How to Build
Reliable, Scalable Filesystem Solution
Using Cloud Infrastructure

—

Sasikanth Eda (sasikanth.eda@in.ibm.com)
Master Inventor, Software Engineer
IBM

Agenda

How Cloud is Falling Short for HPC / Technical Computing

Introduction to Building Blocks

- Parallel Filesystem Architecture [Ex: IBM Spectrum Scale (aka, GPFS)]
- Cloud Infrastructure [Ex: AWS Services / Components]

Solution Models

- Deployment Model
- Management Model

Data Life Cycle Management Practices

- Data export, import Model
- Data migration, tiering Model

How Cloud is Falling Short For HPC - Opportunity

- ✓ According to the latest forecast[1] of Gartner, Worldwide Public Cloud Services revenue is projected to grow in 2020 to total $411.4 billion, up from $219.6 billion in 2016.

- ✓ According to sources[2], The Cloud High Performance Computing (HPC) market accounted to $5.11 billion in 2016 and is expected to reach $15.28 billion by 2022, recording at a CAGR of 20.04% during 2017-2022.

- ✓ [3] 83% of Enterprise Workloads will be in the Cloud by 2020.

  (41% in Public, 22% in Hybrid)

20% CAGR is good, but it can be better !

Sources:
[1] https://www.gartner.com/newsroom/id/3815165
[2] http://bit.ly/MordorIntelligence
[3] http://bit.ly/ForbesEnterpriseArticle

How Cloud is Falling Short For HPC

"The last couple of years have seen cloud computing gradually build some legitimacy within the HPC world, but still the HPC industry lies far behind enterprise IT in its willingness to outsource computational power."

- *Chris Downing, Red Oak Consulting*

Major factors include: Performance, Networking, Data Movement, Storage, Software, Funding & cost management.

Problem with Storage for HPC: *For more demanding users, the problems get worse – none of the built-in storage solutions available across the public cloud providers is going to be suitable for applications with high bandwidth requirements.*

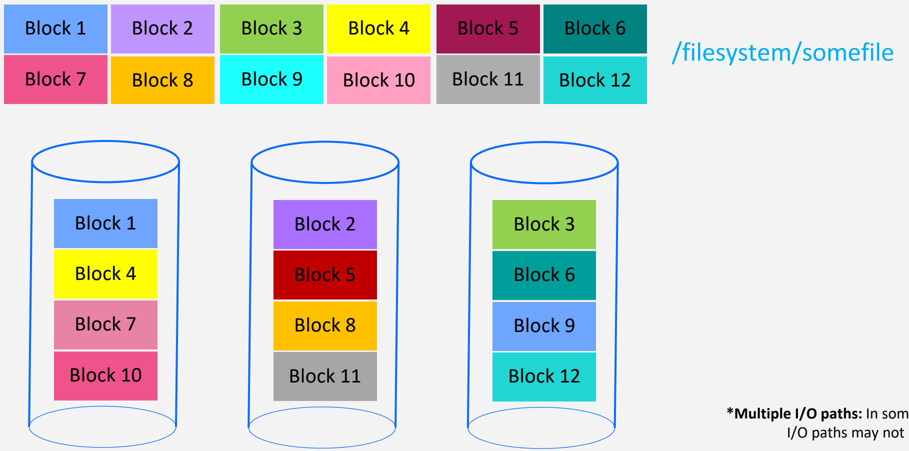
Parallel file systems built on top of block storage are the obvious fix.

Source: http://bit.ly/HPCWireCloudFallingShort

Introduction: Parallel Filesystem Architecture

- ✓ A parallel file system typically breaks up the data set and distributes, or stripes, the blocks to multiple storage drives, which can be located in local and/or remote servers.

- ✓ It can read and write data to distributed storage devices using *multiple I/O paths concurrently (significant performance – high throughput benefit).



*Multiple I/O paths: In some cases (especially with Flash drives), multiple I/O paths may not be needed.

Introduction: Parallel Filesystem Architecture

(Continued..)

- ✓ Parallel file systems are well suited for high-performance computing (HPC/HPTC) environments that require access to large files, massive quantities of data, and simultaneous access from multiple compute clients.

- ✓ Fields: Climate modelling, CAD, data analysis, financial modelling, genomic sequencing, ML / DL, seismic processing and multimedia rendering etc.

- ✓ Over the years, many features were include high availability, information lifecycle management, mirroring, replication, encryption, compression, WAN caching, snapshots and many more.

- ✓ Ex: IBM Spectrum Scale (aka GPFS), Lustre, Gluster, Panasas PanFS etc.

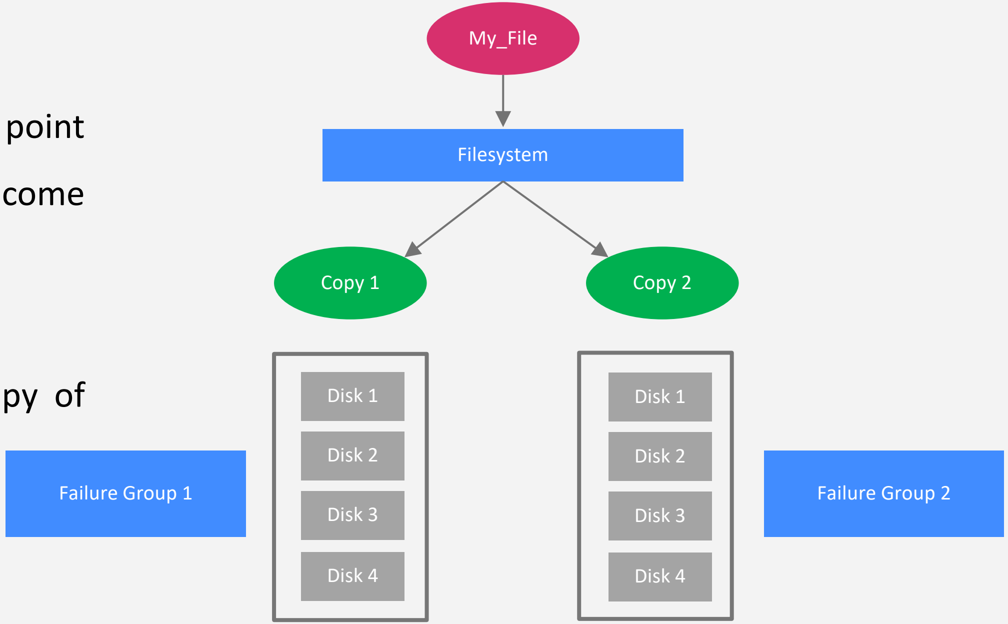
Introduction: Parallel Filesystem Architecture

(Continued..)

Feature relevant for cloud

Failure Group (in context of IBM Spectrum Scale):

- ✓ It is defined as a set of disks that share a common point of failure that could cause them all to become simultaneously unavailable.
- ✓ Filesystem replication ensures that there is a copy of each block of replicated data and metadata on disks in different failure groups.



Introduction: AWS Services / Components

- ✓ VPC (Virtual Private Cloud): Lets you provision a logically isolated section of the AWS Cloud.

- ✓ AMI (Amazon Machine Image): It provides the information required to launch an instance.

- ✓ CloudFormation: Allows user to use a simple text file to model and provision, in an automated and secure manner, all the resources needed for the applications across all regions.

- ✓ AutoScaling: Automatically launch or terminate instances based on user-defined policies, health status checks etc.

Introduction: AWS Services / Components

(Continued..)

- ✓ Auto Recovery: Automatically recovers the instance if it becomes impaired due to an underlying hardware failure.
- ✓ Systems Manager: Provides a unified user interface to view operational data from multiple AWS services and allows to automate operational tasks.
- ✓ IAM Policies, Roles: Identity-based policies are permission policies that can be attached to a principal (or identity), such as an IAM user, role, or group.

Introduction: AWS Services / Components

(Continued..)

- ✓ CloudWatch: Monitoring service for AWS resources. Collect, track metrics and react immediately.

- ✓ Lambda: It is a compute micro service and runs code in response to events such as image uploads, in-app activity, website clicks, or outputs from connected devices.

- ✓ SNS (Simple Notification Service): Pub/Sub messaging and mobile notifications.

Challenges of Integrating Building Blocks

- ✓ Way things are deployed on cloud, monitored, managed (admin, access) are different than on-premise.
- ✓ Determine which feature to be used (Ex: filesystems own monitoring or cloud monitoring services) - Best of both worlds.
- ✓ Rapid elasticity.
- ✓ Tuning & testing - block device, network, sysctl parameters
- ✓ Quite different than application porting.

It is “Software Defined Storage (SDS) over Cloud”

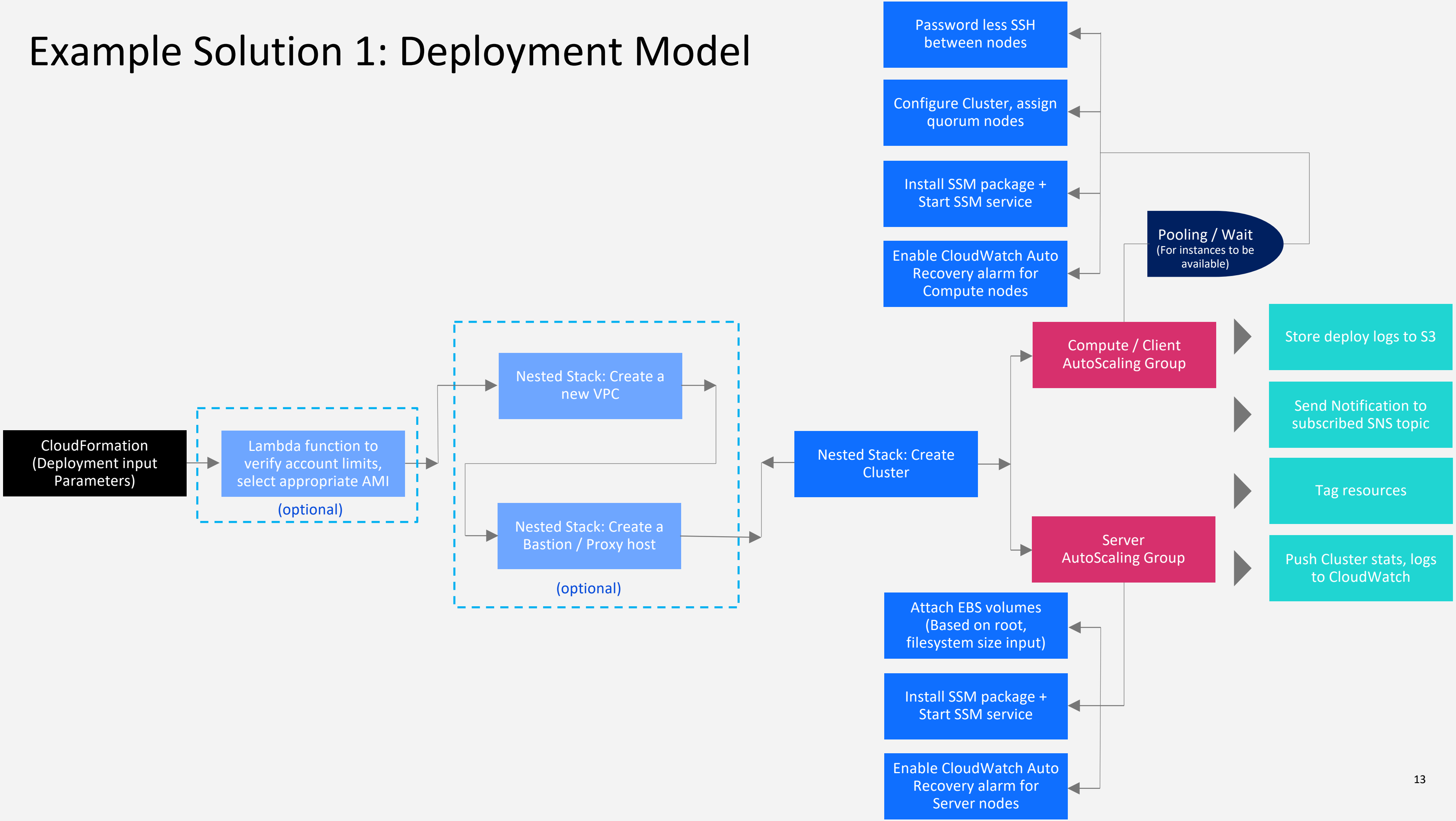
Solution: Deployment Model

- ✓ Largest clusters could include = ~500 nodes or more (storage capacity of [1TB ~ 100PB]).

- ✓ East/Fast to install, East/Fast to cleanup, East/Fast to spin up and spin down.

- ✓ CloudFormation (or similar service provided by the cloud vendor) is the appropriate fit as it;
  - > Provisions cloud resources (allows to parallelize, wait for dependency provisioning)
  - > Modular approach (each nested stack can be optimized - don't keep resources idle!!)
  - > Version control
  - > Allows expansion, contraction of instances (or cluster size).
  - > Enables appropriate security policies and roles.

Example Solution 1: Deployment Model



CloudFormation (Deployment input Parameters)

Lambda function to verify account limits, select appropriate AMI

(optional)

Nested Stack: Create a new VPC

Nested Stack: Create a Bastion / Proxy host

(optional)

Nested Stack: Create Cluster

Compute / Client AutoScaling Group

Server AutoScaling Group

Pooling / Wait (For instances to be available)

Password less SSH between nodes

Configure Cluster, assign quorum nodes

Install SSM package + Start SSM service

Enable CloudWatch Auto Recovery alarm for Compute nodes

Attach EBS volumes (Based on root, filesystem size input)

Install SSM package + Start SSM service

Enable CloudWatch Auto Recovery alarm for Server nodes

Store deploy logs to S3

Send Notification to subscribed SNS topic

Tag resources

Push Cluster stats, logs to CloudWatch

Example Solution 2: Deployment Model

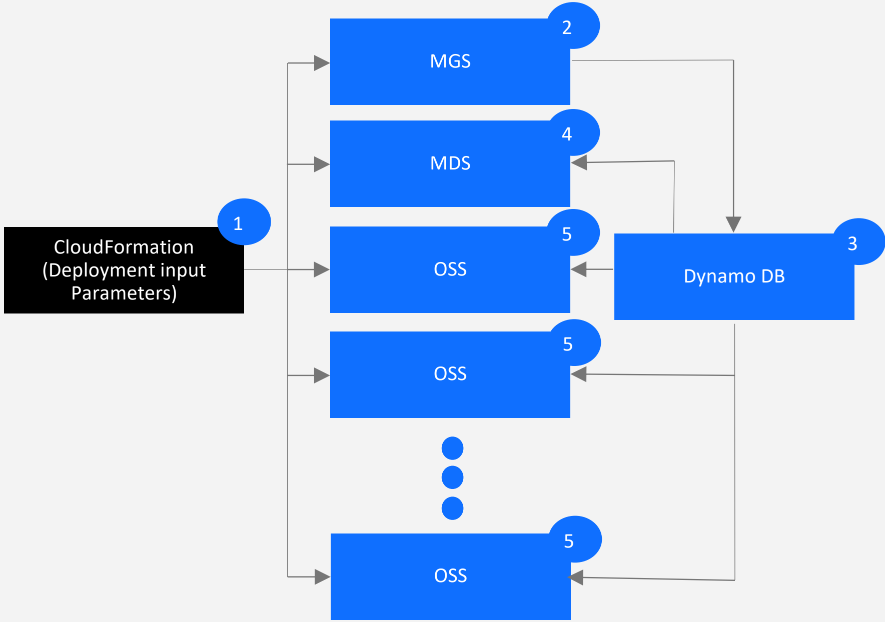
1. CloudFormation creates a stack of AWS resources from AWS resources provided to the template.

2. MGS Initializes itself.

3. MGS updates DB with NID (network identifier).

4. MDS formats MDT, registers with MGS, updates DB

5. OSSs format local targets, updates DB



* Lustre MGS (Management Service): Stores file system configuration information for use by the clients and other Lustre components.

* Lustre MDS (Metadata Service): Provides index, or namespace for Lustre file system

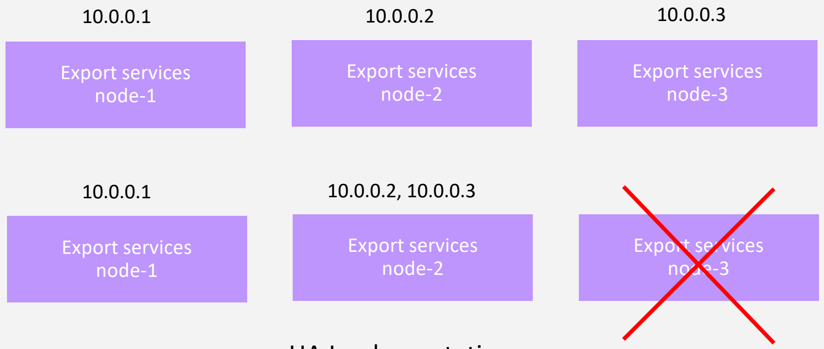
* Lustre OSS (Object Storage Service): Nodes that store file data on one or more object storage target (OST) devices

Sources: http://bit.ly/LustreArc

Solution: Deployment Model (HA)

- ✓ Protocol access (NFS/CIFS/REST/Object – S3/swift) plays a key role in data sharing.

- ✓ High availability solution for export services can be achieved by using AutoScaling and secondary ENI.



10.0.0.1

Export services node-1

10.0.0.2

Export services node-2

10.0.0.3

Export services node-3

10.0.0.1

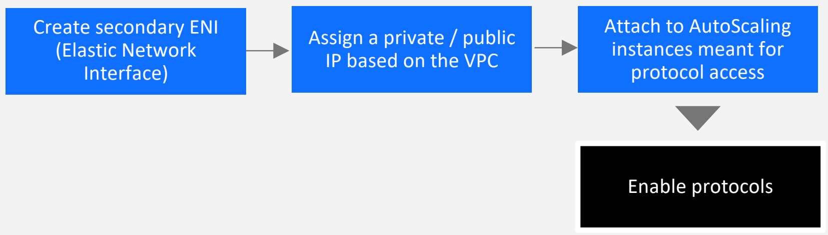
Export services node-1

10.0.0.2, 10.0.0.3

Export services node-2

Export services node-3

HA Implementation



Create secondary ENI (Elastic Network Interface)

Assign a private / public IP based on the VPC

Attach to AutoScaling instances meant for protocol access

Enable protocols

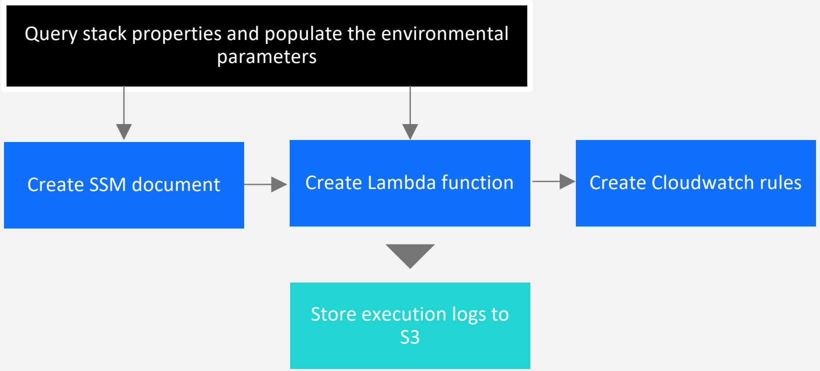
HA Implementation (using Auto Scaling)

* ENI (Elastic Network Interface): It is a logical networking component in a VPC that represents a virtual network card.

Solution: Management Model

(Continued..)

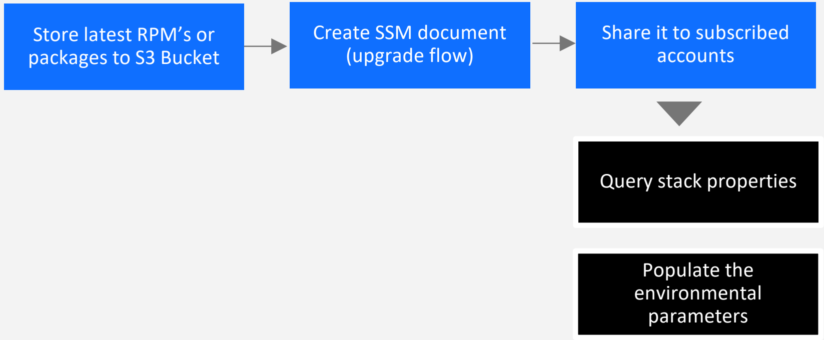
Scenario: Shutdown cluster when not in use and then restart the cluster (preferably schedule based).


Query stack properties and populate the environmental parameters
Create SSM document
Create Lambda function
Create Cloudwatch rules
Store execution logs to S3

Solution: Management Model

(Continued..)

Scenario: Patch or upgrade the cluster with impacting the production workload.



Store latest RPM’s or packages to S3 Bucket

Create SSM document (upgrade flow)

Share it to subscribed accounts

Query stack properties

Populate the environmental parameters

Data Life Cycle Management Practices

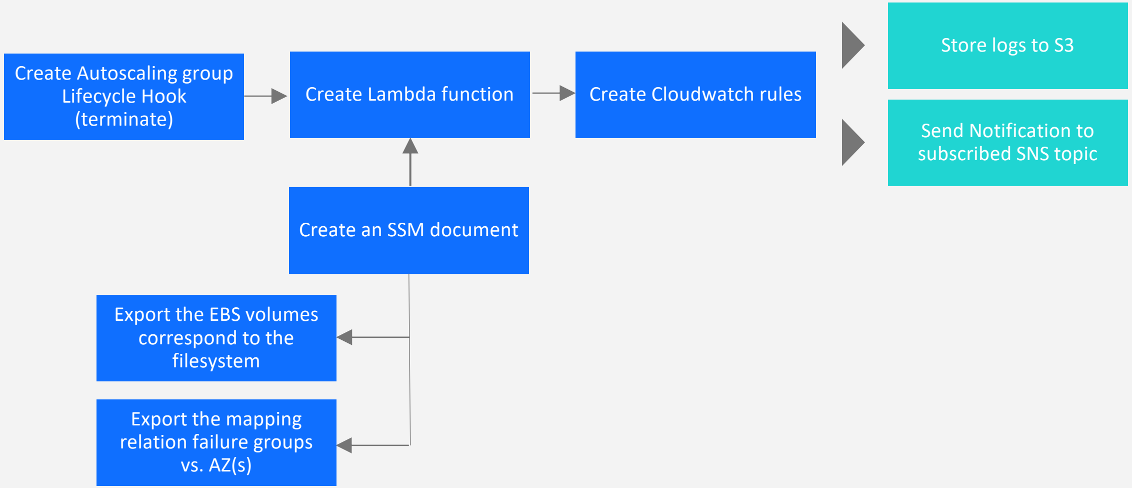
✓ Data to/from methods – multiple options (S3, storage gateways)

✓ Faster way to export / reimport data within cloud environment

✓ Long term cost optimization by placing data on right tier

Data export, import Model

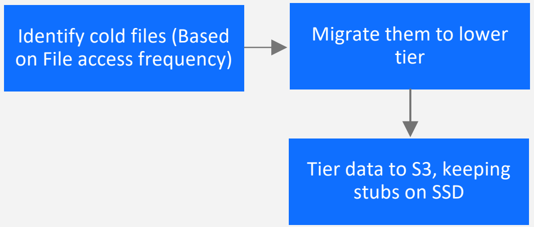
Scenario: Terminate cluster, de-attach data volumes, and allow provision to re-attach data volumes to a new cluster of same or different topology.


Create Autoscaling group Lifecycle Hook (terminate)
Create Lambda function
Create Cloudwatch rules
Store logs to S3
Send Notification to subscribed SNS topic
Create an SSM document
Export the EBS volumes correspond to the filesystem
Export the mapping relation failure groups vs. AZ(s)

Data migration, tiering Model

Scenario: Place (migrate or tier) data on right storage tier.

SSD’s used by instances are expensive (~3x costly than object storage). Tier or migrate non-frequently accessed data to cheaper storages.


Identify cold files (Based on File access frequency)
Migrate them to lower tier
Tier data to S3, keeping stubs on SSD

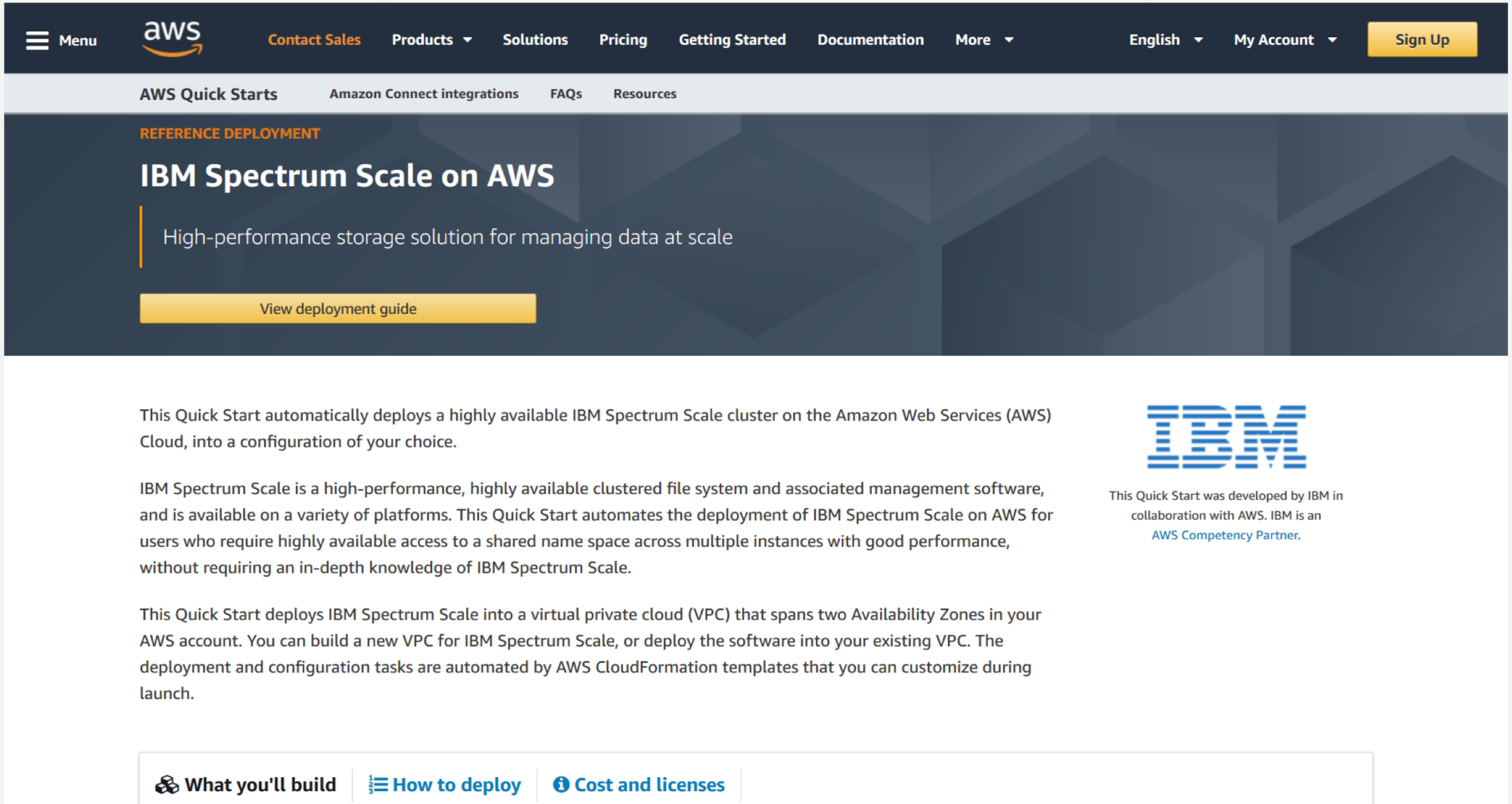
Summary

- One should plan / evaluate appropriate {Deployment, Management , Data life cycle management} approaches before building / adopting solution.
- All the pillars needs to be designed for rapid & repeated deployments, easy administration and utmost security.
- Right infrastructure needs to be selected (as cloud brings in lot of varieties).

Infrastructure Matters for Business Outcomes.

What to try out - Free 90 day Trial – IBM Spectrum Scale on AWS

AWS Quick start: http://bit.ly/scaleonaws | Demo: http://bit.ly/scaleonawsdemo



REFERENCE DEPLOYMENT

IBM Spectrum Scale on AWS

High-performance storage solution for managing data at scale

This Quick Start automatically deploys a highly available IBM Spectrum Scale cluster on the Amazon Web Services (AWS) Cloud, into a configuration of your choice.

IBM Spectrum Scale is a high-performance, highly available clustered file system and associated management software, and is available on a variety of platforms. This Quick Start automates the deployment of IBM Spectrum Scale on AWS for users who require highly available access to a shared name space across multiple instances with good performance, without requiring an in-depth knowledge of IBM Spectrum Scale.

This Quick Start deploys IBM Spectrum Scale into a virtual private cloud (VPC) that spans two Availability Zones in your AWS account. You can build a new VPC for IBM Spectrum Scale, or deploy the software into your existing VPC. The deployment and configuration tasks are automated by AWS CloudFormation templates that you can customize during launch.

This Quick Start was developed by IBM in collaboration with AWS. IBM is an AWS Competency Partner.

THANK YOU

Acknowledgements

John Lewars, Gautam Shah, Dherendra Singh and Sanjay Gandhi - IBM

Vinod Shukla and Scott Kellish - Amazon Web Services