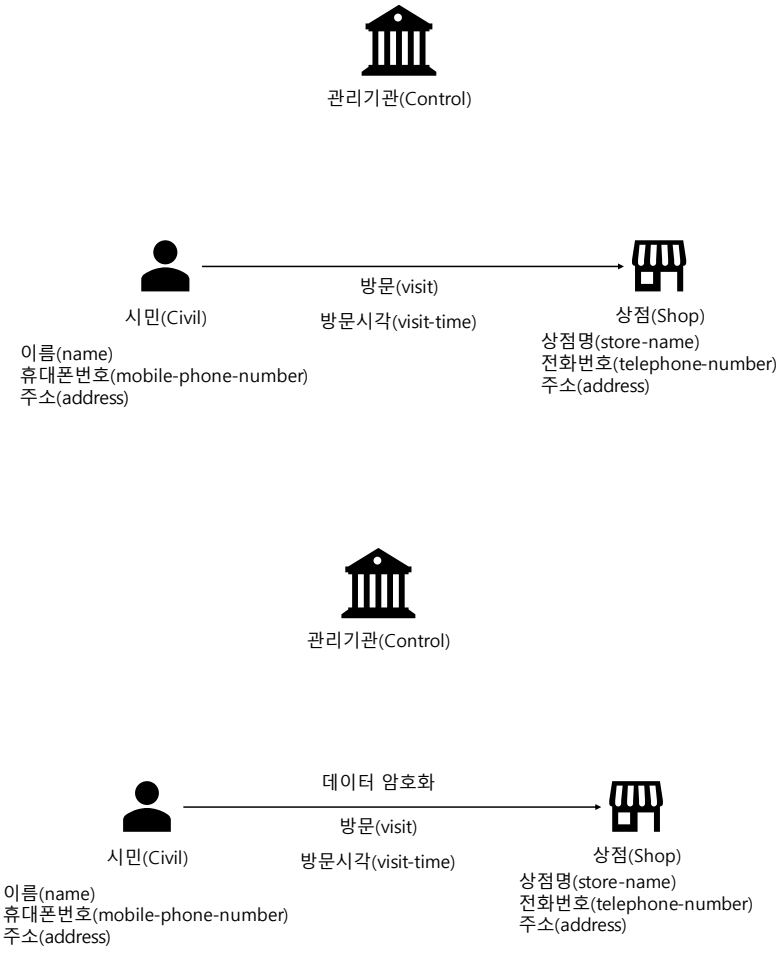
연구 보고서

작성자	김한호	작성일자	2021.04.25.

1. 연구 계획

- Hyperledger fabric asset-transfer-private-data를 참고하여 전자출입명부 개념적 설계
- 공개키 기반 구조를 체인코드 적용방안 구상 후 테스트

2. 논문 연구 진행



공개키 기반구조와 private-data적용한 전자출입명부 프로젝트의 차이는 방문을 남기는 곳을 데이터 암호화를 통해 적용한다. 시민과 상점의 예제 데이터를 설계해보고 예제 데이터를 생각해 봄.

시민(Civil)

일련번호(ID)	이름(name)	휴대폰번호(mobile-number)	주소(address)
1	홍길동	010-4685-9875	경기도 수원시 장안구
2	김철수	019-787-5513	서울특별시 관악구
3	이랑	016-884-7952	대전광역시 중구
...			

상점(shop)

일련번호(ID)	상점명(store-name)	전화번호(telephone-number)	주소(address)
1	버거킹 광교아브뉴프랑점	031-212-0360	경기 수원시 영통구 센트럴타운로 85
2	역전우동0410 아주대점	031-217-0411	경기 수원시 영통구 중부대로 259 예스프라자 107호
3	태화장 본점	031-213-6998	경기 수원시 팔달구 아주로13번길 22
...			

많은 데이터를 넣어서 실제 세상에 적용하는 방법보다는 단순화하여 구현을 쉽게 하기로 계획했다. 시민(Civil)은 일련번호(ID), 이름(name), 휴대폰번호(mobile-number), 주소(address)의 속성(attribute)를 가지고 있음. 상점(shop)은 일련번호(ID), 상점명(store-name), 전화번호(telephone-number), 주소(address)를 정보를 저장하고 있음. 방문이라는 상황이 발생할 때에 저장하는 내용의 차이가 있기 때문에 들어갈 내용을 예측해서 설계를 구상함.

방문(visit)

일련번호(ID)	방문자번호(civil-number)	상점번호(shop-number)	방문시각(visit-time)
1	1	3	2021-04-22 11:30:00
2	2	2	2021-04-24 18:50:24
3	3	1	2021-04-25 19:24:33
...			

방문(visit)

일련번호(ID)	상점번호(shop-number)	방문정보
1	3	!343299fjidRieis88d)(43!!@#ghdjfd
2	2	*783hfuhud(*847372hfjds94uhdsav
3	1	(878fhejr89s89*&(*848302fjdhsjkfds
...		

private-data와 공개키를 사용한 정보를 데이터 저장 차이는 방문자 번호를 가지고 있는 private-data와 방문자번호화 방문시각을 암호화하여 저장한 방문정보의 데이터 저장 차이가 있을 것 같다는 예상이 들었음.

```go
func generateKeyPair(bits int) (*rsa.PrivateKey, *rsa.PublicKey) {
    // This method requires a random number of bits.
    privateKey, err := rsa.GenerateKey(rand.Reader, bits)
    if err != nil {
        fmt.Println("Error: ", err)
    }

    // The public key is part of the PrivateKey struct
    return privateKey, &privateKey.PublicKey
}

func (s *SmartContract) TestGenPubPrivKey(ctx contractapi.TransactionContextInterface) error {
    privateKey, publicKey := generateKeyPair(2048)
    log.Printf("Private Key: %v\n", privateKey)
    log.Printf("Public Key: %v\n", publicKey)
    return nil
}


func (s *SmartContract) EnDecryptTest(ctx contractapi.TransactionContextInterface, pubKeyPath string
, privKeyPath string, message string) error {
    pubKeyFilePath := readKeyPathFromDir(pubKeyPath)
    privKeyFilePath := readKeyPathFromDir(privKeyPath)
    log.Println(pubKeyPath, privKeyPath, message)
    pubKeyPEM := readKeyFromFile(pubKeyFilePath)
    log.Printf("readKeyFrom Public KeyFile")
    pubKeyFile := exportPEMStrToPubKey(pubKeyPEM)
    log.Printf("exportPemStringPubKey")
    cipherText, _ := rsa.EncryptOAEP(sha256.New(), rand.Reader, pubKeyFile, []byte(message), nil)
    log.Printf("exchange message to ciphertext")
    fmt.Printf("Encrypted: %v\n", cipherText)

    privKeyPEM := readKeyFromFile(privKeyFilePath)
    log.Printf("readKeyFrom Private KeyFile")
    privKeyFile := exportPEMStrToPrivKey(privKeyPEM)
    log.Printf("exportPemStringPrivKey")
    decMessage, _ := rsa.DecryptOAEP(sha256.New(), rand.Reader, privKeyFile, cipherText, nil)
    log.Printf("exchange ciphertext to message")
    fmt.Printf("Original: %s\n", string(decMessage))
    return nil
}
```

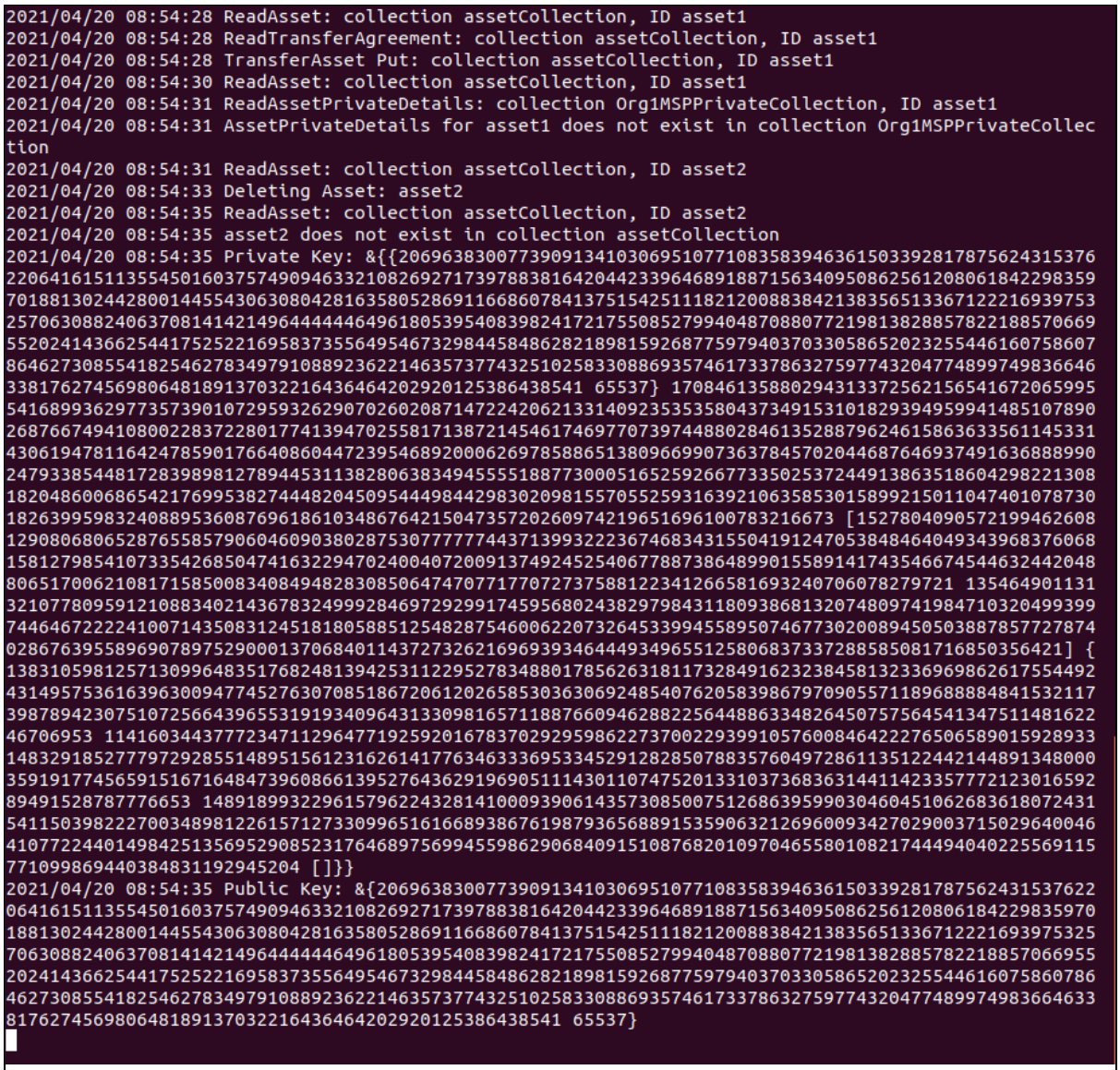
체인코드에 공개키 적용을 위해 테스트하는 EnDecryptTest 함수를 만들어서 asset-transfer-private-data의 application app.js에서 호출하여 테스트하기로 계획함.

```
            statefulTxn.setTransient({
                asset_delete: tmapData
            });
            result = await statefulTxn.submit();

            let data2ForDelete = { assetID: assetID2 };
            try {
                //Non-owner Org2 should not be able to DeleteAsset. Expect an error from DeleteAsset
                console.log('--> Attempt Transaction: as Org2 DeleteAsset ' + assetID2);
                statefulTxn = contractOrg2.createTransaction('DeleteAsset');
                tmapData = Buffer.from(JSON.stringify(data2ForDelete));
                statefulTxn.setTransient({
                    asset_delete: tmapData
                });
                result = await statefulTxn.submit();
                console.log('******** FAILED : expected to return an error');
            } catch (error) {
                console.log(`  Successfully caught the error: \n    ${error}`);
            }
            // Delete Asset2 as Org1
            console.log('--> Submit Transaction: as Org1 DeleteAsset ' + assetID2);
            statefulTxn = contractOrg1.createTransaction('DeleteAsset');
            tmapData = Buffer.from(JSON.stringify(data2ForDelete));
            statefulTxn.setTransient({
                asset_delete: tmapData
            });
            result = await statefulTxn.submit();

            console.log('\n--> Evaluate Transaction: ReadAsset ' + assetID2);
            result = await contractOrg1.evaluateTransaction('ReadAsset', assetID2);
            console.log('  result: ' + prettyJSONString(result.toString()));

            console.log('\n~~~~~~~~~~~~~~~~~ As Org2 Client ~~~~~~~~~~~~~~~~');
            // Org2 can ReadAssetPrivateDetails: Org2 is owner, and private details exist in new own
er's Collection
            console.log('\n--> Evaluate Transaction as Org2: ReadAssetPrivateDetails ' + assetID1 +
' from ' + org2PrivateCollectionName);
            result = await contractOrg2.evaluateTransaction('ReadAssetPrivateDetails', org2PrivateCo
llectionName, assetID1);
            console.log('  result: ' + prettyJSONString(result.toString()));

            await contractOrg1.evaluateTransaction('TestGenPubPrivKey')
            //console.log('\n--> Test CryptoSuite public/private encrypt test');
            /*let Org1PubKey = path.join(Org1PubKeyDir + '/cert.pem');
            *let Org1PrivKKey = path.join(Org1PrivKeyDir + '/
            *console.log('\nOrg1PublicKey:'+Org1PublicKey+'~~~~~~~~~~~~');
            *const fileExists = fs.existsSync(Org1PublicKey);
```

app.js에서 TestGenPubPrivKey함수를 호출하여 실제로 공개키 생성을 할 수 있는지 실행함.

```
h2kim@h2kim-VBox:~/fabric-samples/test-network$ docker ps
CONTAINER ID        IMAGE
                                                                                                                  COMMAND
                CREATED             STATUS              PORTS
 NAMES
f05f9bfd9024        dev-peer0.org1.example.com-private_1.0-75ed6ae2ecb5ea950953d5c91e48dce1556251b3c
5e28afaa960a89bab1d1136-8ab7295b5e6d111973db4c2e4da34e2d82c3889fd85e878b91ca772318e79b78   "chaincod
e -peer.add…"   5 minutes ago       Up 5 minutes
 dev-peer0.org1.example.com-private_1.0-75ed6ae2ecb5ea950953d5c91e48dce1556251b3c5e28afaa960a89bab1d
1136
046b2ef5f598        dev-peer0.org2.example.com-private_1.0-75ed6ae2ecb5ea950953d5c91e48dce1556251b3c
5e28afaa960a89bab1d1136-425abdc9b3a9d5df0064a186089c9de10256f3d9dbc910ec4b73d9e23cbcb12a   "chaincod
e -peer.add…"   5 minutes ago       Up 5 minutes
 dev-peer0.org2.example.com-private_1.0-75ed6ae2ecb5ea950953d5c91e48dce1556251b3c5e28afaa960a89bab1d
1136
0c049fa37f23        hyperledger/fabric-peer:latest
                                                                                                                  "peer nod
e start"        8 minutes ago       Up 8 minutes        0.0.0.0:7051->7051/tcp
 peer0.org1.example.com
88abd5cfb0dd        hyperledger/fabric-peer:latest
                                                                                                                  "peer nod
e start"        8 minutes ago       Up 8 minutes        7051/tcp, 0.0.0.0:9051->9051/tcp
 peer0.org2.example.com
ac27a9ce93e4        couchdb:3.1
                                                                                                                  "tini --
/docker-ent…"   8 minutes ago       Up 8 minutes        4369/tcp, 9100/tcp, 0.0.0.0:7984->5984/tcp
 couchdb1
ad01bd6be81c        couchdb:3.1
                                                                                                                  "tini --
/docker-ent…"   8 minutes ago       Up 8 minutes        4369/tcp, 9100/tcp, 0.0.0.0:5984->5984/tcp
 couchdb0
fcc59a440297        hyperledger/fabric-orderer:latest
                                                                                                                  "orderer"
                8 minutes ago       Up 8 minutes        0.0.0.0:7050->7050/tcp
 orderer.example.com
e5839ac27608        hyperledger/fabric-ca:latest
                                                                                                                  "sh -c 'f
abric-ca-se…"   8 minutes ago       Up 8 minutes        0.0.0.0:7054->7054/tcp
 ca_org1
1caddc561fd0        hyperledger/fabric-ca:latest
                                                                                                                  "sh -c 'f
abric-ca-se…"   8 minutes ago       Up 8 minutes        7054/tcp, 0.0.0.0:9054->9054/tcp
 ca_orderer
66ee336df5ee        hyperledger/fabric-ca:latest
                                                                                                                  "sh -c 'f
abric-ca-se…"   8 minutes ago       Up 8 minutes        7054/tcp, 0.0.0.0:8054->8054/tcp
 ca_org2
h2kim@h2kim-VBox:~/fabric-samples/test-network$
```

체인코드에 올려서 확인함.



docker logs -f <container-number>를 확인을 통해 private-key와 public-key를 출력하는 것을 확인할 수 있음.

```go
func generateKeyPair(bits int) (*rsa.PrivateKey, *rsa.PublicKey) {
    // This method requires a random number of bits.
    privateKey, err := rsa.GenerateKey(rand.Reader, bits)
    if err != nil {
        fmt.Println("Error: ", err)
    }

    // The public key is part of the PrivateKey struct
    return privateKey, &privateKey.PublicKey
}

func ExportRsaPrivateKeyAsPemStr(privkey *rsa.PrivateKey) string {
    privkey_bytes := x509.MarshalPKCS1PrivateKey(privkey)
    privkey_pem := pem.EncodeToMemory(
            &pem.Block{
                    Type:  "RSA PRIVATE KEY",
                    Bytes: privkey_bytes,
            },
    )
    return string(privkey_pem)
}

func ExportRsaPublicKeyAsPemStr(pubkey *rsa.PublicKey) (string, error) {
    pubkey_bytes, err := x509.MarshalPKIXPublicKey(pubkey)
    if err != nil {
            return "", fmt.Errorf("Error: MarshalPKCS1PublicKey")
    }
    pubkey_pem := pem.EncodeToMemory(
            &pem.Block{
                    Type:  "RSA PUBLIC KEY",
                    Bytes: pubkey_bytes,
            },
    )

    return string(pubkey_pem), nil
}

func (s *SmartContract) TestGenPubPrivKey(ctx contractapi.TransactionContextInterface, pkiID string)
 (*PKI, error) {
    privKey, pubKey := generateKeyPair(2048)
    log.Printf("Private Key: %v\n", privKey)
    privateKey := ExportRsaPrivateKeyAsPemStr(privKey)
    log.Printf("Public Key: %v\n", pubKey)
    publicKey, err := ExportRsaPublicKeyAsPemStr(pubKey)
    if err != nil {
        return nil, fmt.Errorf("Error: fun:ExportRsaPublicKeyAsPemStr")
    }
```

TestGenPrivKey 함수는 새로운 키를 만들어내고 키를 통해 암호화를 할 수 있는 상황을 테스트할 수 있도록 구동한다.

```
210 func (s *SmartContract) GetPubPrivKey(ctx contractapi.TransactionContextInterface, privStr string
    , pubStr string) (*rsa.PrivateKey, *rsa.PublicKey, error) {
211     privKey, err := ParseRsaPrivateKeyFromPemStr(privStr)
212     if err != nil {
213         return nil, nil, fmt.Errorf("Error: func:GetPubPrivKey Get privateKey")
214     }
215     pubKey, err := ParseRsaPublicKeyFromPemStr(pubStr)
216     if err != nil {
217         return nil, nil, fmt.Errorf("Error: func:GetPubPrivKey Get publicKey")
218     }
219     return privKey, pubKey, nil
220 }
221
222 func (s *SmartContract) EnDecryptTest(ctx contractapi.TransactionContextInterface, privStr string
    , pubStr string, message string) error {
223     privKey, pubKey, err := s.GetPubPrivKey(ctx, privStr, pubStr)
224     if err != nil {
225         return fmt.Errorf("Error: func:EnDecryptTest")
226     }
227     cipherText, _ := rsa.EncryptOAEP(sha256.New(), rand.Reader, pubKey, []byte(message), nil)
228     log.Printf("exchange message to ciphertext")
229     fmt.Printf("Encrypted: %v\n", cipherText)
230
231     decMessage, _ := rsa.DecryptOAEP(sha256.New(), rand.Reader, privKey, cipherText, nil)
232     log.Printf("exchange ciphertext to message")
233     fmt.Printf("Original: %s\n", string(decMessage))
234     return nil
235 }
```

위에서 만들어진 비밀키와 공개키를 가지고 평문을 암호문으로 만드는 테스트를 하는 EnDecryptTest 함수를 작성함. 위의 함수를 app.js에 적용하여 실제로 암호화하는지 확인하기로 했음.

```
h2kim@h2kim-VBox:~/fabric-samples/asset-transfer-private-data/chaincode-go/chaincode$ docker ps
CONTAINER ID        IMAGE                               COMMAND                  CREATED
STATUS              PORTS                                       NAMES
30943f6797a7        hyperledger/fabric-peer:latest      "peer node start"        3 minutes ago
Up 3 minutes        7051/tcp, 0.0.0.0:9051->9051/tcp            peer0.org2.example.com
2e0e5bdecbaa        hyperledger/fabric-peer:latest      "peer node start"        3 minutes ago
Up 3 minutes        0.0.0.0:7051->7051/tcp                      peer0.org1.example.com
e0a5c1a49bfc        couchdb:3.1                         "tini -- /docker-ent…"   3 minutes ago
Up 3 minutes        4369/tcp, 9100/tcp, 0.0.0.0:5984->5984/tcp  couchdb0
ca5622c49236        hyperledger/fabric-orderer:latest   "orderer"                3 minutes ago
Up 3 minutes        0.0.0.0:7050->7050/tcp                      orderer.example.com
44ba3a770256        couchdb:3.1                         "tini -- /docker-ent…"   3 minutes ago
Up 3 minutes        4369/tcp, 9100/tcp, 0.0.0.0:7984->5984/tcp  couchdb1
1a09f4c888b2        hyperledger/fabric-ca:latest        "sh -c 'fabric-ca-se…"   3 minutes ago
Up 3 minutes        0.0.0.0:7054->7054/tcp                      ca_org1
f24c2ff69b69        hyperledger/fabric-ca:latest        "sh -c 'fabric-ca-se…"   3 minutes ago
Up 3 minutes        7054/tcp, 0.0.0.0:9054->9054/tcp            ca_orderer
9d8f15892043        hyperledger/fabric-ca:latest        "sh -c 'fabric-ca-se…"   3 minutes ago
Up 3 minutes        7054/tcp, 0.0.0.0:8054->8054/tcp            ca_org2
h2kim@h2kim-VBox:~/fabric-samples/asset-transfer-private-data/chaincode-go/chaincode$
```

체인코드가 컨테이너에 올라가 있지 않은 것을 확인함.

```
2021-04-23 05:05:13.713 UTC [committer.txvalidator] Validate -> INFO 071 [mychannel] Validated block
[5] in 0ms
2021-04-23 05:05:13.728 UTC [lifecycle] update -> INFO 072 Updating cached definition for chaincode '
private' on channel 'mychannel'
2021-04-23 05:05:13.736 UTC [lifecycle] update -> INFO 073 Chaincode with package ID 'private_1.0:bbc
4c71504cc2a5164b6f9a30e394f5982d7366058f3e6165e4c889d9612f9e6' now available on channel mychannel for
 chaincode definition private:1.0
2021-04-23 05:05:14.511 UTC [couchdb] createDatabaseIfNotExist -> INFO 074 Created state database myc
hannel_private$$passet$collection
2021-04-23 05:05:14.837 UTC [couchdb] createIndex -> INFO 075 Created CouchDB index [indexOwner] in s
tate database [mychannel_private$$passet$collection] using design document [_design/indexOwnerDoc]
2021-04-23 05:05:14.837 UTC [statecouchdb] ProcessIndexesForChaincodeDeploy -> INFO 076 successfully
submitted index creation request present in the file [META-INF/statedb/couchdb/collections/assetColle
ction/indexes/indexOwner.json] for chaincode [private$$passetCollection] on channel [mychannel]
2021-04-23 05:05:14.837 UTC [cceventmgmt] HandleStateUpdates -> INFO 077 Channel [mychannel]: Handlin
g deploy or update of chaincode [private]
2021-04-23 05:05:15.047 UTC [kvledger] CommitLegacy -> INFO 078 [mychannel] Committed block [5] with
1 transaction(s) in 1334ms (state_validation=1162ms block_and_pvtdata_commit=9ms state_commit=154ms)
commitHash=[ff2d475944fac41f5772e26279f66b9049bda1f1ed6dfee79e46b63bc575abfa]
2021-04-23 05:05:15.060 UTC [comm.grpc.server] 1 -> INFO 079 streaming call completed grpc.service=pr
otos.Deliver grpc.method=DeliverFiltered grpc.request_deadline=2021-04-23T05:05:41.699Z grpc.peer_add
ress=192.168.32.1:59452 error="context finished before block retrieved: context canceled" grpc.code=U
nknown grpc.call_duration=3.361584212s
2021-04-23 05:05:15.980 UTC [lifecycle] Work -> WARN 07a could not launch chaincode 'private_1.0:bbc4
c71504cc2a5164b6f9a30e394f5982d7366058f3e6165e4c889d9612f9e6': chaincode registration failed: contain
er exited with 2
```

로그를 확인해본 결과 체인코드가 등록되지 않은 것을 확인할 수 있었음.

```
/*
func (s *SmartContract) GetPubPrivKey(ctx contractapi.TransactionContextInterface, privStr string
, pubStr string) (*rsa.PrivateKey, *rsa.PublicKey, error) {
    privKey, err := ParseRsaPrivateKeyFromPemStr(privStr)
    if err != nil {
        return nil, nil, fmt.Errorf("Error: func:GetPubPrivKey Get privateKey")
    }
    pubKey, err := ParseRsaPublicKeyFromPemStr(pubStr)
    if err != nil {
        return nil, nil, fmt.Errorf("Error: func:GetPubPrivKey Get publicKey")
    }
    return privKey, pubKey, nil
}

func (s *SmartContract) EnDecryptTest(ctx contractapi.TransactionContextInterface, privStr string
, pubStr string, message string) error {
    privKey, pubKey, err := s.GetPubPrivKey(ctx, privStr, pubStr)
    if err != nil {
        return fmt.Errorf("Error: func:EnDecryptTest")
    }
    cipherText, _ := rsa.EncryptOAEP(sha256.New(), rand.Reader, pubKey, []byte(message), nil)
    log.Printf("exchange message to ciphertext")
    fmt.Printf("Encrypted: %v\n", cipherText)

    decMessage, _ := rsa.DecryptOAEP(sha256.New(), rand.Reader, privKey, cipherText, nil)
    log.Printf("exchange ciphertext to message")
    fmt.Printf("Original: %s\n", string(decMessage))
    return nil
}
*/
```

이번에 새로 작성한 함수를 주석하여 다시 체인코드를 올리는 프로세스를 진행함.

```
h2kim@h2kim-VBox:~/fabric-samples/asset-transfer-private-data/chaincode-go/chaincode$ docker ps
CONTAINER ID        IMAGE
                                                                                          COMMAND
              CREATED             STATUS              PORTS                                         NA
MES
2daad501b36e        dev-peer0.org2.example.com-private_1.0-e32bf5d9c54e711a396450ef0a561e630ca9ecf4f2
bafaad8bee941aad98021f-6d1d3223297456fec24d43974c7d368393e509f64138456db58ab5a776231dfe   "chaincode
-peer.add…"   22 seconds ago      Up 18 seconds                                                   de
v-peer0.org2.example.com-private_1.0-e32bf5d9c54e711a396450ef0a561e630ca9ecf4f2bafaad8bee941aad98021f
9ed87df0a9e1        dev-peer0.org1.example.com-private_1.0-e32bf5d9c54e711a396450ef0a561e630ca9ecf4f2
bafaad8bee941aad98021f-e715551cb6256d504a94383eedf68fa1349cb57d9455e86fcb0d16d4030e1fa3   "chaincode
-peer.add…"   22 seconds ago      Up 19 seconds                                                   de
v-peer0.org1.example.com-private_1.0-e32bf5d9c54e711a396450ef0a561e630ca9ecf4f2bafaad8bee941aad98021f
ad401e5ed7d1        hyperledger/fabric-peer:latest
                                                                                          "peer node
start"        2 minutes ago       Up 2 minutes        0.0.0.0:7051->7051/tcp                        pe
er0.org1.example.com
9a927e44dab4        hyperledger/fabric-peer:latest
                                                                                          "peer node
start"        2 minutes ago       Up 2 minutes        7051/tcp, 0.0.0.0:9051->9051/tcp              pe
er0.org2.example.com
93011b62d76c        hyperledger/fabric-orderer:latest
                                                                                          "orderer"
              2 minutes ago       Up 2 minutes        0.0.0.0:7050->7050/tcp                        or
derer.example.com
bb991dd0ed39        couchdb:3.1
                                                                                          "tini -- /d
ocker-ent…"   2 minutes ago       Up 2 minutes        4369/tcp, 9100/tcp, 0.0.0.0:5984->5984/tcp    co
uchdb0
0cb368b275d5        couchdb:3.1
                                                                                          "tini -- /d
ocker-ent…"   2 minutes ago       Up 2 minutes        4369/tcp, 9100/tcp, 0.0.0.0:7984->5984/tcp    co
uchdb1
84b0f4687524        hyperledger/fabric-ca:latest
                                                                                          "sh -c 'fab
ric-ca-se…"   2 minutes ago       Up 2 minutes        7054/tcp, 0.0.0.0:8054->8054/tcp              ca
_org2
4e0e48a20650        hyperledger/fabric-ca:latest
                                                                                          "sh -c 'fab
ric-ca-se…"   2 minutes ago       Up 2 minutes        7054/tcp, 0.0.0.0:9054->9054/tcp              ca
_orderer
2dd56a624aff        hyperledger/fabric-ca:latest
                                                                                          "sh -c 'fab
ric-ca-se…"   2 minutes ago       Up 2 minutes        0.0.0.0:7054->7054/tcp                        ca
_org1
h2kim@h2kim-VBox:~/fabric-samples/asset-transfer-private-data/chaincode-go/chaincode$
```

제대로 도커에 체인코드를 올라가 있는 것을 확인함. 새로 작성한 코드에 문제가 발생한 것으로 추측됨. 차주에는 독립된 프로젝트로 공개키 암호화한 부분만 구현하여 테스트하는 것으로 계획을 변경.

3. 차주 계획

- 테스트 프로젝트 구현시작
- 공개키 기반 구조 체인코드에 적용

4. 참고 웹페이지

[1] https://ichi.pro/ko/go-eseoui-rsa-amhohwa-gaideu-28694008034141