

# < Hyperledger INDY getting start >

## 1. Hyperledger INDY란?

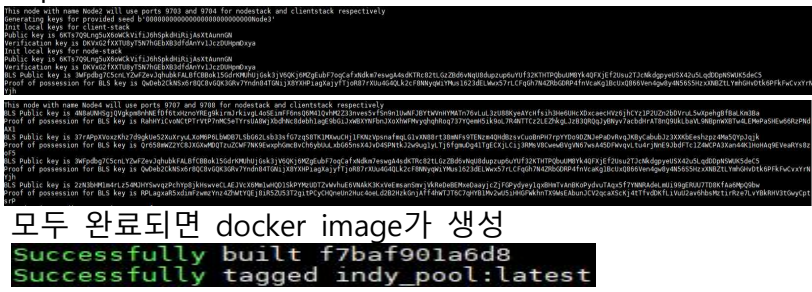

- 리눅스 재단에서 주관하는 블록체인 오픈소스 프로젝트인 Hyperledger의 일부.  
(<https://medium.com/@jw1380/4D/65/88/4C/8D/84/8D/8D/8C/8B/A0/88/4C/A0/80/hyperledger-FA/B0/8C/4C/9A/84/8af3f5>)
- sovrin으로부터 코드를 기증받아 시작된 프로젝트로 블록체인을 통한 DID 생성과 인증 관련  
(<https://hamait.tistory.com/1063>)

## 2. Hyperledger INDY 환경설정


- <https://github.com/hyperledger/indy-sdk>, <https://github.com/hyperledger/indy-sdk/blob/master/cli/README.md>,  
<https://github.com/hyperledger/indy-sdk/blob/master/docs/build-guides/ubuntu-build.md> 참고

<p>① Ubuntu 16.04에서 INDY SDK 및 cli 설치</p>	<pre>\$ sudo apt-key adv --keyserver keyserver.ubuntu.com --recv-keys CE7709D068DB5E88 \$ sudo apt-key list kismi@testpeer1:~/indy\$ sudo apt-key list /etc/apt/trusted.gpg ----- pub 1024D/437D05B5 2004-09-12 uid Ubuntu Archive Automatic Signing Key &lt;ftpmaster@ubuntu.com&gt; sub 2048g/79164387 2004-09-12 pub 4096R/C0B21F32 2012-05-11 uid Ubuntu Archive Automatic Signing Key (2012) &lt;ftpmaster@ubuntu.com&gt; pub 4096R/EFE21092 2012-05-11 uid Ubuntu CD Image Automatic Signing Key (2012) &lt;cdimage@ubuntu.com&gt; pub 1024D/FBB75451 2004-12-30 uid Ubuntu CD Image Automatic Signing Key &lt;cdimage@ubuntu.com&gt; pub 4096R/68576280 2014-06-13 uid NodeSource &lt;gpg@nodesource.com&gt; sub 4096R/AA01DA2C 2014-06-13 pub 4096R/0EBFCD88 2017-02-22 uid Docker Release (CE deb) &lt;docker@docker.com&gt; sub 4096R/F273FCD8 2017-02-22 pub 4096R/68DB5E88 2017-06-29 [expires: 2032-06-25] uid Sovrin-Repo-Master (Master key for repo.sovrin.org) &lt;repo@sovrin.org&gt; sub 4096R/C8C97811 2019-07-01 [expires: 2021-06-30] sub 4096R/B90DD381 2017-06-29 [expires: 2032-06-25]</pre> <pre>\$ sudo add-apt-repository "deb https://repo.sovrin.org/sdk/deb xenial {release channel}" * ubuntu 18.04일 경우 xenial → bionic로 대체 * {release channel}은 master, rc, stable 중 선택 \$ sudo apt-get update \$ sudo apt-get install -y {library} * {library}는 libindy, libnullpay, libvcx, indy-cli 중 선택 \$ sudo apt-get install -y indy-cli * indy-cli 명령어 입력 후 cli test kismi@testpeer1:~/indy\$ indy-cli indy&gt; indy&gt;</pre>
<p>② native libraries와 utilities 설치</p>	<pre>\$ sudo apt-get update \$ sudo apt-get install -y \   build-essential \   pkg-config \   cmake \   libssl-dev \   libsqlite3-dev \   libzmq3-dev \   libncursesw5-dev</pre>

## 2. Hyperledger INDY 환경설정(이어서)

<p>③ libsodium 설치하기</p> <ul style="list-style-type: none"> <li>- 암호화 라이브러리</li> <li>- version은 반드시 1.0.14</li> </ul>	<pre>\$ cd /tmp \$ curl https://download.sodium.org/downloads/release/ob/unsupported/libsodium-1.0.14.tar.gz   tar -xz \$ sudo cd /tmp/libsodium-1.0.14 &amp;&amp; ₩ ./configure --disable-shared &amp;&amp; ₩ make &amp;&amp; ₩ make install &amp;&amp; ₩ rm -rf /tmp/libsodium-1.0.14</pre>
<p>④ libindy build하기</p>	<pre>\$ git clone https://github.com/hyperledger/indy-sdk.git \$ cd ./indy-sdk/libindy \$ sudo apt-get install cargo \$ cargo build → 오래 걸림 \$ cd ..</pre>
<p>⑤ docker로 노드풀 시작하기</p> <ul style="list-style-type: none"> <li>- docker build 명령어를 통해 원하는 image 생성</li> <li>- 가상 INDY 노드 네트워크를 시작해 sdk 테스트</li> </ul>	<pre>\$ cd ./indy-sdk/ci \$ docker build -f indy-pool.dockerfile -t indy_pool . → indy-pool.dockerfile에 따라 indy_pool이라는 nametag를 가지는 docker image를 build하러 → supervisor와 node1 ~ node4 까지 생성</pre>  <pre>\$ docker run --name "indy_pool" -itd -p 9701-9708:9701-9708 indy_pool</pre> 
<p>⑥ Run test</p>	<pre>\$ cd ./indy-sdk/libindy \$ RUST_TEST_THREADS=1 cargo test → 오래 걸림</pre>
<p>참고사항</p>	<p>* docker build 관련 설명  <a href="https://www.slideshare.net/pyrasis/docker-docker-38286477">https://www.slideshare.net/pyrasis/docker-docker-38286477</a> 참고)  <a href="https://sqlmvp.tistory.com/1304">https://sqlmvp.tistory.com/1304</a> 참고)</p> <ul style="list-style-type: none"> <li>• FROM: 어떤 이미지를 기반으로 할지 설정</li> <li>• MAINTAINER: 이미지 작성자 정보</li> <li>• RUN: 이미지에서 스크립트나 명령 실행</li> <li>• CMD: 컨테이너가 시작되었을 때 스크립트나 명령 실행</li> <li>• ENTRYPOINT: 컨테이너가 시작되었을 때 스크립트나 명령 실행(docker run에서 처리 방식이 다름)</li> <li>• EXPOSE: 호스트와 연결할 포트 번호 설정</li> <li>• ENV: 환경 변수 설정</li> <li>• ADD, COPY: 이미지에 파일 추가</li> <li>• VOLUME: 데이터를 호스트에 저장하도록 설정</li> <li>• USER: 명령을 실행할 사용자 계정 설정</li> <li>• WORKDIR: 명령을 실행할 디렉터리 설정</li> <li>• ONBUILD: FROM으로 이미지가 사용될 때 실행할 명령 설정</li> </ul>

### 3. getting start

① docker image 생성하기	<pre>\$ cd ./indy-sdk/docs/getting-started</pre> <pre>\$ docker build -f getting-started.dockerfile -t getting-started .</pre> <p>→ getting-started.dockerfile에 따라 getting-started라는 nametag를 가지는 docker image를 build하라</p>																					
② 생성한 image를 통해 docker container 생성하기	<pre>\$ docker-compose up</pre> <p>→ network가 생성되며 2개의 docker container 실행</p> <pre>kismi@testpeer1:~/indy/indy-sdk/docs/getting-started\$ docker ps</pre> <table><thead><tr><th>CONTAINER ID</th><th>IMAGE</th><th>COMMAND</th><th>CREATED</th><th>STATUS</th><th>PORTS</th><th>NAMES</th></tr></thead><tbody><tr><td>380b541e5fc1</td><td>getting-started</td><td>"jupyter notebook --"</td><td>12 seconds ago</td><td>Up 8 seconds</td><td>0.0.0.0:8888-&gt;8888/tcp</td><td>getting_started</td></tr><tr><td>60dc1057359d</td><td>indy.pool</td><td>"/usr/bin/supervisord"</td><td>16 seconds ago</td><td>Up 11 seconds</td><td>0.0.0.0:9701-9708-&gt;9701-9708/tcp</td><td>indy_pool</td></tr></tbody></table>	CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS	PORTS	NAMES	380b541e5fc1	getting-started	"jupyter notebook --"	12 seconds ago	Up 8 seconds	0.0.0.0:8888->8888/tcp	getting_started	60dc1057359d	indy.pool	"/usr/bin/supervisord"	16 seconds ago	Up 11 seconds	0.0.0.0:9701-9708->9701-9708/tcp	indy_pool
CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS	PORTS	NAMES																
380b541e5fc1	getting-started	"jupyter notebook --"	12 seconds ago	Up 8 seconds	0.0.0.0:8888->8888/tcp	getting_started																
60dc1057359d	indy.pool	"/usr/bin/supervisord"	16 seconds ago	Up 11 seconds	0.0.0.0:9701-9708->9701-9708/tcp	indy_pool																
③ jupyter notebook 실행하기	<pre>\$ docker logs -f getting_started</pre> <pre>kismi@testpeer1:~/indy/indy-sdk/docs/getting-started\$ docker logs -f getting_started</pre> <pre>[I 06:04:19.423 NotebookApp] Writing notebook server cookie secret to /home/indy/.local/share/jupyter/runtime/notebook_cookie_secret</pre> <pre>[I 06:04:19.523 NotebookApp] Serving notebooks from local directory: /home/indy</pre> <pre>[I 06:04:19.523 NotebookApp] The Jupyter Notebook is running at:</pre> <pre>[I 06:04:19.523 NotebookApp] http://380b541e5fc1:8888/?token=1e21b702db6df95e7e23c7f478ac00e6904b04ac116ade61</pre> <pre>[I 06:04:19.524 NotebookApp] or http://127.0.0.1:8888/?token=1e21b702db6df95e7e23c7f478ac00e6904b04ac116ade61</pre> <pre>[I 06:04:19.524 NotebookApp] Use Control-C to stop this server and shut down all kernels (twice to skip confirmation).</pre> <pre>[W 06:04:19.531 NotebookApp] No web browser found: could not locate runnable browser.</pre> <pre>[C 06:04:19.532 NotebookApp]</pre> <p>To access the notebook, open this file in a browser: file:///home/indy/.local/share/jupyter/runtime/nbserver-1-open.html Or copy and paste one of these URLs: <a href="http://380b541e5fc1:8888/?token=1e21b702db6df95e7e23c7f478ac00e6904b04ac116ade61">http://380b541e5fc1:8888/?token=1e21b702db6df95e7e23c7f478ac00e6904b04ac116ade61</a> or <a href="http://127.0.0.1:8888/?token=1e21b702db6df95e7e23c7f478ac00e6904b04ac116ade61">http://127.0.0.1:8888/?token=1e21b702db6df95e7e23c7f478ac00e6904b04ac116ade61</a></p> <p><a href="http://127.0.0.1:8888/?token=1e21b702db6df95e7e23c7f478ac00e6904b04ac116ade61">http://127.0.0.1:8888/?token=1e21b702db6df95e7e23c7f478ac00e6904b04ac116ade61</a> → ip만 해당 클라우드 값으로 변경 후 접속</p> 																					
④ test	하기도 전에 자꾸 kernel error 발생..!																					
트러블슈팅	<p>jupyter notebook kernel error = python version문제로 발생</p> <pre>Traceback (most recent call last):   File "/usr/lib/python3.5/runpy.py", line 174, in _run_module_as_main     mod_name, mod_spec, code = _get_module_details(mod_name, _Error)   File "/usr/lib/python3.5/runpy.py", line 133, in _get_module_details     return _get_module_details(pkg_main_name, error)   File "/usr/lib/python3.5/runpy.py", line 109, in _get_module_details     import _ (pkg_name)   File "/usr/local/lib/python3.5/dist-packages/ipykernel/_init_.py", line 2, in &lt;module&gt;     from .connect import *   File "/usr/local/lib/python3.5/dist-packages/ipykernel/connect.py", line 13, in &lt;module&gt;     from IPython.core.profiledir import ProfileDir   File "/usr/local/lib/python3.5/dist-packages/IPython/_init_.py", line 42, in &lt;module&gt;     """ ImportError: IPython 7.10+ supports Python 3.6 and above. When using Python 2.7, please install IPython 5.x LTS Long Term Support version. Python 3.3 and 3.4 were supported up to IPython 6.x. Python 3.5 was supported with IPython 7.0 to 7.9.  See IPython 'README.rst' file for more information: <a href="https://github.com/ipython/ipython/blob/master/README.rst">https://github.com/ipython/ipython/blob/master/README.rst</a></pre> <p>→ ??</p>																					

#### 4. INDY-SDK Node.js test

- ① write did and query verkey  
- 가장 최초에 DID를 생성하고 이를 바탕으로 query하는 과정

<https://github.com/hyperledger/indy-sdk/blob/master/docs/how-tos/write-did-and-query-verkey/README.md>

INDY pool 생성 → pool open → wallet 생성 → wallet open  
→ 'steward'의 DID 및 verkey 발급 → 'trust anchor'의 DID 및 verkey 발급 → 원장에 기록(현재 pool의 해당 wallet에 steward가 지정한 trustAnchor 등록) → 'client'의 DID 및 verkey 발급 → client가 trustAnchor에 인증 신청(request) → 인증 신청(request) 사실을 원장에 기록 → 'trust anchor'의 verkey와 request에 대한 response 값이 일치하는지 확인(일치-12번) → wallet, pool close → wallet, pool 삭제

```
kismi@testpeer1:~/indy/indy-sdk/docs/how-tos/write-did-and-query-verkey/nodejs$ node write_did.js
Set protocol version 2 to work with Indy Node 1.4
1. Creates a new local pool ledger configuration that is used later when connecting to ledger.
2. Open pool ledger and get handle from libindy
3. Creating new secure wallet
4. Open wallet and get handle from libindy
5. Generating and storing steward DID and verkey
Steward DID: Th7MpTaRZVRYPiabds81Y
Steward Verkey: FymoFw55GeQH7SRFa37dkx1d2dZ3zUF8ckg7wml7ofN4
6. Generating and storing trust anchor DID and verkey
Trust anchor DID: AtxgCDQVhC1CbP22FDQ1wN
Trust anchor Verkey: 6Pq1H5i71ktArgxeCebqGnRTrd2WHGg1ptMqKLvtwU8F
7. Building NYM request to add Trust Anchor to the ledger
8. Sending NYM request to the ledger
9. Generating and storing DID and verkey representing a Client that wants to obtain Trust Anchor Verkey
Client DID: QdR4PrtsfctA7Id4tLd2Bm
Client Verkey: DsvWntSWq8eNjN9G7A943C6SWARRKQ2r2v3Z4hEBBh8C
10. Building the GET_NYM request to query trust anchor verkey
11. Sending the Get NYM request to the ledger
12. Comparing Trust Anchor verkey as written by Steward and as retrieved in GET_NYM response submitted by client
Written by Steward: 6Pq1H5i71ktArgxeCebqGnRTrd2WHGg1ptMqKLvtwU8F
Queried from ledger: 6Pq1H5i71ktArgxeCebqGnRTrd2WHGg1ptMqKLvtwU8F
Matching: true
13. Closing wallet and pool
14. Deleting created wallet
15. Deleting pool ledger config
```

- ② rotate a key  
- 원장에 기록된 DID의 verkey를 변경하는 방법

<https://github.com/hyperledger/indy-sdk/blob/master/docs/how-tos/rotate-key/README.md>

1~8까지 ①과 동일 → 'trust anchor'의 새로운 verkey 생성 → 'trust anchor'가 'trust anchor'에게 verkey가 바뀐 사실을 신고(request) → 원장에 기록 후 wallet에 기록된 'trust anchor'의 verkey 변경 → 'trust anchor'의 DID를 통해 wallet에 기록된 verkey 조회 → 'trust anchor'가 'trust anchor'에게 request 날린 내용을 조회 → 원장에 기록된 값, wallet의 값, 초기값을 비교해 일치하는지 확인(원장에 기록된 값==wallet의 값!=초기값-16번) → wallet, pool close → wallet, pool 삭제

```
kismi@testpeer1:~/indy/indy-sdk/docs/how-tos/rotate-key/nodejs$ node rotate_key.js
Set protocol version 2 to work with Indy Node 1.4
1. Creates a new local pool ledger configuration that is used later when connecting to ledger.
2. Open pool ledger and get handle from libindy
3. Creating new secure wallet with the given unique name
4. Open wallet and get handle from libindy to use in methods that require wallet access
5. Generating and storing steward DID and verkey
Steward DID: Th7MpTaRZVRYPiabds81Y
Steward Verkey: FymoFw55GeQH7SRFa37dkx1d2dZ3zUF8ckg7wml7ofN4
6. Generating and storing trust anchor DID and verkey
Trust Anchor DID: 5bMedoZepqSWEFRtmW4CQQ
Trust Anchor Verkey: 3WApAawmmFTbkaEUZFsRq55afJtg1diyzZR3fLbvbxS
7. Building NYM request to add Trust Anchor to the ledger
8. Sending NYM request to the ledger
9. Generating new verkey of trust anchor in wallet
New Trust Anchor Verkey: 5Q1VoyhXyz81kqDgSpYLpnuL6Ng5VFvLLk4uBayeecoD
10. Building NYM request to update new verkey to ledger
11. Sending NYM request to the ledger
12. Apply new verkey in wallet
13. Reading new verkey from wallet
Trust Anchor Verkey in wallet: 5Q1VoyhXyz81kqDgSpYLpnuL6Ng5VFvLLk4uBayeecoD
14. Building GET_NYM request to get Trust Anchor verkey
15. Sending GET_NYM request to ledger
16. Comparing Trust Anchor verkeys: written by Steward (original), current in wallet and current from ledger
Written by Steward: 3WApAawmmFTbkaEUZFsRq55afJtg1diyzZR3fLbvbxS
Trust Anchor Verkey in wallet: 5Q1VoyhXyz81kqDgSpYLpnuL6Ng5VFvLLk4uBayeecoD
Trust Anchor Verkey from ledger: 5Q1VoyhXyz81kqDgSpYLpnuL6Ng5VFvLLk4uBayeecoD
Matching: true
17. Closing wallet and pool
18. Deleting created wallet
19. Deleting pool ledger config
```



#### 4. INDY-SDK Node.js test(이어서)

<https://github.com/hyperledger/indy-sdk/blob/master/docs/how-to-os/save-schema-and-cred-def/README.md>

Nodejs를 제공하지 않으므로 아래 모듈 설치 후 테스트 진행

```
$ python3 -m pip install python3-indy asyncio
```

1~8까지 ①과 동일 → 'steward'가 credential schema 생성 → schema request 생성 후 원장에 기록 → 'trust anchor'(issuer)가 credential definition 생성 및 저장 → wallet, pool close → wallet, pool 삭제

```

1. Run the following command to save the schema and create the ledger:
$ cp /usr/share/ledger/docs/how-to/save-schema-and-cred-def/python python3 save_schema_and_cred_def.py

2. Opening a new local pool ledger configuration that will be used later when connecting to Ledger.

3. Open pool ledger and get the handle from libindy

4. Creating new secure wallet with the given unique name

5. Open wallet and get handle from libindy to use in methods that require wallet access

6. Generating and storing steward DID and verkey

Steward DID: 7b79tATzCVR9yPabds3Y
Steward Verkey: 4mFmK5S9dGz7Q9fATk1d2d2tUzKcK7zm7JfH4

7. Generating and storing trust anchor DID and verkey

Trust anchor DID: 7uYef18dUq3kz8tAbgpn
Trust anchor Verkey: 4mFQ8C4E4u3z7Q9fAHKAgznd7Sggt5oTXDfQm6d

8. Building WNM request to add Trust Anchor to the ledger

WNM transaction request:
{"identifier": "7b79tATzCVR9yPabds3Y",
 "data": {
   "type": "addTrustAnchor",
   "role": "role",
   "type": "type",
   "verkey": "4mFQ8C4E4u3z7Q9fAHKAgznd7Sggt5oTXDfQm6d"
 },
 "protocolVersion": 2,
 "reqId": 158649613038827783}

9. Sending WNM request to the ledger

```

### ③ Save Schema and Credential Definition

- 원장에 schema와 credential definition을 저장하는 방법
- schema는 name, version, credential에 표시 될 attributes를 포함하는 JSON 문서
- credention definition은 schema를 참고해 issuer, 서명유형(CL), tag, 해지방법 등을 포함

[illegible][illegible]

\* ③ python version 관련 예러  
<https://stackoverflow.com/questions/59842600/importerror-cannot-import-name-sourcetranslation-from-pip-internal-distrib>

```
$ cd /usr/bin
$ sudo wget https://bootstrap.pypa.io/get-pip.py -O ./get-pip.py
$ python3 get-pip.py
$ cd indy/indy-sdk/docs/how-tos/save-schema-and-cred-def/python
$ /usr/bin/python3 -m pip install indy
```

```

Kissmig@tepeeri:~/indy-sdk/docs/how-to/save-schema-and-cred-def/python3$ ./usr/bin/python3 -m pip install indy
Defaulting to user installation because normal site-packages is not writeable
Collecting indy
  Using cached indy-0.1.1.tar.gz (2.7 kB)
Building wheels for collected packages: indy
  Building wheel for indy (setup.py) ... done
Created wheel for indy: filename=indy-0.1.1-py3-none-any.whl size=3523 sha256=b1e25630b6b273595c50e39741ef46f12f20b4bd346d3f26b98d27a
Stored in directory: /home/kissmig/.cache/pip/wheels/85/b5/4d/6b8dbd39a6e25dbf68e26973d6e125f6ae0541c179f2bf2
Successfully built indy
Installing collected packages: indy

```

트러블슈팅

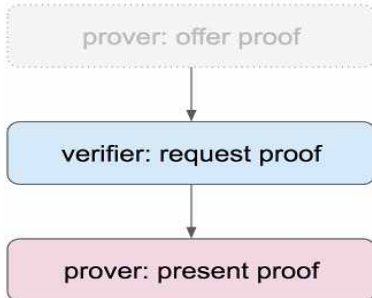




#### 4. INDY-SDK Node.js test(이어서)

##### ⑤ Negotiate Proof

- credential 진위여부 증명 방법



증명서의 진위여부 확인은 보통 verifier의 요청으로 시작되지만, 필요에 따라 prover(당사자, Alice)의 요청에 의해 시작 될 수 있음(생략가능)  
proof request는 어떤 종류의 증명이 신뢰당사자를 만족시키는지 (ex - 00자격증, 성별, 몇 세 이상 등) 설명하는 JSON file형태

proof request를 받은 prover (자격증명 보유자, Alice)는 wallet을 검색해 요청을 충족하는 자격증명 찾을  
대규모의 경우 모든 wallet 검색은 비효율적이기 때문에 새로운 메커니즘 개발을 진행중(indy-sdk)

<https://github.com/hyperledger/indy-sdk/blob/master/docs/how-to-os/negotiate-proof/README.md>

1~8까지 ④와 동일 → Holder(Alice)가 proof request를 받음 (특정 조건을 만족하는지 묻는 JSON file / 예제에서는 나이가 18 세 이상인지) → Holder(자격보유자, Alice)가 request에 대한 증명 생성(wallet에서 검색) → 생성된 증명(proof)을 제시(Holder가 prover(Alice)가 됨) → 검증자(verifier)가 증명 검증 → issuer와 prover의 wallet close → issuer와 prover의 wallet 및 삭제

```

18. Prover gets Credentials for Proof Request
Proof Request:
{'name': 'proof_req_1',
 'nonce': '123432421212',
 'requested_attributes': {'attr1_referent': {'name': 'name',
                                             'restrictions': {'issuer_did': 'UifLzdVpWRlrTrUuAwRH3k',
                                                             'schema_id': 'Th7MpTaRZVRynPiabds81Y:2:gvt:1.0'}}}},
 'requested_predicates': {'predicate1_referent': {'name': 'age',
                                                  'p_type': '>=',
                                                  'p_value': 18,
                                                  'restrictions': {'issuer_did': 'UifLzdVpWRlrTrUuAwRH3k'}}}},
 'version': '0.1'}

19. Prover gets Credentials for attr1_referent and predicate1_referent
Prover credential for attr1_referent:
{'attrs': {'age': '28', 'height': '175', 'name': 'Alex', 'sex': 'male'},
 'cred_def_id': 'UifLzdVpWRlrTrUuAwRH3k:3:CL:Th7MpTaRZVRynPiabds81Y:2:gvt:1.0:TAG1',
 'cred_rev_id': None,
 'referent': 'f216ba65-42b3-4856-8ef4-40cd9276bcb6',
 'rev_reg_id': None,
 'schema_id': 'Th7MpTaRZVRynPiabds81Y:2:gvt:1.0'}
Prover credential for predicate1_referent:
{'attrs': {'age': '28', 'height': '175', 'name': 'Alex', 'sex': 'male'},
 'cred_def_id': 'UifLzdVpWRlrTrUuAwRH3k:3:CL:Th7MpTaRZVRynPiabds81Y:2:gvt:1.0:TAG1',
 'cred_rev_id': None,
 'referent': 'f216ba65-42b3-4856-8ef4-40cd9276bcb6',
 'rev_reg_id': None,
 'schema_id': 'Th7MpTaRZVRynPiabds81Y:2:gvt:1.0'}

20. Prover creates Proof for Proof Request
Requested Credentials for Proving:
{'requested_attributes': {'attr1_referent': {'cred_id': 'f216ba65-42b3-4856-8ef4-40cd9276bcb6',
                                             'revealed': True}},
 'requested_predicates': {'predicate1_referent': {'cred_id': 'f216ba65-42b3-4856-8ef4-40cd9276bcb6'}},
 'self_attested_attributes': {}}

21. Verifier is verifying proof from Prover

22. Closing both wallet_handles and pool

23. Deleting created wallet_handles

24. Deleting pool ledger config
  
```

Schema 요청에 대해 원장에 기록할 때 client의 req가 검증되지 않았다는 AnoncredsMasterSecretDuplicateNameError() 발생

```

10. Sending the SCHEMA request to the ledger
Schema response:
{'identifier': 'Th7MpTaRZVRynPiabds81Y',
 'op': 'REJECT',
 'reason': "client request invalid: UnauthorizedClientRequest('The action is "
           "forbidden',)",
 'reqId': '1580450238603694718'}
  
```

→ <http://pydoc.net/python3-indy/1.3.0/indy.error/> 를 찾아보면,  
"Another master-secret with the specified name already exists" 이미 master-secret이 존재해서 발생하는 오류

→ 해결방법1 = secret\_name 변경

```

# 13.
print_log('\n13. Prover is creating Link Secret\n')
prover_link_secret_name = 'link_secret2'
try:
    link_secret_id = await anoncreds.prover_create_master_secret(prover_wallet_handle,
                                                                prover_link_secret_name)
  
```

해결방법2 = 예외처리를 통해 기존에 존재하던 secret으로 연결  
([https://ais-doudagent.pythonreadthedocs.io/en/latest/modules/ais\\_doudagent/wallet/no/html](https://ais-doudagent.pythonreadthedocs.io/en/latest/modules/ais_doudagent/wallet/no/html))

```

# 13.
print_log('\n13. Prover is creating Link Secret\n')
prover_link_secret_name = 'link_secret'
try:
    link_secret_id = await anoncreds.prover_create_master_secret(prover_wallet_handle,
                                                                prover_link_secret_name)
except IndyError as ex:
    if error.error_code == ErrorCode.AnoncredsMasterSecretDuplicateNameError:
        pprint.pprint("Master secret already exists")
        prover_wallet_handle._master_secret_id = prover_wallet_handle.name
    else:
        raise
  
```

트러블슈팅

#### 4. INDY-SDK Node.js test(이어서)

##### ⑥ Send Secure Message

- indy-sdk의 agent간 통신을 통해 보안 메시지를 주고받는 과정을 보여주는 예제
- varkey를 통해 서로를 식별

<https://github.com/hyperledger/indy-sdk/blob/master/docs/how-to/send-secure-msg/README.md>

두 명의 did를 생성(좌측 - Alice / 우측 - Bob)

→ 서로의 창에 발급받은 DID와 verkey를 붙여넣기(서로 등록하는 과정)

→ 우측-Bob 창에 'prep Hello, world' 입력 ( { 명령어 } { 전달 할 데이터 } )  
데이터의 암호화된 값 출력 및 Alice에게 전송

→ 좌측-Alice 창에 'read' 입력

→ Bob의 verkey와 데이터 출력

```
test@testpc1:~/indy-sdk/docs/how-to/send-secure-msg/python$ python3 send_secure_msg.py
who are you? Alice
wallet = 3
my did and verkey = EtwMfjPKStwvREBBuc3e_BaWtGdHvK8EhU8BdgrUp25UTrP7b66Saj8T17
Enter party's DID and verkey: UgenHwKteidIMe5d0d8_G68Y3H8Qsq4UeFgKCDXaBdpcQWQj2T3x4f0xaff
> read
UgenHwKteidIMe5d0d8_G68Y3H8Qsq4UeFgKCDXaBdpcQWQj2T3x4f0xaff, b'Hello, world'
>
```

```
test@testpc1:~/indy-sdk/docs/how-to/send-secure-msg/python$ python3 send_secure_msg.py
who are you? Bob
wallet = 3
my did and verkey = UgenHwKteidIMe5d0d8_G68Y3H8Qsq4UeFgKCDXaBdpcQWQj2T3x4f0xaff
Enter party's DID and verkey: EtwMfjPKStwvREBBuc3e_BaWtGdHvK8EhU8BdgrUp25UTrP7b66Saj8T17
> prep Hello, world
UgenHwKteidIMe5d0d8_G68Y3H8Qsq4UeFgKCDXaBdpcQWQj2T3x4f0xaff, b'UgenHwKteidIMe5d0d8_G68Y3H8Qsq4UeFgKCDXaBdpcQWQj2T3x4f0xaff:{"msg": "Hello, world"}'
> read
EtwMfjPKStwvREBBuc3e_BaWtGdHvK8EhU8BdgrUp25UTrP7b66Saj8T17, b'Hello, world'
>
```