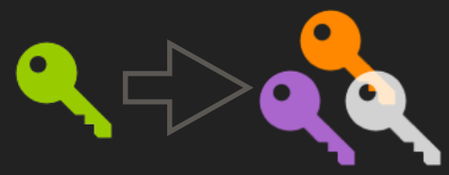
SOLUTIONS FOR DERIVING KEY(S)



```
// Input:
//  Master_key and
//  (DB) record_id target record DB id
// Output:
//  AES-Key and
//  salt for encrypting target record
```

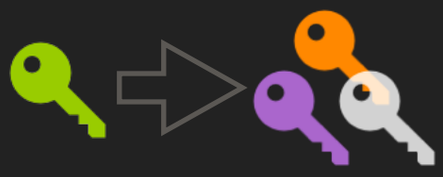
```
// AES-Key and salt for target record. "||" concatenates
// AES-CBC uses 128 bit IV. AES-GCM uses a 96 bit IV
byte[32] keyAndIV = derive_key( master_key ||
                                record_id || record_version, 256 bit)

byte[16] derived_iv   = keyAndIV[0..15]
byte[16] derived_key  = keyAndIV[16..31]
```

- derive_key needs an additional *installation specific* salt of >= 128 bit. PBKDF2 with HMAC sha256 is an example of derive_key, as is scrypt or argon2.
- Use same process for decryption.
- No need to store the *generated* IV value.

IMPORTANT: NEVER USE THE SAME KEY/IV TO ENCRYPT DIFFERENT DATA

MAKE SURE THAT THE MASTER KEY HAS ENOUGH ENTROPY FOR DERIVED KEY AND DERIVED SALT

NEVER REUSE KEYS!



- Encrypting different data with the same key and IV can lead to complete loss of confidentiality / integrity (*)
- **When updating encrypted records a new IV must be used** (better: a new key and IV)
- This can be achieved by incrementing a record-version on each encryption and using it in the key derivation.

(*) This is because of the way CTR/GCM/CBC/… work. See e.g Appendix B of NIST Sp. Pub. 800-38A

IMPORTANT: NEVER USE THE SAME KEY/IV TO ENCRYPT DIFFERENT DATA

MAKE SURE THAT THE MASTER KEY HAS ENOUGH ENTROPY FOR DERIVED KEY AND DERIVED SALT