Amazon Web Server (AWS) training Syllabus

In this 5-days course you will learn how to gradually move part of your infrastructure/application to the cloud, and experience the benefits provided by an IaaS, SaaS.

We will start by creating a simple web-server from scratch, by starting a linux server and install everything by hand. Later we will learn the ways to automate every step in the process. Create some custom images, further customize them with init-script and metadata. Later we add a database backend to the sample (Imagine a custom Wordpress application).

To make the example more highly available, we introduce zones and regions, and explain how the network works in those situations. With monitoring key performance indicators of your application, you can take advantage of autoscaling.

Finally, you can make the whole infrastructure provisioning automated by using CloudFormation.

Product overview:

- EC2: VM as a service:
  - autoscaling (mem/cpu metrics-based instance start/stop)
  - EBS (persistent storage)
  - Images (how to create your own software stack)
  - ELB (managed Load Balancer)
  - Virtual Networks:
    - subnets
    - reserved IP
    - firewall rules
- Management:
  - CloudTrail (audit of all users AWS usage)
  - CloudWatch (metrics/log collection)

    - CloudFormation (aws specific "terraform")
- LightSail: preconfigured stacks: LAMP, Nginx, MEAN and Node.js. Easier to manage than pure ec2 instances. - [optional]
- Databases:
    - DynamoDB (amazon's proprietary schemaless NoSql) written in JAVA, can be self-hosted, billed by per request, fully managed. - [optional]
    - RDS (hosted relational db)
        - postgres
        - mariadb (mysql)
        - oracle
        - sql server
    - DocumentDB (hosted mongodb) - [optional]
    - ElastiCache (hosted redis) - [optional]
- Storage options, availability/speed/price:
    - EBS (disk as a service)
        - snapshots
    - S3 (object storage, rest api, industry standard)
    - EFS (nfs as a service)
- Pricing: AWS makes it really complicated:
    - reserved instances (prepay for 1-3 years)
    - spot price (can be 70% cheaper, with some risk)
- IAM
    - Creating Roles, Policies, binding them
    - authentication best practices (no root account, rotate keys, 2fa)
    - Organizing teams
    - Even create sub accounts
- HTTPS: learn how to use Letsencrypt. Not AWS specific, but important to know about a free automated was of creating certificates.

Optional:

- Lambda (Serverless): no need to pay for a VM if an app is used only rarely:
    - Wide range of language can be used: javascript/python/java/go ...
    - Lot of Infrastructure automation possibility, almost all service can

trigger a lambda function
    - CI/CD hook triggers
    - chatops (bot listens on slack, and deploys a VM...)
    - billed by 100ms
    - API Gateway: rest endpoint implemented as Lambda
- Application Integration:
    - SQS: managed queue. Easy glue between apps / services
    - SNS: manages topic/subscription service:
        - system to system communication: triggers: lambda/sqs/http endpoint
        - user notification: email, sms, push notification
- Elastic Beanstalk: dev uploads a bundle (jar/zip) beanstalk will create VM instance/network/ ... resources automatically.
- Route53: managed DNS
- Containers:
    - ECS (AWS opinionated docker solution)
    - EKS (managed kubernetes)

Exercises:

Webserver - AWS Console

Goal: create a Webserver via the AWS provided browser-based GUI.

- setup shortcuts, bookmarks, resource group
- Create Keypairs (ssh)
- start an Ubuntu/Debian instance

Webserver - cli

Goal: Same as previous exercise, but everything from the terminal.

- aws sdk installation
- shell scripting basics (functions/variables/loops)

Webserver - Spot Instance

Goal: Learn how to save up to 70% of the cost for dev/qa environments.

Webserver - EBS – snapshot

Goal: learn how to use persistent disks, instead of volatile instance storage.

Create an Image

Goal: Instead of having a word docs describing what to install/configure, create a reusable blueprint.

- start with an "artisan" image (quick and dirty)
- automate it with packer

Instance Templates

Goal: Instead writing word docs about how to start an instance (which region, which instance type, how big the disk, networks, subnetworks ...) his is a prerequisite for the next:

Autoscaling and Load Balancing

Goal: how to survive Black Friday? Automation which checks cpu/memory load metrics, and scales up/down.

Database management

Goal: Understand the pro/cons (hint: price is the only issue) of maintaining the DB or let AWS manage it.

WebApplication the AWS way

- Static webhosting on S3
- Serverless Web app: Lambda + S3 + managed DB
- Beanstalk
- LightSail

Serverless

Goal: Understand how Lambda fits into both infrastructure glue code/ automation, and also complete applications (rest API).