03. gRPC 실용사례 python

본 문서 에서는 python 에서 gRPC 를 사용하여 Client 프로그램을 작성 하는 방법을 소개 한다. 사전 조건 으로 python3 및 grpc package 가 설치 되어 있어야 한다. python 에서 grpc 설치는 여기 의 `2. Python 에서 gRPC 사용하기` 를 참조 한다.

1. Interface 를 위한 .proto 파일

실용사례 에서 공통으로 사용할 .proto 파일로 이름은 `xdb_grpc.proto` 로 작성 하였다.
`01. gRPC 실용사례 CPP` 에 있는 내용과 동일 하다.

```
syntax = "proto3";

// Option for Java.
option java_multiple_files = true;
option java_package = "com.hancomsechre.xecuredb.grpc";
option java_outer_classname = "XecuredbProto";

// Option for Ovjective-C.
option objc_class_prefix = "XDB";

package xdbproto;

// The Xecuredb service definition.
service Xecuredb {
  rpc Encrypt (EncRequest) returns (EncReply) {}
  rpc Decrypt (DecRequest) returns (DecReply) {}
  rpc CryptoHash (HashRequest) returns (HashReply) {}
}

// The request message containing the alias and plain text.
message EncRequest {
  string alias = 1;
  string plain = 2;
}

// The response message containing the err code and cipher text.
message EncReply {
  int32 err = 1;
  string msg = 2;
  string cipher = 3;
}

// The request message containing the alias and cipher text.
message DecRequest {
  string alias = 1;
  string cipher = 2;
}

// The response message containing the err code and plain text.
message DecReply {
  int32 err = 1;
```

```
  string msg = 2;
  string plain = 3;
}

// The request message containing the alogorithm id and plain text.
message HashRequest {
  int32 id = 1;
  string plain = 2;
}

// The response message containing the err code and hash string.
message HashReply {
  int32 err = 1;
  string msg = 2;
  string cipher = 3;
}
```

2. protoc 를 이용한 Interface 파일 생성

python 에서 gRPC 를 사용하기 위해서는 protoc 를 이용하여 interface 파일을 생성 하여야 한다. 다음은 interface 파일을 생성 하기 위한 명령 이다. `create_python_src.sh`

```
# python
protoc -I ../proto --python_out=. --plugin=protoc-gen-grpc=`which
grpc_python_plugin` xdb_grpc.proto
protoc -I ../proto --grpc_out=. --plugin=protoc-gen-grpc=`which
grpc_python_plugin` xdb_grpc.proto
```

위 명령을 실행 하면 작업 directory 에 `xdb_grpc_pb2.py`, `xdb_grpc_pb2_grpc.py` 두개의 파일이 생성 된다.

3. Client 코드 작성

python 를 이용한 client 는 python_xdb_client.py 로 작성 하였으며 source code 는 다음과 같다.

```
from __future__ import print_function

import grpc

import xdb_grpc_pb2
import xdb_grpc_pb2_grpc

def run():
    with grpc.insecure_channel('localhost:50052') as channel:
        stub = xdb_grpc_pb2_grpc.XecuredbStub(channel)
        strAlias = 'normal'
        strSource = '1234567890123'
        algo = 6

        resenc =
stub.Encrypt(xdb_grpc_pb2.EncRequest(alias=strAlias,plain=strSource))
```

```
        strEnc = resenc.cipher

        resdec =
stub.Decrypt(xdb_grpc_pb2.DecRequest(alias=strAlias,cipher=strEnc))
        strDec = resdec.plain

        reshash =
stub.CryptoHash(xdb_grpc_pb2.HashRequest(id=algo,plain=strSource))
        strHash = reshash.cipher

    print("Encrypt client received: " + strEnc)
    print("Decrypt client received: " + strDec)
    print("Hash    client received: " + strHash)

if __name__ == '__main__':
    run()
```

`protoc` 에 의해 `xdb_grpc_pb2_grpc.py` 에 `XecuredbStub` stub 코드가 생성 된다. XecuredbStub 는 `xdb_grpc.proto` 에서 정의한 service Xecuredb 를 이용하여 생성 된다. 주의 해서 볼 부분은 stub.Encrypt, stub.Decrypt, stub.Hash 의 parameter로 xdb_grpc_pb2.EncRequest, xdb_grpc_pb2.DecRequest, xdb_grpc_pb2.HashRequest 함수를 사용 하였고 그 함수들의 parameter 로 서버로 전달할 값 들을 넣어서 호출하는 과정 이다. python 에서도 비교적 쉽게 gRPC 를 사용 할 수 있다.

4. python client 프로그램 실행

프로그램의 실행은 다음 명령을 이용한다.

```
python3 python_xdb_client.py
```

`실행 결과`

```
Encrypt client received: AAGo8vet8aHoqsGorhx1CsXL
Decrypt client received: 1234567890123
Hash    client received: vKKOGisl4TfIP+4Oave9Hg9SvVYFg8oHobQvmUTFxQs=
```

5. Remark

위 프로그램은 gRPC examples 에 있는 Helloworld 를 참조 하여 작성 하였다.

- 끝 -