Deployment & Provisioning

Deploy an EC2

Running on spot options:

- Add max price
- Persistent request
- Interruption behaviour:
  - Terminate
  - Stop (lose data on RAM)
  - Hibernate (Keep RAM)
- Request valid from/to
- Launch group (only launches when all can launch)
- Placement group: Add instances to one AZ
- Enable Termination protection
- Shutdown Behavior (stop/terminate)
- Enable detailed monitoring
- Tenancy (Shared - Run a shared hardware)
- T2 unlimited (Burst CPU)
- User data (bootscripts)

EC2 Launch Issues

Common issues:

- InstanceLimitExceeded error: Default limit 20 per region limit (AWS support to raise limit)
- InsufficientInstanceCapacity error: AWS does not have enough available on-demand capacity (wait few minutes / request fewer instance types / select other instance types / purchase reserved instances / Submit request without AZ)

EBS Volumes and IOPS

- gp2 : minimum 100 IOPS to max 16 000 IOPS
- io1 (provisioned iops => databases) : minimum 50 IOPS/Gb to max 64 000 IOPS

Hitting limit gp2 iops => I/O request queuing => Application becomes slow

- raise gp2 volume size
- already 16 000 iops => change to io1

Elatic Loadbalancers

- Application loadbalancer (layer 7)
- Network loadbalancer (layer 4 => Handles millions of request per second)
- Classic loadbalancer (X-forwarded and sticky sessions)

Pre-warming loadbalancer => contact AWs support to pre-warm to handle spikes

- start and enddate
- expected req/sec
- total size of typical request

ALB changes ip addresses when scaling

Network loadbalancers create static ip per subnet (good for firewalling)

Solution: Put an ALB behind a network loadbalancer for static ip

ELB Error Messages

Classic and ALB:

- 200 => success
- 4xx client side error
- 5xx server side error

Client side error:

- 400 => Bad/malformed request (header malformed)
- 401 => unauthorized
- 403 => Forbidden (blocked by WAF access control list)
- 460 => Client closed connection before loadbalancer could respond
- 463 => Loadbalancer received X-forwarded-For header with > 30 ips

Server side error:

- 500 => Internal server error (loadbalancer)
- 502 => Bad Gateway (application server closed connection)
- 503 => Service unavailable (no registered targets)
- 504 => Gateway timeout
- 561 => unauthorized (identity provider)

ELB Cloudwatch Metrics

Loadbalancer have default Cloudwatch metrics and also for the backends

- BackendConnectionErrors => number of unsuccessful connections to the backend instances
- HealthyHostCount
- UnHealthyHostCount
- HTTPCode_Backend_2xx,3xx,4xx,5xx
- Latency => number of second taken for instance to respond
- RequestCount => number of request completed
- SurgeQueueLength => number of pending requests - max of 1024 (Classic only)
- SpolloverCount => number of requests rejected when surge queue is full (Classic only)

AWS Systems Manager

- Management-tool which give you control over AWS infrastructure.
- Integrates with Cloudwatch allowing you view your dashboards, view operation data & detect problems.
- Includes Run command which automates operational tasks across resources - f.e. security patches, package installs.
- Organize your inventorym grouping resources together by application or environment (including on-premise)

Run-command

- Allow you to run pre-defined command on one or more EC2 instances.
- Stop, restart, terminate, resize instance
- Attach/detatch EBS volumes
- Create snapshots, backup DynamoDB tables

- Apply patches and updates
- Run an Ansible playbook
- Run shell scripts

Use

- Create role in IAM for EC2 (EC2RoleforSSM)
- Attach role to EC2
- In SSM, find resource group (create resource group)
    - Build-in insights
        - view Cloudtrail
        - AWS Config
        - Personal Health Dashboard
        - Trusted Advisor
            - Cost optimizations
            - Performance
            - Security recommendations
            - Fault Tolerance
            - Service Limits
    - Dashboards in Cloudwatch
    - Inventory (Top OS, Top Services, ...)
    - Compliance (see patches applied)
    - Automation (Run commands on instances, automated/in steps, create your own documents)
    - Run command (pre-configured or own scripts f.e AWS-RunShellScript document)
        - SNS notification
        - Output to S3
        - Shell script box
        - Apply shows you the output
    - Patch Manager
    - Maintenance Windows (cron scheduler, duration)
    - State Manager (ensure consistent state - reapply when state is no compliant)
    - Managed instances
    - Activations (Register EC2 instances and on-premise - Install ssm agent)
    - Documents (Create your own documents / view existing)

    - Parameter Store (secrets)

##Placement Groups

By default AWS places instances across different physical hardware. This minimizes the impact of a hardware failure. Not so great for low latency, high network throughput applications.

Types:

- Cluster: Instances are all created in a single AZ
    - Full line rate of 10 Gbps
    - Not for high availability
- Partition: Instances are created in logical segments called partitions, each located in a separate rack, with independent network and power. Some instances could be in the same rack.
    - Great for HDFS, HBase, and Cassandra
- Spread: Each instance is created in a separate rack, with independent network and power.
    - Maximum availability