ADVANCED EC2

Bootstrapping EC2 using User Data

EC2 Bootstrapping is the process of configuring an EC2 instance to perform automated install & configuration steps 'post launch' before an instance is brought into service. With EC2 this is accomplished by passing a script via the User Data part of the Meta-data service - which is then executed by the EC2 Instance OS

- tl;dr this allows you to bring and EC2 instance up in a pre-configured state
- EC2 Bootstrapping = EC2 Build Automation
- This process uses User Data via meta-data IP (http://169.254.169.254/latest/user-data) - Note, this is instead of /meta-data -- EC2 doesn't interpret this data, the OS needs to understand the User Data (eg. Linux needs to understand bash, otherwise it won't work)
- This boostrapping `ONLY` happens on Launch. If you update User Data after the fact, it won't be applied
- User Data is OPAQUE to EC2, its just a block of data to EC2
- User Data is NOT secure; don't use it for passwords or long term creds
- User Data limited to 16KB in size
- If you want to update User Data... stop instance, update data, and restart

EXAM: Boot-time-to-Service-Time: how quickly you can bring an instance into service and have it ready for use by customers

- measured in minutes (from AMI to functional instance) if you have to do a lot of manual config, this time increases

DEMO - Bootstrapping WP Installation (parts 1 and 2)

Bootstrap two EC2 instances with WordPress and the 'cowsay' login banner customizations.

- The first, directly using User Data via the console UI
- the second, using CloudFormation

Steps:

1. 1-click deploy > Create stack
2. EC2 > Launch Instance (x2) > Name "A4L-MANUALWORDPRESS" > keypair: proceed without > Network Settings, Edit
3. Instance Network Settings: VPC, select a4l-vpc1 > subnet, select sn-web-A > Select Existing Security Group, BOOTSTRAP-[etc] > Advanced Details, User Data, paste in .txt file provided > Launch Instance
4. EC2 Instance Connect > Will see custom Banner above normal stuff if you wait long enough. Paste in the commands in the command file to see user data Note: Within EC2 Connect, "cd /var/log" you can use the cloud-init-output.log and cloud-init.log to diagnose any bootstrapping issues
5. CloudFormation > 1-click deploy > etc etc
6. Clean up > Terminate EC2 instance > Delete CFN stack

EC2 Instance Roles & Profile

EC2 Instance Roles/Profiles are how applications running on an EC2 instance can be given permissions to access AWS resources on your behalf. Short-term temporary creds are available via the EC2 instance Metadata and are renewed automatically by the EC2 and STS Services

Instance Role Architecture uses an IAM role that allows EC2 instance to assume it. There is an InstanceProfile wrapper that actually attached to the instance. Creds delivered within metadata

- creds used to give InstanceProfile IAM role access are inside meta data -- iam/security-credentials/role-name -- creds always valid/auto-rotated
- BEST PRACTICE: Always use Roles when possible, better security. CLI tools will use role creds automatically

DEMO - Using EC2 Instance Roles

Create an EC2 Instance Role, apply it to an EC2 instance and learn how to interact with the credentials this generates within the EC2 instance metadata.

Steps:

1. 1-click deploy
2. EC2 > instance connect > EC2 Instance Connect > test command `aws s3 ls` > need creds
3. Create IAM Role > IAM > Roles > Create Role > Trust Entity, AWS service: select `EC2` > Next, Add Permissions, search `s3`, select AmazonS3ReadOnlyAccess > Next, Role name "A4LInstanceRole" > Create Role
4. Attach Role to Instance: EC2 > Right-click instance, Security: Modify IAM Role > Choose Role A4LInstanceRole
5. You can now `aws s3 ls` command in EC2 Connect because you have applied the IAM Role that has the S3ReadOnly permission to the EC2 instance
6. Clean up > IAM: Delete A4LInstanceRole > EC2: Security: Modify IAM, select NO IAM Role > CFN: Delete Demo stack

EC2 - SSM Parameter Store

The SSM Parameter store is a service which is part of Systems Manager which allows the storage and retrieval of parameters - string, stringlist or secure string.

- The service supports encryption which integrates with KMS, versioning and can be secured using IAM.
- The service integrates natively with many AWS services - and can be accessed using the CLI/APIs from anywhere with access to the AWS Public Space Endpoints.

As previously mentioned, passing secret data through to EC2 instance with User Data is bad practice.

- SSM is storage for configuration and secrets
- Parameter store can store the following: String, StringList, SecureString -- There are hierarchies and Versioning of parameters -- Can be stored as plaintext and ciphertext (integrating with KMS)
- Public Parameters are available, like the latest AMIs per region

DEMO - Parameter Store

Create some Parameters in the Parameter Store and interact with them via the command line - using individual parameter operations and accessing via paths.

Steps:

1. Nav to Systems Manager in AWS Console > Parameter Store > Create parameter
2. Parameter 1 details: Name "/my-cat-app/dbstring" > value "db.allthecats.com:3306" > description "Connection string for cat app" > create

- the fwd slashes create hierarchy in Parameter Store

3. Parameter 2: Name "/my-cat-app/dbuser" > value: "bosscat" > create
4. Param 3 (secure): Name "/my-cat-app/dbpassword" > Type: SecureString > value: amazingsecretpassword1337 (encrypted)
5. Param 4: Name "/my-dog-app/dbstring" > value: "db.ifwereallymusthavedogs.com:3306"
6. Param 5: Name "/rate-my-lizard/dbstring" > value "db.thisisprettyrandom.com:3306"
7. CloudShell (top menu) > try these commands: aws ssm get-parameters --names /rate-my-lizard/dbstring aws ssm get-parameters --names /my-dog-app/dbstring aws ssm get-parameters --names /my-cat-app/dbstring aws ssm get-parameters-by-path --path /my-cat-app/ aws ssm get-parameters-by-path --path /my-cat-app/ --with-decryption
8. Clean up: Select all params > Delete

System and Application Logging on EC2

CloudWatch is for metrics CloudWatch Logs is for logging and interpreting that logged data -> Neither of these natively capture data `inside` an instance. Use CloudWatch Agent to solve this.

- CloudWatch Agent is required for CloudWatch/CloudWatch Logs to natively capture data in an instance (plus config using Agent Config file and permissions):
  - Configured / Permitted CW Agent pulls logs from Instance and sends to CW / CW Logs
  - The permissions the Agent needs will use an IAM Role Installing and configuring the CloudWatch Agent can be done manually or using CloudFormation if this has to be done at scale. The Agent configuration parameters can be stored in the Parameter Store.

DEMO - Logging and Metrics with CloudWatch Agent

Download and install the CloudWatch Agent and configure it to capture 3 log files from an EC2 instance: /var/log/secure, /var/log/httpd/access_log, /var/log/httpd/error_log

Steps:

1. 1-click deploy
2. Connect to instance: EC2 instance connect
3. Install CW Agent to Instance connect with `wget https://s3.amazonaws.com/amazoncloudwatch-agent/amazon_linux/amd64/latest/amazon-cloudwatch-agent.rpm`
4. Use IAM to create Role/Permissions for Agent: IAM > Roles > Create Role > select options: AWS service/EC2 > select (2) policies `CloudWatchAgentServerPolicy` And `AmazonSSMFullAccess` >

next > name "CloudWatchRole" > Create Role
5. Connect Role to Instance: EC2 Instances > right-click instance, Security, Modify IAM role > Select CloudWatchRole > Update IAM Role
6. EC2 Connect to Start CW Agent config wizard: Default OS, Enter > Default region (EC2), enter > user: root, enter > enter statsd, enter statsd port > interval, enter > enter for everything until "Which default metrics config do you want? input 3 for Advanced... [FOLLOW VIDEO for rest of this crap, it's mostly pressing Enter for defaults] - https://learn.cantrill.io/courses/1820301/lectures/41301605
7. Store Config in Parameter Store: Press enter more for Defaults. Note: Most all of it is defaults, pay attn to the few that aren't
8. CloudWatch Console > Log Groups. Look for the three file paths you created
9. Cleanup > EC2, security: remove CW Role > IAM, Roles, Delete role > CFN, delete Stack

EC2 Placement Groups

Allows you to choose placement of EC2 instances, so you can ensure that they're physically close together or not.

3 placement groups available within AWS:

1. Cluster (PERFORMANCE)
2. Spread (Resilience)
3. Partition (Topology Awareness)

- Cluster. For highest EC2 performance. Launch these all at the same time for similar placement/capacity, will be launched into single AZ. Instances in Cluster group have fast direct connections to each other. Cluster achieves 1-Gbps per single stream instead of the standard 5Gbps/stream. Lowest latency and max Packets-per-second possible in AWS (when paired with Enhanced Networking) EXAM: CLUSTER Placement Groups are ONE AZ ONLY, which is locked when launching the first instance. Close prox is what gives them best Performance of placement groups EXAM: VPC peers in cluster placement groups can span AZs, but this impacts performance EXAM: Cluster PGs not supported on all instance types EXAM: Launch all instances within cluster group at same time as best practice EXAM: 10Gbps single stream performance; highest performance, lowest latency, best throughput

- Spread. For max availability and resilience for an application; span AZs to accomplish this. Use case: small number of critical instances that need to be kept separate from each other EXAM: Spread PG instance LIMIT: 7 INSTANCES per AZ (HARD limit) EXAM: Spread PGs provide "infrastructure isolation" to provide max availability and resilience EXAM: Each instance runs from a different rack, each rack has its own network and power source EXAM: Not supported for Dedicated Instances or Hosts

- Partition. Designed for when you have more than 7 instances per AZ but you still need to be able to separate those instances into separate fault domains. Designed for huge-scale parallel processing systems EXAM: For when you need max availability/resilience but MORE THAN 7 instances per AZ EXAM: Max 7 PARTITIONS per AZ, but each partition can contain more than 7 instances EXAM: Each partition has its own rack EXAM: Instances can be placed into a specific partition or can be auto-placed EXAM: for 'TOPOLOGY' aware applications; Eg: HDFS, HBase, Cassandra

EC2 - Dedicated Hosts

It's an EC2 Host dedicated only to you. Dedicated hosts are EC2 Hosts which support a certain type of instance which are dedicated to your account. Eg: a1, c5, m5

- Billing: No EC2 instance charges, you pay for the host
- You can pay an on-demand or reserved price for the hosts and then you have no EC2 instance pricing to pay for instances running on these dedicated hosts.
- Generally dedicated hosts are used for applications which use physical core/socket licensing (number or cores/sockets requirement) or for compliance reasons.
- Type of instance is fixed (eg. a1, c5, m5) and you can't change it. Depending if Nitro virtualization is used, you can run different instance sizes on the same host. Otherwise you can only run the same size instance on the host (eg. Two R5.large but not 1 R5.large and 2 R5.small)

How they work: Host is designed for a specific family and size of instance. Eg. a1 dedicated host has 1 socket and 16 cores

Dedicated Host Limitations (things that aren't supported)

- AMI limits - Not supported: RHEL, SUSE Linux, Windows AMIs
- Cannot used Amazon RDS instances
- Placement Groups not supported

Note: Using RAM (resource access manager) product, hosts CAN BE shared with other ORG accounts

EC2 - Enhanced Networking & EBS Optimized (EC2 optimization features)

Enhanced networking is the AWS implementation of SR-IOV, a standard allowing a physical host network card to present many logical devices which can be directly utilized by instances.

- This means lower host CPU usage, better throughput, lower and consistent latency
- EBS optimisation on instances means dedicated bandwidth for storage networking - separate from data networking.
- Resource: https://docs.aws.amazon.com/AWSEC2/latest/WindowsGuide/enhanced-networking.html

Enhanced Networking

By default EC2 Network Interface Card (NIC) is not aware of virtualization, so it's not as efficient. Enhanced networking makes the NIC aware of virtualization

Enhanced Networking is designed to improve overall performance of EC2 networking--required for high performance things like Cluster Placement Groups.

- Uses technique called SR-IOV (Single Root I/O Virtualization) so that the network interface inside EC2 is aware of virtualization. --> This all increases IO, lower host CPU usage, more bandwidth, and higher packets-per-second (PPS), consistent lower latency.
- Enhanced networking comes with no charge and is available on most EC2 instance types

EBS Optimized Instances

What we know about EBS already: EBS is block storage over the network. EBS Optimized means DEDICATED capacity for EBS traffic. This is separate from the data traffic. -> Most instances are EBS

optimized by default and enabled by default. Some instances support it but enabling costs extra.