IAM

Users - any individual end user such as an employee, system architect

Groups - any collection of similar people with shared permissions such as system administrators, HR employees, finance teams, etc. Each user within their specified group will inherit the permissions set for the group.

Roles - any software service that needs to be granted permissions to do its job, e.g- AWS Lambda needing write permissions to S3 or a fleet of EC2 instances needing read permissions from a RDS MySQL database.

Policies - the documented rule sets that are applied to grant or limit access. In order for users, groups, or roles to properly set permissions, they use policies. Policies are written in JSON and you can either use custom policies for your specific needs or use the default policies set by AWS.

IAM Policies are separated from the other entities above because they are not an IAM Identity. Instead, they are attached to IAM Identities so that the IAM Identity in question can perform its necessary function.

IAM Key Details:
IAM is a global AWS services that is not limited by regions. Any user, group, role or policy is accessible globally.

The root account with complete admin access is the account used to sign up for AWS. Therefore, the email address used to create the AWS account for use should probably be the official company email address.

New users have no permissions when their accounts are first created. This is a secure way of delegating access as permissions must be intentionally granted.

When joining the AWS ecosystem for the first time, new users are supplied an access key ID and a secret access key ID when you grant them programmatic access. These are created just once specifically for the new user to join, so if they are lost simply generate a new access key ID and a new secret access key ID. Access keys are only used for the AWS CLI and SDK so you cannot use them to access the console.

When creating your AWS account, you may have an existing identity provider internal to your company that offers Single Sign On (SSO). If this is the case, it is useful, efficient, and entirely possible to reuse your existing identities on AWS. To do this, you let an IAM role be assumed by one of the Active Directories. This is because the IAM ID Federation feature allows an external service to have the ability to assume an IAM role.

IAM Roles can be assigned to a service, such as an EC2 instance, prior to its first use/creation or after its been in used/created. You can change permissions as many times as you need. This can all be done by using both the AWS console and the AWS command line tools.

You cannot nest IAM Groups. Individual IAM users can belong to multiple groups, but creating subgroups so that one IAM Group is embedded inside of another IAM Group is not possible.

With IAM Policies, you can easily add tags that help define which resources are accessible by whom. These tags are then used to control access via a particular IAM policy. For example, production and development EC2 instances might be tagged as such. This would ensure that people who should only be able to access development instances cannot access production instances.

Priority Levels in IAM:
Explicit Deny: Denies access to a particular resource and this ruling cannot be overruled.

Explicit Allow: Allows access to a particular resource so long as there is not an associated Explicit Deny.

Default Deny (or Implicit Deny): IAM identities start off with no resource access. Access instead must be granted