Certified SysOps Administrator – Associate 2017

Exam

* Monitoring & Metrics – 15%
* High Availability – 15%
* Analysis – 15%
* Deployment & Provisioning – 15%
* Data Management – 12%
* Security – 15%
* Networking – 13%

**CloudWatch**

* Monitors your AWS resources and applications it runs on.
  + ELB, Route53, Autoscale, EBS Volumes, CloudFront, Storage Gateways, DynamoDB, Elasticache Nodes, RDS Instances, EMR, Redshift, SNS topics, SQS, Opsworks, Cloudwatch logs, and bill.
* Metrics are stored for 2 weeks. Can retrieve longer ones by GetMetric Statistics API.
* Custom Metrics can have a minimum of 1 minute intervals.
* If resources are terminated, data can be retrieved up to 2 weeks.
* Can’t use IAM to control access to Cloudwatch data for specific resources
* Can’t use IAM roles with CloudWatch command line tools
* Minimum resolution is 1-second data points.
* Storage Resolution can be set by the PutMetricData. Can’t delete metrics
* Standard resolution with data having 1 minute granularity (Storage resolution field)
* High resolution with data at a granularity of one second
* Can still retrieve metrics if an EC2 instance is deleted
* EC2Config can be used to send a variety of data like custom text logs, event logs, event tracing, Performance Counter.
* Metric Filters – turn log data into Amazon CloudWatch Metrics for graphing or alarming.
  + Don’t support regular expressions
* Supports text based log data or JSON formatted
* Can search through logs via CLI
* Logs can be stored forever
* SNS integrated with it.
* Alarms history is available for 14 days.
* Events – a stream of system events describing changes in your AWS resources
  + EC2, autoscaling, Cloudtrail support it.
* EC2
  + CPU, Network, Disk, Status Check
  + Custom Metric needed for Harddisk utilization
  + **Default monitoring is 5 minutes or detailed can be enabled by 1 minute.**
  + System Status Checks (Checks underlying physical Host)
    - Check loss of network connectivity
    - Loss of system power
    - Software issues on the physical host.
    - Hardware issues on the physical host.
    - Best way to resolve is to stop and start the VM.
  + Instance Status Checks (Checks VM)
    - Failed system status checks
    - Misconfigured networking or startup configuration
    - Exhausted memory
    - Corrupted file system
    - Incompatible kernel
    - Best way to troubleshoot is by rebooting the instance and make modifications in your OS

Cloudwatch Lab:

yum install perl-Switch perl-DateTime perl-Sys-Syslog perl-LWP-Protocol-https -y

mkdir /CloudWatch

curl http://aws-cloudwatch.s3.amazonaws.com/downloads/CloudWatchMonitoringScripts-1.2.1.zip -O

unzip CloudWatchMonitoringScripts-1.2.1.zip

rm -rf CloudWatchMonitoringScripts-1.2.1.zip

cd aws-scripts-mon/

nano mon-put-instance-data.pl

./mon-put-instance-data.pl --mem-util --verify –verbose

./mon-put-instance-data.pl --mem-util --mem-used --mem-avail

Nano crontab

**EBS Monitoring**

* Throughput Optimized HDD (st1) – streaming workloads requiring consistent, fast throughput at a low price.
* Max volume size is 16 TiB
  + Go from 1GB to 4 GB to 500 GB.
  + 3 IOPS per GB
    - Can burst by using I/O Credits
    - (3000-1500) = may need to increase drive for me.
    - When more baseline performance I/O is required, it uses I/O credits in the credit balance to burst to the required performance level
    - Each volume receives an initial I/O credit balance of 5,400,000 I/O credits. Can sustain 3,000 IOPS for 30 minutes. If you don’t go over, you earn credits.
* Recommended to have SSD as the boot volume.
* **Pre-warming** – maximum performance is started at initialization. Only time this may be needed is for restoring from snapshots. Preliminary action takes times and can cause a significant increase in the latency of an I/O operation the first time each block is accessed.
* **Initialization** - avoid performance hit in production by reading from all of the blocks on your volume before you use it.
* **Volume Read Bytes/Volume Write Bytes –** information on the I/O operations in a specified period of time.
* **Volume Read Ops/Volume Write Ops –** total number of I/O operations in a specified period of time.
* **Volume Total Read Time/Volume Total Write Time –** total number of seconds spent by all operations that completed in a specified period of time.
* **Volume Idle Time –** total number of seconds in a specified period of time when no read or write operations were submitted.
* **Volume Queue Length –** read and write operation requests waiting to be completed in a specified period of time. (indicator of high IOPS)
* **Volume Thoughput Percentage –** Provisioned IOPS (SSD) only. Percentage of I/O operations per second (IOPS) delivered of the total IOPS provisioned for an Amazon EBS volume.
* **Volume Consumed Read Write Ops -** Provisioned IOPS (SSD) only. Total amount of read and write operations consumed in a specified period of time.

**Volume Status:**

* + **OK –** Normal (Volume performance is as expected)
  + **Warning –** Degraded (Volume performance is below expectations). Severely Degraded (Volume performance is well below expectations)
  + **Impaired –** Stalled (Volume performance is severely impacted). Not Available (Unable to determine I/O performance because I/O is disabled)
  + **Insufficient-data**

**Modifying EBS Volumes:**

* Can increase a size or change type and adjust the IOPS on the fly without detaching. Same for detaching.
* **Steps:**
  + Issue the modification command (Console or command line)
  + Monitor the progress of the modification
  + Extend files system to take advantage of the increased storage capacity.

**Monitoring RDS:**

* Metrics:
  + BinLogDiskUsage
  + CPUUtilization
  + **DatabaseConnections**
  + **DiskQueueDepth -** # of read/write I/O to access your RDS instance.
  + FreeableMemory
  + **FreeStorageSpace**
  + **ReplicaLag (Seconds) – lag between RDS instance and read replica’s**
  + SwapUsage
  + ReadIOPS
  + **WriteIOPS**
  + **ReadLatency**
  + **WriteLatency**
  + ReadThroughput
  + WriteTrhoughput
  + NetworkReceiveThroughput
  + NetworkTransmitThroughput
* Events:
  + Creates an SNS topic to send notifications out to.

**Monitoring ELB:**

* Is monitored every 60 seconds if traffic is coming into it. Otherwise, it won’t be reported.
* Metrics:
  + HealthyHostCount
  + UnHealthyHostCount
  + RequestCount
  + Latency
  + HTTPCode\_ELB\_4XX
  + HTTPCode\_ELB\_5XX
  + HTTPCode\_Backend\_2XX
  + HTTPCode\_Backend\_3XX
  + HTTPCode\_Backend\_4XX
  + HTTPCode\_Backend\_5XX
  + BackendConnectionErrors
  + **SurgeQueueLength –** count of total # of requests that are pending submission to a registered instance
  + **Spillovercount –** count of total number of requests that were rejected due to the queue being full.

**Monitoring Elastiche:**

* CPU utilization
  + Memcached
    - Multi-threaded
    - Can handle loads of up to 90%. If it exceeds 90% add more nodes to the cluster
  + Redis
    - Not multi-threaded. To determine the point in which to scale, take 90 and divide by the number of cores
    - Threshold for CPU Utilization (90/4)
* **Swap Usage** – amount of disk storage space reserved on disk if your computer runs out of ram. Size of swap file – the size of the RAM.
  + Memcached
    - Should be around 0 most of the time and not exceed 50Mb
    - If it exceeds 50 Mb you should increase the memcached\_connections\_overhead parameter.
    - **Memcached\_connections\_overhead** – defines that amount of memory to be reserved for memcached connections and other miscellaneous overhead.
  + Redis
    - No SwapUsage metric, instead just uses reserved-memory.
* **Evictions** – occurs when a new item is added and an old item must be removed due to lack of free space in the system.
  + Memcached
    - Can either scale up – increase memory on nodes
    - Scale out – add more nodes
  + Redis
    - Can only scale out – add read replicas
* Concurrent Connections
  + Memcached
    - No recommended setting
    - Set an alarm on the number of concurrent connections
    - Number of concurrent connections can either mean a large traffic spike or the application is not releasing connections as it should be.
  + Redis
    - No recommended setting
    - Set an alarm on the number of concurrent connections
    - Number of concurrent connections can either mean a large traffic spike or the application is not releasing connections as it should be.

**Centralized Monitoring:**

* Enterprises install Zennos, Nimsoft, Splunk, IBM, HP Operations on a centralized server that installs an agent on there.
* Security groups can span multiple AZ’s.
* Basic monitoring is going to use ICMP. Could be SQL (1433) or MySQL (3306)
* Ping is a 2 way street.

**Organizations & Consolidated Billing:**

* **AWS Organizations** – account management service that enables you to consolidate multiple AWS accounts into an organization that you create and centrally manage.
* **Consolidated Billing** - Link Paying account to all other accounts.
  + Limit of 20 linked accounts for consolidated billing
  + Linked accounts are all independent
  + Paying account is independent and can’t access resources
  + Volume pricing discounts
  + Billing alerts are included for all linked accounts if enabled on paying account
  + Cloud trail has to be enabled per account
  + Can consolidate logs on S3 from cloud trail
  + Unused reserved instances for EC2 are applied across the group.
* Billing alarms – helps monitor your account to see when charges reach a certain value.

EC2 Cost Optimization

* Heavy Utilization
* Medium Utilization
* Reserved
* Spot
* On-Demand
* Rule out wrong answers and then choose the best out of the last two.

**Elasticity and Scalability**

* **Elasticity –** allows you to stretch out and retract back your infrastructure, based on your demand. Used for a short period of time.
  + **EC2 –** increase # of EC2 instances based on autoscaling
  + **DynamoDB –** Increase additional IOPS for additional spikes in traffic
  + **RDS –** Not very elastic. Can’t scale based on demand
* **Scalability –** build out infrastructure to meet long term demands. Used for longer time periods.
  + **EC2 –** Increase instance size as required
  + **DyanmoDB –** Unlimited amount of storage
  + **RDS –** increase instance size.

**Scale Up or Scale Out?**

* **Scale Up** – increase resources or size of EC2 instances (bottle neck)
* **Scale Out** – add more EC2 instances with autoscaling (not enough resources)
* **NAT** – Network Address Translation
* **Fleet management** – refers to the functionality that automatically replaces unhealthy instances and maintains your fleet at the desire capacity
  + Replaces failed health checks with new instances automatically
* **Dynamic scaling** – automatically increases or decreases capacity based on load or other metrics.
* **Launch configuration** – template that an autoscaling group uses to launch ec2 instances. Everything you create an autoscaling group you define a launch configuration. If you want to change the instance types in an autoscaling group, adjust the launch configuration to create that instance moving forward.
  + Autoscaling groups can span AZ’s.
  + Can use the AttachInstances API to attach different types of instances to an autoscaling group
  + CodeDeploy or Cloudformation can be used to make changes to multiple instances
  + Autoscaling deploys instances if there is not as much in an AZ where others may be full
  + Lifecycle hooks – let you take action before an instance goes into service or before it gets terminated
  + **Terminate hooks** – useful for collectin important data from an instance before it goes away
  + SetInstanceHealth allows you to change an instance state to unhealthy.
  + Health checks can be suspended by using the SuspendProcesses API
  + Stateful instance – instance has data on it.
  + 5 minutes usually for turnaround time on instance replacements.
  + You can attach existing instances in the console by attaching to the autoscaling group
  + Application autoscaling can scale EC2, EC2 spot fleets, EMR clusters, AppStream 2.0 fleets, dynamoDB

**RDS Multi-AZ Failover:**

* MySQL, Oracle, PostgreSQL use synchronous physical replication on multi-AZ on the standby machine to keep it up to date.
* SQL Server uses synchronous logical replication in multi-AZ which would be mirroring technology
* DNS failover which is essentially the IP address resolve.
* Advantages:
  + HA
  + With the Same region
  + Backups are taken from secondary which avoids I/O suspension
  + Restores are taken from secondary which avoids I/O suspension
  + Can force failover by console or RebootDB Instances API call
  + Read replica’s is used to scale
  + Multi-AZ Failover is not a scaling solution.

**RDS Read Replicas:**

* **Read replicas** – read only copies of your database. Gives you the function to scale out beyond capacity constraints for single DB instance.
  + Can be created by Console or CreateDBInstanceReadReplica API
  + Asynchronously synced.
  + Can have up to 5 Read replicas per primary DB.
  + Supported by MySQL 5.6 (NOT 5.1 or 5.5) and PostgreSQL
  + InnoDB is only supported by AWS for MySQL engine. Can also use MyISAM
  + If multi-az is not enabled, snapshot will be of your primary DB and can cause brief I/O suspensions for around 1 minute
  + If Multi-AZ is enabled, snapshot will be of your secondary database and you will not experience any performances hits on your primary database.
  + When a new read replica is created, you will be able to connect to it using a new end point DNS address.
  + Can promote read replica’s by it’s own standalone but it will break the replication link
  + Read replicas can be in different regions but only for MySQL.
  + Read replica’s can be build of Multi-AZ databases but read replica’s can’t be multi-az currently.
  + Can have read replica’s of read replicas but only for MySQL. Increases latency. Replicates to top tier and then to bottom tier.
  + DB snapshots and automated backups can’t be taken of read replica’s.
  + REPLICA LAG is the key metric.
  + In order to enable read replica, automated backups must be on.
  + Use case:
    - Scaling beyond computer or I/O capacity of a single DB instance. Excess read traffic can be directed to one or more read replicas.
    - Serving read traffic while the source DB instance is unavailable.
    - Business reporting or data warehouse running queries against the read-replica versus the primary DB instance.

Bastion Hosts with Route 53 and 2 public Subnets.

**Troubleshooting Autoscaling:**

* Look for the following:
  + Associated key pair does not exist
  + Security group does not exist
  + Autoscaling config is not working correctly
  + Autoscaling group not found
  + Instance type specified is not supported in the AZ
  + AZ is no longer supported
  + Invalid EBS device mapping
  + Autoscaling service is not enabled on your account
  + Attempting to attach and EBS block device to an instance-store AMI

**Services with OS access:**

* Elastic Beanstalk
* Elastic MapReduce
* OpsWork
* EC2
* Don’t have access to:
  + RDS
  + DynamoDB
  + S3
  + Glacier

**ELB Configurations:**

* Can’t ELB to different regions
* Can ELB across different AZ within the same Region
* Two Types:
  + External ELB (external DNS)
  + Internal ELB (internal DNS)
* Health Check Interval \* Unhealthy Threshold shows whether it is healthy or not
* **Sticky Sessions** – not enabled by default. **Session affinity** – enables load balancer to lock a user down to a specific web server.
  + **Duration Based Session Stickiness –** creates the session cookie by the ELB. No cookie? Sent to the nearest EC2 instance. Cookie is updated after its duration expires.
    - Moves user on a new instance if one goes down.
  + **Application-Controlled Session Stickiness –** uses a special cookie to associate the session with the original server but follows the application generated cookie
    - If the application cookie is removed, session stops being sticky.
    - Moves user on a new instance if one goes down.

**Pre-warming an ELB:**

* Used when flash traffic is expected or where a load test can’t be configured to gradually increase traffic.
* Need to contact AWS to pre-warm the ELB.

**Disaster Recovery & AWS:**

* **Disaster Recovery** – about preparing for and recovering from a disaster. Any event that has a negative impact on a company’s business continuity or finances could be termed a disaster.
  + Traditional approach is N+1
    - Facilities
    - Security
    - Capacity
    - Support
    - Contractual agreements
    - Network Infrastructure
    - DHCNP, etc
  + AWS DR
    - Only minimum hardware is required for data replication
    - Allows you to be flexible
    - Pay as you go
    - Automate Disaster recovery
    - Services?
      * EC2
      * **EC2 VM Import Connector** – virtual appliance which allow syou to import virtual machine images from your existing environment to EC2
      * Orchestration
      * Storage
      * Database
      * Compute
    - **RTO (Recovery Time Objective)** – is the length of time from which you can recover from a disaster. It is measured from when the disaster first occurred as to when you have fully recovered it.
    - **RPO (Recovery Point Objective)** – is the amount of data your organization is prepared to lose in the event of a disaster.
    - **Backup & Restore** – High RTO/RPO, data is backed-up up to tape and sent off-site. Can take a long time to restore. S3 or AWS /Import/Export or Glacier can be used in conjunction with a tiered backup solution.
      * Select appropriate tool to backup up your data to AWS
      * Ensure that you have the appropriate retention policy
      * Ensure that appropriate security measure is in place like ACL and encryption
      * Failing Back:
        + Freeze data changes
        + Take a backup
        + restore
        + repoint users to primary site
        + Unfreeze the changes
    - **Pilot Light** – minimal version of an environment is always running in the cloud. Database is an element of this. Have preconfigured servers AMI’s. You can use **elastic IP addresses or ENI (elastic network interfaces)** with pre-allocated Mac address, or ELB to distribute traffic
      * Failing Back:
        + Establish reverse mirroring/replication from the DR site back.
        + Freeze data changes to the DR site
        + Re-point users to the primary site
        + Unfreeze changes.
    - **Warm Standby** – scaled down version of a fully functional environment is always running in the cloud. Decreases the recovery time. Horizontal scaling is preferred.
      * Key steps:
        + Setup EC2 instances to replicate
        + Create and maintain AMI’s
        + Run application with minimal footprint of EC2
        + Patch and update software and configuration files
        + Manually change DNS records
        + Consider Autoscaling
        + Add resilience or scale up database
    - **Multi-Site** – Active/Active configuration. Runs in AWS and Onsite. Use Route53 to root traffic either symmetrically or asymmetrically
      * Key steps:
        + Setup DNS weighting.
        + Configure traffic away from failure site.
        + Application logic for failover.

**AWS Services and Automated Backups:**

* RDS
  + InnoDB for automated backups only
  + Performance hit if Multi-AZ is not enabled
  + If you delete an instance, all automated backups are deleted
  + Can change engine type with a restore from snapshot.
  + Stored on S3.
* Elasticache (Redis Only)
  + Entire cluster is snapshotted
  + Snapshot will degrade performance
  + Stored on S3.
* Redshift
  + Stored on S3
  + 1-day retention period.
  + Only backups data that has changed.
* Don’t:
  + EC2
    - No automated backups
    - Degrades performance
    - Stored in S3
    - Can create automated backups by CLI or Python. Incremental

**EC2 Types – EBS vs Instance Store:**

* Root Volume ( this is where your operating system is installed)
  + Can either be EBS or Instance Store
  + Instance store size = 10GiB
  + EBS root device is up to 2 TiB depending on OS
* Additional Volumes (/dev/sdb, D:\ E:\)
* EBS Backed
  + Can be stopped
  + Root device volume terminated by default unless otherwise defined.
  + Quicker to boot
* Instance Store
  + Can’t be stopped
  + Instance store root device volume is terminated by Default
  + Data is lost by the following circumstances – terminating an instance, stopping an instance, failure of an underlying drive. If it reboots intentionally or unintentially, data remains.
  + Can’t upgrade the instance
  + Additional store volumes do not persiste with Instance store.
* Always best to stop an instance before attaching a volumet to the instance.
* File –s /dev/xvdf
* Lsblk
* Can’t change magnetic standard on the fly. Wait 6 hours if you do make a change on the fly.
* Volumes must be in the same AZ as the EC2 instances

**Storing Log Files & Backups:**

* Monitor by third party. Stored on S3
* Use CloudWatch logs
* Logging on S3

**OpsWork:**

* **Opsworks** - Cloud based applications require a group of related resources, application servers. Helps you automate operational tasks like code deployment, server scaling with Chef
* **Stack** – collection of instances.
* **Layer** – exists within a stack and consists of things like a web application
  + Need 1 or more layers in a stack
  + Instances must be assigned to at least 1 layer.
  + Preconfigured layers – applications, databases, load balancers, caching
* **Chef** – turns infrastructure into code. Chef server stores your recipes. Chef client is installed on each server. Client polls the chef server.
* Opsworks removes any currently registered instances to an ELB and manages it for you.
* Time-based instances or Load-based instances
* Opsworks security groups has dependencies on different security groups

**Security:**

* Can’t rename a policy with the policy generator.
* Can have 5 different versions per policy.
* Get in the habit of using the region flag with S3 commands.
* **Transmission protection** – can connect to an AWS access point via HTTP or HTTPS using SSL, to protect against tampering, message forgery, eavesdropping.
* **Amazon Corporate Segregation** – AWS production network is segregated from the amazon corporate network by network security.
* **ELB** – SSL Termination is supported. Can identify the originating IP address of a client connecting to your servers.