AMAZON WEB SERVICES (AWS) DATA PIPELINE

WHITEPAPER

PREMIER CONSULTING PARTNER

WWW.CORPINFO.COM

Laith Al-Saadoon

August 2016

CORPINFO

amazon web services | Partner Network

PREMIER CONSULTING PARTNER

TABLE OF CONTENTS

ABSTRACT

Experts estimate that global Internet traffic will exceed a Zettabyte (1 trillion Gigabytes) in 2016 [1] with 40 Zettabytes existing by 2020 [2]. In the current technology climate, most companies both large and small store at least Terabytes of data retrieved from transactional, operational, campaign, and third-party market research data. Some companies embrace even more data sources such as clickstream, event processing, and Internet of Things (IoT) data, thereby exponentially increasing the amount of data ingested. One survey found that 71% of companies have a near-term plan to use even the simplest form of analytics in every day decision-making [3]. The motivation for data usage and storage is clear, as researchers at MIT Sloan Management Review have found that top-performing organizations use analytics five times more than lower performers [4].

Given the surge of data into business technology infrastructure, IT departments may find it difficult to standardize and scale data ingest and extraction, data transformation and cleansing, and data loading into storage that is best suited for advanced analytical processing engines. Some organizations may turn to complex and expensive data integration tools to meet the demands of data operations and data governance. The complexities and cost of these tools may be a show-stopper for some organizations.

Amazon Web Services (AWS) provides AWS Data Pipeline, a data integration web service that is robust and highly available at nearly 1/10th the cost of other data integration tools. AWS Data Pipeline enables data-driven integration workflows to move and process data both in the cloud and on-premises.

AWS Data Pipeline enables users to create advanced data processing workflows that are fault tolerant, repeatable, and highly available. Data engineers, integrators, and system operations staff do not have to worry about ensuring resource availability, provisioning, managing inter-task dependencies, retrying transient failures or timeouts in individual tasks, or creating a failure notification system. With AWS Data Pipeline, IT and data teams can move and process data once locked up in data silos with the benefit of no upfront costs and only paying for what they use.

This whitepaper is intended for big data architects, data engineers, data integrators, and system operations administrators faced with the challenge of orchestrating and Extracting, Transforming, and Loading (ETL) vast amounts of data from across the enterprise and/or external data sources. The aim of this whitepaper is to familiarize readers with AWS Data Pipeline by sharing an overview, best practices, and hands-on examples. This paper also serves to influence technical decision-makers to evaluate their data workflow and orchestration requirements and articulate the benefits of AWS Data Pipeline.

CorpInfo is an AWS Premier Consulting and Big Data Partner. We specialize in guiding our customers with big data challenges on their journey into the cloud. Our data practice is focused on enabling businesses to focus on extracting immediate competitive business insights from their data instead of provisioning servers, storage, and other non-differentiating tasks.

WHAT WE'LL COVER

This whitepaper assumes familiarity with fundamental AWS services such as Elastic Compute Cloud (EC2), Relational Database Service (RDS), Elastic MapReduce (EMR), Virtual Private Cloud (VPC), Simple Storage Service (S3), and Identity and Access Management (IAM). Experience with writing and modifying JSON, familiarity with databases, flat-files, and processing is recommended but not required. We will provide an overview of AWS Data Pipeline, followed by simple and complex pipeline examples in order to internalize the service and patterns that one might encounter.

- **AWS Data Pipeline Overview:** Use-Cases, Architecture & Components, Automation & Agility, Security, Elasticity, and Cost
- **CorpInfo Best Practices and Knowledge Share:** Guidance from CorpInfo's real-world big data scenarios and experience
- **Hands-On Example and Demonstration:** Relational database export Data Pipeline

AWS DATA PIPELINE OVERVIEW

AWS Data Pipeline is a service in the "Analytics" category which provides a simple way to get data integration workflows up and running in the AWS public cloud. The service gives users the ability to submit pipelines in a web browser via the AWS Management Console, the AWS Command Line Interface (CLI), by using an AWS Software Development Kit (SDK), or AWS CloudFormation. The browser-based AWS Management Console has an intuitive Graphical User Interface (GUI) to visually create pipelines shown as directed acyclic graphs (DAGs). Using the AWS CLI, SDK, or CloudFormation, users can create and submit pipelines described in JavaScript Object Notation (JSON) definitions.

USE-CASES

Common use-cases for AWS Data Pipeline include, but are not limited to, the following:

- **Extraction –** for example, to extract relational data from a transactional relational database management system (RDBMS) into S3 object storage for later use
- **Transformation –** run Spark or Hadoop MapReduce jobs to transform and process data sets on EMR
- **Production analytics jobs –** Run MapReduce or Spark jobs on a schedule on EMR for advanced analytics that are SLA-bound
- **Loading –** for example, to load data into AWS's reliable Redshift, an OLAP Data Warehousing solution
- **Scheduled maintenance/administration scripts**

ARCHITECTURE & COMPONENTS

AWS Data Pipeline is made up of many objects that come together to form a solidified data integration architecture.

TABLE 1: AWS DATA PIPELINE COMPONENTS

Component	Example	Description
Definition	Real-world example available in the Hands-on Demonstration on page 13.	A *definition* specifies the business logic of data management using JSON. A definition is made up of data sources, preconditions, resources, activities, parameters, and values.
Pipeline	N/A	A *pipeline* schedules and runs tasks. It includes your definition, a schedule or on-demand activation method, logs, IAM roles, and more.
Objects	N/A	A JSON list of data nodes, resources, activities, schedules, and preconditions
Resources	`{ "objects": [ { "type": "Ec2Resource", "myComment": "EC2 Resource", "id": "ResourceId_VzrKg" … }, … ]}`	An EC2 instance or EMR cluster that is managed by the pipeline on which activities are executed, JDBC connections originate from, and more.
Data Nodes	`{ "objects": [ { "id": "SqlTable", "type": "SqlDataNode", "selectQuery": "#{mySqlSelectQuery}", "table": "#{mySqlTableName}", "runsOn": {"Ref": "ResourceId_VzrKg"}, … }, … ]}`	Native Data Pipeline data source or target in your definition. S3, DynamoDB, SQL databases via SELECT statements, and Redshift are supported.

Component	Example	Description

Activities	`{`
	`  "objects": [`
	`   {`
	`     "type": "CopyActivity",`
	`     "input": {`
	`       "ref": "SqlTable"`
	`     },`
	`     "output": {`
	`       "ref": "DataNodeId_TvTyk"`
	`     },`
	`     …`
	`   },`
	`   …`
	`]}`
		An activity executes your data management tasks. This could include data movement, SQL scripts, Spark and MapReduce jobs, or even arbitrary/custom scheduled programs or tasks run via Shell commands.

In Table 1, we examine in detail the fundamental components that make up a pipeline and its definition. To summarize, pipelines are made up data sources, destinations, and activities to orchestrate the movement. Readers will find a real-world example of a pipeline definition and its components in the Hands-On Demonstration on page 13.

FIGURE 1: EXAMPLE ARCHITECTURE DIAGRAM-EXPORT DATA FROM A RELATIONAL DATABASE INTO S3

In Figure 1 (on the following page), we have shown an architecture for a simple Data Pipeline use-case, extracting data from a SQL relational database into Amazon S3. Activities run on compute resources such as EC2, EMR, or on-premises servers in order to make connections to data nodes, run jobs and scripts, and store results. In this example, we summarize what a Data Pipeline might look like running entirely in AWS, within customer-owned Virtual Private Clouds and compute resources.

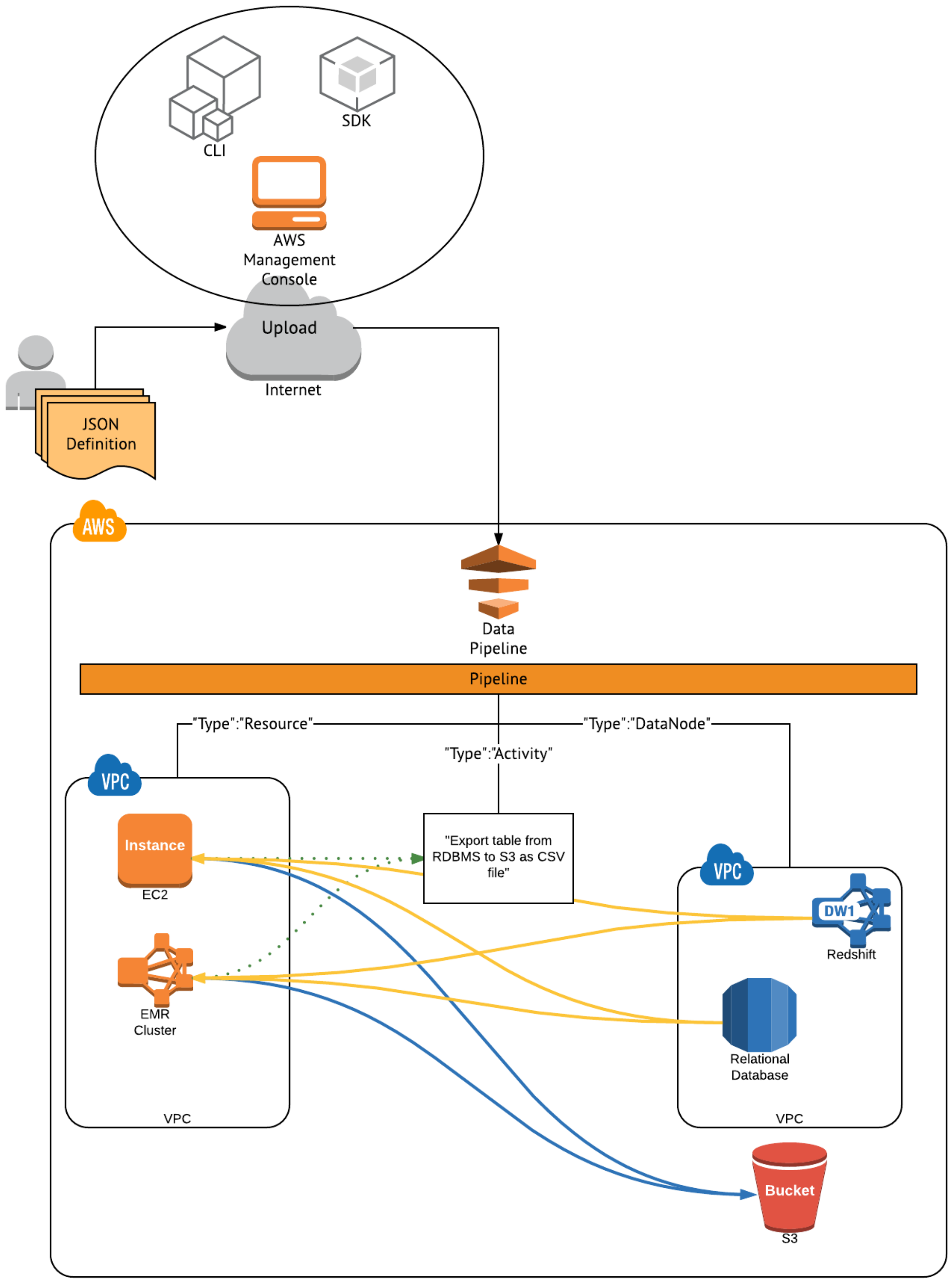

FIGURE 1: EXAMPLE ARCHITECTURE DIAGRAM-EXPORT DATA FROM A RELATIONAL DATABASE INTO S3

AUTOMATION & AGILITY

Data Pipeline has the following features that one should look for to automate and quickly iterate on data integration workflows.

1. Deployment

Automated deployment refers to the ability to provision resources (in this case, pipelines), with little to no human intervention. In the case of AWS Data Pipeline, several pathways exist in order to deploy pipelines automatically. Pipeline engineers can use the natively supported AWS CLI, AWS SDK, and CloudFormation tools to script and automate deployment of Data Pipelines.

2. Schedule & On-Demand

AWS Data Pipelines themselves are, by definition, automated data processing pipelines. In order to achieve that, pipelines utilize scheduling and on-demand activation mechanisms to activate automatically.

3. Preconditions

In AWS Data Pipeline, a precondition is a pipeline component containing conditional statements that must be true before an activity can run. This allows users to check that a certain table exists, S3 key exists, or even use custom Shell scripts for endless precondition scenarios. With respect to automation, this allows a pipeline engineer to set a condition as code once and not have to check data sources manually before activating a pipeline.

4. Notifications & Retries

The Data Pipeline service natively supports unattended retries. Additionally, it supports notifications for success, failure, late start, and more by integrating with Amazon SNS topics. Support staff could be notified proactively through SMS, e-mail, or other HTTP integrated notification systems.

5. Automated Provisioning

When running Data Pipeline in an AWS VPC, the service will provision and decommission EC2 and EMR resources on your behalf, within your private network in the cloud. Data Pipeline runs on Linux instances. These resources are ephemeral and temporary, meaning data engineering or IT operations are not required to preconfigure EC2 instances or EMR clusters to run AWS Data Pipeline tasks by default. This allows for rapid experimentation, iteration, and elasticity of pipelines.

With regards to these transient Data Pipeline resources in a production pipeline, troubleshooting and configuration should occur “off-server” through reviewing logs and efficient configuration scripting. In a development and testing scenario, given that the resources are transient and ephemeral by default, it is advisable that you run custom activity code (bash scripts, Python, etc.) on a manually provisioned EC2 instance so you can quickly debug, view logs, etc.

For high-frequency or high-concurrency data workflows (hourly or more), one may choose to use persistent resources to achieve shorter job run times. Read more about this in the CorpInfo Best Practices and Knowledge Share section on page 13.

6. Pipelines as Code

At their core, AWS Data Pipelines are defined in JSON-syntax code. For example, workflows created in the AWS Management Console user interface can be exported and downloaded as code definitions. Code definitions can be shared and collaborated on in source version control repositories, can be self-documenting, and are less error-prone to manual human intervention in automated systems.

7. Logging

Automated systems must have robust logging in order to trace errors, audit activities, and so on. Data Pipeline can shuttle logs to S3 storage natively.

Security

IT and technology management should evaluate security before adopting any new systems and services. We've outlined AWS's and Data Pipeline's security controls below.

1. Shared Responsibility Model

When evaluating security in AWS, it is important to understand the AWS Shared Responsibility Model (https://aws.amazon.com/compliance/shared-responsibility-model/). The underlying facilities, physical hypervisor resources, and the services themselves in the AWS cloud have undergone many extensive third-party audits and certification for security compliance. To summarize, the Shared Responsibility Model identifies that AWS is responsible for the security of the AWS infrastructure as a service, while the customer is responsible for the secure use of those services.

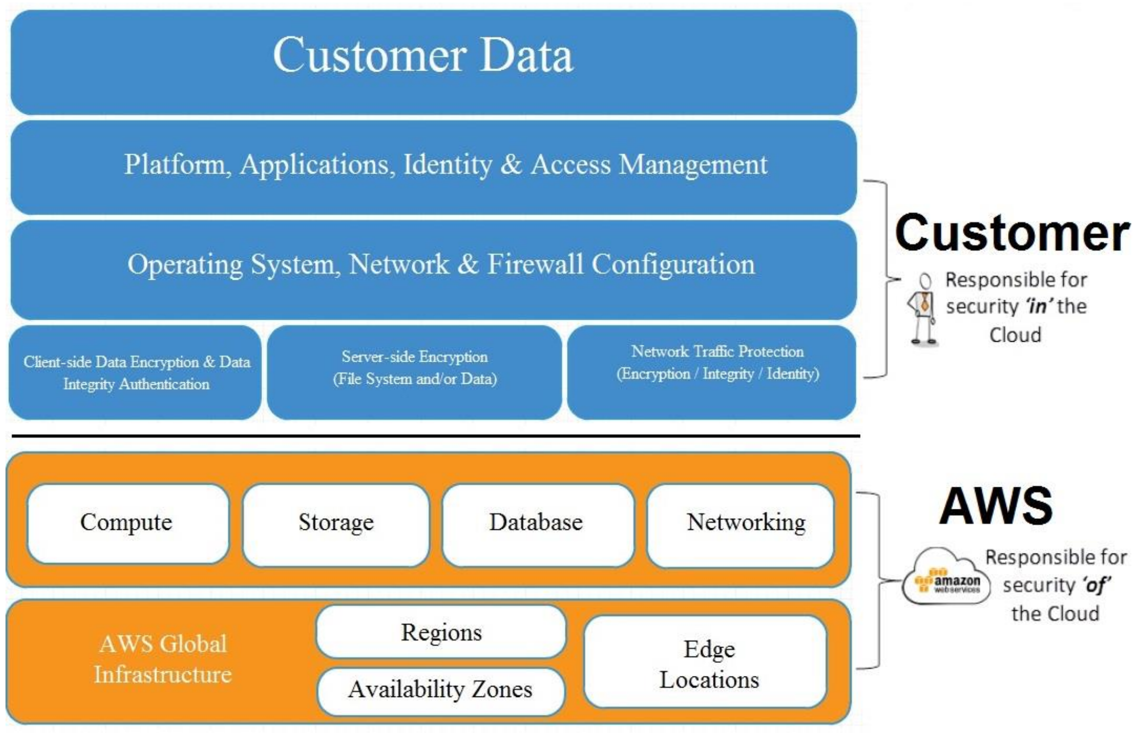
Customer Data
Platform, Applications, Identity & Access Management
Operating System, Network & Firewall Configuration
Client-side Data Encryption & Data Integrity Authentication
Server-side Encryption (File System and/or Data)
Network Traffic Protection (Encryption / Integrity / Identity)
Customer
Responsible for security 'in' the Cloud
Compute
Storage
Database
Networking
AWS Global Infrastructure
Regions
Availability Zones
Edge Locations
AWS
Responsible for security 'of' the Cloud

FIGURE 2: AWS SHARED RESPONSIBILITY MODEL

With respect to AWS Data Pipeline, customers are not responsible for the security of the Data Pipeline web service. They are, however, responsible for the secure coding practices for custom script activities and code, proper utilization of AWS Virtual Private Cloud (VPC) components to secure the environment, securing the guest operating systems, and utilization of encryption protocols. AWS Data pipeline can utilize custom VPC networks to control source/destination traffic with Security Groups and Network ACLs. This allows the Data Pipeline's EC2 and EMR resources to run within your private network to access databases and other restricted data sources. Advanced configurations are possible in order to customize the EC2 and EMR resources to implement host-based IDS, anti-malware, file integrity management, and other security software mandated in a business's security operations.

2. Security in the Virtual Private Cloud

In reference to the architecture shown in Figure 1, AWS Data Pipeline resources should reside in "private" subnets, rather than "public" subnets. The Data Pipeline EC2 and EMR resources do not have publically accessible IP addresses attached to them, and they achieve outbound Internet access through a Network Address Translation (NAT) server on EC2, or AWS NAT Gateway. JDBC and S3 connections can occur over the private IP space and using a VPC S3 Endpoint. In most cases, no remote administrator access (SSH) is enabled to pipeline resources. This greatly reduces the attack surface of the environment.

3. Identity & Access Management Roles

When using EC2 and EMR resources running in an AWS VPC, AWS actions such as downloading and uploading data in S3, provisioning instances, and so forth are all authorized and authenticated using Identity and Access Management (IAM) Roles. This allows security administrators to implement least privileges policies to Pipelines to only have access to specific S3 locations and other resources as needed. Additionally, using IAM Roles allows these actions without storing AWS API keys anywhere. IAM Role keys are rotated automatically at regular intervals. Any and all API actions to an AWS account can be logged and monitored using AWS CloudTrail.

4. Storage Security

Customers are also responsible for their storage security. It's recommended that customers use the encryption options at their disposal. For example, S3 has server-side encryption, which will encrypt objects at the storage layer for protection. Relational, SQL databases running in Amazon RDS should have encryption options enabled, as well.

Be advised: AWS Data Pipeline does not have a built-in feature to retrieve authentication "secrets" from a service such as AWS Key Management Service (KMS). When you submit a pipeline definition, your database credentials are submitted in clear text. We have provided some tips to keep your credentials and data safe, below.

- Avoid storing and committing credentials in your Data Pipeline definitions – pass them as parameters when you submit a pipeline, instead.
- In your JSON definitions, prefix your authentication parameters with the '*' (asterisk) special character to encrypt the credentials in transit and in the Data Pipeline console.

- It's recommended that you create a user on your RDBMS systems specifically for Data Pipeline to control access to certain schemas, databases, and tables and audit the activities regularly. Use a cryptographically strong password and rotate regularly.
- Limit access to the AWS Data Pipeline API using AWS Identity and Access Management (IAM). We recommend permitting the "GetPipelineDefinition" action only to privileged IAM users such that underprivileged users cannot retrieve submitted passwords from the AWS Console or API.

Elasticity

Elasticity refers to the ability to scale servers in and out (horizontal scaling), only when load demands it. Historically, systems architects would design and plan for peak utilization and storage, thus purchasing a resource pool that was mostly underutilized in normal conditions.

1. Compute

Using AWS Data Pipeline, EC2 and EMR resources are provisioned and decommissioned automatically: on-demand or based on a schedule. EMR provides a managed Hadoop platform on AWS, supporting the linear, horizontal scaling capabilities that one would expect from Hadoop.

2. Storage

Data Pipeline, when combined with S3, allows for virtually unlimited storage. S3 storage will expand as your store data in buckets, and one must only pay for what is used. Database Services in AWS such as RDS and Redshift support on-demand storage scaling in minutes.

Cost

One of the most enticing aspects of AWS Data Pipeline is its low cost. There are no upfront charges or license fees to use the service. Data Pipeline has an on-demand, frequency-based pricing model, which we will describe in detail.

AWS Data Pipeline is billed based on how often your activities and preconditions are scheduled to run and where they run (AWS or on-premises). High Frequency activities and preconditions are ones scheduled to execute more than once a day; for example, an activity scheduled to execute every hour or every 12 hours is High Frequency. Low Frequency activities are ones scheduled to execute one time a day or less. Inactive pipelines are those in PENDING, INACTIVE, and FINISHED states.

	High Frequency	***Low Frequency***
Activities or preconditions running on AWS	$1.00 per month	$0.60 per month
Activities or preconditions running on-premises	$2.50 per month	$1.50 per month
Inactive pipelines: $1.00 per month		

In addition to Data Pipeline fees, your Amazon EC2, Amazon Elastic MapReduce, Amazon S3, Amazon RDS, Amazon DynamoDB, Amazon Redshift, and Amazon SNS activities associated with AWS Data Pipeline are billed separately according to those services' normal hourly, per GB, or per request rates.

At CorpInfo, we've developed and encountered many intricate and complex Data Pipelines, and have learned and shared the follow cost saving tips:

1. Use Amazon EC2 Spot Instances for transient and cost-driven workflows, and for testing workflows

Data Pipeline includes native capabilities for Amazon EC2 Spot Instances for EC2 and EMR resources. Spot Instances allows one to bid on underutilized EC2 resources and achieve a lower hourly rate for EC2. At a high level, batch data processing makes a great use-case for utilizing Amazon EC2 Spot Instances, with the caveat that the business's data processing must tolerate potential failure due to a lost bid. Learn more about Amazon EC2 Spot Instances in the Further Reading section.

2. Use Amazon EC2 Reserved Instances for long-running, very high frequency workflows

In the event of having a high number of activities spread throughout the day, or very frequent activities, one may find that one or more EC2 and EMR resources are always running at a given time. In this case, consider purchasing Amazon EC2 Reserved Instances for the minimum amount of EC2 instances running at a given time. Amazon EC2 Reserved Instances allow a business to make a 1- or 3-year commitment on EC2 resources with zero, partial, or full upfront payment options in order to save on EC2 spend. Learn more about Amazon EC2 Reserved instances in the Further Reading section.

Consider these cost benefits of using AWS Data Pipeline:

- Get started quickly with no upfront cost or license fees
- 1/10th the cost of competing commercial Data Integration products
- Pay-per-use, frequency-based pricing that grows when you need it, and shrinks when you don't
- Built-in resource cost saving optimization capabilities with Amazon EC2 Spot Instances in your JSON definitions

CORPINFO BEST PRACTICES & KNOWLEDGE SHARE

Through hands-on experience developing and guiding customers using Data Pipeline, CorpInfo has determined a few custom configurations and best practices.

When to Use Existing Resources Using Task Runner

AWS provides Data Pipeline Task Runner, a Java executable that you can install on EC2, EMR, or even on-premises resources to execute Data Pipeline activities. In contrast to the EC2 and EMR resources that Data Pipeline creates and terminates by default on your behalf, these resources persist beyond the life of a workflow. The user has the responsibility of maintaining these resources.

For situations in which you have high frequency or highly concurrent data workflows, you may find that EC2 and EMR provisioning times are a bottleneck to your workflows. In these situations, you may find increased throughput by using Task Runner on existing resources. This bypasses the provisioning step that Data Pipeline-managed resources entail, cutting pipeline times to complete significantly. Read more on Executing Work on Existing Resources Using Task Runner at https://docs.aws.amazon.com/datapipeline/latest/DeveloperGuide/dp-how-task-runner-user-managed.html.

Use a Custom AMI for EC2Resource

By default, Data Pipeline uses a dated Amazon Machine Image (AMI) that utilizes paravirtualization (PV) for its EC2 resources. AWS recommends choosing hardware virtualized machines (HVM) for EC2. Read more on HVM versus PV: Linux AMI Virtualization Types (https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/virtualization_types.html).

In accordance with AWS's recommendation, CorpInfo recommends that all Data Pipeline consumers use a custom AMI for Data Pipeline EC2 Resources. By using a custom HVM AMI, customers can achieve lower times to complete a pipeline activity due to decreased provisioning times for resources. Additionally, customers will have more EC2 instance families and instance types to choose from. CorpInfo encourages that costumers should take advantage of using the various instance types to achieve lower costs with EC2 Spot Instances, where applicable.

Read more about using custom AMI with Data Pipeline:
http://docs.aws.amazon.com/datapipeline/latest/DeveloperGuide/dp-custom-ami.html

HANDS-ON DEMONSTRATION

PRE-REQUISITES

To follow along with the hands-on demonstration, you will need the following files, software, and resources:

- Download and unzip the example AWS Data Pipeline definition to your local machine: https://s3-us-west-2.amazonaws.com/cis-samples/awsdatapipeline/simplepipeline.zip

- Access to an AWS account

- AWS CLI tool installed on your local machine

  Instructions: http://docs.aws.amazon.com/cli/latest/userguide/installing.html

- A simple text editor, such as Sublime Text, Atom, or Notepad++

To maintain focus on Data Pipeline and data integration topics, we kindly ask that you please refer to the links above to download, install, and set up the required software, tools, and accounts before moving on.

Total Hands-On Estimated Duration: 1.5 hours

- VPC and RDS creation: **approximately 45 minutes passive wait time to complete**

- Simple Pipeline setup: **10 minutes active**

- Simple Pipeline run time: **approximately 15-20 minutes passive wait time to complete**

- Clean-up: **approximately 10 minutes active**

Disclaimer

Users are advised, throughout this demonstration, you will be creating AWS resources in your own AWS account. This includes: Data Pipeline, VPC, EC2, NAT Gateway, S3, RDS, and more. Your account will be charged for these resources accordingly at a standard hourly rate. With that in consideration, all templates and parameters were designed for minimum specifications to minimize costs.

For maximum cost efficiency, we recommend completing the Hands-On Demonstration end-to-end in one session, and immediately deleting S3 content, terminating all templates and Data Pipeline pipelines when you have completed the demo.

I. PREPARE A VIRTUAL PRIVATE CLOUD (VPC) AND RELATIONAL DATABASE (RDS)

In this first section, we are going to build essential infrastructure for our pipelines. Using CloudFormation, we can create scripted templates of resources such as VPC, S3, RDS, IAM, and more. The RDS instance houses the Employees database with makeshift employee records. Read more about the Employees database here: https://dev.mysql.com/doc/employee/en/employees-introduction.html. To proceed, follow the steps below:

1. Log into the AWS Management Console for an account that you own.

2. Click the blue rectangle shape below. This will open up the a CloudFormation page in the “US West (Oregon)” region, also known as “us-west-2”.

Click here to launch a base VPC and RDS infrastructure required for this tutorial

3. You should have been take to the “Select Template” page after clicking the above link. Click “Next” to accept the defaults and proceed.

4. On the “Specify Details” page, scroll down and click “Next” to accept the defaults and proceed.

5. On the “Options” page, you can ignore the tagging options and click “Next” to proceed.

6. You should now be on the “Review” page. Scroll down to find a checkbox with the following text: “**I acknowledge that AWS CloudFormation might create IAM resources.”** Check the box and click “Create”.

7. You should have been redirected to the CloudFormation management page. You should see a Stack Name such as “CISWDPWhitepaper” that has a status `CREATE_IN_PROGRESS`

8. Wait for the status to update to `CREATE_COMPLETE` in the AWS Management Console to complete the template. This may take about 40 minutes to complete.

9. Click on the check box next to the Stack Name “CISWDPWhitepaper” and click on the “Output” tabs on the lower half other browser window.

10. Copy the Key and Value table into a text file, as you will need these as parameters in the subsequent section below.

II. SET UP AND ACTIVATE A SIMPLE PIPELINE

In this second section, we will demonstrate how easy it is to set up and run a simple AWS Data Pipeline. The pipeline will extract the Employees table (300,024 rows) from the MySQL database on RDS to a comma separated values (CSV) file stored in S3. We will use the AWS CLI to create and publish the pipeline. We highly encourage you to open the “.json” files in the “simplepipeline” folder and examine the pipeline definition code.

1. Browse to the “simplepipeline” folder that you downloaded in the Pre-Requisites section

2. Open the file “parameter_values.json” in a text editor. This JSON document contains parameter values which the pipeline uses.

3. Replace the text inside the <> characters with the corresponding values from the Values that you saved from Step 10 in Section I above. Make sure to delete the <> characters. Note the red text below as an example:

```
{
  "values": {
    "myRDSInstanceId": "cm10f8g5n2dzrgn",
    "myRDSUsername": "root",
    "myEC2InstanceType": "t1.micro",
    "myOutputS3Loc":"s3://ciswdpwhitepaper-mys3bucket-9119hd8w98wz/pipelineOutputs",
    "*myRDSPassword": "Password1!",
    "myEc2RdsSecurityGrps": "sg-23375545",
    "myRDSTableName": "employees.employees",
    "myEc2Subnet": "subnet-1497ff70",
    "myLogS3Uri":"s3://ciswdpwhitepaper-mys3bucket-9119hd8w98wz/logOutputs"
  }
}
```

4. Save the file

5. Open a command prompt of your choice on your local machine with the AWS CLI installed

6. Change directories (“cd” command) in the terminal to the “simplepipeline” folder where there should be three “.json” documents

7. Run the following command:

```
aws --region us-west-2 datapipeline create-pipeline --name whitepaperdp --unique-id mywhitepaperdatapipeline
```

8. Copy the pipeline identifier that is printed in the output. It should look similar to: dp-123456789

9. Run the next command, replacing the text <pipeline-id> with the pipeline identifier you copied in the previous step. Remove the <> characters. This command uploads our JSON pipeline definition, parameters, and values to the empty pipeline we created in step 7.

```
aws --region us-west-2 datapipeline put-pipeline-definition --pipeline-id <pipeline-id> --parameter-objects
file://parameter_objects.json --parameter-values-uri file://parameter_values.json --pipeline-definition
file://pipeline_definition.json
```

10. Run the last command below to activate the pipeline, again replacing the text <pipeline-id> with your unique pipeline identifier from step 7. Remove the <> characters.

```
aws --region us-west-2 datapipeline activate-pipeline --pipeline-id <pipeline-id>
```

11. Congratulations, you've activated your first AWS Data Pipeline. It will now process the definition, start up an EC2 instance in the VPC, copy data from RDS, store it in S3, and shut the EC2 instance down automatically when it is done. You can watch the pipeline status from the AWS Data Pipeline console. After about 10-15 minutes, it should complete.

12. To examine the output content, go to the AWS S3 console page, and look for the bucket that was created and used for outputs in your pipeline. Look in the "pipelineOutputs" folder and drill down to see the result. You should find a ~13 Megabyte CSV file which you can download and view in spreadsheet software such as Microsoft Excel.

III. CLEAN UP

1. When you are done, delete the contents from the S3 bucket so we can terminate all resources on the CloudFormation page. Browse to the S3 console page, and right-click (Windows) or control+click (Mac) the bucket that was created.
2. Click "Empty Bucket"
3. Copy and paste the bucket name into the field, and then click "Empty Bucket"
4. Go back to the CloudFormation page from the AWS Management Console main page.
5. Select the CloudFormation Stack we used to create the VPC and RDS, click "Actions", click "Delete Stack", and then click "Yes, Delete"
6. Browse to the Data Pipeline page from the AWS Management Console main page.
7. Select the check box next to the pipeline named "whitepaperdp"
8. Click "Actions", then "Delete", and in the pop-up window, click "Delete".

CONCLUSION

Businesses around the globe are seeking to tap into a growing number of data sources and volumes in order to make better, data-driven decisions, advanced analysis, and make predictions. AWS Data Pipeline is a service provided to simplify those data workflow challenges in order to bring many data into and out of the AWS ecosystem such as Amazon S3, RDS, EMR, and Redshift. Data Pipeline runs on top of a highly scalable and elastic architecture, and data is stored and moved inside costumer-managed AWS account and Virtual Private Cloud networks. Data Pipeline comes with zero upfront cost and on-demand based pricing that is up to 1/10th the cost of competitors. Data Pipeline manages many complex parts of workflows, leaving data engineers and big data architects to focus on what matters most – the business logic and source and target systems behind the data flows.

FURTHER READING

CorpInfo Data & Analytics Practice
https://www.corpinfo.com/cloud-services/aws-big-data-solutions

AWS Data Pipeline
https://aws.amazon.com/datapipeline/

EC2 Spot Instances
https://aws.amazon.com/ec2/spot/

EC2 Reserved Instances
https://aws.amazon.com/ec2/purchasing-options/reserved-instances/

REFERENCES

1. *The Zettabyte Era—Trends and Analysis*, Cisco, http://www.cisco.com/c/en/us/solutions/collateral/service-provider/visual-networking-index-vni/vni-hyperconnectivity-wp.html

2. *Surprising Facts and Stats about the Big Data Industry*, Cloudtweaks.com, Daniel Price, http://cloudtweaks.com/2015/03/surprising-facts-and-stats-about-the-big-data-industry/

3. *Advanced Analytics & Big Data Adoption Report*, International Institute for Analytics, http://iianalytics.com/analytics-resources/market-research/advanced-analytics-big-data-adoption-report

4. *Big Data, Analytics and the Path from Insights to Value*, MIT Sloan Management Review, http://sloanreview.mit.edu/article/big-data-analytics-and-the-path-from-insights-to-value/