## Weekly Goal

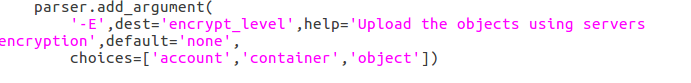
1.Finish the procedure between client and proxy. The client sets the encryption setting and the proxy gets it and passes it to the server with an IV.

2.The token stored in Database now is hashed(md5). So, the Database will be more secure because no one could match the key in database server to the objects in storage node.

1.The encryption level is set by client by using command -E

On the client-end,

In shell.py



There are three choices(levels) ‘account’, ‘container’ and ‘object’. The default is ‘none’, which means it won’t use any encryption.

In service.py



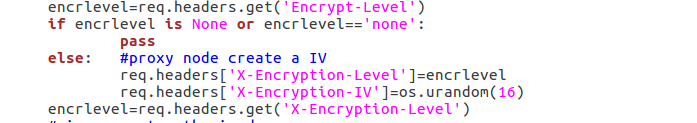
In client, options[‘encrypt\_level’] will be written into request.headers.

On the proxy-end,

In obj.py method PUT

The request from above will be passed into this method

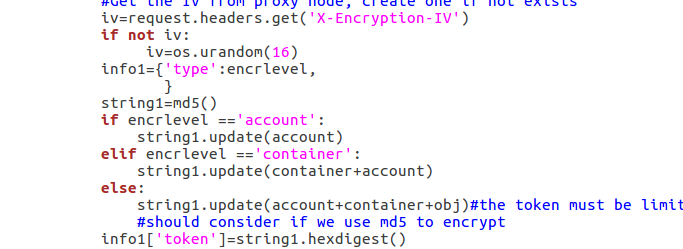




The proxy processes this request, generate an IV, then pass all to the storage node (Server)

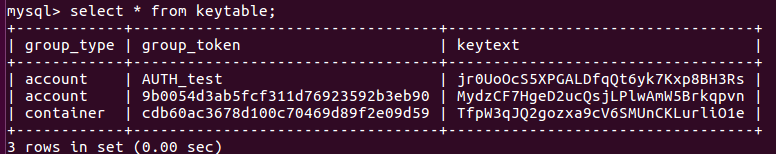
On the server-end

In obj.py method PUT

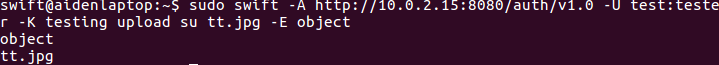


Now the token will be md5 of the names(account/container/obj). So, if one wants decrypt the object, it will need the key and the token. They are separately saved in server and database, which means one must attack both of them to decrypt the object.

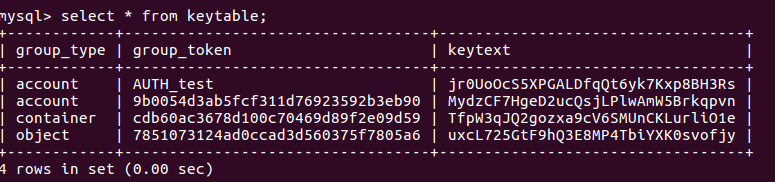
## Function test



The database



In command, we send a new object with encryption-level of object.



After that, a new row inserted into the database. And the token is the md5 of the obj, so no one could decrypt the object just getting the database’s data.

## Problem

Our project implements the encryption method on server-end. And Swift originally have an encryption middleware on client-end. So, this two can cooperate together, but what will be the most advantage for encryption on server-end over client-end?

## Possible improvement

I am wondering if it will be better to keep the count of key recorded. Like if we delete an object that using a key, maybe we should decrease the count of the key by 1.If the count reduced to 0, we will delete the key. So the resource can be saved and also more secure the database will be.

## Next Step

The core project has been completed. Next is to implement all the functions on iCloudLab, test them.