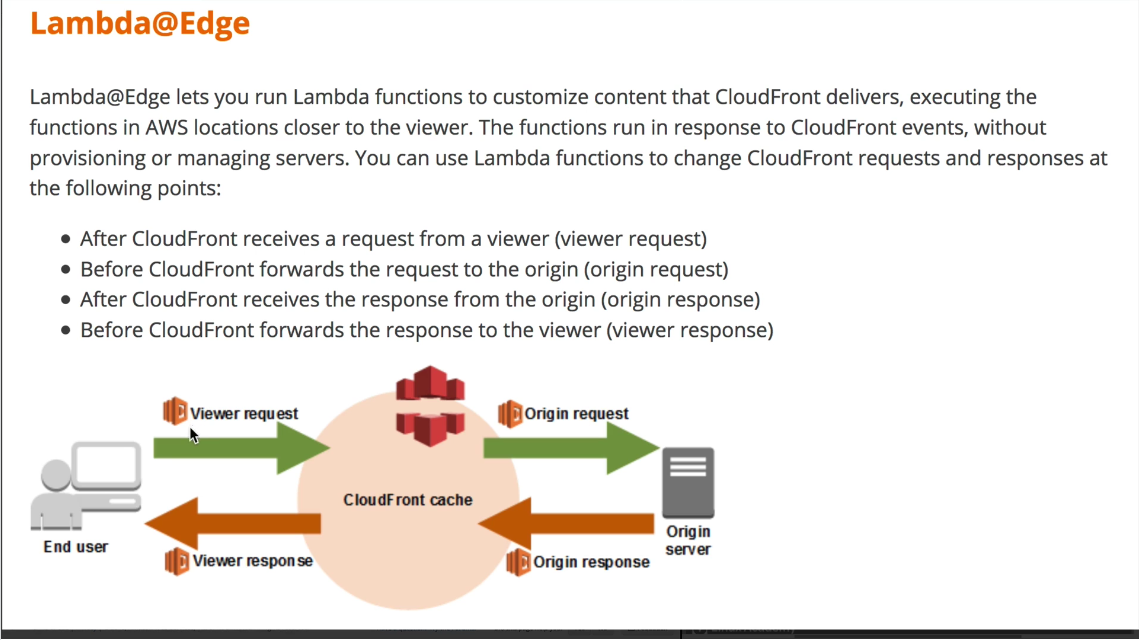
**How can you Restrict Viewer Access to the Distribution?**

* Use pre-signed URL’s
* Use signed cookies
  + By default, you have no signed cookies and signed cookies are on a per distribution basis

**Lambda Function Associations:**

**Lambda@Edge-** Allows you to run Lambda functions to customize content that CloudFront delivers, executing functions in AWS locations closer to the viewer.

* Perform compute operations as request come in or as they go out



**Why?**

* You can use it to inspect cookies to rewrite URL’s to different versions of a site
* Send different objects to your users based on the user-agent header, which contains information about the device that submitted the request
* Inspect headers or authorized tokens, inserting a corresponding header and allowing access control before forwarding a request to the origin
* Add, delete, modify headers, and rewrite the URL path to direct users to different objects in the cache
* Generate new HTTP responses to do things like redirect unauthenticated users to login pages, or create and deliver static webpages right from the edge

**Other Things You Should Know:**

* WAF intercepts requests before it gets to your CloudFront edge locations
* A custom origin needs to be on the SSL certificate for end to end encryption
* You cannot use self-signed certificates; your certificates need to be 3rd party trusted
* Custom SSL client is supported
* The default configuration uses SNI which is only supported for modern browsers
* All browsers requirements need to be dedicated IP which allows everyone to see your content

**Field Level Encryption-** Allows you to protect information end to end

* It encrypts any data sent from users or clients all the way through to your origin server
* This allows the choice to supply your own end to end encryption that CloudFront cannot view
* Only you can see it

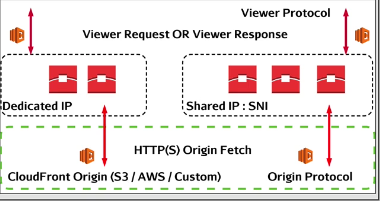
CloudFront supports logging to S3 buckets

**Origin Protocol Policy-** Allows you to select what protocol is used between a custom origin and the CloudFront distribution.

* 3 types, HTTP only, HTTPS, only, and Match Viewer

**Viewer Protocol Policy and Origin Protocol Policy Differences:**

* The Viewer protocol policy is between the internet and the edge location
* The origin protocol policy is between your edge locations and your origin
* With the origin protocol policy, you can specify ports numbers and you can use other ports beside 80 and 443 if you have a bespoke configuration

****

**Restricting S3 to CloudFront**

How do you restrict content to where it can only be accessed by CloudFront?

You need to crate an Origin Access Identity (OAI).

What is an OAI?

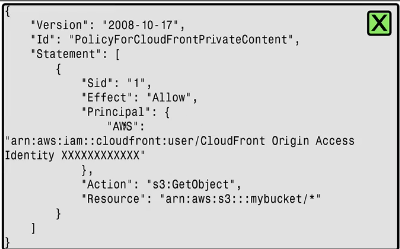
When accessing S3, CloudFront assumes this identity.

How is an OAI used?

* Public permissions are removed from your S3 bucket policy and permissions for the OAI to access your bucket are added
* Only the CloudFront using that OAI can access your S3 bucket

**Permissions on the bucket policy:**

The permissions on the bucket needs to only allow access to the bucket if gone through the single principle of the OAI being used.



**Grant Read Permissions on Bucket-** This option will automatically update the bucket policy; otherwise you will have to do it yourself.

**Signed URL’s and Cookies**

**Signed URL’s –** Allows an entity (generally and application) to create a URL which includes the necessary information to provide the holder of that URL with read/write access to an object even if they have no permission on that object.

**Cookies-** Allows access to an object type or area/folder and it does not need a specifically formatted URL.

**Features/Limits:**

* Signed URL’s/Cookies are linked to an existing identity (User/Role) and they have the permission of that entity
* They can have their own validity period; The default is 60 minutes
* They expire either at the end of the period or when the roles temp credentials expire
* Anyone can create a signed URL even if they do not have object permissions
* With CloudFront you define the account which can sign the key pair TrustedSigners which defines who can create cookies
* Signed cookies do not work with RTMP distributions

**Lecture:**

* When a singed URL is generated by an IAM user, that signed URL assumes the same access policy as the user who create the URL
* You can use Signed URL’s for online training courses that need paid subscriptions

**Test Tips:**

* You do not need any permission to generate pre-signed URL’s

How can a pre-signed URL generate an access denied message?

* It may be expired
* If it is not expired the entity that generated that URL may have not had appropriate permission to view the object or his permission changed after creating the URL
* The roles temp credentials may have expired

It is recommended for pre-signed URL’s to be generated by users.

**TrustedSigners:**

* Defined on a per distribution basis
* Once enabled it makes your distribution private
* It defines a list of accounts that it trusts to sign URL’s
* After your distribution is private a signed URL or cookies is required to access any objects provided by that distribution
* This only works if you use an OAI to deny bucket access unless it comes from the define OAI

RTMP can only use signed URL’s

**Deciding which one to use:**

* Pre-signed URL’s are for single objects
* Cookies allow you to grant access to a specific type of object; Maybe a JPEG, or you can give access to an area/folder for your distribution

**CloudFront Geo Restrictions**

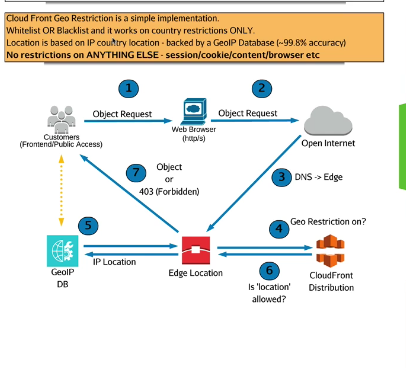
CloudFront can restrict content in one of 2 ways

1. Using CloudFront geo restrictions
2. You can also use a 3rd party geolocation service

**CloudFront Geo Restriction:**

* Has a simple implementation
* Has whitelists or blacklists and it only works for country restrictions
* Location is based on IP country location
* No restrictions on anything else

You cannot combine whitelists and blacklists



**3rd Party Geolocations:**

* To implement it needs a server or application and signed URL’s are used
* Use for extra accuracy
* Also use to apply additional restrictions other than just country
* You can apply Session, Browser, Account level restrictions
* You can also have more granular location restrictions
* City, Local, and longitude and latitude



**Exam Tips:**

* Anything beyond IP location blacklists or whitelists you need to use 3rd party geolocations
* Reasons to use 3rd party include, app account, security, browser, OS, State, Region, City, and Longitude, and Latitude.

**Forcing S3 Encryption**

S3 does not encrypt buckets, but instead S3 encrypts objects and the settings for that encryptions are defined at an object level.

* You can setup S3 default encryption on a bucket level. When you do this any objects uploaded without an encryption header are encrypted using the default settings
* You can also use bucket policies to deny put objects if no encryption is used

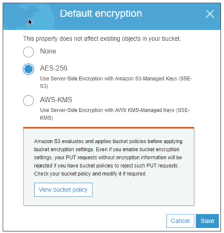
How can you force encryption or specify a type of encryption due to company standards?

You can add a bucket policy to force encryption that denies puts if the condition of StringNotEqual AES256 is not met.



You can also do a different bucket policy that does the same thing.



The new way of doing it is now you can specify the default type of encryption on a bucket by bucket basis. When an encryption method is not defined it will default to the specified type.

S3 evaluate and applies bucket policies before applying bucket encryption settings. Using both at the same time does not work due to the fact that the bucket policy will be read first and deny the Put operation before evaluating the bucket default encryption.

**Cross Region Replication Security**

Cross region replication is configured on a bucket level and provides an asynchronous replication of objects from are source bucket to one destination bucket in a different AWS region.

**Features and Limits:**

* Only replicates objects after enabling CRR (Cross Region Replication)
* Replicates unencrypted and SSE-S3 encrypted objects by default
* SSE-C is not supported
* SSE-KMS is supported only if specifically enabled
* By default, ownership and ACL’s are replicated and maintained, but CRR can adjust these
* The storage class is maintained by default
* Only customer actions are replicated (Human uploads or APP) lifecycle events are not replicated
* When the bucket owner has no permissions on the object, that object is not replicated

**Standard CRR:**

* Both buckets are in the same account
* You need to use an IAM role to get objects from the source bucket and replicate objects to the destination bucket

Replication is uni-directional which mean it is only from the source to the destination.

Only puts and deletes in the source bucket will be replicated.

It also replicates object permissions so if you make an object in your source bucket public it will become public in your destination bucket.

**Other Account CRR:**

Setting it up is essentially the same except you need to add replication permissions on the bucket policy of the destination bucket.

* That policy allows the IAM role to replicate objects
* Object owner by default will own any objects put into the destination bucket
* CRR allows the ability to owner change during replication
* You can also allow CRR to use KMS encryption to encrypt the objects outside of S3
* You need to change the IAM role to use the keys in KMS to encrypt the objects
* You also need to explicitly allow this in the replication config.

**Exam Tips:**

* Objects only replicate after CRR is enabled
* Uni-Directional
* You need an IAM role so that it can replicate objects from the source to the destination
* For cross account replication you need a bucket policy on the destination bucket
* If you are uploading objects using KMS you need to update the replication config, and the IAM role needs access to the necessary keys to encrypt the objects

**Web Application Firewall and AWS Shield**

**AWS Web Application Firewall-** Allows for conditions or rules to be set on CloudFront web traffic or an Application Load Balancer.

* It can look for cross-site-scripting attacks, specific IP addresses, the location of the requests, certain query strings, and SQL injections
* When multiple conditions exist in a rule, the results must include all conditions
* Example- A block request from a specific IP address that looks to contain SQL code for SQL injections
* Both conditions must match, the specific ip and it containing SQL code for it to block



**AWS Shield Standard-** The basic level of DDOS protection for you web applications. This is included with WAF with no additional costs to you.

**AWS Shield Advanced-** Expands services protected by DDOS attacks to include Elastic Load Balancers, CloudFront distribution, Route 53 Hosted Zones, and resources with EIP’s

**Advantages of AWS Shield Advanced Over Standard:**

* You can contact a 24 by 7 DDOS team for assistance during a DDOS attack
* It has cost protection against spike due to DDOS attacks
* Has expanded protection against many other types of attacks

For AWS Shield Advanced it costs 3,000 a month per org and you are charged for data transfer and usage fees. The basic AWS Shield is free.

**WAF:**

* Sits in between customers and your AWS CloudFront or Application Load Balancer
* It is a layer 7 firewall
* If used in CloudFront it is global
* If used with Application Load Balancers it is regional

**Conditions-** A basic match that can be based on different things. Locations, IP, presence of SQL code

**Rules-** Rules consists of multiple conditions and all conditions can be reused

**Web ACL’s-** Consists of multiple rules

**Rule Types:**

**Regular Rules-** Use to match specific traffic

**Rate Based Rules-** This one counts the number of matching traffic and after 5 matches in an hour or however much in however amount of time. Block if rule matched 3 times in an hour.

**3 Action Types of Rules:**

* Allow all traffic that match
* Block all traffic that matches
* Count any incidents of this rule being invoked
  + Use this when you want to initially evaluate the effectiveness of a rule

**Exam Tips When to Use WAF VS. Shield:**

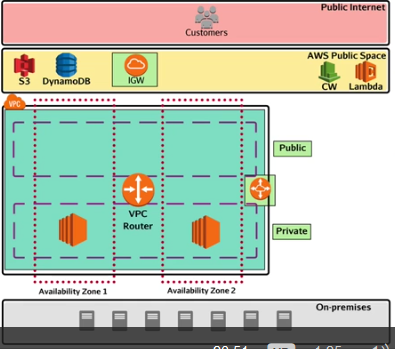
* WAF Functions as a product that can filter know specific traffic, gives you the flexibility on what types of traffic you want to block and where its coming from, as well as rate limit certain types of traffic, and can use monitoring only capabilities. (Count traffic that match)
* Shield is protection against network attacks
  + Protection products besides CloudFront and ALB’s
  + Real time response team
  + Money protection

**3.2 Design and Implement a Secure Network Infrastructure**

**VPC Design and Security**

VPC’s provide a separate security domain and limit the infrastructure layer blast radius.

They do not provide any restrictions for account level exploits.



This video is mostly a review about VPC’s so the one thing you should note is that AWS services are not on the internet. They actually live in-between your VPC and the internet in something called the immediate space. This is where the AWS Endpoints live.

**Security Groups**

Security groups are made at the VPC level and can only be assigned to resources that are in the same VPC as the security group.

Security groups are associated to the interface ID and that interface may be associated with an instance. Instances can have multiple interfaces and interfaces can have multiple security groups attached allowing for super granular permissions.

**Security Groups-** Applied to one or more network interfaces and evaluates traffic before the traffic is allowed ingress or egress.

* Stateful firewall
* Can only allow traffic
* Has an implicit deny
* Security groups can allow traffic from IP/CIDR ranges or from other security groups. Security groups can even reference themselves
* You can use the same security group for all applications to form a bespoke application allowing for every app to communicate only to each other

**VPC Peering**

* You can reference security groups across peers using the security group ID, but only if both peers are in the same region

**Traffics Journey:**

1. Comes from the internet or maybe somewhere else
2. Goes to the Internet Gateway
3. The internet gateway sends it to the VPC router
4. The router forwards it through the subnet to the instance
5. Then it arrives

**Filter Points Inbound:**

1. The instance has a public IP
2. Filters through the subnet which has a network access control list attached
3. Filters through the Security group

**Filter Points Outbound:**

1. Goes out the Security group
2. Goes out the subnet which has a NACL
3. Then goes out the IGW

**Network Access Control Lists (NACLs)**

**NACLs-** Can be associated with 0 or more subnets

* A VPC has a default NACL which is associated with any subnets that do not have an explicit alternate subnet association
* 1 NACL per subnet, but 1 NACL can be associated with many subnets

**Key Facts and Limitations:**

* NACLs are stateless
* NACLs are processed in a rule order, the lowest rule number will be processed first
* Has a default deny all traffic
* You can add explicit allows and denies, including an explicit default allow
* NACLs are processed when data enters or leaves a subnet
* NACLs can only work with IP/CIDR ranges
* NACLs can only be associated to the VPC in which they were created in

Processed in rule order. The lower numbered rule has a higher priority.

Deny overrides an allow. The rule processing stops once it finds a rule that either denies or allows the traffic.

When traffic goes from one subnet to another that traffic due to a NACLs statelessness will be processed 4 times.

You can control traffic to and from individual instances if they have static IP’s

**VPC Peering**

**VPC Peering-** A secure routed connection between 2 VPC’s; These VPC’s can be in the same account/same region, different account/same region, or same account/different region.

* Peers can be in different regions known as inter-region peers
* VPC peering opens your VPC to external access

**What is a peer?**

* A 1 to 1 connection between a source VPC and a destination VPC
* There is no transitive peering meaning if there is 1 VPC with 2 separate peers, the 2 VPC’s peered to the main 1 cannot talk to each other
* You also cannot create a peer with overlapping CIDR ranges

**Handshaking architecture:**

* The source VPC sends a request to the destination VPC and the destination VPC needs to accept that request

You can logically reference security groups of peers within the same region, but no inter-region peering.

When opening firewalls based on CIDR/IP addresses it is always more secure to allow access via one IP rather than a whole CIDR range.

**VPC Endpoints**

**VPC Endpoints-** Allows access to public AWS services, or services provided by 3rd parties without requiring an internet gateway to be attached to the VPC, or a NAT gateway.

* VPC endpoints are delivered in one of 2 forms; Interface endpoints, Gateway endpoint

**Gateway Endpoints:**

* Think of it more like a gateway
* It requires a route in the route table
* The destination is a prefix list. A prefix list is a logical endpoint for a given service in a given region
* The target is the gateway endpoint
* The supported services are DynamoDB and S3

**Gateway Endpoint Policy:**

* With gateway endpoints you can only restrict access to specific resources via a policy document
* You can use gateway endpoints to access buckets with update repositories stored in the bucket then only allow certain private instances access to those buckets via the bucket or the gateway policy
* Gateway endpoints are regional

**Key Features and Limitations:**

* Gateway endpoints cannot restrict via security groups or NACLs
* They do not work across regions
* Gateway endpoints only work with IPV4
* You cannot use Gateway Endpoints to connect to resources via direct connect, peering, or a VPN
* DNS resolution must be enabled inside the VPC for it to work

**Interface Endpoints:**

* They actually inject an interface into your VPC
* It is a single interface in a single AZ like a NAT Gateway. You can elect to put the endpoints in multiple AZs for fault tolerance purposes.
* **Private link-** allows for private traffic from AWS services and 3rd party resources into your VPC
* Has a single ENI which allows for security group attachment

**Key Features and Limitations:**

* You can connect to interface endpoints via direct connect and peering connections
* Only can be accessed by intra-region peering. You cannot connect to them via different region peering or VPN connections
* You can use interface endpoints to access non AWS 3rd party services via your VPC

**Serverless Security**

Lambda functions run in a temporary runtime environment, which is used for a limited number of function executions.

**Lambda Function Policy-** This policy controls who are what can invoke it

* Any event source which invokes functions like CloudWatch Events will need permissions to invoke functions via the functions policy
* For poll-based services (kinesis, DynamoDB, SQS) lambda polls on your behalf so permissions are gained via the execution policy
* For anything else, or for external entities/accounts, the push execution is used
* Changes to the function policy will be required for services that push through lambda
* Under normal circumstances this function policy is handled by AWS when you define an event-source
* Push services generally deliver the event to lambda, so extra permissions aren’t always needed

**IAM Execution Role-** The role assigned to the function when it executes

* Much like an EC2 role, it provides the runtime environment and function with the ability to interact with other AWS products via a temporary credential managed by STS
* This role ensures that it has enough permissions to log to CloudWatch logs
* It also makes sure that it can access and use any resources it needs to pull from or push too during the function
* For event-driven invocation, the execution role does not need permissions to access it
* For pull based sources it does

**Exam Tips:**

* By default, the lambda function policy gives the account that created the function the ability to invoke the function
* You can edit this function policy to grant other accounts the ability to run the function on your behalf
* You can modify it via the CLI
* Lambda invocation issues are usually down to the function policy

**Egress Only Internet Gateways**

**Egress Only Internet Gateway-** A features limited internet gateway, specifically for IPv6 traffic, and only allows outbound connection and return traffic. Think of it as a NAT gateway for IPv6 traffic.

* No incoming Traffic connections can be initiated to your VPC while using this type of gateway
* By default, IPv6 is publicly routable
* This gateway only works with IPv6 instances
* Stateful device

**Exam Tip:**

* You can’t do any DNS restrictions, IP restrictions, or authentication restrictions while using this gateway
* You need to restrict the subnets and the resources sending the traffic using NACLs and security groups

**Bastion Hosts**

**Bastion Host-** A bastion host is a security concept which allows the access of otherwise private resources via a hardened and secure public connection point; Generally, a virtual server or EC2 instance.

* Most commonly used when accessing an otherwise fully private VPC, or when a single publicly accessible management server is required to reduce management overhead.
* A bastion host has a public IP address allowing you to connect to isolated instance in private subnets
* They are super locked down
* Security group around the bastion hosts involves single rules that allows port 22 from a known CIDR block or for extra security a single IP
* Also, you can protect a bastion host with a NACL denying all other traffic and IP’s

**Increasing Security:**

* For increased security you can only allow connection to the private subnet from the security group in the public subnet
* You can also log failed SSH connections and use port forwarding
* You could install an intrusion detection/protection system

Configuring the bastion host to allow ID federation using direct connect or a VPN is not an option and the org needs the ability to connect into private instances using existing identities. How can you go about setting this up?

* Use a bastion host to authenticate against an external IDP
* You can install admin tools
* You can also have a VPN endpoint functioning as a bastion host

**3.3 Troubleshoot a Secure Network Infrastructure**

**Troubleshoot a VPC**

**Routing:**

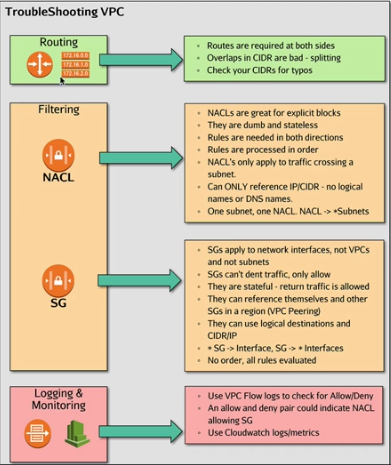
* You can configure routing from a specific IP address to limit traffic
* The most specific IP the more secure it is
* Make sure to configure routes on both sides of VPC peering so the source VPC and destination VPC
* Also remember you can’t peer form overlapping CIDR blocks
* Check for typos on your routes

**Filtering:**

* **NACL:** They product subnets and products that go into subnets like EC2 instances, NAT Gateways, and RDS Instances
* They can do explicit denies
* Stateless
* Rules are processed in a number order. Once a NACL finds a rule that matches it will not look at any more rules
* If 2 instances in the same subnet are communicating NACL rules will never take effect
* Can only use IP/CIDR blocks
* A single subnet can only have a single NACL, but one NACL can belong to multiple subnets
* **Security Groups:** Applied at the network interface level
* All rules are processed at the same time and security groups can reference themselves as long as everything is in the same region
* Security groups can be attached to interface endpoints

**Logging and Monitoring:**

* VPC Flow Logs can be used to check for allow and deny messages
* If you get both an allow and a deny it means you have both a NACL and a security group in place
* Use CloudWatch logs and metric to log everything

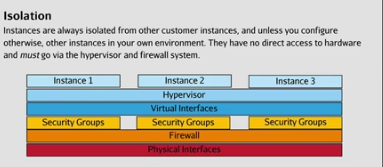


From a routing perspective, routes always use the most specific route in place A /32 route will be used over a /24 so keep that in mind for troubleshooting.

**3.4 Design and Implement Host-Based Security**

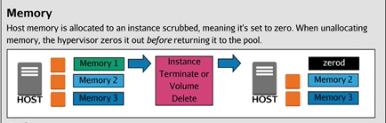
**AWS Host/Hypervisor Security (disk/memory)**

**Isolation-** Instances are always isolated from other customer instances, and unless you configure otherwise, instances in your own account. They have no direct access to hardware and must go via the hypervisor and firewall system.



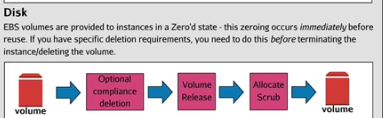
**Memory-** Host memory is allocated to an instance scrubbed. When unallocating memory, the hypervisor zeros it the memory out before returning it to the available pool for other customers.

* The memory gets scrubbed right after termination and does not get sent back into the pool till after
* Already 0 before use



**Disk-** Disk is provided to customers in a zerod state. The zeroing occurs immediately before reuse. If you have specific deletion requirements, you need to zero out your own disk before terminating the instance or deleting the volume.

* When you delete a volume, the physical space that the volume occupies is released back into the pool of available data
* When that storage is re allocated that is when the scrub is performed
* If you have any requirements for formal data deletion such as mist 180088 then you will need to perform some process of scrubbing yourself before deleting the volume
* You can wipe you own data before deleting it
* You can set volumes to not delete when an instance is terminated
* You can use data deletion tools that deletes unused volumes to automate scrubbing and deletion



**Host Proxy Servers**

**Host Proxy Servers-** In AWS filtering happens at 2 points, your security groups and your NACLs. These can filter based on protocols, ports, and IP/CIDR ranges. They cannot filter on DNS names, nor can they decide between allowing and denying traffic based on any form of authentication. If any of this is required you should use a proxy server or an enhanced NAT architecture.

* Proxy servers can be much more granular in allowing or denying traffic by using authentication or higher-level traffic analysis
* These can include DNS name, web path, user name, session state

**Typical Exam Question:**

You need to filter traffic by DNS name, whether an app is logged on or logged off, or whether it is using certain elements inside its session state. What should you use? **Host Proxy Server**

**Host-Based IDS/IPS**

**IDS-** Intrusion Detection System

**IPS-** Intrusion Protection System  
A host-based IDS/IPS compliments the features available within AWS such as examples below

**WAF-** Provides edge security before a threat arrives at your edge environment

**AWS Config-** Ensures a stable and compliant config of account level aspects

**SSM-** Ensures compute resources are compliant with patch levels

**Inspector-** Reviews resources for known exploits and questionable OS/Software config

**IDS Appliance-** Monitor and analysis data as it makes it into your platform

**Host Based IDS-** Handles everything else

**IDS-** Host, Instance, VM based solutions which monitors traffic and it actively looks for malicious or potentially malicious alerts

* Looks at system files for malware or OS level exploits
* Includes advanced behavioral analysis
* Looks for unusual traffic or process activity
* Can see errors, increased CPU usage, increased memory usage, odd networking patterns and more

IDS systems can run off box, but that limits their features to only networking analysis.

**IPS-** Supports a full set of protection resources

* They can block malicious traffic
* Analyze traffic and notify admins
* It allows you to automate mitigations

You can find these tools in the AWS marketplace

You can inject logs and metrics into CloudWatch using a host IDS.

* Off of those logs you can use lambda to generate automated functions
* SNS to notify elasti search to have a granular search of those log files

**Typical Exam Question:**

What should you use for a True Intrusion Detection System? **IDS/IPS**

**Systems Manager**

**Systems Manager-** A systems management product that provides insight, so information gathering and management, or action services to compute resources at scale. Its 2 core functional areas are insights and actions.

**Insights-** Gather information

**Actions-** Perform tasks

It provides a wide range of additional functions with which assist with these core abilities.

**Insights:**

Inventory and compliance

**Inventory-** Systems manager will periodically scan EC2 instances, or on-prem servers, retrieving details of installed applications, AWS components, network config, windows updates, detailed info on running services, windows roles, and optional custom data SSM can collect on your behalf.

**Compliance-** Allows that data to be compared against a baseline, providing a compliant or non-compliant state to a resource. Compliances uses state manager, SSM patching, and custom compliance types.

**Managed Instances-** Can be EC2 or on-prem servers and it becomes managed by installing the SSM agent and providing the instances with a role to deliver info to SSM.

The SSM agent is installed on most modern AMI’s of EC2

* Once managed it gives you the ability to attach a patch baseline to run regular patch assessments on your fleet of instances based off of the patch baseline
* SSM is enabled regionally

You can configure SSM to write all your data to an S3 bucket for each region and use centralized logging and prefixes to fit it into one bucket.

* **Once in the bucket you can use amazon Athena and amazon Quicksites to get a global overview of your state (Test Question)**

**Actions-** You can perform collections, run commands, control patching, and manage the general state of your managed instance fleet.

**Automations-** The role can be enabled in the automations and that role may have greater permissions then the user.

**Executing the Automation:**

* Give automations a lot of permissions
* Give a JR. Admin just the permission to run tasks using SSM
* This allows for role separation of security

**Run Command-** A simple method of running commands on managed instances at scale.

**Documents-** A unit of consumption for run command

Commands which can be executed on 1 or more instances

**Patch Manager-** Applies patches to managed instances

**State Manager-** A desired state engine

* You define the desired state in the form of a system manager document
* Command document or policy document
* A document is associated with one or more managed instances and a run command document defines a list of things to run, a state manager document defines the desired state
* Systems manager will od what is required to get those instances into the desired state
* It makes 100 percent of your instances conform to that state

**Shared Tooling:**

**Managed Instances-** A machine configured to use system manager

**Activations-** A method used to activate non EC2 instances within systems manager

**Documents-** Scripts or lists of commands that can run against managed instances

**Parameter Store-** A place to store config data and secrets. You can store KMS encrypted managed secrets.

**Limitations:**

The SSM store requires public space connectivity which means it needs some sort of gateway. For secure connectivity you can use and interface endpoint.

You can use Build-In insights with systems manager

**Personal Health Dashboard-** A filtered personalization indications of how the state of your infrastructure is.

**Packet Capture on EC2**

Packet capture or packet sniffing is a process where network traffic can be intercepted, analyzed, and logged.

Unlike VPC Flow Logs, sniffed packets can be inspected at a data level providing that the packets are unencrypted.

**Common Scenarios:**

* You need to review data flows between components to identify networking problems
* Packet sniffing supports IDS/IPS systems by helping them detect and remediate intrusion attempts
* Helps you debug connections between clients and the edge components of an environment
* Helps debug communication between tiers of your app
* Verifies the functionality of other network components such as firewalls, NAT’s and proxies

VPC Flow Logs meet a subset of the above scenarios, but don’t allow for traffic capture.

In AWS if you install a packet sniffer you can only see traffic local to that instance.

Compliments VPC Flow Logs. VPC Flow Logs offer the ability to see meta data about IP traffic and allows you to monitor an entire VPC.

AWS Security Specialty Notes Linux Academy Domain 4

**Identity and Access Management**

**4.1 Design and Implement a Scalable Authorization and Authentication System to Access AWS Resources**

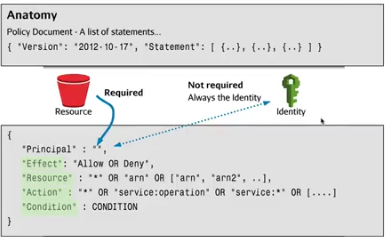
**IAM Policies**

**IAM Policies-** A policy document that defines if one or more actions, on one or more resources, with one or more conditions, is allowed or denied.

**2 Types:**

* **Identity Policies-** Attached to identities (Users, Groups, Roles)
* **Resource Policies-** Attached to resources

**Anatomy:**

****

With resource policies like bucket policies you always need a principle, but for identity policies you only need it sometimes.

**Principle-** The identity the policy applies too. This defines a bucket or other resources.

**Effect-** A single allows ore deny that applies to everything in that statement. Above shows 1 statement, but you can have multiple statements in 1 policy. Policies have an implicit deny and a deny overrules an allow always.

**Resources-** Defines which resources the statement applies too. A wildcard means everything, it can be a single ARN, or a list of ARN’s.

**Action-** What actions/operations is the policy allowing or denying for the given resource. Examples can be s3:GetObject, s3:Get\*

**Condition-** An optional line in policies that allow for additional control of weather the statement applies.

**Uses of Statements:**

These are just some common examples that may come up on the test.

* Restricting based on IP
* Restricting based on time
* Allowing access to only certain DynamoDB attributes
* Restricting based on username

**Users, Groups, and Roles**

**User-** A true identity within IAM.

* Has an ARN that can be referred to in the principle policies
* Has a friendly name
* Can be references in policies directly

**Users and Access Keys:**

* A user can have 2 access keys at once
* You can make access keys inactive, active again, and delete them
* It is always best practice to rotate access keys regularly
* You can only rotate manually
* Keys have their own set of records
* Use CloudTrail to understand when operations happen revolving around access keys
* If you make a pre-signed URL around an access key, if that key becomes inactive or gets deleted then the pre-signed URL expires

**Policy Types:**

* **Inline Policy-** A policy applied directly to a resource. A 1 to 1 relationship
* **AWS Managed Policy-** Pre-built policies that AWS makes and provides everyone with access too. Can change if AWS wants or feels the need to change them.
* **Customer Managed Policies-** Policies you create and can apply to multiple resources.

IAM has user limits

* Use external identities if user count gets too high in IAM
* You can use MFA with a user

**Access Advisor-** An overview of exactly what permissions that IAM user has access too and the last time that access was used.

**Credentials Report-** A list of all the account users and the state of their various credentials.

Always work with the lowest number of IAM users possible.

**Groups-** A container for multiple users.

* Groups can have many members and many users can be a part of many groups
* Groups cannot be references in policies
* Groups are not real identities

**Roles-** Assumed

* You can’t log into a role
* An app or an AWS service assumes a role

**Trust Policy-** The trust policy for roles defines who is allowed to assume that role

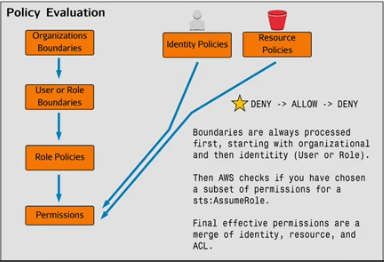
* If allowed you get a set of temporary credentials that expire
* You can use roles for organizational account switching
* This is how AWS services access other AWS services
* Gives a delegation of access
* Allows for cross account access
* You can chain role assumptions

**Revoking permissions:**

* You can change the permissions in the policy or you can use the revoke sessions button
* The revoke active sessions will revoke any active sessions using the role
* It does this by adding a condition that denies access to any resources based on the TokenIssueTime. If the temp token was issues before the time the revoke active sessions button was applied then the role will deny further access.

**Permission Boundaries and Policy Evaluation**

**Permission Policy-** A permission policy that allows or denies actions on resources.



**Permission Boundaries:**

* This applies a permission boundary that the user can’t go over
* It only restricts rights that are available
* If you set up the boundary to include S3 and SQS you still need to give that user access in their policy document
* It also is flipped, if you have permissions in the policy that are not in the permission boundary then that user will not be able to access it

**Permission Boundaries Apply to:**

Users, roles, and organizations. When applied to organizations you can limit the child accounts and the root user of those child account. This is the only time when you can limit a root user permission.

**Organizations and Service Control Policies**

**Organizations-** A multi-account management system

* Allows you to manage multiple account within an organization which can be structured into OUs
* Can be created directly or you can add existing accounts
* Offers centralized management of accounts, consolidated billing, and hierarchical control of accounts from a security perspective using service control policies.

**Service Control Policies-** A permission boundary applied to an AWS account within and organization.

**Organizations:**

* Starts with a single master account (The account being billed)
* This is the account where the root user can’t be restricted
* When creating an org you can choose just consolidated billing or all features
* All features include service control policies and role switching between accounts
* On the root account service control policies can first be applied and this will impact the whole org

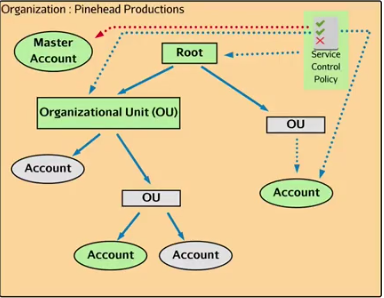
**Organizations Units-** Containers for accounts and other OUs

By default, all access to an account made in an org is accessed via role shift.

The second an account joins an org it gets effected by any policies on any OUs in its chain.

**Where can service control policies be put?** A service control policy is a permissions boundary that can be attached to accounts, OUs, or root accounts in an organization.

A service control policy does not affect the master account.



**Resource Policies: S3 Bucket Policies**

**S3 Bucket Policies-** These policies are attached to the bucket resource and controls who has access to a specific resource from the bucket perspective.

**Some common conditions:**

You want to restrict access to a specific web page. Use the condition key of aws:Refer and deny access if the StringNotLike the website.

You can also add a bucket policy that requires MFA by adding the aws:MultiFactorAuthAge policy.

You can restrict access to a bucket via IP using the aws:SourceIp condition and specifying an IP/CIDR range.

**Resource Policies: KMS Key Policies**

**Key Policies-** resource policies which control access to CMK’s that they are associated with.

* The root you by default has no access
* If you delete the policy you will get locked out of that key and the only way to restore that key is to contact AWS

**Key Admin-** An IAM user or role that has the ability to admin that KMS key. Key admins are permitted to perform admin actions on keys.

* Kms:Create, Kms:Describe, Kms:Enable, Kms:Put, Kms:Update, Kms:Delete, and more

**Key Usage-** An IAM user or role who has the ability to encrypt and decrypt data using that KMS key.

* Kms:Encrypt, Kms:Decrypt, Kms:Reencrypt, Kms:GenerateDataKey, Kms:DescribeKey.

**CMK-** Customer master key. By default, nobody in an account has CMK access.

* CMK’s generally are not used to encrypt data, they instead are used to generate a data key which perform the encryption and decryption. This process is known as envelope encryption.
* CMK’s can only encrypt 4KB of data so only use them to encrypt small files
* During envelope encryption there are 2 keys that are made, a data key in plaintext which is used to do the initial encryption and then is discarded, and then an encrypted version of the data key which is stored along with the encrypted data
* In order to implement cross account capabilities with KMS you need to edit the key policy

**Cross-Account Access to S3 Buckets and Objects**

There are 3 main ways to provide access to your S3 buckets from external AWS accounts

1. IAM Roles
2. Bucket Policies
3. Bucket ACL’s

**ACL’s-** An ACL is a legacy access method for S3 which allows and restricts access to a bucket or an object. They do not allow other advanced conditions.

* The owner of an object is the entity that uploaded that object
* If account B puts an object into a bucket in a different account, the entity that put the object are considered the owner of that object
  + Remember with Cross Region Replication, objects only replicate if the bucket owner also owns the object

Changing an ACL has to be done via the CLI.

**Bucket Policy-** You can use a bucket policy to allow a second account put access with a condition that only allows the put to happen if they bucket owner is granted full control over that uploaded object.

**Role-** Users of account B assume a role in account A which provides them with temporary access credentials. This makes them an identity for a limited amount of time and any puts to a bucket will give account A automatic ownership to that object.

**Identity Federation**

**Identity Federation-** Is where an AWS account is configured to allow external identities from an external identity provider access to your account.

**Supports 2 Types:**

* SAML 2.0 (For Active Directories)
* OpenID connect (Use for granting via social media)

**3 Ways:**

1. Web identity federation (Twitter, Google, ETC..)
2. SAMLE 2.0 federation
3. Custom ID Broker Federation (Used when SAML 2.0 is not available)

**Why Would You Not Want to Use IAM?**

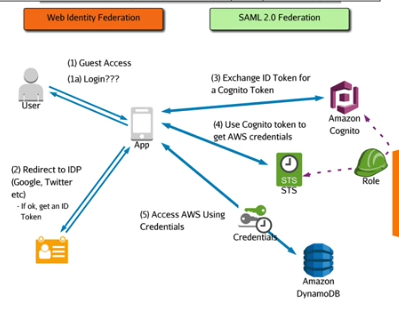
Maybe you have existing identities you want to use because you are migrating to the cloud, or maybe IAM is not an option due to the limited number of users aloud per account.

**Web Identity Federation:**

* This is commonly used with mobile apps
* This way offload users to an IDP and is web facing

**Steps in Web Identity Federation:**

1. You get a mobile app
2. When a user opens that app to log in, they can choose to login as guest or login via a web identity which will redirect to the IDP. If allowed you are granted an ID token
3. You are now redirect to Amazon Cognito and you exchange the ID token for a Cognito token
4. That Cognito Token is used to get the AWS credentials. These credentials give an IAM role for the authenticated user or gives a different limited role for a guest
5. Then you use these temp credentials to access the resources

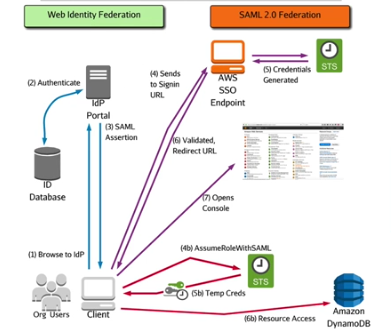


**SAML 2.0 Federation:**

* Use it when you own an ID provider
* Active directories
* Requires and ADFS server

**Steps in SAML 2.0 Federation:**

1. Your first initiate the connection by browsing to the IDP portal (ADFS)
2. You authenticate against the ID database (AD)
3. This returns a list of roles you can assume and you pick the role you want and this returns a SMAL assertion (ID token)
4. The client passes the SAML assertion to the AWS SSO endpoint
5. STS generates temporary credentials
6. The console opens or the CMD login opens
7. You call the AssumeRoleWithSAML API call and you get the Temp Credentials
8. You can then use the role to access resources that it allows



**When Would You Use Each?**

Web identity should be used for mobile development using large scale authentication, and when users are already likely to have a web identity.

SAML 2.0 should be use in an enterprise situation, and when you have an internal IDP or you have existing ID’s.

**Exam Tip:**

Users in an AWS account is limited to 5,000 but when you get near the few hundreds in users you should start considering using an identity federation.

**AWS Systems Manager Param Store**

**AWS Systems Manager Parameter Store-** Provides secure storage for configuration data and secrets. You can store values as plain text or you can encrypt it with KMS. Data is references using a unique name provided you have access granted from your permissions.

**Key Features:**

* The configuration and data are separated from the code
* Data is stored hierarchically
* Data is versioned, and access can be controlled and audited
* It integrates with many AWS services such as EC2, ECS, Lambda, and CodeBuild/Deploy
* Can be used for automated deployments like CloudFormation
* Serverless, resilient, and scalable

**4.2 Troubleshoot an Authorization and Authentication System to Access AWS Resources**

**Troubleshooting Permission Union (IAM//Resource//ACL)**

**Permission Union-** A situation where an identity accesses a resource and multiple permissions are in effect.

* IAM User policy
* Resource Policy
* ACL
* S3 and KMS

**Check Permission Boundaries-** This boundary can be on users, roles, or on child account of AWS organizations.

* Look if the service trusts the account
  + Service in an account by default trust it
  + Exception is KMS, make sure the key policy trusts the account you are in
  + Make sure the key policy is still there
  + If you are using an external account, make sure the account has been granted appropriate permissions

**Resource policy-** Specify whether an external account is trusted

**IAM Policy-** Check for denies

Check for groups that the identity might be in.

**Troubleshooting Cross-Account Roles**

IAM roles in one account are assumed by identities in another account.

There are 2 things you need to check, the trust policy, and the permission policies.

**Looking in the Account That Owns the Role:**

* Look at the trust policy, make sure it trusts the service and the account trying to use that role

**Determine:**

* Can I assume the role? If not check the trust policy.
* If you can assume the role you may just have wrong permissions. Check the permission policy on the role.

**Can I Assume the Role?**

* Use CloudTrail to check if you are getting AssumeRole requests.
* If you are not getting AssumeRole request than there is a problem with account B
* If you do get AssumeRole requests that fail than look at the trust policy
* Pay attention to conditions

**Account B:**

* Make sure the account can assume the role
* IAM users need the ability to do the STS:AssumeRole API in their permissions policies
* Verify that the identity is the one you are expecting

**Condition:**

You can use the Sts:ExternalID condition which acts as an MFA. If you use the StringEquals with that condition you can present a password that a user has to input in order to assume the role. This stops 3rd accounts from assuming a role into account B, and then assuming a role into your account. You put this condition in the trust policy.

Once you have determined someone can assume the role, then it becomes a permissions policy issue.

**Troubleshooting Identity Federation**

Web Identity Federation uses the AssumeRoleWithWebIdentity API call

SAML uses the AssumeRoleWithSAML

Regular Roles use the Assume Role API call

The name of the API call matches the federation being uses

**Troubleshooting KMS CMK’s**

KMS is architected to separate encryption keys, from the products or services that use them.

Remember the role separation between Key Admins and Key Users

* There is no trust between a CMK and the account
* The permission can be removed, resulting in an unusable key and you will need to contact AWS to reset that policy

CMK’s are generally not used to encrypt data and they can only encrypt data smaller then 4KB

You use CMK’s to generate, encrypt, and decrypt data keys.

**Problems with CMK’s:**

* It is not always the CMK
* Do you have permissions to decrypt the data keys?

**Rate Limit:**

* There is a per second limit for AWS KMS API operations
* For decryption it is 5,500
* For encryption it is 10,000, but only in 3 regions, Virginia, Oregon, and Ireland

What should you do if you have an application that needs to do a lot of encryption or decryption?

AWS Security Specialty Notes Linux Academy Domain 5

**5.1 Design and Implement Key Management and Use**

**Key Management System (KMS)**

**KMS-** The key management service uses FIPS 140-2 compliant hardware modules to manage access to key material.

* Integrates fully with CloudTrail and IAM for permissions and auditing
* Can be used with most services that support encryption

The primary function of KMS is to generate and manage CMK’s

**Customer Master Key-** A logical representation of a key

* Keys can be generated by KMS or imported
* CMKs never leave KMS and never leave a region
* CMKs can encrypt or decrypt data up to 4KB in size
* They are FIPS 140-2 level 2 compliant
* Controlled by resource and identity policies
* By default, no one has access to a CMK
* By default, the key policy will trust the account

**KeyID-** A unique identifier

**Key Manager-** AWS managed keys or customer managed keys

**Alias-** A friendly name that you can point to a specific key ID

* You can have the same alias for different keys if they are located in different regions
* An alias points to a specific key in that region

**CMKs:**

* CMKs generate a data encryption key by using the GenerateDataKey API call
* This creates 2 keys a plaintext key, and an encrypted data key
* The plaintext key is used for the initial encryption and then discarded
* The encrypted key is stored along with the encrypted data key
* This process is known as envelope encryption
* KMS is used to decrypt the encrypted key returning plaintext, and data is decrypted

A CMK is a logical entity and is a wrapper for one or more backing keys.

**Backing Keys-** A physical cryptographic material used to encrypt and decrypt data

* Each CMK has a current backing key
* Backing keys stay linked to the same CMK
* CMKs know what backing key is needed to decrypt objects

**Why would you want to generate your own key material?**

* Governance and auditing
* Prove you generated the material with randomness
* Set expiration time and to manually delete immediately, but also have the option for future use
* Use key material from your own infrastructure
* Own an original copy of the key material

**Rotation of Keys Automatically:**

* AWS managed keys have rotation enabled by default and that happens every 1095 days or 3 years
* Customer managed keys do not have rotation enabled by default, once enabled rotation occurs every 1 year
* For imported key material you need to do manual key rotation which involves you creating a new KMS key and changing the alias to point at the new key id

**ReEncrypt API:**

This passes KMS a particular piece of encrypted data then requests KMS to ReEncrypt that data using a different key.

With this you can give admins the ability to ReEncrypt data without giving them decrypt abilities.

They can’t see the data.

**Encryption Context-** Sets of keys and values that are tied to the encryption process

* This adds an extra layer of protection
* Example would be an object stored in S3 is encrypted and you use the encryption context with that bucket
* Moving the files will deny decryption
* Movement or changes to data KMS will refuse decryption
* Key value pairs pass along with the encryption or decryption requests

**Grants-** Another way to control key permissions

* Think of it like a pre-signed URL for KMS
* Grants an entity the ability to work with a key

**Deleting Keys:**

* Schedule key deletion- default is 30 days but can be as low as 7
* After deletion CMKs are non-retrievable
* Any data encrypted cannot be retrievable
* Exceptions include if you use your own key material you don’t need to schedule key deletion and you can reupload the key material and reuse it

**KMS in a Multi-Account Configuration**

CMKs can be configured to allow other accounts to use them.

* The key wont appears in the external account
* If configured in the key policy, that account can interact with the key for cryptographic functions

To setup you need to add the account ARN in the key policy under the principle section for key usage.

**External Accounts:**

* The external accounts also need to grant key permissions in the IAM policies of their users/roles using the keys.

**Why?**

* You may use it for cross account logging using KMS to separate encryption completely using an isolated security account
* Isolated key account that only holds your CMKs, other accounts can use your CMKs, but you can be sure that you maintain control of the keys

**CloudHSM**

**CloudHSM-** Is a dedicated HSM which runs within your VPC and accessible only to you.

* It uses industry standard APIs as opposed to AWS APIs
* Keys can be transferred between CloudHSM and other hardware solutions
* Apps outside the VPC can talk to CloudHSM
* CloudHSM complies with FIPS 140-2 level 3 standards

**5.2 Troubleshoot Key Management**

**Troubleshooting KMS Permissions**

Permissions in KMS are centered around CMKs

* The default policy trusts the account in which the key was created within, and this trust can be provided to IAM users via identity policies or the key policy itself

CMKs are a logical entity that represents the physical backing keys

You can lock yourself out of a CMK by deleting the key policy, and in order to get it back you need to contact the AWS support team.

To give yourself cross account access to CMKs you need to add that account ID in the key policy, then apply identity policies to allow key usage in the account that the key is not in,

**Key Admin Operations:**

* Create, Describe, Enable, List, Put, Update, Revoke, Disable, Get, Delete, Tag, UnTag, Schedule key deletion, Cancel key deletion

**Key Usage Operations:**

* Encrypt, Decrypt, ReEncrypt, GenerateDataKey, DescribeKey, CreateGrant, ListGrants, RevokeGrants

**KMS Limits**

3 Limits

1. Simple limits
2. Rate limits
3. Cross account limits

**Simple Limits:**

* 1,000 CMKs per region in any state
* 1,100 Aliases per account
* 2,500 grants per CMK

Breaking the shared, or per operation limits result in KMS throttling the requests.

Cross account applies to the account that is the source. This means if you have 1 isolated account for keys, your limits will not add up.

There is a 5,500 shared API limit

* The default is 5,500, but for select regions it gets up to 10,000. Those regions are Virginia, Ireland, and Oregon
* This applies to key usage operations

You can contact AWS support to raise your rate limits

**Rate Limits-** The number of operations an API can be called per second

* 5,500 shared API limit for key usage operations such as decrypt and encrypt
* For some regions this is 10,000 a single second
* If you exceed these limits you will get a ThrottlingException error
* You will get the error in the account using the API’s
* In a single account scenario obviously that error will be in the one account
* In a multi-account situation if the keys are stored in account A, and an app is using keys in account B, then you will get the ThrottlingException error in account B

**5.3 Design and Implement a Data Encryption Solution for Data at Rest and Data in Transit**

**Data at Rest: KMS**

KMS integrates with many other AWS products to provide encryption services and key management.

The way it integrates is service specific.

As soon as KMS generates a data key and hands it over to a service, it does not manage that data key anymore.

**Data at Rest: S3 Client-Side Encryption Options**

**SSE-C-** This is where S3 handles the cryptographic operations, but does so with keys that you as the customer manage and supply with every object operation.

**Why?**

* Governance and security policy perspective
* Maintain key internally
* You own an HSM appliance

**Data in Transit: ACM**

**ACM-** A managed service providing x509v3 ssl/tls certificates

* Asymmetric
* Generates certificates that are valid for 13 months

**Key Features:**

* Native integration with ELBs, CloudFront, ElasticBeanstalk, and API Gateway
* No cost
* Auto renewed when actively used within supported services
* Integrates with Route 53 to perform DNS checks as part of certificate issuing process
* Regional
* KMS is used. Certs are never stored unencrypted

**Encryption SDKs**

**Encryption SDKs-** A software development kit encryption library that makes it easier for you to implement KMS

* Used to interact with KMS and different HSMs
* Allows for data key caching
* Bypass KMS API limits

**Compliance Examples**

**Compliance-** You as an org follow certain standards

**AWS Artifact-** Features a comprehensive list of access-controlled documents relevant to compliance and security in the AWS cloud.

How do you gain access to the documents that prove compliance frameworks from AWS perspective? AWS Artifact

* Proofs of compliance
* Explicitly acknowledge account or org agreements for particular compliance frameworks

AWS Security Section 2 Notes

CIA is a security model in IT that stands for confidentiality, integrity, and availability.



Confidentiality = privacy

* Think about health care data. You’ve had a DNA test done that measures your disease risk.
* You want to keep this data private to yourself, but you might want to share it to your wife and children.
* Does not have to be absolutely secret to yourself.
* Data that you want to keep confidential, but you want to expose to 3rd parties where needed.
* Examples= Health, a bank statement

How to ensure confidentiality

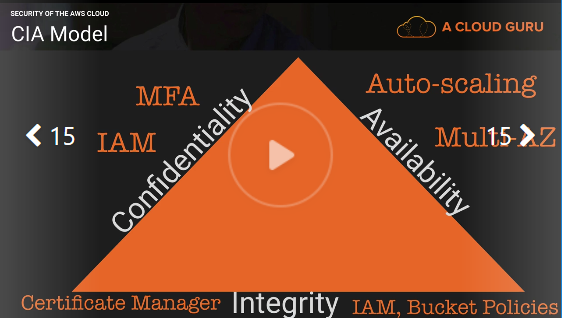
* Data encryption- encrypting data at REST and in Transit
* User ID’s and passwords
* 2 Factor Authentication
* AWS services- IAM, MFA, Bucket policies, ACL’s, Security Groups, encryption

Integrity = maintaining consistency, accuracy, and trustworthiness of your data over its entire lifecycle. Data can’t change in transit and ensure that data can not be altered by unauthorized people.

* File permissions
* User access controls
* Version control
* Example= Checksum- when you download a piece of software it comes with a Checksum. You can check the hash of that software with the checksum and if they match that means your data is integral.
* AWS services- amazon certificate manager, IAM, Bucket Policies, encryption, versioning, MFA delete

Availability= Keeping your systems available

* Redundancy
* Raided disks, HA clusters, multi AZ, multi regions, and design for failure
* AWS services- Auto-Scaling, Multi-AZ, Multi-Regions, Route 53 with health checks



AAA- Extends and compliments the CIA model

Authentication- When you log in to the AWS console the first thing you do is enter your user name and password. This is authenticated against IAM and checks whether there is a user with that user name and password.

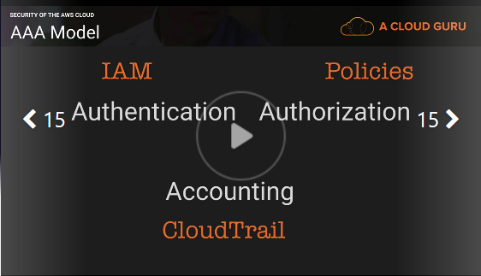
IAM

Authorization- The ability to use the console depends on how much permissions that you have.

Policies

Accounting- What is it that you are doing on the AWS platform?

Cloudtrail



The Security of AWS

Physical and environmental security

* Fire detection and suppression
* Power
* Climate and temperature
* Management
* Storage device decommissioning

Business Continuity Management

* Availability
* Incident response
* Companywide executive review
* communication

Network Security

* Secure Network Architecture
* Secure Access Points
* Transmission protection
* Amazon corporate segregation
* Fault tolerant design
* Network monitoring and protection

AWS access

* Account review and audit
* Background checks
* Credential policy

Secure design principles

* AWS development process follows secure software development best practices

Change management

* Software
* Infrastructure

Security of the Cloud- AWS responsibility

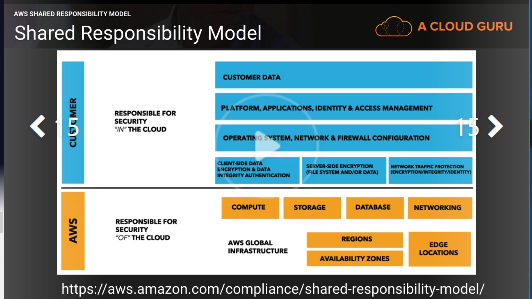
Security in the Cloud- your responsibility

AWS responsibility

* Global infrastructure
* Hardware, software, networking, and facilities
* “managed services”
* Hyper visors

Customer security responsibilities

* Infrastructure as a services (IAAS)
* Including updates and security patches
* Configuration of the AWS provided firewalls



The model changes for different service types:

* Infrastructure
* Container
* Abstracted

Infrastructure: This category includes compute services, such as amazon EC2, EBS, auto scaling, and amazon VPC. With these services, you can architect and build a cloud infrastructure using technologies similar to and largely compatible with on-premises solutions. You control the operating system, and you configure and operate any identity management system that provides access to the user layer of the virtualization stack.

EC2 you are responsible for;

* AMI’s
* Operating systems
* Applications
* Data in transit
* Data at rest
* Data stores
* Credentials
* Policies and configuration

Container services: Services in this category typically run on separate Amazon Ec2 or other infrastructure instances, but sometimes you don’t manage the operating system of the platform layer. AWS provides a managed service for these application “containers”. You are responsible for setting up and managing network controls, such as firewall rules, and for managing platform-level identity and access management separately from IAM. Examples of container services include RDS, EMR, and Elastic Beanstalk.

Abstracted Services: This category includes high-level storage, database, and messaging services, such as S3, Glacier, DynamoDB, SQS, SES. These services abstract the platform or management layer on which you can build and operate cloud applications. You access the endpoints of these abstracted services using AWS APIs, and AWS manages the underlying service components or the operating system on which they reside.

Security in AWS

Cloud controls

* Visibility
* Auditability
* Controllability
* Agility
* Automation
* Scale

Visibility

What assets do you have? (AWS Config)

Auditability

Do we comply with policies and regulations? (AWS CloudTrail)

Controllability

Is my data controlled? (AWS KMS and AWS CloudHSM)

Agility

How quickly can we adapt to changes? (AWS CloudFormation and AWS Elastic Beanstalk)

Automation

Are our processes repeatable? (AWS CodeDeploy and AWS OpsWorks)

Scale

In AWS the scale is on your side!

Services that help with all control

AWS IAM AWS CloudWatch AWS Trusted Advisor

Summery

Non-repudiation- you can’t deny that you did something.

* Cloudwatch
* Cloudtrail
* IAM
* MFA

Why should we trust AWS?

Compliance programs

* Pci DSS
* ISO 27001
* HIPPA

AWS Security Specialty Section 3 Notes

IAM allows you to manage users and their level of access to the AWS Console.

What does IAM give you?

* Centralized control of your AWS account
* Shared access to your AWS account
* Granular permissions
* Identity federation
* MFA
* Provides temporary access for users/devices and services where necessary
* Allows you to set up your own password rotation policy
* Integrates with many different AWS services
* Supports PCI DSS compliance

**Users-** End users. (people)

**Groups-** A collection of users under one set of permissions

**Roles-** you create roles and can then assign them to AWS resources

**policies-** a document that defines one or more permissions

IAM is global

Root user scenario

1. Change the password
2. Delete root keys
3. Disable and reenable MFA
4. Evaluate existing users

IAM policies overview

**IAM policies**- sepecify what you are allowed to do with any AWS resources. They are global and apply to all areas of AWS. You can attach IAM policies to IAM users, groups, or roles. These users, groups, and roles are then subject to the permissions you define in the policy. In other words, IAM policies define what a principle can do in your environment.

Types of policies

* AWS Managed policies
* Customer managed policies
* Inline Policies

AWS Managed policies

* An AWS managed policy is a standalone policy that is created and administered by AWS
* Literally 100,000 or even millions of AWS accounts use these policies. They are the same applied across multiple accounts.
* They can change, but AWS is very careful as a small change could impact a lot of people as everyone has access to these policies.

Customer Managed policies

* Standalone policies that you admin in your own AWS account
* You can then attach the policies to multiple principle entities in your AWS account.
* When you attach a policy to a principle entity, you give the entity the permissions that are defined in the policy

Inline policy

* Are useful if you want to maintain a strict one-to-one relationship between a policy and the principle entity that it’s applied to
* Use it if you want to be sure that permissions in a policy are not inadvertently assigned to a principle entity other than the one they are intended for

What is the difference between the Root user, an Admin user, and a Power user?

* Root user has access to everything
* Admin has access to everything except billing
* Power user has access to everything except billing and IAM

Bucket policies

**Bucket policies-** S3 bucket policies are attached only to S3 buckets. S3 bucket policies specify what action are allowed or denied on the bucket. They can be broken down to a user level, so Alice can PUT but not DELETE and John can READ but not PUT. Bucket level only, S3 only.

Use cases

* You want a simple way to grant cross-account access to your S3 environment, without using IAM roles.
* Your IAM policies bump up against the size limit. (2kb for users, 5kb for groups, 10kb for roles). S3 supports bucket policies of up to 20kb.
* You prefer to keep access control policies in the S3 environment

A Bucket policy overwrites an IAM policy unless the IAM policy has an explicit deny. For example if we had a bucket policy allowing access and an IAM policy with nothing in it that user would be able to do everything the bucket policy was allowing.

At the end of the resource line stating your bucket you need to add a / followed by either a \* for all objects or followed by an object name. This is not talked about in the video so here is an example from my account.



I have 2 separate statements in a bucket policy highlighted by the red squares.

On the first statement the resource has the /\* which means allow access to every action stated for every file in the bucket.

The second statement has a deny effect on a specified file in my bucket called index.txt

This means everything is allowed in my bucket except for on that object in which I placed a deny.

**Principles-** when making a bucket you are asked to choose a principle that it effects and these are often IAM users, but can also be specific instances or DynamoDB tables and so on.

**Resources-** when using the policy generator your also required to choose a resource. This can be your overall bucket if you add the /\* or it can be an exact path name in which you want the statement to affect.

A bucket policy can extend extra access to IAM users who have no denies.

A bucket policy can also overwrite a root user

**Explicit deny-** an explicit deny is a deny in any policy and it overwrites all allows.

All policies are at a default deny state.

S3 ACL

**S3 ACL-** are a legacy access control mechanism that predates IAM. AWS recommend sticking to IAM policies and S3 bucket policies. However, if you need to apply policies on the objects themselves, then use S3 ACL’s. Bucket policies can only be applied at the bucket level where as S3 ACL’s can be applied to individual files.

Use cases

* If you need fine grained permissions on individual files/objects within S3
* Bucket policies are limited to 20kb in size, so consider using S3 ACL’s if you find that your bucket policy grows too large.

Setting up ACL’s

* You can go into a specific file and do basic grant read access/write access permissions
* If you want to do it at a user level you need to do it via the CLI or the API
* You need your AWS account ID and an owner canonical user id

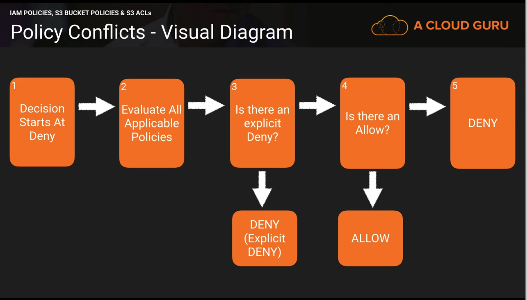
Confliction policies

Whenever an AWS principle (user, group, or role) issues a request to S3, the authorization decision depends on the union of all the IAM policies, S3 bucket policies, and S3 ACL’s that apply.

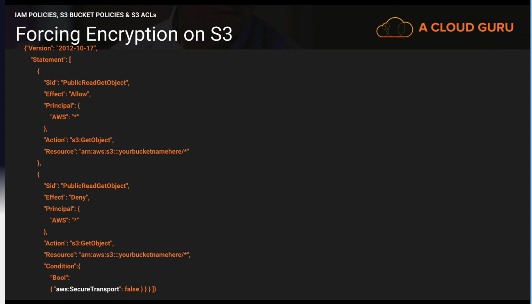
**Least privilege-** decisions in AWS always default to a deny meaning that the only way a user can do something on your account is because somewhere someone has given that user access to do so.

An explicit Deny always trumps an allow

Only if no method specifies a Deny and one or more methods specify an Allows will the request be allowed.



Forcing Encryption on S3



You need to have 2 statement. The first statement will allows access to get the object from S3 from the bucket you want.

The second statement will deny access only if the condition of Bool SecureTransport is false.

Cross Region Replication

**Cross Region Replication-** cross region replication replicates objects from one region to another

* By default, security in transit is used
* You can only replicate objects from a source bucket to only one destination bucket

Cross region replication requirements

* Versioning must be enables
* Both buckets must be in different AWS regions
* Amazon S3 must have permissions to replicate objects from that source bucket to the destination bucket on your behalf. Will create a role for you
* If the source bucket owner also owns the object, the bucket owner has full permissions to replicate the object. If not the object owner must grant the bucket owner the READ and READ\_ACP permissions via the object ACL

Cross region replications across accounts

* The IAM role must have permissions to replicate objects in the destination bucket
* In the replication configuration, you can optionally direct S3 to change the ownership of object replica to the AWS account that owns the destination bucket

What is replicated?

* Any new objects created after you add a replication configuration
* Unencrypted objects and SSE-S3 encrypted objects
* SSE-KMS encrypted objects if you grant permission
* Object metadata
* Any object ACL updates
* Any object tags
* Amazon S3 replicates only objects in the source bucket for which the bucket owner has permissions to read objects and read access control lists.

What is replicated? Deletes

* If you just use a delete marker, then the delete marker is replicates

What is not replicated?

* Anything create before CRR is turned on
* Objects create with SSE-C
* Objects with SSE-KMS unless you enable it
* Objects in the source bucket for which the bucket owner does not have permissions
* Deletes to a particular version of an object

Forcing S3 to use CloudFront

**Origin Domain Name-** The source from where your distribution is getting its content

**Origin Path-** any sub folder you want inside your bucket

**Restrict bucket access-** restricts users from accessing your S3 bucket through S3 URL’s and forces them to use CloudFront URL’s only.

Steps to restrict bucket access after a distribution has been created

1. Click on the distribution and click distribution settings
2. Click on origins
3. Click origin and click edit
4. Change and restrict bucket access
5. Create or choose and origin access identity
6. Click grant permissions on bucket
7. Save

**Origin access identity-** a special user that forces people to use cloudfront URL’s to access your amazon S3 content

**Grant read permissions-** grants read permissions to your bucket automatically so that you do not need to manually change the bucket policy.

S3 Presigned URL’s

**Presigned URL’s-** typically done using the SDK’s, but you can also do it via the command line. Allows you to share objects inside S3 without changing permissions and allows for good confidentiality.

The command to do this is

aws s3 presign s3://bucketname/object

The default time for presigned URL’s is 1 hour. The max is 7 days.

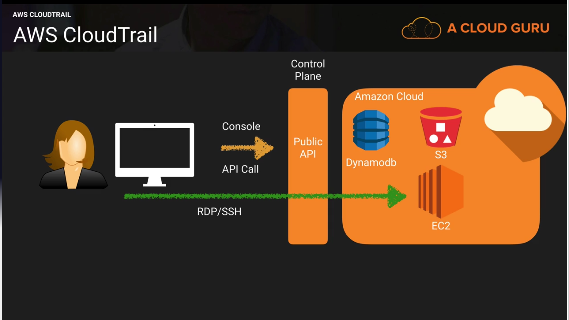
You can change the time by doing the –expires-in followed by the time you want in seconds. This looks like this.

aws s3 presign s3://bucketname/object –expires-in 300

this would change it to make it expire after 5 minutes.

AWS Security Specialty Section 4 Notes and Elaborations

CloudTrail- a web service that records AWS API calls for your account and delivers log files to you



Enables:

* After the fact incident investigation
* Near real-time intrusion detection
* Industry and regulatory compliance

Provides:

* Logs API call details (for supported service)

What is logged?

* Metadata around the API calls
* The identity of the API caller
* The time of the API call
* The source IP address of the API caller
* The request parameters
* The response elements returned by the service

CloudTrail Event Logs:

* Sent to an S3 bucket
* You manage the retention in S3 (life cycle management policies)
* Delivered every 5 (active) minutes with up to 15 minute delay
* Notifications available
* Can be aggregated across regions and across multiple accounts (use CRR for best practice)

Setup:

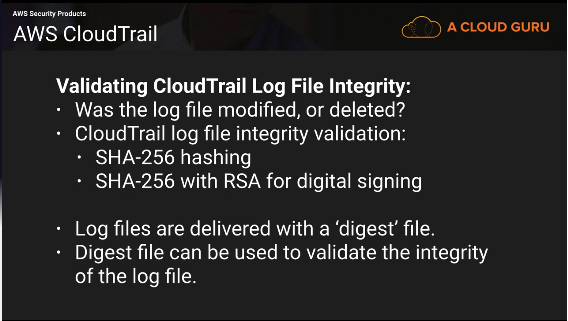
* Enabled by default (7 days)
* For longer you need to provision it in the console

**Management events-** operations that occur on your AWS account and resource, such as EC2 RunInstance API

* You can set up to log all events
* Read only events
* Write only events
* No events

**Digest File-** a way to validate if your cloudtrail logs are actually valid.

* Creates a hash for every log file that is delivered
* Hash is generated use public keys and private keys
* Amazon has access to the private keys
* Check if the hash is correct. If the hash matches then the log files have not been altered



CloudTrail Protecting Your Logs

Q) CloudTrail logs contain metadata not application data. Why should you consider security them?

A) CloudTrail logs may contain personal identifiable data such as usernames and even team memberships. Also, detailed configuration information such as DynamoDB table and key names may be stored. This information may provide valuable to an attacker and it is considered best practice to secure CloudTrail logs.

Q) How do we stop unauthorized access to log files?

A) Use IAM policies and S3 bucket policies to restrict access to the S3 bucket containing the log files.

Use SSE-S3 or SSE-KMS to encrypt the logs.

Q) How do we restrict log access to only employees with a security responsibility?

A) Place the employees who have a security role, into and IAM group with attached policies that enable access to the logs.

2 AWS CloudTrail policies that are AWS managed.

AWSCloudTrailFullAccess- Grant this policy to people who are going to be actually managing your AWS environment. Setting up CloudTrail and changing the policies of the buckets.

AWSCloudTrailReadOnlyAccess- Give this policy to auditors.

Q) How can we be notified that a log file has been created, and validate that it’s not been modified?

A) Configure SNS notifications and log file validation on the ‘Trail’. Develop a solution that when triggered by SNS will validate the logs using the provided digest file.

Set up a Lambda function that will compare digest files from a month ago and compare them with the CloudTrail logs they are associated with. If you compare files immediately then someone changes it you may not notice the change.

Q) How can we prevent logs from being deleted?

A) Restrict delete access with IAM and bucket policies. Configure S3 MFA delete. Calidate that logs have not been deleted using log file validation.

Q) How can we ensure that logs are retained for X years in accordance with our compliance standards.

A) By default, logs will be kept indefinitely. Use S3 object lifecycle management to remove the files after the required period of time, or move the files to AWS Glacier for more cost effective long term storage.

CloudWatch 101

**CloudWatch-** a monitoring service for AWS cloud resources and the applications you run on AWS

Enables:

* Resource utilization, operational performance monitoring (CPU, Disk, and custom metrics)
* Log aggregation and basic analysis

Provides:

* Real- time monitoring within AWS for resources and applications
* Hooks to event triggers

Key components:

* CloudWatch
* CloudWatch Logs
* CloudWatch Events

CloudWatch

* Real time
* Metrics
* Alarms
* Notifications
* Custom metrics (Can be on premise servers)

CloudWatch Logs

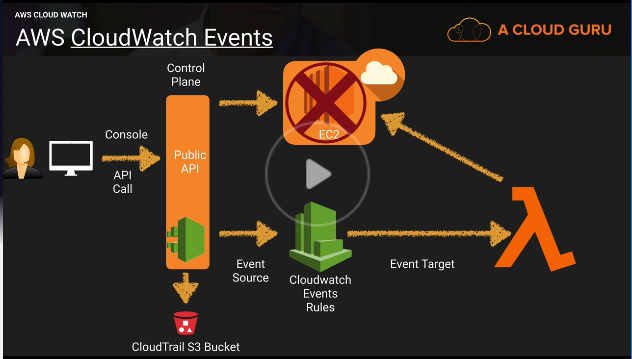
* Pushed from some AWS service (including cloudtrail)
* Pushed from your application/systems
* Metrics from log entry matches
* Stored indefinitely behind the scenes (not user S3)

Monitor HTTP response code in apache logs

Receive alarms for errors in kernel logs

Count exceptions in application logs

These are examples, but you can log just about anything in your application that you know how to script.

CloudWatch Events

* Near real-time stream of system events

Event types

* AWS resources state change (what happens when an instance gets stopped?)
* AWS CloudTrail (API Calls)
* Custom events (Code)
* Scheduled

Rules- match incoming events and route them to one or more targets

Targets- AWS Lambda, SNS topics, SQS queries, kinesis streams

Q) How do we control who can access CloudWatch and what they can do?

A) Use IAM policies to restrict access to CloudWatch and the actions they can perform.

However, remember that data is decoupled from its source, therefore you are not able to restrict access by the originating resource. (restrict access to CloudWatch and the originating source separately in your IAM policies)

Q) How are unauthorized users prevented from accessing CloudWatch?

A) Users need to be authenticated with AWS and have the appropriate permissions set via IAM policies to gain access.

Config 101

**Config-** A fully managed service that provides you with an AWS resource inventory, configuration history, and configuration change notifications to enable security and governance

Enables:

* Compliance auditing
* Security analysis
* Resource tracking

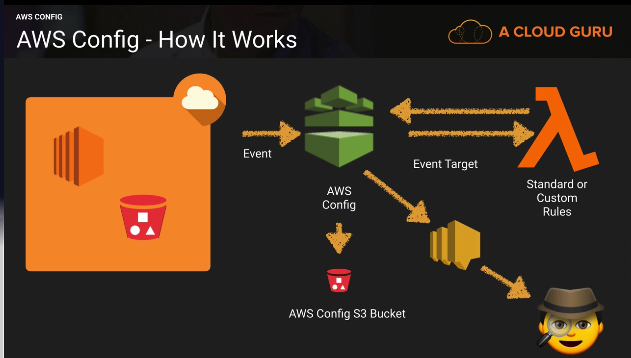
Provides:

* Configuration snapshots and logs config changes of AWS resources
* Automated compliance checking

Key components:

* Config dashboard
* Config rules (managed and custom)
* Resources
* Settings

You cannot just turn it on for all regions with one click of the button



Key Terminology:

Configuration Items- point in time attributes of a resource

Configuration snapshots- a collection of config items

Configuration stream- stream of changed config items

Configuration history- a collection of config items for a resource over time

Configuration recorder- the configuration of config that records and stores config items

Recorder setup:

* Logs config for account in a region
* Stores in S3
* Notifies SNS

What can we see:

* Resource Type
* Resource ID
* Compliance
* Timeline
  + Configuration details
  + Relationships
  + Changes
  + Cloudtrail events

Compliance checks:

* Trigger
  + Periodic
  + Configuration snapshot delivery

Managed rules

There are 60 or so right not (the video says 40)

Managed rules are basic, but fundamental.

Permissions needed for config:

* AWS config requires an IAM role with (console will optionally create this for you)
  + Read only permissions to the recorded resources
  + Write access to S3 logging bucket
  + Publish access to SNS

Restrict Access:

* Users need to be authenticated with AWS and have the appropriate permissions set via IAM policies to gain access
* Only admins needing to set up and manage config require full access
* Provide read only permissions for config day-to-day use

Monitoring config:

* Use cloudtrail with config to provide deeper insight into resources
* Use cloudtrail to monitor access to config such as someone stopping the config recorder

Set up an alert if the root user logs in

1. Turn on Cloudtrail and CloudWatch logs integration
2. Create a metric filter
3. Assign a metric
4. Create an alarm
5. Push it out to an SNS topic
6. Look up the event and take corrective action

CloudHSM- A service that helps you meet corporate, contractual and regulatory compliance requirements for data security by using dedicated Hardware Security Module appliances within the AWS cloud

Enables:

* Control of data
* Evidence of control
* Meet tough compliance controls

Provides

* Secure key storage
* Cryptographic operations
* Tamper resistance Hardware Security Module



Key control:

* AWS does not have access to your keys
* Separation of duties and role based access control is part of the design of the HSM
* AWS can only administer the appliance, not the HSM partitions where the keys are stored
* AWS can (but probably won’t) destroy your keys. But otherwise they have no access

Tampering:

* If the CloudHSM detects physical tampering the keys will be destroyed
* If the CloudHSM detects five unsuccessful attempts to access an HSM partition as Crypto Officer the HSM appliance erases itself
* If the CloudHSM detects five unsuccessful attempts to access an HSM with CryptoUser credentials, the user will be locked and must be unlocked by a crypto officer

Monitoring:

Use CloudTrail to monitor your API calls to your HSM to see who is doing what.

Inspector and Trusted Advisor

Inspector- automated security assessment service that helps improve the security and compliance of applications deployed on AWS. Inspector automatically assesses applications for vulnerabilities or deviations from best practices. After performing an assessment, inspector produces a detailed list of security findings prioritized by level of severity. These findings can be reviewed directly or as part of detailed assessment reports which are available via the amazon inspector console or API.

How does it work?

* Create “assessment target”- the instances you want inspector to run on
* Install agents on EC2 instances
* Create “Assessment template”- the type of rules you want to run and for how long
* Perform “Assessment run”
* Review “Findings” against “rules”

Create a new role- gives inspector read only access of your instances in your account

Uses tags to define which instances inspector will check

Install the agent on your instances

Define an assessment target by using your tags

Define an assessment template

Rule packages

* Security best practices 1.0
* Runtime Behavior analysis 1.0
* Common vulnerabilities and exposures 1.1
* CIS operating system security configuration benchmark 1.0

Duration

* 15 minutes
* 1 hour
* 8 hours
* 12 hours
* 24 hours

Severity levels

* High
* Medium
* Low
* Informational

After a run you can download the findings report or a full report in the form of HTML of a PDF

You can run all the rules packages at the same time

AWS Trusted Advisor- an online resource to help you reduce cost, increase performance, and improve security by optimizing your AWS environment. Advisor will advise you on cost optimization, performance, security, fault tolerance, and not stated in the video but service limits for free tier account.

By default, if you do not get full trusted advisor and will only get a few security checks, one performance issue, and all service limit checks. To unlock full trusted advisor, you need to have a business or enterprise AWS account.

Core checks and recommendation.

The main default check is security groups that are opened to the world.

Inspector will be needed if you want to check security on your instances, but if it is trusted advisor security it will usually ask about Security Groups or MFA. Also remember that trusted advisor is more than just security.

Inspector will:

* Monitor the network, file system, and process activity within the specified target
* Compare what it ‘sees’ to security rules
* Report on security issues observed within target during run
* Report findings and advise remediation

It will not:

* Relive you of your responsibility under the shared responsibility model
* Perform miracles

Logging with AWS

Services:

* AWS CloudTrail- logs API calls- after the fact investigation
* AWS Config- logs configuration changes- point in time logging tool
* VPC Flow Logs- logs ENI traffic in and out of your VPC
* AWS CloudWatch Logs- logs anything on your EC2 instances or on prim servers that you can code in python. Usually around the application. You can also send AWS services like cloudtrail to cloudwatch logs. Logs performance of your AWS assets

Prevent unauthorized access:

* IAM users, groups, roles, and policies
* Amazon S3 bucket policies
* MFA

Ensure role-based access:

* IAM users, groups, roles and policies- security people should be able to look at your logs
* Amazon S3 bucket policies

Alerts when logs are created or fail:

* CloudTrail notifications
* AWS Config Rules

Alerts are specific, but don’t divulge detail:

* CloudTrail SNS notification only point to log file location

Log changes to system components:

* AWS Config Rules
* CloudTrail

Controls exist to prevent modification to logs

* IAM and S3 control and policies
* CloudTrail log file validation
* CloudTrail log file encryption

Storage of log files

* Logs are stored for at least one year
* Store logs for an org-defined period of time
* Store logs Realtime for resiliency
* Lifecycle policies

AWS Security Specialty Section 5 Notes

KMS- a managed service that makes it easy for you to create and control the encryption keys used to encrypt your data, and uses HSMs to protect the security of your keys.

IAM is global- you can use any user in any region

KMS is regional- a key can only be used in the region you created it in.

3 Users

1st user= system admin= policy is administrator access

2nd user= financial controller= S3FullAdminAccess and ReadOnlyAccess to the console. He gets a KMS key.

3rd user= account team but not CFO= S3FullAdminAccess and ReadOnlyAccess to the console

Key material orgins

* KMS= AWS provides the key material
* External= You provide the key material

Key Administrators- defines which user or role can admin the key through the KMS API

Key Deletion- you can allow key admins to delete the key, but if you delete the key and the key is encrypted you will lose that data.

Key Usage- the IAM user and roles that can use the key to encrypt and decrypt the data

You can give external AWS accounts the ability to use your key

* Just add the account ID you want to give access too

2nd user uploads the 3 files using the KMS key that is made for the 2nd user.

What if you make one object public?

Because it is encrypted people cannot view it over the web without the KMS key

What happens if we change that encryption to SSE-S3?

You can now access it over the web because amazon has the key.

In conclusion KMS has an added layer of security.

If you went to click open and or download the object that is encrypted in S3 with the user who has the KMS key you would be able to open or download it.

3rd user tries to access the same files without the KMS key.

The 1 file with SSE-S3 will still be accessible by this user, but the files with the KMS key cannot be accessed without the KMS key usage permissions. Same with open or downloading the file even though this user has full S3 access.

1st user with admin access but no KMS key usage permissions

The file with KMS will get denied if you try to access it thru S3 URL’s because you are trying to access it as a random user, but if you try to open or download it you can decrypt it without the key usage permissions

* Be careful who you give admin access too
* Use least privilege
* Admins can also change themselves to a key administrator or a key user if they want

1st user with system administrator policy access

You can no longer add yourself to a key as an administrator or a user. You can only add roles

If you go back to S3 you can also no longer open or download the KMS file.

SysAdmin access takes away some important privileges in terms of KMS

What happens if the 2nd user leaves the company and decides to delete the encryption key?

* You can only schedule a key deletion- deleting a key makes all data under that encryption key unrecoverable and it forces a minimum waiting period of 7 days and a max of 30
* You can also disable it- if you disable it that means any data encrypted with it will not be viewable until your enable it as the data stays encryption but the key is disabled. No matter which user
* If you let the key delete any data encrypted is unrecoverable

KMS Part 3

Create a new key with External Key Material Origin and check the box saying I understand the Security, Availability, and durability implications of using an imported key.

The key has been created but you cannot use it until you import key material.

* Download wrapping key and import token you are going to use to generate that key material
* Wrapping algorithm= RSAES\_OAEP\_SHA\_1 and download it
* Wrapping key and import token expire after 24 hours
* Click I am ready to upload my key material
* Download and unzip openSSL. Move your import parameter into your openSSL directory
* 3 file wrapping key, import token, and read me

Ready to upload:

* Upload the encrypted key material you generated
* Upload the import token
* Then you need to choose an expiration option and that can either be a date you specify or you can make it so key material does not expire
* Then click finish

Can you import and generate a new CMK with the same wrapping algorithm and the import token?

No, you cannot use somebody else’s import token to generate a new key or the same encrypted key material.

You cannot enable automatic key rotation for a CMK with imported key material, however you can manual rotate a CMK with imported material. To do so create a new CMK then change the CMK identifier to your new CMK you just created.

Ciphertext are not portable between CMKs. Data cannot be encrypted with a separate CMK only the CMK it is associated with.

If you use SSE-S3 it will generate a new key in KMS, but you cannot do anything with that key because AWS manage it.

With external key material origin, you can delete the key material immediately and bypass the 7 day wait, and that data will become unusable immediately. You will still need 7 days to delete the customer master key from the KMS screen.

KMS Part 4

KMS integrates with EBS, S3, Amazon Redshift, Amazon Elastic Transcoder, Amazon WorkMail, Amazon RDS, and other to make it simple to encrypt your data with encryption keys that you manage.

The customer master key:

* Alias
* Creation date
* Description
* Key state
* Key material (customer provided or AWS provided)
* Can never be exported

AWS managed CMK for each service that is integrated with AWS KMS

Customer-managed CMK that you generate by providing AWS with key material

Setup a customer master key:

* Create Alias and Description
* Choose material option
* Define key administrator permissions
  + IAM user/Roles that can admin the key through the KMS API
* Define key usage permissions
  + IAM users/roles that can use the key to encrypt and decrypt data

Why import your own key material?

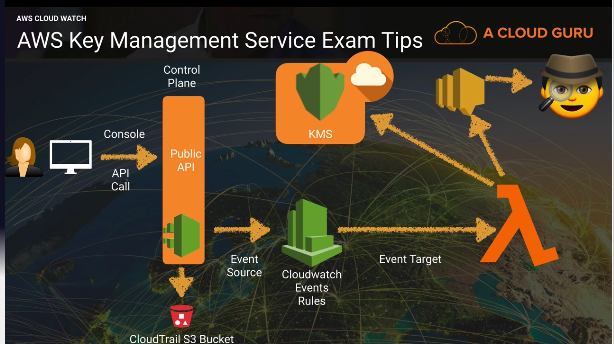
* Prove that randomness meets your requirements (Compliance)
* Extend your existing processes to AWS
* To be able to delete key material without a 7-30 day wait. Then be able to import them again
* To be resilient to AWS failure by storing keys outside AWS

How to import your own key material:

* Create a CMK with no key material
* Download a public key (wrapping key) and import token
* Encrypt the key material
* Import the key material

Considerations for imported key material:

* Availability and durability is different
* Secure key generation is up to you
* No automatic rotation
* Ciphertext are not portable between CMKs



Event driven security on KMS. Have cloudwatch events monitor KMS API via cloudtrail and if someone were to disable a key have that trigger an event. Send that event to the event target of lambda and have lambda reenable that key. Also have lambda send an SNS notification to your sysadmin.



Use config with KMS to monitor if anyone changes the key admin or usage permissions on a key or maybe to monitor any new key provisions. Have config send an SNS notification to your sysadmin.

Using KMS with EBS

By default, root device volumes are not encrypted

If you add additional volumes you can encrypt those, but those additional volumes you encrypt will be treated like SSE-S3 but for EBS. Fully managed by AWS.

Once you provision a volume using KMS that volume will always be linked to that key

Create a snapshot of the root device volume

Create an image of the snapshot

AMI options:

* You can modify the image permissions
* By default, all images you take will be private, but you can change them to public or share them with specific AWS accounts by providing the AWS account number

Copy the AMI and copy it to the exact same region

Check the encrypt target EBS snapshot and select your own KMS CMK

If you copy your AMI to a different region you need to make sure that the key you want to use has to be in the destination region.

You can use KMS to encrypt EBS volumes, but you cannot use KMS to generate a public key/private key to log in to EC2.

You can import public keys into EC2 key pairs, but we cannot use EC2 key pairs to encrypt EBS volumes, you must use KMS or 3rd party application/tools.

EC2 and key pairs

If you cd .ssh you will be able to go to the hidden directory of .ssh. this directory contains the authorized\_keys file. If you cat that file you will be able to see the public keys that are stored in that instance.

You can also get this using the meta data

Curl <http://169.254.169.254/latest/meta-data/public-keys/0/openssh-key/>

Add a new public key to an instance.

Add a full access S3 role for your instance

Ssh-keygen -t rsa

Enter a file name for it. This will generate 2 files mynewkp and mynewkp.pub.

You need to add the mynewkp.pub to the authorized\_keys file

Cat mynewkp.pub >> .ssh/authorized\_keys

The private key is the other file without the .pub

Copy they private key to an S3 bucket

Aws s3 cp mynewkp s3://bucketname

Download it from your bucket

Can you connect to your instance using that new key?

Yes you can

EC2 Key Pairs part 2

What happens if you delete all your key pairs in the AWS console? Will you still be able to access your instance?

Yes, you can. When you delete it in the console it does not affect your EC2 instances

Nothing will change in the meta data or the authorized\_keys file.

Can you add a new public key to your instance if you can no longer access an instance?

Create an AMI of you instance.

Launch that AMI instance and add a new key pair.

Will this append the new key on the authorized keys file or will it overwrite the whole file?

It will append the key and keep the existing keys. You should clean up this file so you can only use the brand new private key.

AWS Marketplace Security Products

You can purchase 3rd party AMI’s from the marketplace that are preconfigured

* You sort by OS
* Sort by billing
  + Free
  + Monthly
  + Annual
  + Bring your own license

Types of products:

* Firewalls
* Kali Linux
* CIS Red Hat Linux 7 benchmark

AWS WAF and Shield

WAF- a web application firewall that lets you monitor the HTTPS and HTTPS requests that are forwarded to Amazon cloudFront or an application load balancer. AWS WAF also lets you control access to your content.

You can configure conditions such as what ip addresses are allowed to make this request or what query string parameters need to be passed for the request to be allowed and then the application load balancer or cloudfront will either allow this content to be received or to give a HTTP 403 status code.

At its most basic level, AWS WAF allows 3 different behaviors:

* Allows all requests except the ones that you specify
* Block all requests except the ones that you specify
* Count the requests that match the properties that you specify

Additional protection against web attacks using conditions that you specify:

* IP addresses that requests originate from
* Country that requests originate from
* Values in request headers
* String that appear in request, either specific string or string that match regular expressions
* Presence of sql code that is likely to be malicious
* Presence of a script that is likely to be malicious

WAF integration:

* Cloudfronts distributions (global)
* Application load balancers (regional)

Associating a WAF with a Cloudfront distribution:

1. Go to your WAF and click rules
2. Click add association
3. Find the resource you want to add and click add

Manual IP block set- you can add IP addresses or IP addresses ranges with this option. You can have /8 /16 /24 /32

When you associate your cloudfront distribution with your WAF it redeploys the distribution so this can cause delays and may take 15 to 20 minutes to become active.

You can use WAF to protect web sites not hosted in AWS via CloudFront. Cloudfront supports custom origins outside of AWS.

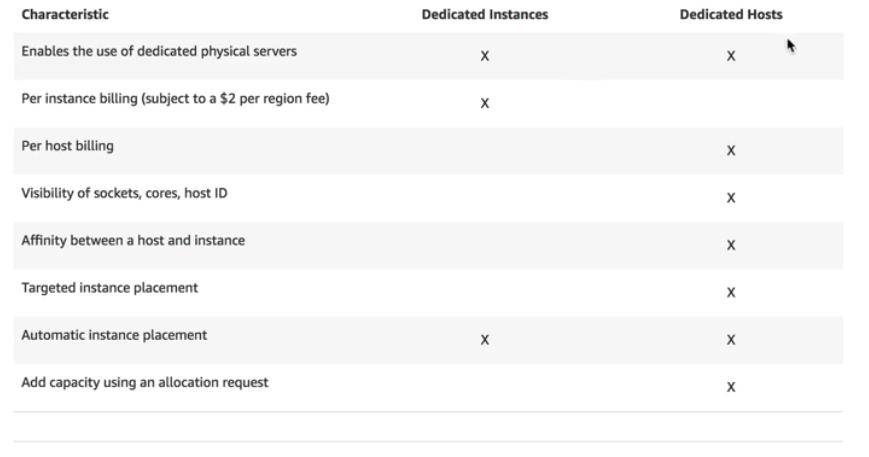
Dedicated Instances vs Dedicated Hosts

Dedicated Instance- Amazon EC2 instances that run in a VPC on hardware that is dedicated to a single customer. Your dedicated instances are physically isolated at the host hardware level from instances that belong to other AWS accounts.

Dedicated instances may share hardware with other instances from the same AWS account that are not dedicated instances.

Pay for dedicated instances on-demand, save up to 70% by purchasing reserved instances, or save up to 90% by purchasing spot instances.

Dedicated hosts- the main difference is that dedicated hosts give you additional visibility and control over how instances are placed on a physical server, and you can consistently deploy your instance to the same physical server over time. As a result, dedicated hosts enable you to use your existing server-bound software licenses and address corporate compliance and regulatory requirements.



Both have dedicated hardware to you. Single tenancy.

Dedicated instances are charged by the instance, dedicated hosts are charged by the host.

If you have specific regulatory requirements or licensing conditions, choose dedicated hosts.

Dedicated instances may share the same hardware with other AWS instances from the same account that are not dedicated

Dedicated hosts give you much better visibility in to things like sockets, cores, and host id.

AWS Hypervisors

Hypervisor- a computer software, firmware or hardware that creates and runs virtual machines. A computer on which a hypervisor runs one or more virtual machines is called a host machine, and each virtual machine is called a guest machine.

EC2 currently runs on Xen Hypervisors. Xen hypervisors can have guest operating systems running either as Paravirtualization (PV) or using Hardware Virtual Machine (HVM).

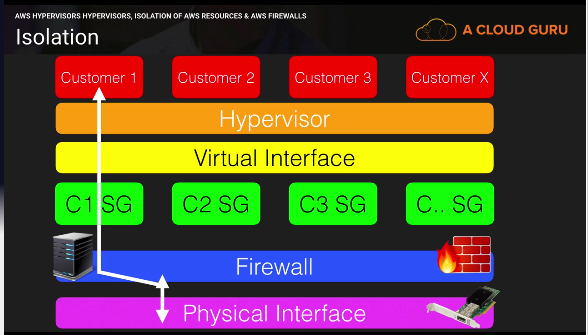
**HVM-** are fully virtualized. The VMs on top of the hypervisors are not aware that they are sharing processing time with other VMs.

**PV-** is a lighter form of virtualization and it used to be quicker (not anymore so use HVM whenever possible)

Windows instances can only be HVM where as Linux can be both PV and HVM

Paravirtualized guests rely on the hypervisor to provide support for operations that normally require privileged access, the guest OS has no elevated access to the CPU. The CPI provides four separate privilege modes: 0-3, called rings. Ring 0 is the most privileged and 3 is the least. The host OS executes in ring 0. However, rather than executing in ring 0 as most OS do, the guest OS runs in a lesser-privileged ring 1 and applications run on ring 3.

Isolation



Customers are completely isolated from one another.

Hypervisor Access:

Administrators with a business need to access the management plane are required to use MFA to gain access to purpose-build administration hosts. These administrative hosts are systems that are specifically designed, build, configured, and hardened to protect the management plane of the cloud. All such access is logged and audited. When an employee no longer has a business need to access the management plan the privileges and access to these hosts and relevant systems can be revoked.

Guest EC2 Access:

Virtual instances are completely controlled by you, the customer. You have full root access or administrative control over accounts, services, and applications. AWS does not have any access rights to your instances or the guest OS.

Memory Scrubbing:

EBS automatically resets every block of storage used by the customer, so that one customers data is never unintentionally exposed to another. Also memory allocated to guests is scrubbed by the hypervisors when it is unallocated to a guest. The memory is not returned to the pool of free memory available for new allocations until memory scrubbing is complete.

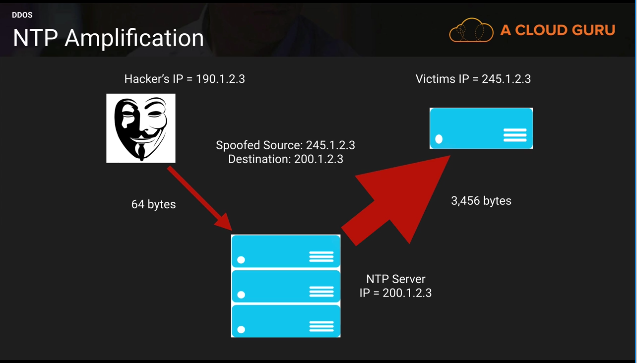
AWS Security Specialty Section 8 notes

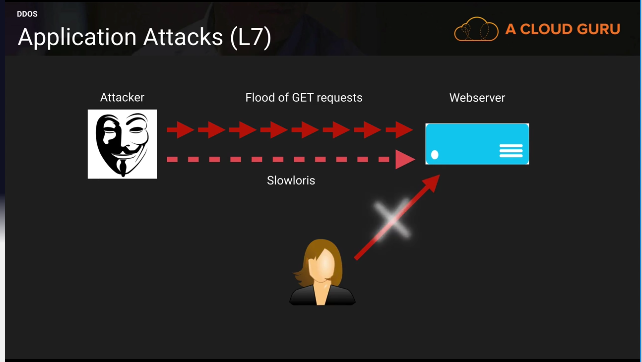
DDOS Overview

DDoS- A distributed Denial of Service attack is an attack that attempts to make your website or application unavailable to your end users.

This can be achieved by multiple mechanisms, such as packet floods, by using a combination of reflection and amplification techniques, or by using large botnets.

Amplification/Reflection Attacks- can include things such as NTP, SSDP, DNS, Chargen, SNMP attacks, and more. It is where an attacker may send a third-party server a request using a spoofed IP address. That server will then respond to that request with a greater payload than initially requested to the spoofed IP address.

This means that if the attacker sends a packet with a spoofed IP address of 64bytes the NTP server would respond with up to 3,456 bytes of traffic. Attackers can co-ordinate this and use multiple NTP servers a second to send legitimate NTP traffic to the target.



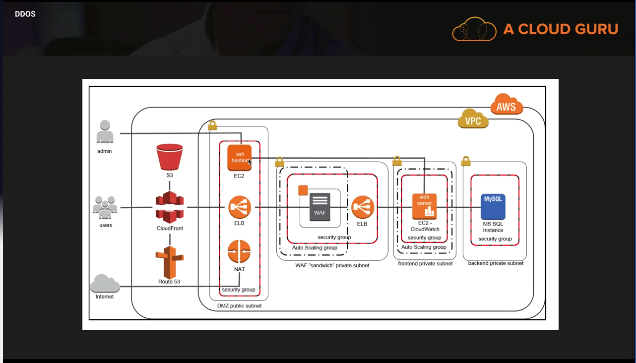
How to mitigate DDos attacks?

* Minimize the attack surface area
* Be ready to scale to absorb the attack
* Safeguard exposed resources
* Learn normal behavior
* Create a plan for attacks

Minimize the attack surface area

Some production environments have multiple entry points in to them. Perhaps they allow direct SSH or RDP access to their web servers/application and DB server for management.

This can be minimized by using a Bastion/Jump Box that only allows access to specific white listed IP addresses to these bastion servers and move the web, application, and DB servers to private subnets. By minimizing the attack surface area, you are limiting your exposure to just a few hardened entry points.



Be ready to scale to absorb the attack.

The key strategy behind a DDoS attack Is to bring your infrastructure to a breaking point. This strategy assumes one thing: that you cant scale to meet the attack.

The easiest way to defeat this strategy is to design your infrastructure to scale as, and when it is needed.

You can scale both horizontally and vertically.

Scaling Benefits:

* The attack is spread over a larger area
* Attackers then have to counter attack, taking up more of their resources
* Scaling buys you time to analyze the attack and to respond with the appropriate countermeasures
* Scaling has the added benefit of providing you with additional levels of redundancy

Safeguard Exposed Resources

In situations where you cannot eliminate internet entry points to your applications, you will need to take additional measures to restrict access and protect those entry points without interrupting legitimate end user traffic.

3 resources that can provide this control and flexibility are CloudFront, Route 53, and WAFs.

CloudFront:

* Geo Restriction/Blocking- Restrict access to users in specific countries (using whitelists or blacklists)
* Origin Access Identity- Restrict Access to your S3 bucket so that people can only access S3 using CloudFront URLs.

Route53:

* Alias Record Sets- you can use these to immediately redirect your traffic to an Amazon cloudFront distribution, or to a different ELB with higher capacity EC2 instances running WAFs or your own security tools. No DNS change, and no need to worry about propagation.
* Private DNS- Allows you to manage internal DNS names for you application resources (web servers, application servers, DBs) without exposing this information to the public internet

WAFs: DDoS attacks that happen at the application layer commonly target web applications with lower volumes of traffic compared to infrastructure attacks. To mitigate these types of attacks, you will want to include a WAF as part of your infrastructure.

* New AWS WAF service
* Find WAF’s in the AWS market place for specific needs

Learn normal behavior:

* Be aware of normal and unusual behavior
  + Know the different types of traffic and what normal levels of this traffic should be
  + Understand expected and unexpected resource spikes

What are the benefits?

* Allows you to spot abnormalities fast
* You can create alarms to alert you of abnormal behavior
* Helps you to collect forensic data to understand the attack

Create a plan for attacks:

* Having a plan in place before an attack ensures that
  + You’ve validated the design of your architecture
  + You understand the costs for your increased resiliency and already know what techniques to employ when you come under attack.
  + You know who to contact when an attack happens

AWS Shield:

* Free service that protects all AWS customers on ELB’s, CloudFront, and Route 53
* Protects against SYN/UDP floods, reflection attacks, and other layer3/layer 4 attacks
* Advanced provides enhanced protections for your apps running on ELB’, CloudFront, and Route 53 against larger more sophisticated attacks for only 3000 per month.
* Shield for layer 3 and layer 4 attacks. WAF’s for layer 7 attacks.

AWS Shield Advanced provides:

* Always on, flow-based monitoring of network traffic and active applications monitoring to provide near real time notification of DDoS attacks
* DDoS response team 24x7 to manage and mitigate applications layer DDoS attacks.
* Protects your AWS bill against higher fees due to ELB, CloudFront, and Route 53 usage spikes during a DDoS attack.

Technologies you use to mitigate a DDoS attack:

CloudFront, Route 53, ELBs, WAFs, Autoscaling, CloudWatch

WAF Integration

WAF integrates with both Application Load Balancers and CloudFront. It does not integrate with EC2 directly, nor Route 53 or any other services.

EC2 has been hacked. What should you do?

What steps should you take?

1. Stope the instance immediately
2. Take a snapshot of the EBS volume
3. Terminate the instance (ryan states in the video)
4. Deploy the instance in to a totally isolated environment. Isolated VPC, no internet access- ideally into a private subnet. Also monitor using VPC Flow Logs
5. Access the instance using an isolated forensic workstation. Wireshark or kali linux to investigate.
6. Read through the logs to figure out how it happened. (Windows Event Logs or Linux logs)

I’ve leaked my keys on GitHub accidentally

1. Go to IAM and find the user whose keys have been leaked
2. Navigate to the security credentials
3. First make it inactive
4. Make a new key
5. Then test the new key and delete the old one

For the root user to the same thing, but the keys are located under my security credentials. Also remember it is best practice to just delete your root keys as soon as you make your AWS account.

Reading CloudTrail Logs

Every API call will be logged into CloudTrail

CloudTrail logs are in JSON or key value pairs

Pen Testing AWS Market Place

Whenever you want to do pen testing you always need permission. Just a little note if you actually read this, there was a whiz labs question that stated there are now 2 ways to do pen testing and the second way is to use pre-approve tools in the AWS market place. Obviously on the test as it may not have been updated always choose the older way, but if you have a question that says choose 2 well now you know.

Allowed resources:

* EC2
* RDS
* Aurora
* CloudFront
* API Gateway
* Lambda
* LightSail
* DNS zone walking

RDS instance types:

The policy does not permit testing on m1.small, t1.micro, or t2.nano EC2 instances types. Same for EC2 instance types.

Other simulated events:

* Security simulations or security game days
* Support simulations or support game days
* War game simulations
* White cards
* Red team and blue team testing
* Disaster recovery simulations
* Other simulated events

Pen testing tools in the AWS marketplace

Kali Linux- gold standard for pen testing

AWS Certificate Manager

Need a Registered Route 53 Domain name to use Certificate Manager

You can extend your domain name for up to 9 years, but it will cost you money. You can also turn on auto renew and amazon will automatically renew the domain for you granted that you pay for it.

Certificate Manager- Makes it easy to provision, manage, deploy, and renew SSL/TLS certificates on the AWS platform.

* You can import your own SSL certificates into ACM
* You can never export an AWS certificate out of AWS

Validation method:

* DNS validation- choose this option if you have or can obtain permissions to modify the DNS configuration for the domains in your certificate request
* Email validation- choose this option if you do not have permissions or cannot obtain permissions to modify the DNS configuration for the domain in you certificate request

To validate:

* You need to add the CNAME record to your Route 53 configuration
* You can also choose create record in Route 53 if you want amazon to update your DNS configuration for you

Steps:

1. Go to your Route 53 and click create a record set
2. Choose CNAME and copy the name and paste it into the name text box
3. Copy the value and past it into the value of your record
4. Click create
5. Click continue on your ACM page and it will update yourself

ACM automatically renews unless:

* It does not auto renew imported certificates
* It does not auto renew certificates associated with Route 53 private hosted zones

ACM does not have to be used through Route 53. You can use it for like go daddy URLs and those will automatically renew.

ACM services that use it:

CloudFront and Application load balancers

To associate it with cloudfront:

* Go to edit your distribution
* Click custom SSL certificates
* And click on your certificate and save it

To associate it with ALB:

* Click on listeners and choose HTTPS
* Click next
* Choose a certificate from ACM and apply the certification name

If you want to apply them to both you need to do it separately. It still uses the same certificate, but it is just used at separate locations

You cannot export ACM certificates due to the fact that they are free, but you have to use them on AWS services essentially making them not free.

Perfect Forward Secrecy and ALBs

Perfect Forward Secrecy- a property of secure communication protocols in which compromises of long-term keys do not compromise past session keys.

* Old- all of your traffic to and from a load balancer would be recorded and you could take a compromised key and you could go in and decrypt old traffic to and from both points
* New- even if a key is compromised a year in the future people won’t be able to go back and decrypt that old traffic

If a private key is compromised in the future and they recorded all the traffic to and from the load balancer, they won’t be able to use that private key to decrypt later traffic in the future.

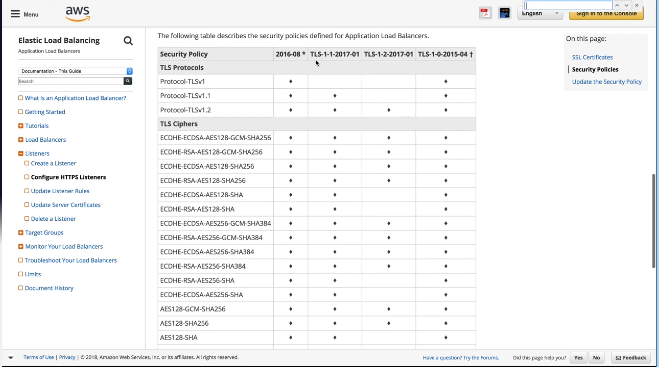
Securing your load balancer with perfect forward secrecy:

* Select Application load balancer
* Choose HTTPS
* Click next and choose select security policy
* Choose the policy with 2016-08

The TLS cipher that needs to be enables on the load balancer is the ECDHE cipher.

HTTP=80 HTTPS=443

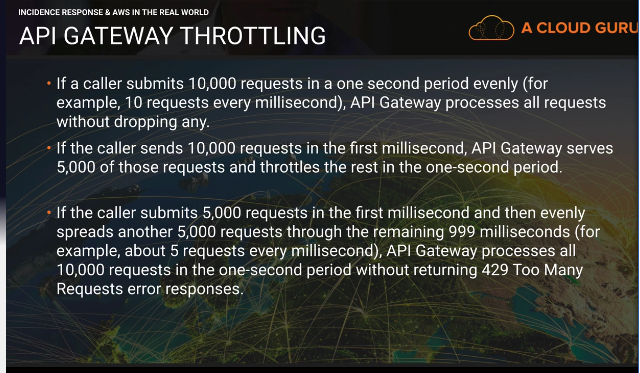
SSL/TLS=HTTPS



API Gateway – Throttling and Caching

API Gateway Throttling:

* To prevent your API from being overwhelmed by too many requests, amazon API gateway throttles request to you API
* When request submission exceeds the steady-state request rate and burst limits, API Gateway fails the limit-exceeding requests and returns 429 Too many requests error responses to the client
* By default, API gateway limits the steady-state request rate to 10,000 requests per second
* It limits the burst to 5,000 requests across all APIs within an AWS account.
* DDoS mitigation technique



API gateway is throttled by default

How to raise it:

* Raise a ticket with AWS to raise the rate limit and burst limit

API Gateway Caching:

What if you are getting the same request 10,000 times a second?

* You can enable API caching in Amazon API Gateway to cache your endpoints response
* With caching, you can reduce the number of calls made to your endpoint and also improve the latency of the requests to your API.
* When you enable caching for a stage, API Gateway caches responses from your endpoint for a specified time-to-live period, in seconds
* API gateway then responds to the request by looking up the endpoint response from the cache instead of making a request to your endpoint. The default TTL value for API caching is 300 seconds
* The max Is 3600 seconds
* TTL=0 means caching is disabled.

AWS Systems Manager Parameter Store

Systems Manager- A way to control your EC2 fleet at scale

Parameter Store- a way to store confidential information such as users, passwords, license keys etc. this information can be passed to EC2 as a bootstrap script, while maintaining the confidentiality of the information.

How to use it:

1. Create a parameter
2. Store sensitive data
3. Access parameters across services

Create parameter:

* Choose name- name has to be unique as this is what your resources will ref

Types:

* String- enter any data
* String list- enter multiple values separated with a comma
* Secure string- encrypts the data using KMS. Used to store sensitive information. Other system admins can ref the parameter, but won’t be able to see the data.

Services that can ref the parameter:

* EC2
* CloudFormation
* Lambda
* EC2 Run Command

AWS System Manager Run Command

EC2 Run Command

* You work as a systems administrator managing a large number of EC2 instances and on-premise systems
* You would like to automate common admin tasks and ad hoc configuring changes e.g. installing applications, applying the latest patches, joining new instances to a windows domain without having to login to each instance

What do you need to make it work?

* You need to create a role and you need to apply that role to the EC2 instances you want it to run on
* EC2 role for simple system manager
* The SSM agent needs to be installed on all your managed instances

How can you base what instances it runs on?

* You can base it off tags
* You can manually select the instances you want

Commands can be issued using AWS console, AWS CLI, AWS Tools for Windows Powershell, Systems Manager API or Amazon SDKs

You can use this service with your on-premise systems as well as EC2 instances

The commands and parameters are defined in a system manager document

Compliance in AWS

PCI DSS

ISO 27001

HIPPA

Security Specialty Outside of ACloudGuru Notes

KMS:

Key Rotation:

AWS Managed CMK’s (SSE-S3)- Rotation is already enabled and it is rotated automatically every 3 years (1095 days)

AWS KMS (SSE-KMS)- Rotation is disabled by default and has to be enabled. Once enabled the backing key will rotate every year.

Customer Managed CMK’s (SSE-C)- Rotation has to be done manually.

**Customer Master Keys-** The primary resources in AWS KMS are customer master keys. You can use a CMK to encrypt and decrypt up to 4 kilobytes (4096 bytes) of data. Typically, you use CMKs to generate, encrypt, and decrypt the data keys that you use outside of AWS KMS to encrypt your data. This is a strategy known as envelope encryption.

* Encrypting with the actual Customer Master key is for small data like passwords or RSA Keys
* For encryption of bigger file, you need a data key

**Data keys-** encryption keys that you use to encrypt data, including large amounts of data and other data encryption keys.

To use this you need to use the GenerateDataKey operation.

**Key policies-** when you create a CMK, you determine who can use and manage that CMK. These permissions are contained in a document called the key policy. (think like bucket policy) you cannot edit the key policy for an AWS-managed CMK.

<https://docs.aws.amazon.com/kms/latest/developerguide/concepts.html>

**Data key caching-** stores data keys and related cryptographic material in a cache.

* Improves performance, reduce costs, helps you stay within the services limits as your application scales

Reasons to use it:

* Allows you to reuse data keys
* It generates numerous data keys
* Your cryptographic operations are unacceptably slow, expensive, limited, or resource intensive
* Reduces costs of accessing the keys in the AWS KMS Service

The min set of permissions to encrypt and decrypt data using CMK keys:

Encrypt, Decrypt, Rencrypt, GenerateDataKey, DescibeKey.

Auditing CMK Usage:

* Use AWS CloudTrail to audit key usage
* See who is assigned key usage permissions to the CMK. The reason to check this is to see how many people have the option to use it. The more people the more likely it is or may be getting assigned to data

**Alias-** an optional display name for a CMK

* Each CMK can have multiple aliases, but each alias points to only one CMK
* Alias names must be unique in the AWS account and region

Kinesis:

**Kinesis-** used to collect and process large streams of data records in real time.

* Kinesis streams
* Kinesis firehose
* Kinesis analytics

**Kinesis Client Library (KCL)-** helps you customize and process data from a kinesis stream.

* KCL needs a policy allowing access to DynamoDB and CloudWatch

**Kinesis Enhanced Monitoring-** allows you to monitor your kinesis streams at the shard level.

Guard Duty:

**Amazon GuardDuty-** a continuous security monitoring service that analyzes and processes the following data sources: VPC Flow Logs, CloudTrail Event Logs, DNS Logs.

* Uses threat intelligent feeds and machine learning to identify unexpected and potentially unauthorized and malicious activity within your AWS environment
* Can monitor malicious port scans

Amazon Macie:

**Amazon Macie-** a security service that uses machine learning to automatically discover, classify, and protect sensitive data in AWS.

Cognito:

**Amazon Cognito-** provides authentication, authorization, and user management for your web and mobile apps.

* Sign in directly with a user name and password
* Sign in through a 3rd party such as facebook

**User pools-** a user directory in Cognito

* Sign up and sign in services
* Customizable web UI to sign in users
* Social sign in with Facebook, SAML sign in from your user pool
* Users directory management and user profiles
* MFA, checks for compromised credentials, account takeover protection, and phone/email verification
* Customized workflows and user migration through AWS Lambda triggers

**Identity pools-** enables you to create unique identities for you users and federate them with identity providers

Cognito Use Cases

* Use AWS Cognito to manage user profiles to apps that users sign up via web federation
* App sign in via an external ID provider
* If you have different types of users such as paid subscribers and guest you can create different Cognito groups, 1 for subscribers and 1 for guests.

VPC Endpoint Types:

**VPC Endpoint Gateway-** A gateway that is a target for a specified route in your route table, used for traffic destined to supported AWS services.

* S3
* DynamoDB

**VPC Interface Endpoint-** An elastic network interface with a private IP address that serves as an entry point for traffic destined to supported services.

* Config
* CloudWatch
* API Gateway
* KMS
* Kinesis Data Streams
* SNS

I only named a few, basically use this one unless you need to access resources in S3 or DynamoDB

Extra knowledge from whiz labs tests:

Best practices for carrying our Security Audits:

* On a periodic basis
* If there are changes in your organization, such as people leaving
* If you have stopped using one or more individual AWS services. Removing permissions on policies to keep least privilege
* If you’ve added or removed software in your accounts, such as applications, Opsworks stacks, CloudFormation templates
* If you ever suspect that an unauthorized person might have accessed your account

If you have a NACL and you need to send traffic to the web you should allow outbound traffic to the web on 443 or 80 depending on requirements (usually 443 for security) and your inbound traffic should be allowed traffic on ephemeral ports. You may receive traffic from a weird port so if you just open up 443 the traffic may not be allowed back in.

**Direct connect-** use this for consistent low latency from your data center to AWS

**VPN-** use this for encryption in transit

If you combine direct connect and a VPN you can get low latency with encryption in transit

**DHCP Option Set-** you can create a new option set to use your own managed DNS.

By default, a VPC has one that comes with the amazon DNS. You cannot change an existing DNS option set. So, to point to your own DNS server you need to create a new option set and replace the existing one.

AWS KMS and CloudHSM both are FIPS 140-2 compliant, but CloudHSM is level 3 outta 4 and AWS KMS is level 2.

Pen Testing:

* Get prior approval from AWS for conducting the test
* You can also use a pre-approved pen testing tool from the AWS marketplace

To encrypt DynamoDB tables you need to specify encryption at creation of the table

**WAF Sandwich-** EC2 instances running your WAF software is included in an Auto Scaling group and placed in between 2 ELB’s.

KMS Keys with the Redshift Cluster Service:

Amazon Redshift uses a four-tier, key-based architecture for encryption.

* Master key encrypts the cluster key
* Cluster key encrypts the database key
* Database key encrypts the data encryption keys

CloudTrail:

**Data Events-** these events provide insight into the resource’s operations performed on or within a resource. These are also known as data plane operations.

**Management Events-** Management events provide insight into management operations that are performed on resources in your AWS account. These can log things like non-API call events that occur in your account such as ConsoleLogin.

You can configure multiple CloudTrails logging either of these if needed or you can log all of them in one cloudtrail.

Best practice for multiple environments is to have a separate account for each environment.

Production, Test, Dev all have a separate AWS account.

EBS vocab:

**Amazon Data lifecycle manager-** you can use this to automate the creation, retention, and deletion of snapshots taken to back up your Amazon EBS volumes.

**EBS Snapshots-** are more durable then EBS volume replications as snapshots are stored in S3.

**Lambda authorizer-** a lambda function that you provide to control access to your API methods.

**Trust policy-** specifies which trusted account members are allowed to assume the role.

**IAM Credential Reports-** audits IAM credentials such as Access keys and passwords.

CloudFront:

**Viewer protocol policy-** This allows you to specify how users access cloudfont. You can restrict for your cloudfront to only be accessed via ssl with this.

**Signed Cookies-** allows you to control who can access your content when you don’t want to change your current URLs or when you want to provide access to multiple restricted files.

Use this when you want to restrict a section of files, but use pre signed URLs when you want to restrict individual files.

**Lambda@edge-** lets you run lambda functions to customize content that CloudFront delivers, executing the functions in AWS locations closer to the viewer.

Hybrid environments integration:

**Remote Desktop gateway-** used to deploy Microsoft windows based workload on its highly reliable cloud infrastructure

To make sure all traffic is encrypted in transit you need to enable LDAP over SSL

Condition policies:

**kms:ViaService-** Limits use of a customer-managed CMK to requests from particular AWS services. You can use it to assign a CMK to only S3, then a different CMK to only Redshift.

**aws:Referer-** You can use this condition to only allow bucket access from a specific web address.

**aws:PrincipalOrgID-** This condition can be used to grant access to your full organization to a specific bucket or resource.

AWS SSM:

Troubleshooting your instances that cannot properly use SSM:

* Install the SSM agent on your instances
* Associate the SSM role to the instances
* Check the instance status by using the health API
* Check the /var/log/amazon/ssm/errors.log.file on linux
* Check the %PROGRAMDATA%\Amazon\SSM\Logs\error.log

**AWS System Manager Patch Manager-** can be used to generate the report of compliance instances and then you can use Patch Manager to install the missing patches of your instances at scale.

**AWS System Manager Run Command-** use to run commands of instances at scale.

* Does not interfere with the continuous running of instances
* You can check the running processes of your instances and output it to an S3 bucket.

**AWS SSM Parameter Store-** Used to store confidential information and can be passed securely to AWS services like cloudformation

Inspector Vs Trusted Advisor:

Trusted Advisor

* From a Security perspective you can use Trusted Advisor to check which security groups allow unrestricted access to a resource

Inspector

* Checks security on actual EC2 instances
* Check for insecure server protocols such as telnet can be analyzed with the runtime behavior analysis rules package
* Check for security vulnerabilities

Monitoring:

CloudTrail

* API calls
* PCI compliance and API history investigation (you can go through logs and see API calls that happened in the past)
* Use CloudTrail to monitor KMS usage

Config

* Configuration changes
* History of configurations
* Monitors compliance of your configurations

VPC Flow Logs

* Web traffic in your VPC’s

CloudWatch Logs

* Application system monitoring
* Application logs

AWS Marketplace

* If you want to monitor and inspect actual security threats of your packets you should look at the marketplace to use a host-based intrusion detection system. Also use a third-party firewall installed on a central EC2 instance.

AWS GuardDuty

* To monitor malicious port scans
* Can monitor CloudTrail EventLogs, VPC Flow Logs, and DNS Logs