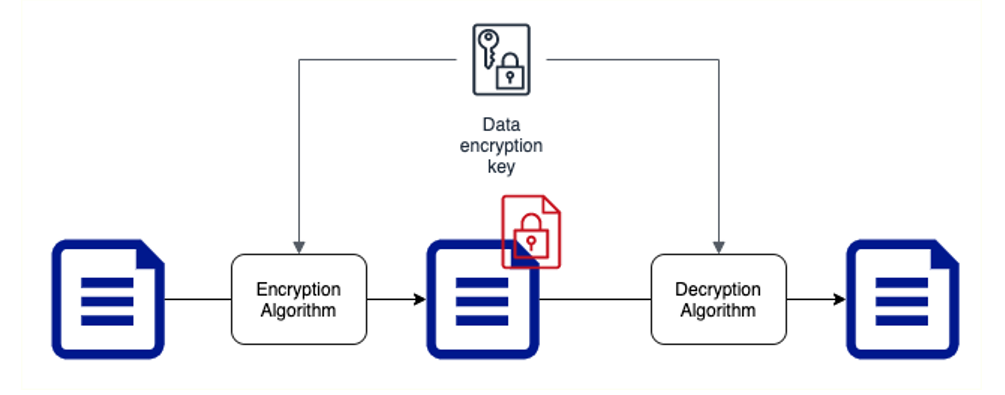
**Data States**

* Data at rest: Stored on a device or a backup
  + Examples : data on a hard disk, in a database, backups and archives
* Data in motion: Being transferred across a network
  + Also called Data in transit
  + Examples :
    - Data copied from on-premise to cloud storage
    - An application in a VPC talking to a database
  + Two Types:
    - In and out of AWS
    - Within AWS
* Data in use: Active data processed in a non-persistent state
  + Example: Data in your RAM

**Encryption**

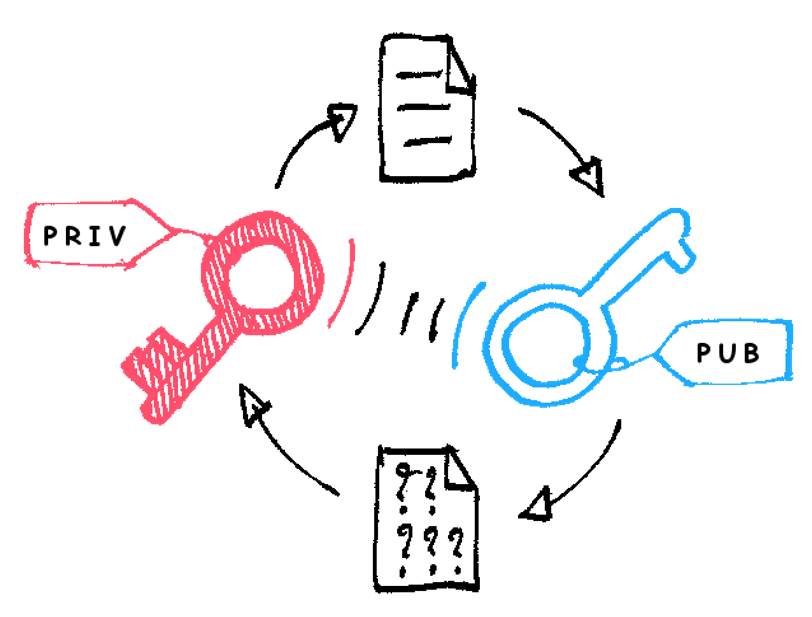
* If you store data as is, what would happen if an unauthorized entity gets access to it?
  + Imagine losing an unencrypted hard disk
* **First law of security** : Defense in Depth
* Typically, enterprises encrypt all data
  + Data on your hard disks
  + Data in your databases
  + Data on your file servers
* Is it sufficient if you encrypt data at rest?
  + No. Encrypt data in transit - between application to database as well.

**Symmetric Key Encryption**



* Symmetric encryption algorithms use the same key for encryption and decryption
* Key Factor 1: Choose the right encryption algorithm
* Key Factor 2: How do we secure the encryption key?
* Key Factor 3: How do we share the encryption key?

**Asymmetric Key Encryption**



* Two Keys : Public Key and Private Key Also called Public Key Cyptography
* Encrypt data with Public Key and decrypt with Private Key
* Share Public Key with everybody and keep the Private Key with you(YEAH, ITS PRIVATE!)
* No crazy questions:
  + Will somebody not figure out private key using the public key?
* How do you create Asymmetric Keys?

**KMS and Cloud HSM**

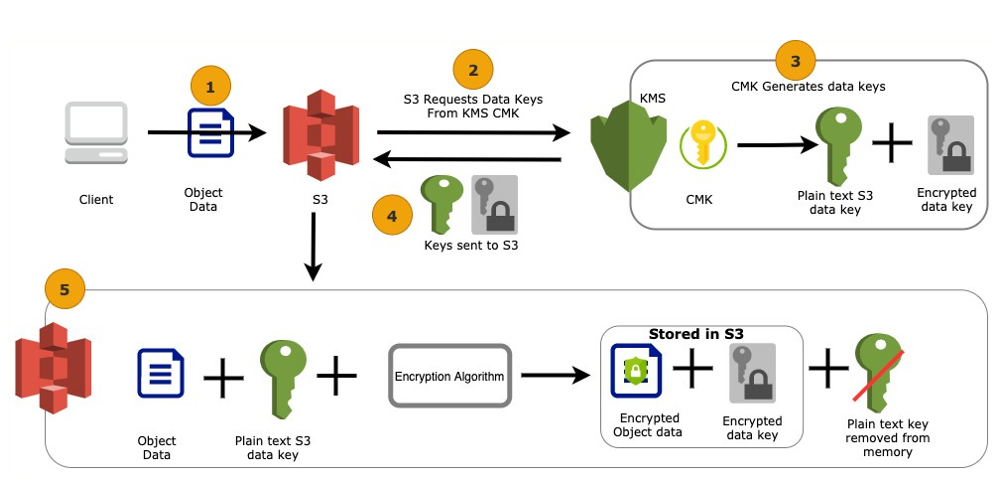
* How do you generate, store, use and replace your keys?
* AWS provides two important services - **KMS and Cloud HSM**
  + Manage your keys
  + Perform encryption and decryption

**AWS KMS**

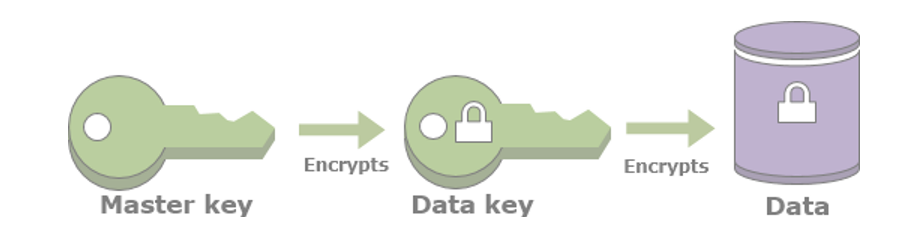
* Create and manage cryptographic keys (symmetric and asymmetric)
* **Control their use** in your applications and AWS Services
* Define key usage permissions (including cross account access)
* Track key usage in AWS CloudTrail (regulations & compliance)
* Integrates with almost all AWS services that need data encryption
* Automatically rotate master keys once a year
  + No need to re-encrypt previously encrypted data (versions of master key are maintained)
* Schedule key deletion to verify if the key is used
  + Mandatory minimum wait period of 7 days (max-30 days)

**Server Side Encryption with KMS**

* Create Customer Master Key. Map to AWS service (S3)
* Steps
  + Data sent to S3
  + S3 receives data keys from KMS S3 encrypts data
  + Stores encrypted data & data key
* Remember
  + CMK never leaves KMS
  + Encryption of data key – KMS using CMK
  + Encryption of data - AWS Service - Amazon S3 using data key



**Envelope Encryption**



* The process KMS uses for encryption is called Envelope Encryption
  + Data is encrypted using data key
  + Data key is encrypted using Master key
  + Master key never leaves KMS
* KMS encrypts small pieces of data (usually data keys) <4 KB

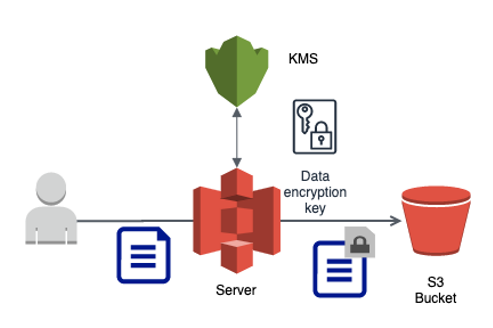
**Decryption of data using KMS**

* AWS service (Amazon S3) sends encrypted data key to KMS, KMS uses Customer Master Key (CMK) to decrypt and return plain-text data key
* AWS service (Amazon S3) uses the plain-text data key to perform decryption
* (TIP) Remove plain-text data key from memory asap
* (TIP) AWS service needs IAM permissions to use the CMK

**AWS CloudHSM**

* Managed (highly available & auto scaling) dedicated single- tenant Hardware Security Module(HSM) for regulatory compliance
  + (Remember) AWS KMS is a multi-tenant service
* FIPS 140-2 Level 3 compliant
* AWS CANNOT access your encryption master keys in CloudHSM
  + In KMS, AWS can access your master keys
  + Be ultra safe with your keys when you are using CloudHSM
  + (Recommendation) Use two or more HSMs in separate AZs in a production cluster
* AWS KMS can use CloudHSM cluster as "custom key store" to store the keys:
  + AWS Services can continue to talk to KMS for data encryption
  + (AND) KMS does the necessary integration with CloudHSM cluster
* (Best Practice) CloudWatch for monitoring and CloudTrail to track key usage
* Use cases
  + (Web servers) Offload SSL processing
  + Certificate Authority
  + Digital Rights Management
  + TDE for Oracle databases

**Server Side Encryption**

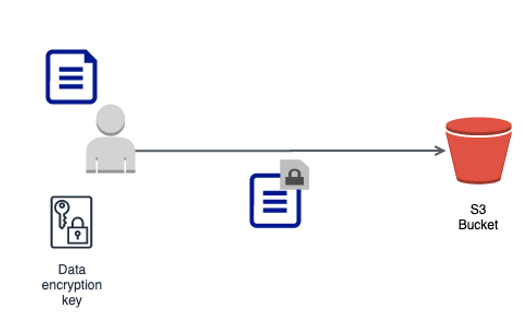


* Client sends data (as is) to AWS service
* AWS service interacts with KMS to perform encryption on the server side
* Recommended to use HTTPS endpoints to ensure encryption of data in transit
  + All AWS services (including S3) provides HTTPS endpoints
  + Encryption is optional with S3 but highly recommended in flight and at rest

**Server Side Encryption - S3**

* SSE-S3:
  + AWS S3 manages its own keys
  + Keys are rotated every month
  + Request Header - x-amz-server-side-encryption(AES256)
* SSE-KMS:
  + Customer manages keys in KMS
  + Request Headers - x-amz-server-side-encryption(aws:kms) and x-amz-server-side- encryption-aws-kms-key-id(ARN for key in KMS)
* SSE-C:
  + Customer sends the key with every request
  + S3 performs encryption and decryption without storing the key
  + HTTPS is mandatory

**Client Side Encryption**



* Client manages encryption process and sends encrypted data to AWS service
  + AWS will not be aware of master key or data key
* AWS service stores data as is
* For Amazon S3, you can use a client library (Amazon S3 Encryption Client)