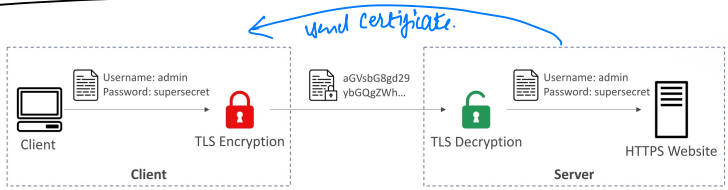
Security & Encryption

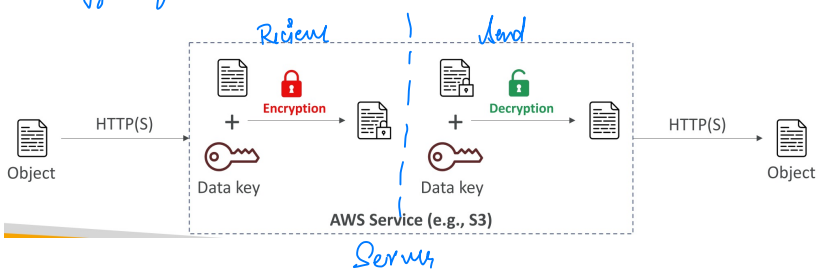
* In flight Encryption: (using TLS) { protects MITM attack }



* server side encryption

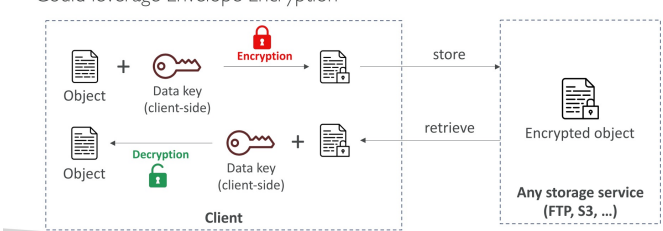
{ data encrypted after reaching @ server
data decrypted before leaving — " —

→ encrypted using data key, need to store the key securely for decrypting later.



* Client side encryption

- Could leverage Envelope Encryption



* automatic encryption { • S3 • storage Gateway • Cloudtrail logs

AWS KMS

→ manage encryption Keys for us

→ fully integrated with IAM for authorization.

→ able to audit KMS Key usage using cloud trail

→ integrates with most services

* never store secrets in plain text { use KMS Key encryption to store.

KMS Keys → Symmetric (AES-256 Keys)
=) single encryption Key for encrypt & decrypt
→ AWS services uses this!
→ we never get access to Keys only API calls.

↓

Assymmetric
(RSA, ECC Key pairs)

=> public Key to encrypt
private Key to decrypt } → sign/verify

→ public Key easily available

→ encryption outside of AWS by users can't call the KMS API calls.

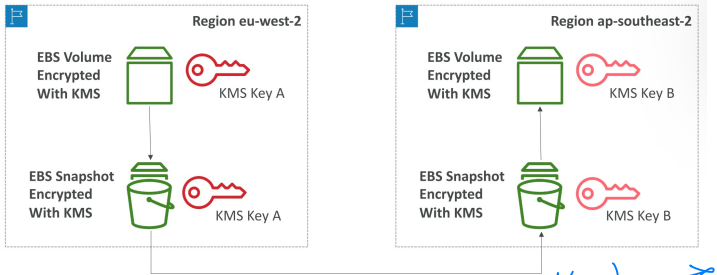
① AWS Owned Keys : SSE-S3, SSE-DDB, SSE-SQS { free }

② AWS Manage Keys :- (aws / service-name) { free }
{automatic rotation every 1 yr} created, managed & used on customer's behalf by AWS.

③ AWS customer managed Keys : custom Keys { must enable AR -ly }
{1$ /month} + imported Keys { manually rotate Keys }

* pay per API call ($0.03 / 10,000 calls)

Copying Snapshots across regions



(done by AWS) (diff Key) → because same KMS Keys cannot live in same region

KMS Key Policies

no access w/o policies!

- Control access to KMS keys, "similar" to S3 bucket policies
- Difference: you cannot control access without them

- Default KMS Key Policy:
  - Created if you don't provide a specific KMS Key Policy
  - Complete access to the key to the root user = entire AWS account
- Custom KMS Key Policy:
  - Define users, roles that can access the KMS key
  - Define who can administer the key
  - Useful for cross-account access of your KMS key

use case

Copying Snapshots across accounts

custom Key policy बनी के लिए बनाते हैं।

1. Create a Snapshot, encrypted with your own KMS Key (Customer Managed Key)
2. Attach a KMS Key Policy to authorize cross-account access
3. Share the encrypted snapshot
4. (in target) Create a copy of the Snapshot, encrypt it with a CMK in your account
5. Create a volume from the snapshot

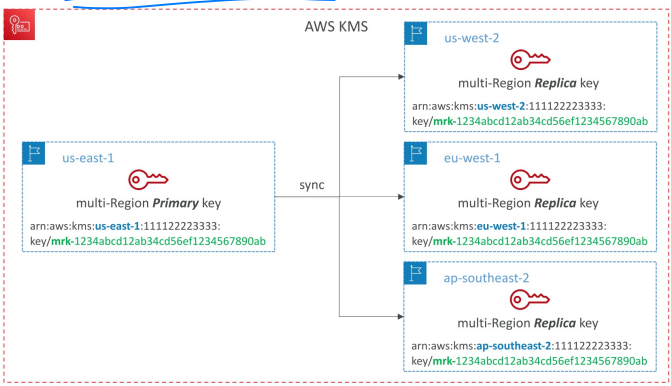
different Key

```
{
    "Sid": "Allow use of the key with destination account",
    "Effect": "Allow",
    "Principal": {
      "AWS": "arn:aws:iam::TARGET-ACCOUNT-ID:role/ROLENAME"
    },
    "Action": [
      "kms:Decrypt",
      "kms:CreateGrant"
    ],
    "Resource": "*",
    "Condition": {
      "StringEquals": {
        "kms:ViaService": "ec2.REGION.amazonaws.com",
        "kms:CallerAccount": "TARGET-ACCOUNT-ID"
      }
    }
}
```

KMS Key Policy

go through the KMS hands on! for CLI

KMS Multi-Region Keys



* Key id needs to be same *

(1) Same Key ID
(2) Same Key material
(3) Same automatic rotation feature.

→ for encrypt in 1 region & decrypt in another.

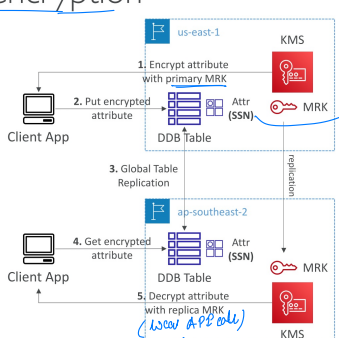
→ no need to re-encrypt when moving data from one region to another (like we did in EBS snapshots)

→ NOT global (Primary + Replicas)
(each multi-region key is managed independently)
+ not much recommended!

use case ⇒ global CSE, Global DynamoDB tables, Global Aurora.

DynamoDB Global Tables and KMS Multi-Region Keys Client-Side encryption

- We can encrypt specific attributes client-side in our DynamoDB table using the Amazon DynamoDB Encryption Client
- Combined with Global Tables, the client-side encrypted data is replicated to other regions
- If we use a multi-region key, replicated in the same region as the DynamoDB Global table, then clients in these regions can use low-latency API calls to KMS in their region to decrypt the data client-side
- Using client-side encryption we can protect specific fields and guarantee only decryption if the client has access to an API key

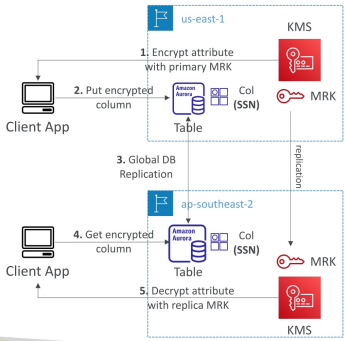


(social security Numbers = SSN)

- if the DB admins don't have access to the KMS Key then they cannot read the SSN
- protection even against DB admins.

Global Aurora and KMS Multi-Region Keys Client-Side encryption

- We can encrypt specific attributes client-side in our Aurora table using the AWS Encryption SDK
- Combined with Aurora Global Tables, the client-side encrypted data is replicated to other regions
- If we use a multi-region key, replicated in the same region as the Global Aurora DB, then clients in these regions can use low-latency API calls to KMS in their region to decrypt the data client-side
- Using client-side encryption we can protect specific fields and guarantee only decryption if the client has access to an API key, we can protect specific fields even from database admins



S3 Replication Encryption Considerations

- Unencrypted objects and objects encrypted with SSE-S3 are replicated by default
- Objects encrypted with SSE-C (customer provided key) can be replicated
- For objects encrypted with SSE-KMS, you need to enable the option
  - Specify which KMS Key to encrypt the objects within the target bucket
  - Adapt the KMS Key Policy for the target key
  - An IAM Role with kms:Decrypt for the source KMS Key and kms:Encrypt for the target KMS Key
  - You might get KMS throttling errors, in which case you can ask for a Service Quotas increase
- You can use multi-region AWS KMS Keys, but they are currently treated as independent keys by Amazon S3 (the object will still be decrypted and then encrypted)

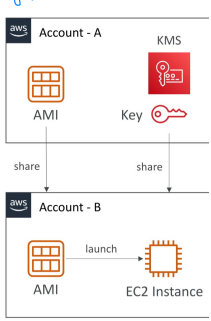
for tg bkt
- specify which Key
- adapt to Key policy
- IAM role to encrypt & decrypt.

using the same key even though its multi-region.

AMI sharing process (via KMS)

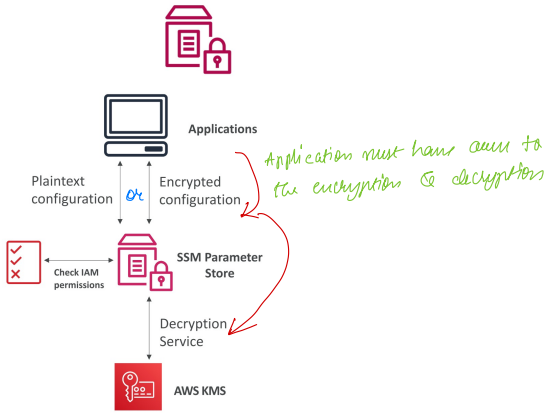
using Key policy!

1. AMI in Source Account is encrypted with KMS Key from Source Account
2. Must modify the image attribute to add a Launch Permission which corresponds to the specified target AWS account
3. Must share the KMS Keys used to encrypted the snapshot the AMI references with the target account / IAM Role
4. The IAM Role/User in the target account must have the permissions to DescribeKey, ReEncrypted, CreateGrant, Decrypt
5. When launching an EC2 instance from the AMI, optionally the target account can specify a new KMS key in its own account to re-encrypt the volumes



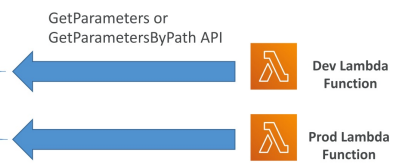
SSM Parameter Store

- Secure storage for configuration and secrets
- Optional Seamless Encryption using KMS
- Serverless, scalable, durable, easy SDK
- Version tracking of configurations / secrets
- Security through IAM
- Notifications with Amazon EventBridge
- Integration with CloudFormation



SSM Parameter Store Hierarchy

- /my-department/
  - my-app/
    - dev/
      - db-url
      - db-password
    - prod/
      - db-url
      - db-password
  - other-app/
- /other-department/
- /aws/reference/secretsmanager/secret_ID_in_Secrets_Manager
- /aws/service/ami-amazon-linux-latest/amzn2-ami-hvm-x86_64-gp2 (public)



Parameters Policies (for advanced parameters)

- Allow to assign a TTL to a parameter (expiration date) to force updating or deleting sensitive data such as passwords
- Can assign multiple policies at a time

Expiration (to delete a parameter)

```
{
    "Type":"Expiration",
    "Version":"1.0",
    "Attributes":{
        "Timestamp":"2020-12-02T21:34:33.000Z"
    }
}
```

ExpirationNotification (EventBridge)

```
{
    "Type":"ExpirationNotification",
    "Version":"1.0",
    "Attributes":{
        "Before":"15",
        "Unit":"Days"
    }
}
```

NoChangeNotification (EventBridge)

```
{
    "Type":"NoChangeNotification",
    "Version":"1.0",
    "Attributes":{
        "After":"20",
        "Unit":"Days"
    }
}
```

AWS Secrets Manager

→ exclusively meant for storing secrets!

→ force rotations in n days

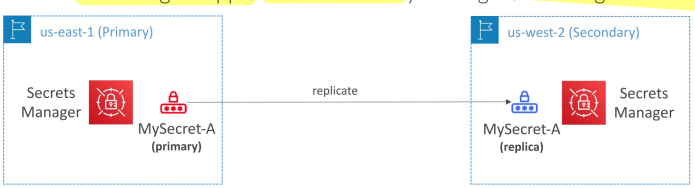
→ 2 integrations to generate new secrets.

→ RDS, Aurora & other DB services के लिए the username & password can be stored in secrets manager!

** integration with RDS for storing secrets! **

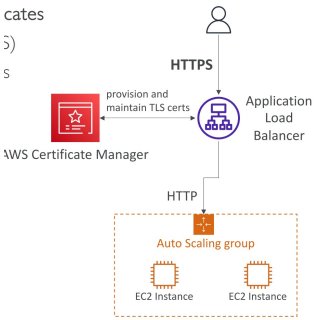
AWS Secrets Manager – Multi-Region Secrets

- Replicate Secrets across multiple AWS Regions
- Secrets Manager keeps read replicas in sync with the primary Secret
- Ability to promote a read replica Secret to a standalone Secret
- Use cases: multi-region apps, disaster recovery strategies, multi-region DB…



AWS Certificate Manager (ACM)

→ Easily provision & manage "TLS certificates" (to provide in-flight encryption (HTTPS))



→ attach @ load balancers to act as an HTTPS endpoint for users to connect to.

→ supports public & private certs. (free public certs)

→ TLS certs renewal

→ load TLS certs into → ELBs, API gateways & APIs, cloudfront distros.

* cannot create public certs for "EC2" using ACM

* ACM – Requesting Public Certificates → how to request certificates?

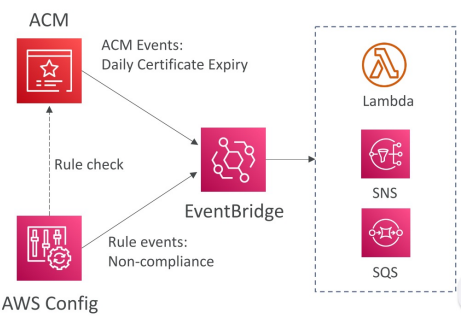
1. List domain names to be included in the certificate
   - Fully Qualified Domain Name (FQDN): corp.example.com
   - Wildcard Domain: *.example.com
2. Select Validation Method: DNS Validation or Email validation
   - DNS Validation is preferred for automation purposes
   - Email validation will send emails to contact addresses in the WHOIS database → registrar for domain
   - DNS Validation will leverage a CNAME record to DNS config (ex: Route 53) → if route 53 enabled then automatic!
3. It will take a few hours to get verified
4. The Public Certificate will be enrolled for automatic renewal
   - ACM automatically renews ACM-generated certificates 60 days before expiry

→ When importing certificates generated outside of ACM then we need to take care of certificate renewal. (No automatic renewal)

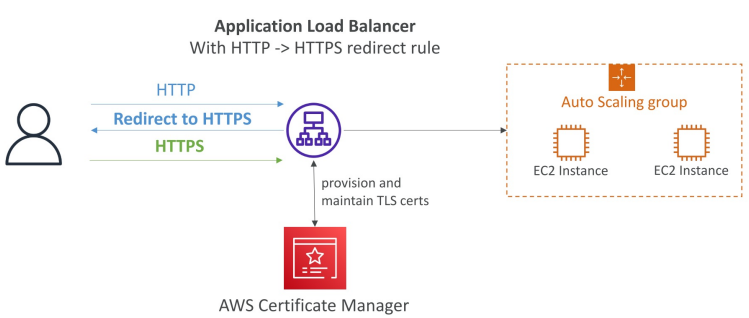
Two ways get automated alws.

→ ACM will send expiration events starting 45 days prior
(1) expiration (configurable) to the event bridge which can be further pushed to SNS topic to recieve notifications
or
2 or SQS queue.

→ AWS Config has a managed rule called acm-certificate-
(2) expiration-check to check for expiring certificates & alert (any certificate not compliant) event bridge.



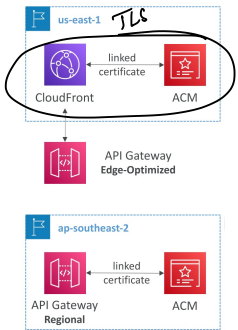
(1) ACM – Integration with ALB



2

ACM – Integration with API Gateway

- Create a **Custom Domain Name** in API Gateway
- **Edge-Optimized (default):** For global clients
  - Requests are routed through the CloudFront Edge locations (improves latency)
  - The API Gateway still lives in only one region
  - The TLS Certificate must be in the same region as CloudFront, in us-east-1 (CloudFront is located here)
  - Then setup CNAME or (better) A-Alias record in Route 53
- **Regional:**
  - For clients within the same region
  - The TLS Certificate must be imported on API Gateway, in the same region as the API Stage
  - Then setup CNAME or (better) A-Alias record in Route 53



ACM

Reminder !

API Gateway - Endpoint Types

- **Edge-Optimized (default):** For global clients
  - Requests are routed through the CloudFront Edge locations (improves latency)
  - The API Gateway still lives in only one region
- **Regional:**
  - For clients within the same region
  - Could manually combine with CloudFront (more control over the caching strategies and the distribution)
- **Private:**
  - Can only be accessed from your VPC using an interface VPC endpoint (ENI)
  - Use a resource policy to define access

to integrate we first need to create a resource in API gateway called the custom domain name.

AWS WAF

→ Layer 7 (Application layer) exploits. protects web applications

→ deployed at

1. ALB (Not NLB)
2. API gateway
3. cloudfront
4. App sync GraphQL API
5. Cognito user pool

* Remember only these !

AWS WAF – Web Application Firewall

- Define Web ACL (Web Access Control List) Rules:
  - **IP Set:** up to 10,000 IP addresses – use multiple Rules for more IPs
  - HTTP headers, HTTP body, or URI strings Protects from common attack - SQL injection and Cross-Site Scripting (XSS)
  - Size constraints, geo-match (block countries)
  - Rate-based rules (to count occurrences of events) – for DDoS protection

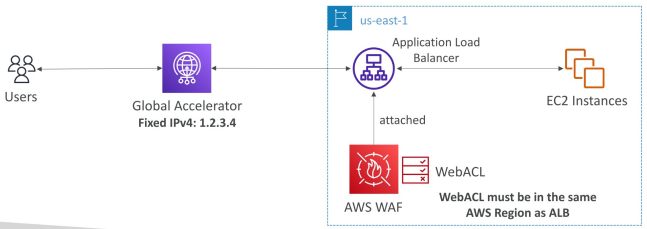
filters

prevent particular IPs from sending too many requests

- Web ACL are Regional except for CloudFront
- A rule group is a reusable set of rules that you can add to a web ACL

WAF – Fixed IP while using WAF with a Load Balancer

- WAF does not support the Network Load Balancer (Layer 4) → so need to use ALB & but it does not have fixed ip 3
- We can use Global Accelerator for fixed IP and WAF on the ALB



AWS Shield { DDOS protection }

① AWS Shield Standard : (free & activated for every AWS user)
- → provides protection from SYN / UDP floods,
- → layer 3 / layer 4 attacks protection.

② AWS Shield Advanced : ($3000 per month per org)
- → 24/7 access to DDOS response team (DRP)
- → mitigate attacks on
  - EC2
  - ELB
  - cloudfront
  - Global Accelerator
  - Route 53
- → mitigates DDOS (means protect against DDOS ↓ ₹210 price spikes)
- → automatic application layer DDOS mitigation

(Access to SRT (Shield Response team))

* AWS firewall manager

→ manage all firewall rules in all AWS accounts in an AWS organization.

Security policy: set of rules { created the policy @ region level }

1. WAF rules
2. AWS shield adv.
3. SGs for EC2, ALB & ENI in VPC
4. AWS Network firewall (VPC level)
5. Route 53 DNS firewall

→ Rules get automatically applied to new resource as they are created (for compliance) in all future accounts of the AWS organization

WAF vs. Firewall Manager vs. Shield


AWS WAF


AWS Firewall Manager


AWS Shield

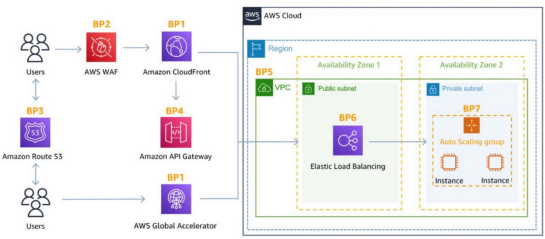
- WAF, Shield and Firewall Manager are used together for comprehensive protection
- Define your Web ACL rules in WAF
- For granular protection of your resources, WAF alone is the correct choice
- * If you want to use AWS WAF across accounts, accelerate WAF configuration, automate the protection of new resources, use Firewall Manager with AWS WAF
- Shield Advanced adds additional features on top of AWS WAF, such as dedicated support from the Shield Response Team (SRT) and advanced reporting.
- If you're prone to frequent DDoS attacks, consider purchasing Shield Advanced

DDOS best practice & solution Architecture

AWS Best Practices for DDoS Resiliency

Edge Location Mitigation (BP1, BP3)

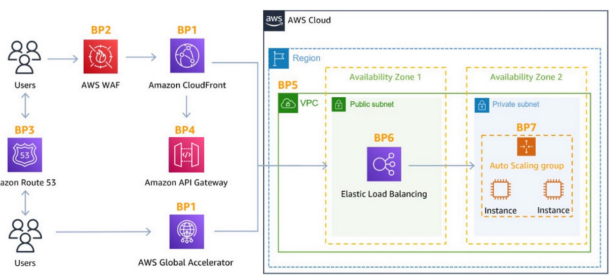
- BP1 – CloudFront
  - Web Application delivery at the edge
  - Protect from DDoS Common Attacks (SYN floods, UDP reflection…) & shield
- BP1 – Global Accelerator
  - Access your application from the edge
  - Integration with Shield for DDoS protection
  - Helpful if your backend is not compatible with CloudFront
- BP3 – Route 53
  - Domain Name Resolution at the edge
  - DDoS Protection mechanism



better DDOS protection @ edge.

- Infrastructure layer defense (BP1, BP3, BP6)
  - Protect Amazon EC2 against high traffic
  - That includes using Global Accelerator, Route 53, CloudFront, Elastic Load Balancing
- Amazon EC2 with Auto Scaling (BP7)
  - Helps scale in case of sudden traffic surges including a flash crowd or a DDoS attack
- Elastic Load Balancing (BP6)
  - Elastic Load Balancing scales with the traffic increases and will distribute the traffic to many EC2 instances

if some reach EC2



- Detect and filter malicious web requests (BP1, BP2)
  - CloudFront cache static content and serve it from edge locations, protecting your backend
  - AWS WAF is used on top of CloudFront and Application Load Balancer to filter and block requests based on request signatures
  - WAF rate-based rules can automatically block the IPs of bad actors
  - Use managed rules on WAF to block attacks based on IP reputation, or block anonymous Ips
  - CloudFront can block specific geographies
- Shield Advanced (BP1, BP2, BP6)
  - Shield Advanced automatic application layer DDoS mitigation automatically creates, evaluates and deploys AWS WAF rules to mitigate layer 7 attacks

Attack surface reduction

- Obfuscating AWS resources (BP1, BP4, BP6)
  - Using CloudFront, API Gateway, Elastic Load Balancing to hide your backend resources (Lambda functions, EC2 instances)
- Security groups and Network ACLs (BP5)
  - Use security groups and NACLs to filter traffic based on specific IP at the subnet or ENI-level
  - Elastic IP are protected by AWS Shield Advanced
- Protecting API endpoints (BP4)
  - Hide EC2, Lambda, elsewhere
  - Edge-optimized mode, or CloudFront + regional mode (more control for DDoS)
  - WAF + API Gateway: burst limits, headers filtering, use API keys

& filtering any http req &

Amazon Guard duty

→ intelligent threat discovery to protect AWS account

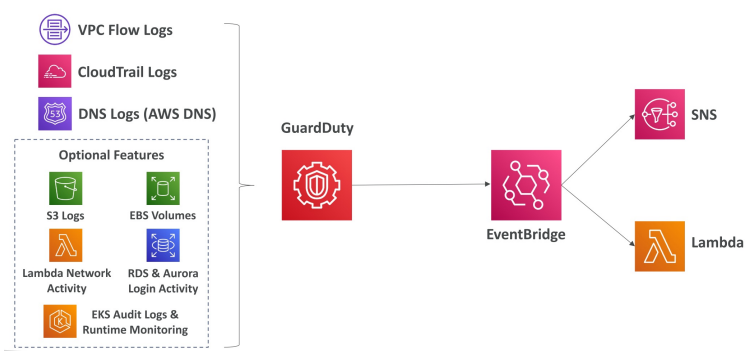
→ uses ML for anomaly detection.

→ one click enable

1. cloud trail events → management events / data events
2. VPC flow logs → unusual traffic, ips
3. DNS logs → compromised EC2
4. other logs events.

→ for findings add eventbridge rules

→ protect against "Crypto attacks!"



Amazon Inspector

→ run automated security assessments!

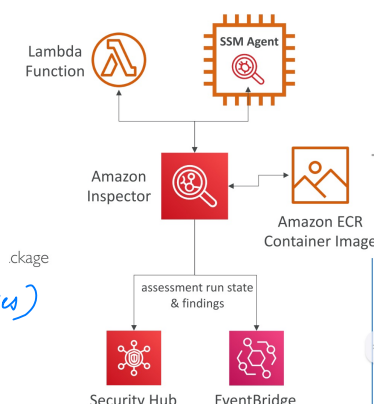
(1) For EC2 instances
→ Analyze unintended network accessibility, running OS, against known vulnerabilities.
→ using AWS SSM agent (System)

(2) For containers
→ container assessment as being pushed to ECR.

(3) For Lambda
→ as the functions are being developed.
→ Reports the finding to AWS Security Hub.
& set events to Event Bridge.

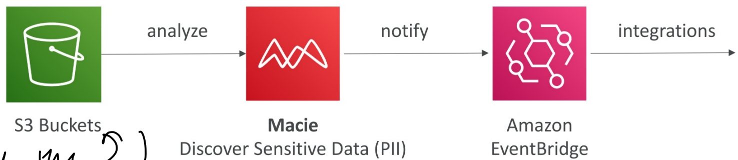
What does Amazon Inspector evaluate?

- Remember: only for EC2 instances, Container Images & Lambda functions
- Continuous scanning of the infrastructure, only when needed!
- Package vulnerabilities (EC2, ECR & Lambda) – database of CVE (vulnerabilities)
- Network reachability (EC2)
- A risk score is associated with all vulnerabilities for prioritization



Amazon Macie

- → fully managed data security & data privacy service
- → uses pattern matching & machine learning!
- → protects sensitive data in AWS and alerts about PII (personally identifiable info)



(specify IAM ?)
done!