Overview

Security

Data needs to be protected, and any possible threats must be clearly identified.

Identity

Any user should only be able to access the minimum required resources.

> !! Various solutions to achieve above

Relevant services

Data Protection

1. **Amazon Macie** - Discover and protect sensitive data
2. **AWS KMS (Key Management Service)** - Store and manage Encryption keys
3. **AWS CloudHSM** - Hardware based key storage
4. **AWS Certificate Manager** - Issue SSL and TLS certificates
5. **AWS Secrets Manager** - Rotate, manage and retrieve secrets

Infrastructure Protection

1. **AWS Shield** - DOS (Denial of Service) Protection
2. **AWS Web Application Firewall** - Filter malicious website traffic
3. **AWS Firewall Manager** - Centrally manage firewall rules

Threat Detection

1. **Amazon GuardDuty** - Automatically Detect threats
2. **Amazon Inspector** - Analyze app security
3. **AWS Config** - Record and evaluate configurations of AWS resources.

4. **AWS CloudTrail** - Track usage of activity and/or API

Indentity Management

1. **AWS IAM** - Manage access to AWS account services and resources
2. **AWS SSO** - Single sign on implementation service
3. **Amazon Cognito** - Manage identity inside apps
4. **AWS Directory Service** - Implement and manage, Microsoft Active Directory
5. **AWS Organizations** - Centrally Govern and manage multiple AWS accounts

IAM

!! AWS Identity and Access Management

!! Free for all AWS accounts

- Manage who can access what in an AWS account
- Create users and groups
- Allow or Deny access via policies

IAM Users

1. **Root user** - Granted on account creation
2. **IAM user** - Created by a root user, or an IAM user with permission to create other IAM Users.

!! Created via the IAM console

Policies

1. **Effect**: Allow or Deny
2. **Action**: What can be done
3. **Resource**: What it applies to

Example

1. Allow all actions to all resources (admin access to account)

```json
{
    "Version" : "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "*",
            "Resource": "*"
        }
    ]
}
```

1. Let user read and store files from aws s3 buckets.

```json
{
    "Version" : "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "s3:ListBucket"
            ],
            "Resource": "arn:aws:s3:::bucket-name/*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "s3:PutObject",
                "s3:GetObject"
            ],
            "Resource": "arn:aws:s3:::bucket-name/**"
        }
    ]
}
```

IAM Roles

- Both a user and a service can assume a role.
  Ex: Assume a role that grants access to a service (like a database). The role can grant a user (DB admin) to access the service, or another service (A website/backend) to access the service as well.
- Can enable users from different AWS accounts as well.

Secrets Manager

- Protects the secrets required to access your resources.
- Rotates automatically.
- Stores passwords, keys, and tokens.
- Code requiring said secrets can fetch them at runtime via several kinds of APIs.
- Secrets can also be automatically rotated as necessary.

Directory Service

- Used for networks like managed computers (for office use for example).
- Helps out in providing the Microsoft Active Directory service for Windows systems.
- Provides a simple active directory for opting out of Microsoft service as well.
- Provides an AD connector, which allows users to access AWS applications with their AD login.
- Redundancy via distributed service (automatic failover).
- Compatible with other AWS services.