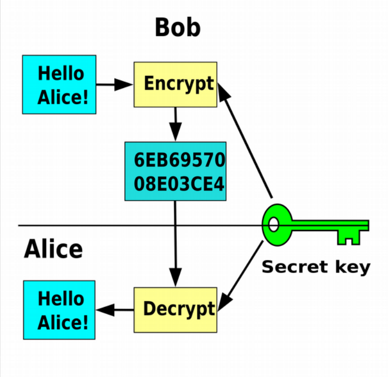
REA AWS Training

KMS (aka crypto time)

What is encryption

- In cryptography, encryption is the process of encoding messages or information in such a way that only authorized parties can read it.
- Encryption does not of itself prevent interception, but denies the message content to the interceptor.

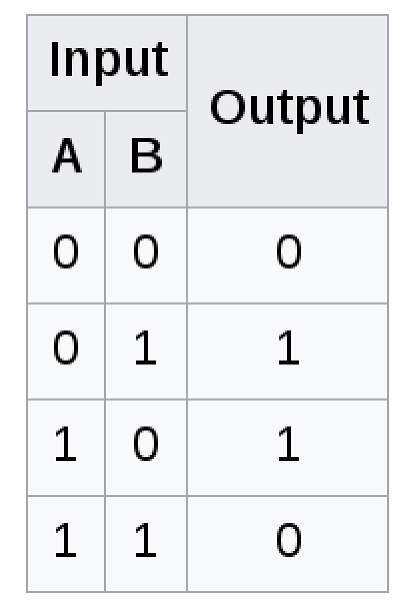
Symmetric Key



- Same key to encrypt/decrypt
- Fast (compared to RSA)
- Two main types of cipher (block and stream)
- Stream ciphers produce a constant stream of pseudo-random bits which the data is XORed with
- Block ciphers work on fixed size blocks of data. Your data is cut/padded into ***N*** sized blocks and encrypted
  - There are a number of block cipher modes of operation and block ciphers are more versatile than stream ciphers (for example they can provide authentication of encrypted data), however stream ciphers are often faster
  - There are also some bad weaknesses in certain block cipher operation modes (Google search for “ECB Penguin”)
  - Also consider if we have a repeating word input and use a fixed key block cipher, will our ciphertext reveal anything that may compromise the secrecy of the data?
- Can provide authenticity guarantees in some cases (KMS does)
  - (AEAD/Authenticated Encryption)
- Suffers from key sharing problem
- EG: AES (block [but can work like a stream in CTR mode]), RC4 (stream), Blowfish (block)

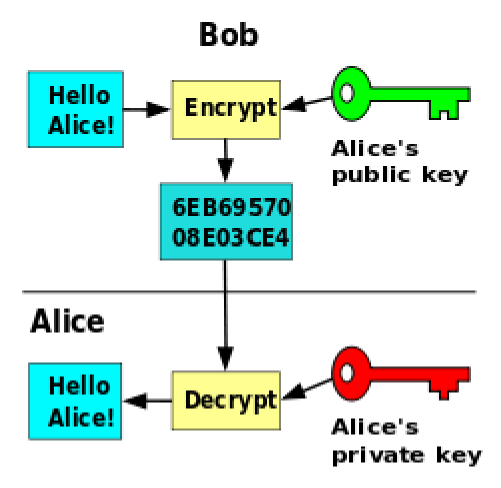
XOR (exclusive or operation)

0 == False
1 == True



Input		Output
A	B	
0	0	0
0	1	1
1	0	1
1	1	0

Asymmetric (Public Key)



- One key to encrypt, one key to decrypt
  - Keys are mathematically related
  - Security based in hardness of factoring large prime numbers (NP-Hard)
  - The discrete logarithm problem can also be used as a basis for asymmetric cryptography (research this)
- Solves key sharing problem
- Can be used for encryption/signatures/both
- Can only encrypt a small amount of data [(key size in bits / 8) - padding]
- Slow (computationally when compared to symmetric key algorithms)
- EG: RSA

How RSA works

- http://sergematovic.tripod.com/rsa1.html

Question

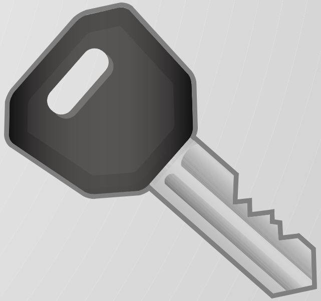
- How can they be used together?
  - Think about how SSL works :)

KMS: Key Management Service

- Manages crypto keys for you (you don't need to understand cryptography well to use it securely)
  - Handles key rotation/revocation
  - Handles access control to keys (using “grants” and IAM)
  - Provides a nice API for encrypting/decrypting
  - Can encrypt up to 4K of data by itself (master key)
    - Can generate data keys to encrypt more
- Used by many other services (EC2, S3, RDS)
- Under the IAM page in the console (make sure to set reg
  - IAM → Encryption Keys

KMS access control for users

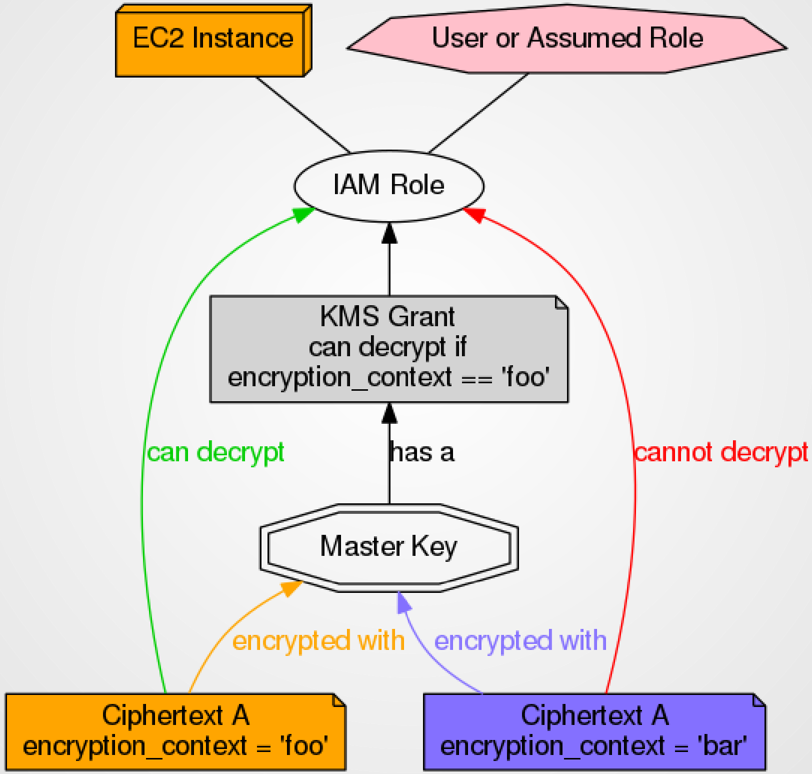
- Key management
  - Create/delete/rotate
  - access management: update policies/grant/revoke
- Key Use
  - encrypt/decrypt/re-encrypt
  - generate data key
- Cross-account support



KMS for applications

- Grants provide long-term finer-grained control
  - restrict to specific principal
  - optional encryption context (supplied by the client)
- Application can provide Grant Token for faster access

Encryption contexts and grants



KMS: encrypting/decrypting small data

- Generate master key (with policies/grants)
  - returns key ID and/or alias
- Call Encrypt API with your plaintext, get back a ciphertext blob
  - Blob includes key reference (you don't need to pass in the key ID when calling the Decrypt operation)
- Authorized users call Decrypt API with ciphertext blob
  - Returns plaintext

KMS: encrypting/decrypting large data

- Max of 4KB per master key (explained earlier)
- Generate data key
  - Returns plaintext and enciphered versions of the same key
- Encrypt large data with plaintext key
  - Then throw away plaintext key
- Store encrypted data and ciphertext of key together
- Pass ciphertext of data key to Decrypt API (policy/grant controls)
  - It will be decrypted by the master key
  - Returns plaintext of data key
- Use decrypted data key to decrypt data

KMS: Integration with AWS services

- S3
- EBS
- RDS (actually using EBS underneath)
- Etc.

KMS caveats

- Keys can deleted, but only after a (minimum) 7-day cooling-off period
- Don't rely on encryption context to authenticate non-AWS applications
  - fine for grants against S3, for example
- AWS uses default per-region key if you don't specify
- Grants may not become active instantly
  - use Grant Tokens to work around this

A Final Caveat

- The master keys can never be extracted from within KMS (as they are stored in a Hardware Security Appliance).
- This means if your app secures customer data with KMS you are coupling yourself to AWS.
- If you want to move to another cloud you can take the ciphertexts with you, but you will not be able to decrypt them outside of AWS!
- Read this to better understand https://docs.aws.amazon.com/kms/latest/developerguide/importing-keys.html

KMS hands-on

- Follow the hands-on steps in `…/kms/hands-on.md`