**AWS SA Pro Cert Notes**

**Introduction:**

* 2.7 minutes per question
* 170 minutes
* 80 questions
* scenario based questions
* FAQ’s – cloudformation, direct connect, kinesis, Lambda
* Read Exam blue print
* Take practice exam
* Make a note of technologies in the exam.
* Key Tech: KMS, STS, Import/Export, cloudformation, cloudfront, cloudhsm, cloudsearch, cloudwatch, datapipeline, direct connect, dynamodb, EBS, EC2, ELB, EMR, ElastiCache, Elastic Beanstalk, Elastic Transcoder, Glacier, IAM, Kinesis, Opsworks, RDS, Redshift, Route 53, S3, SES, SNS, SQS, SWF, Storage Gateway, VPC.
* Test 4 things – comprehension, knowledge of AWS, How you cope under pressure, time management.
* Exam strategy – look at sample questions on AWS.com. Read question first.

**High Availability & Business Continuity:**

**\*Read DR whitepaper - http://d36cz9buwru1tt.cloudfront.net/AWS\_Disaster\_Recovery.pdf**

* **DR** – preparing for and recovering from a disaster
  + Usually uses a N+1 approach
  + S3
  + Glacier
  + EBS
  + Direct Connect
  + Storage Gateway
  + EC2/VM Import Connector
  + Route53
  + Elastic Load Balancing
  + VPC
  + RDS
  + DynamoDB
  + Redshift
  + CloudFormation
  + ElasticBeanstalk
  + OpsWork
* **RTO** – time it takes to recover from an outage or disruption
* **RPO** – maximum period of time in which data might be lost from an IT service due to a major incident.
* **Backup/Restore** – Cheapest/longest RPO/RTO
* Pilot Light \
* Warm Standby |-> in SAA Notes
* MultiSite /
* **Know different RTO/RPOS for different AWS services.**
  + **S3**
  + **Glacier**
  + **EBS**
  + **DynamoDB –** offers cross region replication
  + **RDS –** can haveread replica in another region
  + **RedShift –** can copy to another region

DR & BC for Databases

* SQL Server – AlwaysOn Availability Groups, SQL Mirroring
* MySQL – Asynchronous replication
* Oracle – Oracle Data Guard, Oracle RAC
* RDS Multi-AZ Failover
* Automatic failover in case of
  + Loss of availability in **primary AZ**
  + Loss of connectivity to **primary DB**
  + Storage or host failure to **primary DB**
  + Software patching
  + Rebooting of **primary DB**
* Oracle, PostgreSQL, MySQL, and MariaDB use Amazon’s failover technology
* SQL Server DB uses **SQL Server mirroring**
* Amazon Aurora instances stores copies of the data in a DB cluster across multiple **AZ’s**
* MySQL
  + MySQL 5.6 (**NOT 5.1 or 5.5)**
  + Can use both **MyISAM and InnoDB** however InnoDB is supported by AWS
* PostgreSQL
  + PostgreSQL 9.3.5 or newer
* MariaDB
  + All current versions
* Oracle MSSQL
  + All current versions
* **Synchronous replication**
* If application doesn’t require transaction support, consider using DynamoDB if it doesn’t need ACID compliance (Atomicity, Consistency, Isolation, Durability)

**Read Replicas** – elastically scale out beyond the capacity constraints of a single DB instance for read-heavy database workloads

* Can only have **5 read replicas** of a DB at a moment.
* Can be created by console or CreateDBInstanceReadReplica API
* **Asynchronous replication**
* Used for i/o read heavy workloads or serving read traffic when source DB is unavailable **due to i/o suspension for backups or scheduled maintenance** or business reporting or data warehousing scenarios.
* Scaling = read replicas / BC = Multi-AZ
* If Multi-AZ not enabled
  + Snapshot will be of your **primary database** causing a brief I/O suspension for around 1 minute
* If Multi-AZ enabled
  + Snapshot will be of your **secondary database** and won’t have any performance hits.
* Get a new **DNS** endpoint address.
* Can promote a read replica but it does break the replication link between both DB’s.
* Can create read replica’s with RDS (MySQL, PostgreSQL, MariaDB) in different **regions**
  + Improve DR capabilities
  + Scaling read operations into a region closer to you
  + Make it easier to migrate from a DC to another.
  + Can create an Amazon Aurora DB cluster as read replicate in a different region.
  + Can be encrypted or unencrypted DB clusters
  + Read replica must be encrypted if the source DB is encrypted.
  + SQL Server and Oracle doesn’t have it in different regions.
  + **Read replicas can’t be Multi-AZ currently**
  + Can be built off Multi-AZ’s
  + Can have read replicas of read replicas with **only MySQL**
  + DB snapshots and automated backups **can’t** be taken of read replicas.

Storage Gateway

* How can I backup by data?
  + To S3 via **API calls**
  + To Storage Gateway and then replicate to S3.
* **File Interface**
  + NFS(Network File System) protocol stored in S3.
  + S3 buckets as mount points. File Server. Used data is cached on the gateway.
* **Volume interface**
  + **Gateway-Cached Volumes**
    - iSCSI based block
    - store primary data in S3 but retain IA locally
    - Unlimited amount of storage. **File size max is 5 TB**.
  + **Gateway-Stored Volumes**
    - iSCSI based block
    - Store primary data locally and asynchronously backup point in time snapshots to S3.
* **Tape Interface**
  + **Gateway-Virtual Tape Library**
    - iSCSI based virtual tape solution
    - Backed by S3 or VTS (Virtual Tape Shelf) and then backed by Glacier.
    - VTS and VTL retrieval times are different.

Snowball

* **Import/Export Disk** – move large amounts of data in and out of AWS using a portable storage device using Amazon’s high speed internal network.
* **Snowball** – PB scale transport to transfer data in and out of AWS. 80 TB snowball in all regions.
  + TPM enclosures
  + 256 encryptions.
  + Data is erased
  + Need a client to connect to the snowball.
* **Snowball Edge** – contain 100TB of storage with transfer device capability. Comes with compute capability whereas snowball does not. Lamba functions can be run from this.
* **Snowmobile** – Diesel truck can transfer up to 100PB of data.

Know snowball, import/export, what snowball can do(export to S3, etc)

Automated Backups

* Have
  + RDS
    - MySQL you need **InnoDB**
    - Performance hit Multi-AZ is not enabled
    - If you delete an instance, **ALL** automated backups are deleted
    - Manual DB snapshots will **NOT** be deleted.
    - Stored on S3
    - When you do a restore, you can change the engine **type (SQL Standard to SQL Enterprise)**
  + Elasticache (**redis only)**
    - Entire cluster is snapshotted
    - Snapshot will degrade performance
    - Set your snapshot window during the least buy part of the day.
    - Stored on **S3**
  + Redshift
    - Stored on **S3**
    - Enables automated backups of your data warehouse cluster with a **1-day** retention period
    - Incremental backup that has changed so most snapshots only use up a small amount of your free backup storage.
* Don’t Have
  + EC2
    - Backups **degrade** performance
    - Need automated script by python or CLI
    - **Incremental** backups
      * Only charged for incremental storage
      * Contains base snapshot
    - Stored in **S3**
    - Not automated.
* **Domain 1.0 – Summary :HA and BC**
  + Demonstrate ability to architect the appropriate level of availability based on stakeholder requirements
  + 1.2 Demonstrate ability to implement **DR** for systems based on **RPO** and **RTO**
    - **Know how to backup each service and look closely at RPO/RTO**
    - Backup/Restore
    - Pilot Light
    - Warm Standby
    - Multi-Site
    - S3
      * **11 9s**
    - EBS
    - Archives
      * **3 hours or longer**
    - **DynamoDB – always go for this first.**
    - **RDS**
    - **Redshift**
  + 1.3 Determine appropriate use of **Multi-AZ vs. Multi-region architectures**
    - Know different use cases
      * **Multi-AZ** for DR and BC
      * **Read Replicas** for scaling
    - Know different types of replication
      * **Synchronous** replication for **Multi-AZ**
      * **Asynchronous** replication **for read replicas.**
    - App doesn’t require **ACID**, use **DynamoDB**.
    - Know **4** different Storage Gateways
      * **File Gateway**
      * **Volume Gateway**
        + Cached
        + Stored – store data as **EBS snapshots in S3.**
      * **Tape Gateway**
        + Virtual tapes access are **instantaneous**
        + Virtual tape shelf can take **24 hours.**
    - Encrypted using SSL for transit and encrypted at rest in **Amazon S3** using **AES-256**
    - Snapshots can be scheduled
    - Bandwidth can be throttled **(good for remote sites)**
    - **Know Snowball and Import/Export**
      * Snowball can **import** to S3 and **export** from S3.
  + 1.4 Demonstrate ability to implement self-healing capabilities
  + **15%** of the exam

**Domain 2- Costing & Account Management:**

Cross Account Access – Roles & Permissions

* Makes it easier to operate in a multi-role or multi-account environment.

Consolidated Billing & Organization

* **Organization** – account management service that enables you to consolidate multiple AWS accounts into an organization.
  + Consolidated Billing
  + All features
* Paying account is independent. All linked accounts are independent. Only **20** accounts can be linked. Accounts have to be linked to get unused RI usage
* Can have billing alerts. Montoring at the paying account will include all linked accounts.
* **One bill, easy to track, discount volume pricing**
* Paying account should be used for billing purposes only.
* Use **strong** and **complex** password
* Always **enable** MFA
* CloudTrail is **per account** and **per region**
  + Can consolidate logs using an **S3 bucket**
  + Turn on cloudtrail
  + Create bucket policy for cross account access
  + Turn on cloudtrail in the other accounts and use the bucket in the paying account.

Tagging & Resource Groups

* **Tags**
  + **KeyValue Pairs** attached to AWS resources
  + Metadata (**Data about Data)**
  + Tags can sometimes be **inherited.**
* **Resource groups –** make it easy to group your resources by tags
  + Include information iike region, name, health checks
  + EC2 – Public & Private IP addresses
  + ELB – Port configurations
  + RDS – Database engine
  + Can export all columns
  + Tag editor is used to find resources without tags

Reserved instances for EC2 & RDS

* **On-Demand** – allowy ou to pay a fixed rate by the hour with no commitment
  + low cost and flexibility with no upfront
  + applications with short terms, spiky, unpredictable workloads
  + applications being developed or tested on EC2
* **Reserved** – provide you with a capacity reservation and offer a significant discount on the hourly charge for an instance 1 or 3 year terms
  + Applications with steady state or predictable usage
  + Applications that require reserved capacity
  + Users can make upfront payments to reduce total computing costs
  + Pay all upfront = **Largest Discount** (Up to 75%)
  + Partial Up Front = **Middle Discount**
  + No Upfront = **Least Discount** (Still cheaper than on demand)
  + Can modify these reserved
    - Switch AZ
    - **Change instance type within the same instance family.**
  + **RDS – RI**
    - DB Engine, EB instance class, Deployment type, region, license model
    - Reservation can be applied to DB with the s**ame** attributes. If the attributes are changed, pricing **reverts** to on-demand
    - If you change them back to match the **same** attributes in the beginning, RI pricing will be applied until the end of the term.
    - Can reserve instances for **Multi-AZ** and **read replicas**. Read replicas must be in the same region.
    - Reservations **can** be used in any available AZ within the region.
  + **Standard RIs**
    - Best for steady state usage and has the most discount (**up to 75% off On-Demand)**
    - Can only sell these on the marketplace.
    - Can **change** these but need a modification request but based on the footprint remaining the same. Calculated by using **normalization factors.**
    - **Normalization factors** 
      * Each **RI** has an instance size which is determined by the normalization factor and the number of instances in the reservation.
      * A modification request is **not** processed if the footprint of the target configuration does not match the size of the original configuration. Footprint is measure in units in the **EC2 console**.
      * Footprint = (instance type & # of instances \* normalization factor)
      * Always reserve a large instance in case you need to dice it up.
      * Only for linux (excluding RedHat and Suse)
  + **Convertible RI’s**
    - Capability to change the attribute of the RI as long as the exchange results in a creation of RI is equal or greater value. Best for stead-state. **Up to 45% off On-Demand**
    - Offers Change instance families, OS, tenancy, and payment option
  + **Scheduled RI’s**
    - Available to launch within the time windows you reserve. Match capacity reservation to a predictable recurring schedule that only requires a fraction of a day.
* **Spot** – enable you to bid whaterver price you want for instance capacity, providing for even greater savings if your applications have flexible start and end times.
  + Applications that have start and end times
  + Applications feasible with low compute prices
  + Urgent computing needs for large amounts of capacity
* **Dedicated** – run in a VPC on hardware that is dedicated to a single customer. Isolated at the host hardware level from your instances that aren’t dedicated and from instances that belong to other AWS accounts.

KNOW YOUR EC2 INSTANCE TYPES

* D2(Dense)
* R4(Memory)
* M4(General Purpose)
* C4(Compute)
* G2(Graphic Intensive)
* I3(High Speed Storage)
* F1(Field Programmable)
* T2(General Purpose)
* P2 (General Purpose GPU)
* X1(Memory Optimized)

**Domain 2.0 - Costing:**

* 2.1 Demonstrate ability to make architectural decisions that minimize and optimize infrastructure cost
  + Resource groups
  + Consolidated Billing
  + CloudTrail
  + RI
  + Cross account access
  + EC2 instance types
* 2.2 Apply the appropriate AWS account and billing set-up options based on scenario
* 2.3 Ability to compare and contrast the cost implications of different architectures
* 5% of exam
  + read the question. See if it is asking you to commericially feasible way to design a solution or HA with low RTO/RPOs

**Domain 3 – Deployment Management:**

Cloudformation

* **Read the FAQ**
* Allows you to take what was once traditional hardware and convert it to code.
* Gives developers and systems administrators an easy way to create and manage a collection of related AWS resources, provisioning and updating them in an orderly and predictable fashion.
* Look at all the services supported by Cloudformation - <http://docs.aws.amazon.com/AWSCloudFormation/latest/UserGuide/cfn-supported-resources.html>
* **Template** – architectural diagram
* **Stack** – end result of that diagram
* Built in JSON or YAML
* **Mandatory Elements:**
  + List of Resources and configuration values
* **Option Elements:**
  + Template file format & version number
  + Parameters,
    - Input value that are supplied at stack creation time. Limit of 60.
  + Output Values
    - Required once a stack has finished building. Limit to 60
    - **Fn:GetAtt** to output data
  + List of data tables
    - Used to look up static configuration files.
* **Supports Chef, Puppet Integration, Bootstrap scripts**.
* **Automatic rollback on error enabled by default**
* **Charged for resource** but using it is **free**.
* **Wait condition** - acts as a barrier, blocking the creation of other resources unitl a completion signal is received from an external source
* Can define **deletion** policies.
* Can preserve a resource when the stack is deleted.
* Can update a stack after it is created by **modify or update.**
* Can be used to create **IAM roles** in AWS or granting access.
* Can specify **CIDR ranges** or individual IP addresses or **preexisting EIP’s**.
* Can create **multiple VPC’s in a template** and **enable VPC Peering** but only in the **same** account.
* Can create **new hosted zones or update existing** like Route53. Can add or change records.
* **Know all services supported**
* **Remember what is mandatory for a template.**

Elastic Beanstalk

* Makes it easier to quickly deploy and manage applications in AWS.
* EB **will not provision** CloudFormation templates
* CloudFormation supports EB.
* Meant to get environments up and running
* Supported Languages
  + Packer Builder
  + Apache
  + Docker
  + Ruby
  + Go
  + Java
  + .NET
  + Node.js
  + Python
  + PHP
* Built in **Cloud Watch** monitoring
* Adjust **JVM settings** and pass variables.
* Can run **memory caching** service
* Can access **log files** without logging in.
* **Visual Studio or Elicpse** allows you to deploy straight into AWS without leaving your IDE
* Can push out updates from GIT: only modified files are transmitted
* Supports multiple environments like one for pre-production/production. Each environment runs on its own separate AWS resources.
* Stores application files in S3. Files uploaded will be **automatically** copied from your local client to S3. Can **optionally** set EB to copy your server log files **every hour.**
* Can store **application data** but best to include **AWS SDK** as part of the WAR file for the application.
* Can deploy RDS instances **automatically**. Connectivity is **exposed via environment variables.**
* Can be configured to be **fault tolerant** within a **region using multiple** AZ’s. **Not fault** tolerant between regions.
* App will be publicly available. Integrates with **VPC**. Can restrict via **white-listing** at SG or NACL. Fully supported by **IAM.**
* Full access to resources under EB.
* Can **roll** back changes. **Multiple** environments are allowed to support version control. **Charged for errors**. Cloudformation to use is **free**. **Only** changes from **Git Repo** are replicated.
* Windows 2012 R2 and Windows 2008. 2016 isn’t **supported** yet. Linux AMI

Opsworks

* **Stack** – collection of instances
* **Opsworks** – straightforward way to create and manage stacks. Makes it easy to automate operational tasks like code deployment, software using Chef.
  + Flexibility to define application architecture, resource configuration and provisioning resources. Scale based on time or load to make it easier.
* **Chef** – turns infrastructure into code. You can automate, build, deploy, and manage infrastructure.
  + **Nodes** – chef client installed on each server
  + **Stacks** – container of resources such as ELB, EC2
  + **Recipes** -
  + **Layers** – exisdts in a stack and consisting of different layers.
    - OpsWorks takes care of the manual configuration.
    - **1 or more** layer in a stack
    - **at least 1** layer with an instance.
    - Preconfigured – applications, databases, Load balancers, caching
  + Chef 11 for builtin functions
  + Sample stack/Chef 12 doesn’t create all the SG like Chef 11
  + After you attach an ELB, Opsworks removes any registered instances and then manages the ELB. If the ELB is modified, shanges will not be performed.

**Domain 3** – High Availability and Business Continuity

* **3.1** – Ability to manage the lifecycle of an application on AWS
* **3.2** – Demonstrate ability to implement the right architecture for development, testing, and staging environments
* **3.3** – Position and select most appropriate AWS deployment mechanism based on scenario
* **10%** of exam

**Domain 4 – Network Design:**

VPC Refresher

* Know how to build a **VPC**, **public/private subnet**
* Know what a **NAT** is
  + Disable Source/Destination Checks
* Know what a **route table** is
  + Subnets can communicate with each other by default

Build your own customer VPC

* Subnets can communicate with different AZ by default
* SG can span AZ but subnet can’t span AZ

Deploying a NAT LAB

* Use a larger nat instance if you’re not getting enough throughput.
* Disable source destination checks

VPC Peering

* **Peering** – connection between two VPC’s that enables you to route traffic between them using Private IP addresses **within a single region.** Can be done across regions.
  + Transitive Peering NOT Supported
  + Can Peer to 50 VPC’s if needed by a soft limit. Can go up to **125** by request
* Limits:
  + Can’t create a Peering connection between VPCS having CIDR blocks
  + Can’t create VPC Peering connection between VPC in different regions
  + A placement group can span peered VPCs; however, you will **not** get full bandwidth between instances in peered VPCs.
  + Private DNS values **can’t** be resolved between instances in peered VPCs.
* **Setup**
  + Owner of local VPC sends a request to the owner of the second VPC to peer
  + Owner of the second VPC has to accept
  + Owner of the local VPC adds a route to their route table allowing their subnets to route out to the peer VPC
  + Owner of the peer VPC adds a route to their route table allowing their subnets to route back to the other VPC
  + Security Groups in both VPC have to both allow traffic.
* **Troubleshooting**
  + Are the VPCs in the same region?
  + If you can’t create a VPC peer, check to see if the CIDR blocks are overlapping
  + Check that the relevant security groups and NACLS are allowing traffic through
  + Check that a route has been create in BOTH VPC’s routing tables.

Direct Connect

* Makes it easy to establish a dedicated network connection from your premises to AWS
* **Uses 802.1q VLANS**. Can be partitions into multiple virtual interfaces **(VIFS).** Can use the **same** connection to access public resources
  + Allow **public connections** to **EC2** or **S3** using public IP addresses
  + Allows **private connections** to **VPC** using internal IP addresses
* Doesn’t involve the internet instead uses a private connection
* VPN uses the internet with low to modest bandwidth requirements.
* Available in **10Gbps, 1 Gbps, Sub 1 Gbps** can be purchased through AWS Direct Connect Partners
* For Disaster, use **BGP** to Fail over automatically from Direct Connect to S2S VPN
  + Can also have 2 connections (2 routers, 2 direct connects) or S2S
* **Customer Gateway (CGW)** – anchor on your side of that connection which can be physical or software appliance
* **Virtual Private Gateway** – anchor on the AWS side of the VPN connection
* In the US, you only nee**d 1 direct connect connection to connect in to all 4 US regions**. Data goes over AWS internal Lines.
* **Layer 2** connections are **not** supported.
* **Direct Connect FAQ**
* Benefit:
  + Increase **reliability**
  + Increase **bandwidth**
  + Reduce **costs** when using large volumes of traffic.

HPC & Enhanced Networking

* Involves batch processing with large and compute intensive workloads
* Demands HPCU, Network & Storage
* Usually Jumbo Frames are required
  + **Jumbo Frames** – Ethernet frames with more than 1500 bytes of payload. Can carry up to 9000 bytes. Supported through enhanced networking.
  + **NFS** uses Jumbo Frames. HPC uses a lot of Disk I/O
  + **Enhanced Networking** uses **SR-IOV(Single Root- I/O Virtualization)** on HVM.
    - C3, C4, D2, I2, M4, R3
* **Placement group** – logical grouping of instances within a single **AZ**. Participation in low-latency, 10 Gbps. Best to choose an instance that supported enhanced networking.
  + Doesn’t span diferent AZ’s. 1 PG = 1 AZ. Can span subnet.
  + **Existing instances can’t** be moved in a placement group.
  + Always have enough capacity in the placement group as there may not be sufficient capacity
  + Use homogenous instance types with placement groups.
  + <http://docs.aws.amazon.com/AWSEC2/latest/UserGuide/enhanced-networking.html>

ELB

* automatically distributes incoming application traffic across multiple EC2 instances.
* Provides greater fault tolerance
* Classic Load Balancer (assume CLB in exam)
  + HA deploy in multiple or single.
  + Health Checks
  + Security Features
  + SSL Offloading. Cipher support
    - If using multiple SSL certs, you should use multiple ELBS unless you have a wildcard cert.
  + Sticky Sessions (Cookie based)
  + IPv6 & IPv4 (VPC’s don’t currently support IPv6)
  + Layer 4 or Layer 7 Load Balancer features (X-Forwarded/Sticky)
  + Operational Monitoring
  + Logging
  + EC2-VPC (1-65535)
  + EC2-Classic (25,80,443,465,587, 1024-65535)
  + **Can’t assign an EIP** to an ELB
  + Can load balance to **Zone Apex** of your domain name.
  + Get History of ELB API calls via Cloudtrail
* Application Load Balancer
  + HA - Must be deployed in multiple AZ and not single.
  + **Content-Based Routing** – route requests to a service based on the content of the request.
  + **Host-Based Routing** – route client requests on host field of HTTP to multiple domains
  + **Path-based Routing** – route a client request based on URL path of HTTP header
  + Containerized Application Support (integrates with ECS)
  + HTTP/2 Support – single multiplex
  + WebSockets Support – real-time messages delivered
  + Native IPv6 Support
  + Sticky Sessions
  + Security Features
  + Integrates with WAF
  + Layer 7 LB
  + HTTPS Support (SSL Offload)
  + Operational Monitoring
  + Logging
  + Delete Protection
  + Request Tracing – custom identifier “X-Amzen-Trace-Id” HTTP header on all requests
  + Health Checks

Scaling NATS

* Reducing bottlenecks:
  + **Scale Up**
    - Increase instance size
    - Choose instance family with enhanced networking
  + **Scale Out**
    - 1 NAT can only route to 1 subnet. Add an additional NAT and migrate half your workloads to the new subnet.
  + **HA**
    - Can failover to another subnet
    - [**http://nineofclouds.blogspot.ca/2013/01/vpc-migration-nats-bandwidth-bottleneck.html**](http://nineofclouds.blogspot.ca/2013/01/vpc-migration-nats-bandwidth-bottleneck.html)
    - [**https://aws.amazon.com/articles/high-availability-for-amazon-vpc-nat-instances-an-example/**](https://aws.amazon.com/articles/high-availability-for-amazon-vpc-nat-instances-an-example/)

**Domain 4 – Summary**

* 4.0 Network Design for a complex large scale deployment
* 4.1 Demonstrate ability to design and implement networking features of AWS
* 4.2 Demonstrate ability to design and implement connectivity features of AWS
* 10% of exam

**Domain 5 – Data Storage:**

Optimizing S3

* two options
  + Strong internet connection, with fast speeds
  + Less reliable internet connection with inconsistent network performance
* Optimize for PUTS
  + Take advantage of the network itself and make the network the bottleneck
  + Weaker networks, we want to prevent large files having to restart their uploads
  + Parallelizing your puts by dividing them
  + If one fails, it can be restarted. Helps to increase aggregate throughput
  + 25-50 MB size of chunks on high bandwidth networks. 10 MB on mobile networks
* Optimizing for GETS
  + Use Cloudfront
  + RTMP & Web distribution
  + Use range-based GETS with multithreaded performance via HTTP header.
  + Can send multiple GETS at once
* Lexicographical – dictionary order
  + Introduce randomness to spread it across S3
  + Better performance with more random