# PASS인증서 서비스 중계서버 연동가이드

**- 이용기관 / 대행사 용 -**

**2023.03**

ATON
LIFE INNOVATOR GROUP

| 버전 | 설명 | 작성자 | 작성일자 |
|---|---|---|---|
| | 생략 | | |
| 1.2 | **이용기관(제휴사/대행사) PASS인증서 연동가이드v1.2** | DevOps1팀 | 2019.06.07 |
| 1.3 | **보안관련 적용사항 추가** | DevOps1팀 | 2019.07.20 |
| 1.4 | **생년월일 필수에서 옵션 처리 추가** | DevOps1팀 | 2020.01.09 |
| 1.5 | **내용 현행화** | DevOps1팀 | 2020.12.02 |
| 1.6 | **App to App 연동 방법추가** | DevOps1팀 | 2021.08.02 |
| 1.7 | **공개키로 암호화된 CI 복호화 방법 추가** | DevOps1팀 | 2021.08.26 |
| 1.8 | **PASS인증서서비스 중계서버 명칭 등 변경** | DevOps1팀 | 2021.12.24 |
| 2.0 | **PASS인증서 연동가이드(App2App연동 포함) 내용 추가** | DevOps1팀 | 2022.04.13 |
| 2.1 | **연동가이드 및 규격서 통합** | DevOps1팀 | 2022.06.19 |
| 2.2 | **테스트 사용자 등록방법 추가** | DevOps1팀 | 2022.06.24 |
| 2.3 | **- 전자서명 서비스(S1001,S1002,S1003,S2001) 성공시 사용자연계정보(CI) 전송을(옵션->필수)로 변경**<br>**- 전자서명 요청 응답 시 telcoTxId 옵션값으로 변경** | DevOps1팀 | 2022.11.03 |
| 2.4 | **- 이용기관 공개키 및 개인키 관련 문구 변경** | DevOps1팀 | 2023.01.09 |
| 2.5 | **- 에러코드 및 요청/결과조회 전문 현행화** | DevOps1팀 | 2023.03.09 |

# 목차

# 1. 개요

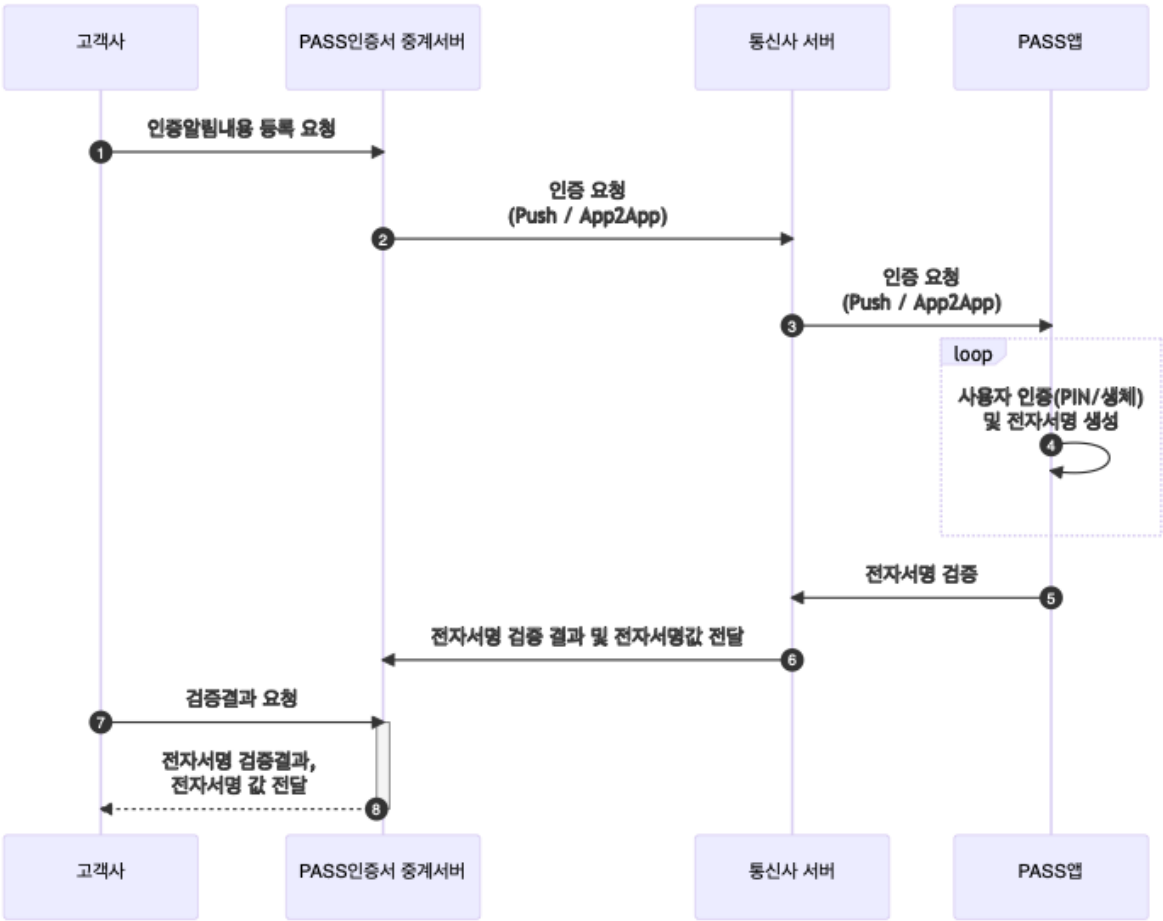
## 1.1 목적

본 문서는 통신사 PASS인증앱을 통해 제공되는 PASS인증서 서비스를 연동 하고자 하는 이용기관(제휴사/대행사 등, 이후 '이용기관')이 보다 용이하게 연동을 할 수 있도록 하기 위해 제공되는 연동가이드 입니다.

## 1.2 범위

본 문서는 이용기관 서버에서 PASS인증서 서비스 중계서버에서 제공하는 API 규격에 따라 연동을 하기 위한 절차에 대한 가이드로서 PASS인증서 서비스 중계서버와 연동 개발을 해야 하는 이용기관 서버 애플리케이션 개발 담당자에게 연동에 대한 이해를 높이고자 제공 됩니다.

## 1.3 PASS인증서 프로세스



# 2. 사전 준비 사항

## 2.1 서비스 도메인

**PASS인증서 전자서명을 위해서는 TLS 1.2 이상 적용이 필수입니다.**

도메인은 스테이징, 상용에 따라서 아래에 맞게 설정이 필요합니다.

| 서버 | 도메인 | IP | PORT |
|---|---|---|---|
| 스테이징 | https://api-stg.passauth.co.kr | 443 | 유동 IP |
| 상용 | https://api.passauth.co.kr | 443 | 유동 IP |

=> 위 도메인의 IP는 주기적으로 변경되며 고정 IP로 연동이 필요할 경우 아래 도메인으로 연동

| 서버 | 도메인 | IP | PORT |
|---|---|---|---|
| 스테이지 | https://pubstg-api.passauth.co.kr | 3.36.4.194 | 443 |
| 상용 | https://pub-api.passauth.co.kr | 52.79.209.35/<br>3.34.60.181 | 443 |

## 2.2 네트워크 연결 (방화벽 확인)

PASS인증서서비스를 이용하기 위해 이용기관의방화벽허용 작업을 기본으로 합니다.
이용기관에서 사용중이신 IP/방화벽 정보를 담당자에게 전달 부탁드립니다.

**담당자 메일 (PASS인증서 그룹 : passhelp@atoncorp.com)**

## 2.3 제휴사 식별코드 / API 접근토큰 / 전송 데이터 암복호화키 발급

PASS인증서서비스 중계서버와 연동을 하기 위해서는 다음과 같은 정보를 발급받으신 후 이용가능합니다.
접근 토큰 및 암호화키는 검증(스테이지)용/상용(운영)용으로 각각 구분하여 발급됩니다.

- 제휴사 식별코드 (이용기관 코드)
- API 연동을 위한 접근 토큰 (Access Token)
- 양방향 데이터 암복호화를 위한 암호화키

**토큰(Access Token)과 암호화키에 대하여**

- 잘못된 토큰/암호화키를 사용할 경우 연동 실패입니다.
- 접근 토큰(Access Token)은 20자리의 문자열로서 PASS인증서 API와 연동 시 API 권한 체크에 사용되는 키값 입니다.
  PASS인증서서비스 중계서버에서 제공하는 접근 토큰을 양방향 API 통신에 모두 사용해도 되며, 이용기관에서 제공하는 접근 토큰과 함께 각각 적용하여 사용 가능 합니다
- 암호화키는 2가지의 키가 제공 됩니다.
  1. 대칭키(AES256 or AES128) : 16 또는 32 자리의 문자열로서 연동 시 암호화 대상 파라미터 암복호화에 사용되는 키 입니다.
  2. 개인키 : 전자서명 완료 후 수신한 CI(사용자 연계정보 식별값) 값을 복호화 할때 사용되는 키 입니다.

## 2.4 이용기관 등록 정보 전달

PASS인증서서비스 중계서버 연동을 위한 이용기관 등록 시 아래의 정보를 운영 담당자에게 전달해 주어야 합니다

**담당자 메일 (PASS인증서 그룹 : passhelp@atoncorp.com)**

### [이용기관 등록을 위한 요청자료]

```
1. 이미지 파일 (이미지 2개 모두 필요)
 1)  정사각형 : 가로 168(고정) * 세로 168(고정) pixel, jpg/png 파일
 2)  직사각형 : 가로 1300pixel 이하(변동가능) * 세로 192(고정) pixel, jpg/png 파일
2. 서비스명 이름
 -  고객에게 보여줄 발신자명에 들어갈 내용(15자 내외)
3. 개인정보 암호화키 방식 선택
 -  AES 256 (AES 128로 해야하는 경우, AES 128 선택)
4. 사업자 등록번호
5. 회사명
6. 대표자명
7. 회사주소
8. 고객센터 번호
9. 키수령 담당자 정보 (담당자명 / 연락처(휴대폰번호) / 이메일)
```

해당 이미지는 전자서명 요청 알림 등을 PASS인증앱에 표시될 때 다음과 같은 형태로 노출되는데 이용될 수 있습니다.


[서명 요청 예시 화면]

※ 스테이징서버 테스트 시 필요사항

- 개발용 PASS앱
- 테스트 사용자 등록 (휴대폰번호 / 통신사 / 생년월일 / 성별 / 이름을 담당자에게 전달 부탁드립니다.)
- KT 테스트 사용자 및 통신사별 IOS 개발용 PASS앱은 지원하지 않습니다.

## 2.5 테스트사용자 등록방법

- 사용자 등록시 필요 정보
    - 이름, 통신사, 알뜰폰여부, 생년월일, 전화번호, 성별
- SKT, LGT 사용자 등록가능

> 참고사항
> **KT 사용자 및 SKT 알뜰폰 사업자중** KCT **사용자 지원 불가**
> **LGT 개통후 3개월 지나야 등록가능**

# 3. API 연동

**PASS인증서 전자서명을 위해서는 TLS 1.2 이상 적용이 필수입니다.**

## 3.1 인증요청 API

**(이용기관/대행사/체험서비스 ▶ PASS인증서 서비스 중계서버)**

사전에 담당자에게 접근 토큰(Access Token)을 전달받아야 합니다.

### URL

| URL | PROTOCOL | METHOD | Content-Type | DESCRIPTION |
|---|---|---|---|---|
| /v1/certification/notice | HTTPS | POST | Application/json;charset=utf8 | |

### REQUEST HEADER

| PARAMETER | TYPE | LENGTH | REQUIRED | DESCRIPTION |
|---|---|---|---|---|
| Authorization | String | 20 | M | - 전달받은 접근토큰(Access Token) 값을 적용<br>- ex) Authorization: Bearer **{Access Token}** |

### REQUEST BODY (REQUIRED - M : Mandatory / O : Optional)

| PARAMETER | TYPE | LENGTH | REQUIRED | DESCRIPTION |
|---|---|---|---|---|
| companyCd | String | 5 | M | - 이용기관 코드 |
| channelTycd | String | 2 | O | - 인증 알림내용 등록을 요청하는 이용기관의 서비스 채널을 구분에사용<br>- PW : PC Web (Browser)<br>- MW : Mobile Web<br>- PA : PC Application<br>- MA : Mobile App |
| channelNm | String | 40 | O | - 이용기관의 서비스 채널 이름<br>- 리브똑똑, 현대해상 보상서비스 앱, 현대 HTS |
| agencyCd | String | 2 | O | - PASS인증서서비스에서 발급한 대행사 코드 |
| serviceTycd | String | 5 | M | - 인증서 서비스 유형 코드<br>- S1001 : 간편증빙<br>- S1002 : 간편날인<br>- S1003 : 간편통지<br>- S2001 : 출금이체동의<br>- S3001 : 간편로그인<br>- S3002 : 간편인증 |
| | | | | - 통신사 구분 코드<br>- S: SKT |

| | | | | |
|---|---|---|---|---|
| telcoTycd | String | 1 | O | - K: KT<br>- L: LGU+ |
| phoneNo | String | 40 | M | - 휴대폰 번호<br>- AES128 또는 AES256으로 암호화 |
| userNm | String | 300 | M | - 사용자의 이름<br>- AES128 또는 AES256으로 암호화 |
| birthday | String | 40 | O | - 생년월일로 “YYMMDD” 형식을 사용<br>- AES128 또는 AES256으로 암호화 |
| gender | String | 40 | O | - 성별로 0~9의 값을 가짐<br>- AES128 또는 AES256으로 암호화<br>- 9: 1800 ~ 1899년에 태어난 남성<br>- 0: 1800 ~ 1899년에 태어난 여성<br>- 1: 1900 ~ 1999년에 태어난 남성<br>- 2: 1900 ~ 1999년에 태어난 여성<br>- 3: 2000 ~ 2099년에 태어난 남성<br>- 4: 2000 ~ 2099년에 태어난 여성<br>- 5: 1900 ~ 1999년에 태어난 외국인 남성<br>- 6: 1900 ~ 1999년에 태어난 외국인 여성<br>- 7: 2000 ~ 2099년에 태어난 외국인 남성<br>- 8: 2000 ~ 2099년에 태어난 외국인 여성 |
| reqTitle | String | 50 | M | - 인증요청 알림 제목 |
| reqContent | String | 500 | O | - 인증요청 알림 내용 |
| reqCSPhoneNo | String | 12 | M | - 인증을 요청하는 이용기관의 고객센터 연락처 |
| reqEndDttm | String | 20 | M | - 인증요청의 유효 만료일시 “YYYY-MM-DD hh:mi:ss”형식사용 |
| isNotification | String | 1 | O | - 단말로 인증 알림내용에 대한 통지(Notification) 여부<br>ex)<br>- N: App 에서 호출할 경우(모바일앱, 모바일웹 가능)<br>- Y: Push로 호출하는 경우(PC웹, 모바일웹, 모바일앱 모두 가능)<br><br>**-> 사용자 인증 간소화/Push 미수신 등 고려시, 모바일 환경에서 App to App 으로 호출하는 경우(“N”)으로 설정** |
| isPASSVerify | String | 1 | M | - 서명검증 주체에 대한 식별값<br>- serviceTycd(인증서 서비스 유형 코드)값이 “S3001: 간편로그인 서비스”, “S3002: 간편인증 서비스”인 경우 “Y”값을 가져야 함<br>- **기본값 : "Y"**<br>- Y : 서명검증을 통신사 PASS인증서 시스템에서 수행 |

| | | | | |
|---|---|---|---|---|
| | | | | - N : 서명검증을 이용기관 또는 대행사에서 수행 |
| verifyURL | String | 100 | O | - 서명검증을 요청받을 이용기관 또는 대행사의 URL<br>**- 반드시 SSL이 적용된 https(안드로이드, iOS 정책)**<br>**-> isPASSVerify 값이 N인 경우 필수** |
| signTargetTycd | String | 1 | M | - 서명대상 유형 코드<br>- 1 : 서명대상이 원문 PlainText 인 경우<br>- 2: 서명대상이 원본Hash인 경우<br>- 3: 서명대상이 원본URL인 경우<br>- 4: 서명대상이 nonce인 경우(serviceTycd값이 S3001, S3002인 경우 사용) |
| signTarget | String | 500,000 | M | - 서명대상 정보<br>- 최대 500,000자리(약 1MB)의 값을 가짐<br>- 서명대상 유형 코드(signTargetTycd)값이 1번(원문), 3번(원본URL)인 경우 AES128 또는 AES256 으로 암호화<br><br>- 인증서 서비스 유형 코드(serviceTycd)값이 S3001(간편로그인 서비스), S3002(간편인증 서비스)인 경우 이용기관/대행사에서 1회용으로 생성한 nonce값(재사용 불가)을 사용<br><br>- 인증서 서비스 유형 코드(serviceTycd)값이 S2001(출금이체동의 서비스)인 경우, 서명대상정보에 다음과 같은 출금관련 정보가 포함되어 있어야함.<br>**- 오픈뱅킹 출금동의인 경우 : 이름, 금융기관명, 계좌번호**<br>**- 자동이체출금동의(CMS출금동의)인 경우 : 이름, 금융기관명, 계좌번호, 금액** |
| isUserAgreement | String | 1 | O | - 사용자 동의 필요 여부<br>**- 기본값 : "N"**<br>- Y : 통신사 PASS 앱에 노출하여 사용자 동의를 받고자 하는 경우<br>- N : 통신사 PASS 앱에 노출할 필요 없는 경우. |
| originalInfo | jsonObject | | O | - 서명대상 원본 정보<br>**- serviceTycd값 또는 signTargetTycd에 따라 필수, 옵션 구분**<br>**- 필수값인 경우 :** serviceTycd가 S1001, S1003, S2001 이고<br>signTargetTycd 2 또는 3인 경우<br>**- 아래의 경우 사용되지 않음**<br>- serviceTycd값 : S1002, S3001, S3002 |
| | | | | - 이용기관 또는 대행사에서 생성한 트랜잭션 ID |

| | | | | |
|---|---|---|---|---|
| reqTxId | String | 20 | M | **- 반드시 20자리의 값으로 요청 (특수문자 사용 불가)** |
| isDigitalSign | String | 1 | O | - 인증서 서비스 결과 알림 수신 시 전자서명값 포함 여부<br>**- 기본값: "Y"**<br>- Y : 서명 완료 후 전자서명값 수신<br>- N : 서명 완료 후 전자서명값 수신하지 않음 |
| isCombineAuth | String | 1 | O | - 전자서명 완료 시 사용자 정보 포함 여부 (serviceTycd : S1001, S1002, S1003, S2001 사용 시)<br>**- 기본값: "N"**<br><br>사용자 정보(이름, 휴대폰번호, 생년월일, 성별)<br>- Y : 서명 완료 후 사용자정보 수신<br>- N : 서명 완료 후 사용자 정보 수신하지 않음 |

> **참고**: jsonObject 의 세부항목은 아래를 참고하시기 바랍니다.

```
originalInfo : {
    originalTycd : {
        "type": "string",
        "length" : 2,
        "description":  AG: Agreement(동의서)
                        AP: Application(신청서)
                        CT: Contract(계약서)
                        GD: Guide(안내서)
                        NT: Notice(통지서)
                        TR: Terms(약관)
    },
    originalURL : {
        "type": "string",
        "length" : 100,
        "description": 반드시 SSL이 적용된 https여야함(안드로이드, iOS 정책임)
    },
    originalFormatCd : {
        "type": "string",
        "length" : 1,
        "description":  1: Plain Text
                        2: HTML
                        3: Download Image
                        4: Download Document
    }
}
```

**REQUEST SAMPLE (JSON)**

```
Authorization: Bearer {Access Token}
{
    “companyCd”: “이용기관코드”,
    “agencyCd”: “대행사코드”,
    “serviceTycd”: “S1001”,
    "telcoTycd": "S",
    "phoneNo": “01012345678”,
    “userNm”: “홍길동”,
    “birthday”: “801031”,
    “gender”: “1”,
    "reqTitle": "인증요청 알림 제목",
    “reqCSPhoneNo”: “1833-1234”,
    “reqEndDttm”: “2018-12-31 23:59:59”,
    “isNotification”: “Y”,
    “isPASSVerify”: “Y”,
    “verifyURL”: “https://example.com/verify”,
    “signTargetTycd”: “2”,
    “signTarget”: “1728FB69B522C169EAFE27E49B…",
    “isUserAgreement”: “N”,
    "originalInfo": {
        “originalTycd”: “AG”,
        “originalURL”: “https://example.com/example/agreement”,
        “originalFormatCd”: “4
    },
    “reqTxId”: “abcdefghij0123456789”,
    “isDigitalSign”:”Y”
}
```

### RESPONSE BODY

| PARAMETER | TYPE | LENGTH | REQUIRED | DESCRIPTION |
|---|---|---|---|---|
| reqTxId | String | 20 | M | - 요청시 전달받은 값을 사용 |
| certTxId | String | 20 | M | - PASS인증서서비스 중계서버에서 생성한 트랜잭션 ID<br>- PASS 앱 실행 시 전달할 경우 해당하는 트랜잭션의 인증 알림내용을 조회할 수 있으며, 전달하지 않을 경우 모든 인증 알림내용을 조회 |
| telcoTxId | String | 50 | O | - 통신사 PASS 시스템에서 생성한 트랜잭션 ID<br>- 통신사 암호화 키로 암호화된 값<br>- 요청 시 isNotification = “N”으로 보낼경우 전달 |

### RESPONSE SAMPLE (JSON)

```
HTTP 200 OK
{
    “reqTxId”: “abcdefghij0123456789”,
    “certTxId”: “1234567890klmnopqrst”
    “telcoTxId”: “암호화된 데이터”
}
```

## 3.2 인증 처리상태 조회 요청

**(이용기관/대행사 ▶ PASS인증서 서비스 중계서버)**

### URL

| URL | PROTOCOL | METHOD | Content-Type | DESCRIPTION |
|---|---|---|---|---|
| /v1/certification/status | HTTPS | GET | Application/json;charset=utf8 | |

### REQUEST HEADER

| PARAMETER | TYPE | LENGTH | REQUIRED | DESCRIPTION |
|---|---|---|---|---|
| Authorization | String | 20 | M | - 전달받은 접근토큰(Access Token) 값을 적용<br>- ex) Authorization: Bearer **{Access Token}** |

### REQUEST BODY

| PARAMETER | TYPE | LENGTH | REQUIRED | DESCRIPTION |
|---|---|---|---|---|
| reqTxId | String | 20 | M | - 인증 알림내용 등록 요청시 전달했던 이용기관 또는 대행사에서 생성한 트랜잭션 ID |
| certTxId | String | 20 | M | - 인증 알림내용 등록 요청의 응답으로 전달했던 PASS인증서서비스 중계서버에서 생성한 트랙잰션 ID |

### REQUEST SAMPLE (JSON)

```
/certification/status?reqTxId=abcdefghij0123456789&certTxId=1234567890klmnopqrst
HTTP/1.1
Authorization: Bearer {Access Token}
```

### RESPONSE BODY

| PARAMETER | TYPE | LENGTH | REQUIRED | DESCRIPTION |
|---|---|---|---|---|
| reqTxId | String | 20 | M | - 요청시 전달받은 값을 사용 |
| certTxId | String | 20 | M | - 요청시 전달받은 값을 사용 |
| statusCd | String | 1 | M | - 처리 상태<br>- W: Waiting(대기중)<br>- C: Complete(인증처리 완료)<br>- R: Reject(인증요청 거절(취소))<br>- F: Fail(인증(검증) 실패)<br>- F01: 인증서 유효성 검증 실패<br>- F02: 폐기된 인증서<br>- F03: 만료된 인증서<br>- F04: 인증내역이 존재하지 않음 |
| requestTime | String | 20 | M | - 인증요청이 PASS인증서서비스 중계서버에 등록된 시각으로 “YYYY-MM-DD hh:mi:ss”형식을 사용 |
| viewTime | String | 20 | O | - 인증요청 알림을 최초로 조회한 시각으로 “YYYY-MM-DD hh:mi:ss”형식을 사용 |
| rejectTime | String | 20 | O | - 인증을 거절(취소)한 시각으로 “YYYY-MM-DD hh:mi:ss”형식을 사용 |
| completeTime | String | 20 | O | - 인증을 완료한(서명 및 검증이 완료된) 시각으로 “YYYY-MM-DD hh:mi:ss”형식을 사용 |

### RESPONSE SAMPLE (JSON)

```
HTTP 200 OK
{
    “reqTxId”: “abcdefghij0123456789”,
    “certTxId”: “1234567890klmnopqrst”,
    “statusCd”: “C”,
    “requestTime”: “2018-09-01 13:17:20”,
    “viewTime”: “2018-09-01 13:17:21”,
    “rejectTime”: “2018-09-01 13:17:21”,
    “completeTime”: “2018-09-01 13:17:25”
}
```

## 3.3 서명 검증 처리 요청

**(PASS인증서 서비스 중계서버 ▶ 이용기관/대행사 서버)**

**이용기관/대행사에서 직접 서명검증 처리 할 경우에만 사용되며 3.1 인증 알림내용 등록 요청 시 isPASSVerify 값이 “N” 인 경우 전송**

### URL

| URL | PROTOCOL | METHOD | Content-Type | DESCRIPTION |
|---|---|---|---|---|
| 인증요청시 전달한 verifyURL | HTTPS | POST | Application/json;charset=utf8 | |

### REQUEST BODY

| PARAMETER | TYPE | LENGTH | REQUIRED | DESCRIPTION |
|---|---|---|---|---|
| reqTxId | String | 20 | M | - 인증 알림내용 등록 요청시 전달했던 이용기관 또는 대행사에서 생성한 트랜잭션 ID |
| certTxId | String | 20 | M | - PASS인증서서비스 중계서버에서 생성한 트랙잰션 ID |
| telcoTxId | String | 전달받은 값 길이 | O | - 통신사 PASS 시스템에서 생성한 트랜잭션 ID<br>- 통신사 암호화 키로 암호화된 값<br>- 요청 시 isNotification = “N” 으로 보낼경우 전달 |
| reqTycd | String | 1 | M | - 3: 고정값 |
| digitalSignature | String | 가변 | M | - 전자서명 값<br>- 최대값은 DB상의 CLOB 용량(1GB) |

### REQUEST SAMPLE (JSON)

```
{
    “reqTxId”: “abcdefghij1234567890”,
    “telcoTxId”: “전달받은 telcoTxId”,
    “certTxId”: “1234567890klmnopqrst”,
    "digitalSignature": "전자서명 값"
}
```

### RESPONSE BODY

| PARAMETER | TYPE | LENGTH | REQUIRED | DESCRIPTION |
|---|---|---|---|---|
| reqTxId | String | 20 | M | - 요청시 전달받은 값을 사용 |
| telcoTxId | String | 20 | M | - 요청시 전달받은 값을 사용 |
| certTxId | String | 20 | M | - 요청시 전달받은 값을 사용 |

**RESPONSE SAMPLE (JSON)**

```
HTTP 200 OK
{
    “reqTxId”: “abcdefghij1234567890”,
    “telcoTxId”: “abcdefghij0123456789”,
    “certTxId”: “1234567890klmnopqrst”
}
```

## 3.4 PASS인증서 발급 여부 조회

**(이용기관/대행사 ▶ PASS인증서 서비스 중계서버)**

### URL

| URL | PROTOCOL | METHOD | Content-Type | DESCRIPTION |
|---|---|---|---|---|
| /v1/certification/notice/inquiry/subscriber | HTTPS | POST | Application/json;charset=utf8 | |

### REQUEST BODY

| PARAMETER | TYPE | LENGTH | REQUIRED | DESCRIPTION |
|---|---|---|---|---|
| companyCd | String | 5 | M | - 인증 알림내용 등록을 요청하는 이용기관 코드로 PASS인증서서비스 중계서버에서 발급한 이용기관 코드를 사용함 |
| agencyCd | String | 2 | O | - 인증 알림내용 등록을 요청하는 대행사 코드로 PASS인증서서비스 중계서버에서 발급한 대행사 코드를 사용함 |
| userNm | String | 300 | M | - 사용자 이름<br>- AES128 또는 AES256으로 암호화 |
| birthday | String | 40 | O | - 사용자의 생년월일로 “YYMMDD” 형식을 사용<br>- AES128 또는 AES256으로 암호화 |
| gender | String | 40 | O | - 성별로 0~9의 값을 가짐<br>- AES128 또는 AES256으로 암호화<br>- 9: 1800 ~ 1899년에 태어난 남성<br>- 0: 1800 ~ 1899년에 태어난 여성<br>- 1: 1900 ~ 1999년에 태어난 남성<br>- 2: 1900 ~ 1999년에 태어난 여성<br>- 3: 2000 ~ 2099년에 태어난 남성<br>- 4: 2000 ~ 2099년에 태어난 여성<br>- 5: 1900 ~ 1999년에 태어난 외국인 남성<br>- 6: 1900 ~ 1999년에 태어난 외국인 여성<br>- 7: 2000 ~ 2099년에 태어난 외국인 남성<br>- 8: 2000 ~ 2099년에 태어난 외국인 여성 |
| phoneNo | String | 40 | M | - 사용자 휴대폰번호<br>- AES128 또는 AES256으로 암호화 |
| reqTxId | String | 20 | M | - 이용기관 또는 대행사에서 생성한 트랜잭션 ID |

### REQUEST SAMPLE (JSON)

```
Authorization: Bearer {Access Token}
{
    “companyCd” : “이용기관코드”,
    “agencyCd”: “대행사코드”,
    “userNm”: “홍길동”,
    “birthday”: “801031”,
    “gender”: “1”,
    "phoneNo": “01012345678”,
    “reqTxId”: “abcdefghij0123456789”
}
```

### RESPONSE BODY

| PARAMETER | TYPE | LENGTH | REQUIRED | DESCRIPTION |
|---|---|---|---|---|
| reqTxId | String | 20 | M | - 요청시 전달받은 값을 사용 |
| isSubscribed | String | 1 | M | - PASS 인증서 가입 여부 Y: 가입, N: 미가입<br>(PASS인증서서비스에 등록된 PASS 인증서 가입 정보 기준으로 응답) |

### RESPONSE SAMPLE (JSON)

```
HTTP 200 OK
{
    “reqTxId”: “abcdefghij0123456789”,
    " isSubscribed ": "Y"
}
```

### 3.5 검증 결과 요청

**(이용기관/대행사 ▶ PASS인증서 서비스 중계서버)**

#### URL

| URL | PROTOCOL | METHOD | Content-Type | DESCRIPTION |
|---|---|---|---|---|
| /certification/result | HTTPS | POST | Application/json;charset=utf8 | |

#### REQUEST BODY

| PARAMETER | TYPE | LENGTH | REQUIRED | DESCRIPTION |
|---|---|---|---|---|
| companyCd | String | 5 | M | - 인증 알림내용 등록을 요청하는 이용기관 코드로 PASS인증서서비스 중계서버에서 발급한 이용기관 코드를 사용함 |
| reqTxId | String | 20 | M | - 인증 알림내용 등록 요청시 전달했던 이용기관 또는 대행사에서 생성한 트랜잭션 ID |
| certTxId | String | 20 | M | - 인증 알림내용 등록 요청의 응답으로 전달받은 PASS인증서서비스 중계서버에서 생성한 트랜잭션 ID |
| phoneNo | String | 40 | M | - 사용자 휴대폰번호<br>- AES128 또는 AES256으로 암호화 |
| userNm | String | 300 | M | - 사용자 이름<br>- AES128 또는 AES256으로 암호화 |

#### REQUEST SAMPLE (JSON)

```
Authorization: Bearer {Access Token}
{
    “companyCd”: “이용기관코드”,
    “reqTxId”: “abcdefghij1234567890”,
    “certTxId”: “1234567890klmnopqrst”,
    "phoneNo": “01012345678”,
    “userNm”: “홍길동”
}
```

#### RESPONSE BODY

| PARAMETER | TYPE | LENGTH | REQUIRED | DESCRIPTION |
|---|---|---|---|---|
| reqTxId | String | 20 | M | - 요청시 전달받은 값을 사용 |
| telcoTxId | String | 20 | O | - 통신사 PASS 시스템에서 생성한 트랜잭션 ID<br>- 서명 대기 시 전송되지 않음 |
| certTxId | String | 20 | M | - PASS인증서서비스 중계서버에서 생성한 트랜잭션 ID |
| | | | | - 서비스 알림 유형 코드 |

| | | | | |
|---|---|---|---|---|
| resultTycd | String | 1 | M | (PASS인증서서비스에 등록된 PASS 인증서 가입 정보 기준으로 응답)<br>1: 인증 완료<br>2: 인증 대기중<br>3: 서명검증 Fail<br>4: 인증요청 거절(취소)<br>5: 인증요청 만료<br>6: 인증서 유효성 검증 실패<br>7: 폐기된 인증서<br>8: 만료 인증서 |
| resultDttm | String | 20 | O | - 결과 일시로 resultTycd에 해당하는 일시정보가 전달되며 “YYYY-MM-DD hh:mi:ss” 형식을 사용<br>- 서명 대기 시 전송되지 않음 |
| digitalSign | String | 가변 | O | - 전자서명 값<br>- 최대값은 DB상의 CLOB 용량(1GB)<br>- 인증 알림내용 등록 요청 시 isDigitalSign값이 “Y” 인 경우에만 값이 전달됨<br>- resultTycd값이 “1: 인증 완료”인 경우에만 값이 전달됨 |
| CI | String | 200 | O | - 사용자 연계정보 식별값(Connectiong Information)<br>- resultTycd값이 “1: 인증 완료”인 경우 전달됨<br>- 이용기관의 공개키로 암호화된 CI (RSA/ECB/PKCS1Padding)<br><br>**※ 아래 userNm, birthday, gender 전달 조건**<br>- serviceTycd값이 S3001, S3002이고 resultTycd값이 "1" 일 경우<br>- serviceTycd값이 S1001, S1002, S1003, S2001 이고 isCombineAuth값이 "Y" 이고 resultTycd값이 "1" 일 경우 |
| userNm | String | 300 | O | **- CI 설명 참조**<br>- 사용자 이름<br>- AES128 또는 AES256으로 암호화 |
| birthday | String | 40 | O | **- CI 설명 참조**<br>- 사용자의 생년월일로 “YYMMDD” 형식을 사용<br>- AES128 또는 AES256으로 암호화 |
| gender | String | 40 | O | **- CI 설명 참조**<br>- 사용자의 성별로 0~9의 값을 가짐<br>- AES128 또는 AES256으로 암호화<br><br>- 9: 1800 ~ 1899년에 태어난 남성<br>- 0: 1800 ~ 1899년에 태어난 여성<br>- 1: 1900 ~ 1999년에 태어난 남성<br>- 2: 1900 ~ 1999년에 태어난 여성<br>- 3: 2000 ~ 2099년에 태어난 남성<br>- 4: 2000 ~ 2099년에 태어난 여성<br>- 5: 1900 ~ 1999년에 태어난 외국인 남성 |

| | | | | |
|---|---|---|---|---|
| | | | | - 6: 1900 ~ 1999년에 태어난 외국인 여성<br>- 7: 2000 ~ 2099년에 태어난 외국인 남성<br>- 8: 2000 ~ 2099년에 태어난 외국인 여성 |
| phoneNo | String | 40 | O | - 전자서명 (serviceTycd : S1001, S1002, S1003, S2001) 시 isCombineAuth 값이 “Y’ 이고 resultTycd값이 “1: 인증 완료”인 경우 전달됨.<br>- AES128 또는 AES256으로 암호화 |
| telcoTycd | String | 1 | O | - 통신사 구분 코드<br>- resultTycd값이 “1: 인증 완료”인 경우에만 값이 전달됨<br>- S: SKT<br>- K: KT<br>- L: LGU+ |

**RESPONSE SAMPLE (JSON)**

```
{
    “reqTxId”: “abcdefghij1234567890”,
    “telcoTxId”: “abcdefghij0123456789”,
    “certTxId”: “1234567890klmnopqrst”,
    “resultTycd: “1”,
    "digitalSign": "전자서명 값",
    "CI": "ALKDJF17KHJ11AC1T...",
    “userNm”: “홍길동”,
    “birthday”: “801031”,
    “gender”: “1”
}
```

### 3.6 PASS인증서서비스 에러 응답 및 에러 코드

#### RESPONSE BODY

| PARAMETER | TYPE | LENGTH | REQUIRED | DESCRIPTION |
|---|---|---|---|---|
| errorCd | Integer | 4 | M | - 에러 코드 |
| errorMessage | String | 100 | M | - 에러 메세지 |
| errorPointCd | String | 5 | M | - 에러 발생 지점 코드<br>- TLPAS: 통신사 PASS 서버(TeLco PASS Server)<br>- PACPR: PASS인증서서비스 요청 관리(PASS Certification Platform Request manamgent)<br>- UOSYS: 이용기관 시스템(Using Organization SYStem)<br>- AGSYS: 대행사 시스템(AGency SYStem) |
| telcoTxId | String | 20 | O | - 요청시 전달받은 값을 사용 |
| reqTxid | String | 20 | O | - 요청시 전달받은 값을 사용 |
| certTxId | String | 20 | O | - 요청시 전달받은 값을 사용 |

#### RESPONSE SAMPLE (JSON)

```
HTTP 400/500 Bad Request/Internal Server Error
{
    “errorCd”: xxxx,
    “errorMessage”: “필수 요청함목이 누락되었습니다.”,
    “errorPointCd”: “PACPM”,
    “telcoTxId”: “abcdefghij0123456789”,
    “reqTxId”: “abcdefghij1234567890”,
    “certTxId”: “1234567890klmnopqrst”
}
```

### o 공통 에러(9xxx)

| 에러 코드 | 에러 메시지 | HTTP Status Cd |
|---|---|---|
| 9000 | 권한이 없습니다 | 401 Unauthorized |
| 9001 | 요청 Body가 없습니다. | 400 Bad Request |
| 9002 | 요청 Body 형식이 잘못되었습니다. | 400 Bad Request |
| 9003 | 지원하지 않는 HTTP Method입니다. | 400 Bad Request |
| 9004 | 일시적인 오류가 발생했습니다. 담당자에게 문의하세요. | 400 Bad Request |
| 9099 | 일시적인 오류가 발생했습니다. 잠시 후 다시 요청해 주세요 | 500 Internal Server Error |

### o 인증 알림내용 등록 에러(3xxx)

| 에러 코드 | 에러 메시지 | HTTP Status Cd |
|---|---|---|
| 3101 | 필수항목 {요청항목}이 누락되었습니다. | 400 Bad Request |
| 3102 | {요청항목} 값이 유효하지 않습니다. | 400 Bad Request |
| 3103 | 인증서가 발급되어 있지 않은 사용자입니다. | 400 Bad Request |
| 3104 | 인증앱에 가입되어 있지 않은 사용자입니다. | 400 Bad Request |
| 3105 | {통신사 PASS 서버 오류코드}-{통신사 PASS 서버 에러 메시지} | 400 Bad Request |
| 3106 | 통신사에 가입되어 있지 않은 사용자입니다. | 400 Bad Request |
| 3107 | 통신사 시스템 오류로 가입자 정보 확인이 불가능합니다. | 500 Bad Request |
| 3199 | 일시적인 오류가 발생했습니다. 잠시 후 다시 요청해 주세요. | 500 Internal Server Error |

### o 검증 요청 에러(4xxx)

| 에러 코드 | 에러 메시지 | HTTP Status Cd |
|---|---|---|
| 4101 | 필수항목 {요청항목}이 누락되었습니다. | 400 Bad Request |
| 4102 | {요청항목} 값이 유효하지 않습니다. | 400 Bad Request |
| 4103 | 유효기간이 만료된 인증서 입니다. | 400 Bad Request |
| 4104 | 삭제(폐기)된 인증서 입니다. | 400 Bad Request |
| 4105 | 기기 고유번호가 일치하지 않습니다. | 400 Bad Request |
| 4106 | 서명값이 유효하지 않습니다. | 400 Bad Request |
| 4107 | 인증 요청 유효기간이 만료되었습니다. | 400 Bad Request |
| 4108 | 이미 인증이 완료 되었습니다 | 400 Bad Request |
| 4109 | 이용기관 검증 중 오류가 발생했습니다. | 500 Internal Server Error |
| 4110 | 인증 요청과 일치하는 데이터가 없습니다. | 400 Bad Request |
| 4111 | 인증 요청 만료 일자가 유효하지 않습니다. | 400 Bad Request |
| 4112 | 이미 거절된 서명입니다. | 400 Bad Request |
| 4113 | 이용기관 연동중 오류가 발생했습니다. | 500 Internal Server Error |
| 4199 | 일시적인 오류가 발생했습니다. 잠시 후 다시 요청해 주세요. | 500 Internal Server Error |

### o 인증 알림내용 상세 조회 에러(5xxx)

| 에러 코드 | 에러 메시지 | HTTP Status Cd |
|---|---|---|
| 5101 | 필수항목 {요청항목}이 누락되었습니다. | 400 Bad Request |
| 5102 | {요청항목} 값이 유효하지 않습니다. | 400 Bad Request |
| 5103 | 인증 요청 유효기간이 만료되었습니다. | 400 Bad Request |
| 5104 | 이미 인증이 완료 되었습니다. | 400 Bad Request |
| 5105 | 이미 거절(취소)한 인증 요청입니다. | 400 Bad Request |
| 5106 | 인증 요청 유효기간이 만료되었거나, 일치하는 데이터가 없습니다. | 400 Bad Request |
| 5199 | 일시적인 오류가 발생했습니다. 잠시 후 다시 요청해 주세요. | 500 Internal Server Error |

### o 인증 처리상태 조회 에러(6xxx)

| 에러 코드 | 에러 메시지 | HTTP Status Cd |
|---|---|---|
| 6101 | 필수항목 {요청항목}이 누락되었습니다. | 400 Bad Request |
| 6102 | {요청항목} 값이 유효하지 않습니다. | 400 Bad Request |
| 6103 | 인증 요청 유효기간이 만료되었거나, 일치하는 데이터가 없습니다. | 400 Bad Request |
| 6199 | 일시적인 오류가 발생했습니다. 잠시 후 다시 요청해 주세요. | 500 Internal Server Error |

### 3.7 통신사 PASS 서버 에러 코드

| CODE | MESSAGE | HTTP Status Cd | DESCRIPTION |
|---|---|---|---|
| E0000 | | 200 | 성공 |
| E0001 | 허용되지 않는 accessToken 입니다. | 401 | |
| E0002 | 필수항목 {요청항목}이 누락되었습니다. | 400 | |
| E0003 | {요청항목} 값이 유효하지 않습니다. | 400 | |
| E0004 | 요청한 내용과 일치하는 데이터가 없습니다. | 400 | 인증서 발급완료 알림 시 telcoTxId, certTxId, CI와 일치하는 데이터가 없는 경우 |
| E0005 | 요청한 내용과 일치하는 인증 알림 내역이 없습니다 | 400 | 인증서 만료 예정 알림 시 certTxid에 해당되는 데이터가 존재하는 않은 경우 |
| E0200 | {SKT/KT/LGU+} 가입자가 아닙니다. | 403 | SKT/KT/LGU+ 가입자가 아니고,<br>KT/LGU+ 알뜰폰 가입자도 아닌 경우<br><br>(SKT는 알뜰폰 가입자 판단 불가) |
| E0201 | {SKT/KT/LGU+} 2G/3G 가입자 입니다. | 403 | SKT/KT/LGU+ 2G 가입자인 경우 |
| E0202 | 일시 정지된 휴대폰 번호입니다. | 403 | |
| E0203 | 시스템 오류로 가입자 정보 확인이 불가능합니다. | 403 | |
| E0204 | PASS 앱 미가입자 입니다. | 403 | |
| E0205 | PASS 인증서 미가입자 입니다. | 403 | |
| E0206 | 전달 휴대폰 번호와 다른 휴대폰 번호가 조회됩니다. | 403 | CI 및 이름 생년월일은 동일하나 전화번호가 다른 경우가 조회될 때 |
| E0207 | 분실 신고된 휴대폰 번호 입니다. | 403 | |
| E0208 | 사용자가 일시적으로 집중되어 서비스 지연 중입니다. 잠시 후 다시 시도해주세요. | 403 | |
| E8000 | 알림 발송 오류입니다. 잠시 후 다시 시도해주세요. | 500 | |
| E9999 | 시스템 오류입니다. 잠시 후 다시 시도해주세요. | 500 | |

### 3.8 코드 정의

| CODE TYPE | PARAMETER | LENGTH | DESCRIPTION |
|---|---|---|---|
| 운영 체제(OS) 유형 코드 | deviceOsTycd | 1 | A - Android<br>I - IOS |
| 통신사 구분 코드 | telcoTycd | 1 | S - SKT<br>K - KT<br>L - LGT+ |
| 인증서 서비스 유형 코드 | serviceTycd | 5 | S1001 - 간편증빙<br>S1002 - 간편날인<br>S1003 - 간편통지<br>S2001 - 출금이체동의<br>S3001 - 간편로그인<br>S3002 - 간편인증 |
| 서명대상 유형 코드 | signTargetTycd | 1 | 1 - 서명대상이 원문 Plain Text인 경우<br>2 - 서명대상이 원본Hash인 경우<br>3 - 서명대상이 원본URL인 경우<br>4 - 서명대상이 nonce인 경우(serviceTycd값이 S3001, S3002인 경우 사용)<br>5 - 서명대상이 원문 HTML인 경우 |
| 원본 유형 코드 | originalTycd | 2 | AG - Agreement(동의서)<br>AP - Application(신청서)<br>CT - Contract(계약서)<br>GD - Guide(안내서)<br>NT - Notice(통지서)<br>TR - Terms(약관) |
| 원본 형태 유형 코드 | originalFormatCd | 1 | 1 - Plain Text<br>2 - HTML<br>3 - Download Image<br>4 - Download Document |
| 인증 처리 상태 코드 | statusCd | 1 | W - Waiting(대기중)<br>V - Viewed(조회완료)<br>C - Complete(인증처리 완료)<br>R - Reject(인증요청 거절(취소))<br>F - 서명검증 실패<br>F01 - 인증서 유효성 검증 실패<br>F02 - 폐기된 인증서<br>F03 - 만료된 인증서<br>F04 - 인증내역이 존재하지 않음 |
| 대행사 코드 | agencyCd | 2 | 담당자에게 문의 요망 |
| 이용기관 코드 | companyCd | 5 | 담당자에게 문의 요망 |

# 4. 상용서비스 적용 및 참고사항

상용서비스 적용은 스테이지에서의 연동 **테스트 완료 후 아톤과 접근토큰 및 암호화키의 적용시점 협의**가 필요합니다.

PASS인증서 서비스UI 구현 시 샘플 화면은 전달드린 파일들 중 아래 항목 참고부탁드립니다. 개인정보 제3자 제공 동의 내용은 별도로 요청주시면 전달드립니다.

o **(PASS인증서_화면템플릿.zip 파일 참고)**
o **[공통]PASS인증서*표준화면&대기화면 가이드*v2.0.pdf**

### PASS인증서 관련 명칭 가이드

- **공식 명칭** : PASS인증서, 패스인증서
- **서비스 명칭**: PASS인증서 (간편)로그인 / (본인)인증 / 전자서명 / 출금(이체)동의 / 통합인증
- **인증요청 버튼 명칭** : 로그인 요청 / 간편인증(본인인증) 요청 / 전자서명 요청 / 출금(이체)동의 요청

### 참고 Link

- PASS인증서서비스 중계서버 Swagger Doc. Page
  http://52.78.120.77:8080/swagger-ui.html
- AES-128 (or AES-256) 데이터 암복호화 결과 확인
  https://www.devglan.com/online-tools/aes-encryption-decryption

### 암호화 샘플 소스 (AES-128 or AES-256)

```java
import org.apache.commons.codec.binary.Base64;
import javax.crypto.Cipher;
import javax.crypto.SecretKey;
import javax.crypto.spec.IvParameterSpec;
import javax.crypto.spec.SecretKeySpec;

public class AESCipher {
    // 알고리즘/모드/패딩
    private static final String algorithm = "AES/CBC/PKCS5Padding";
    // 암호화 키
    private SecretKey secretKey;
    // 초기화 벡터
    private IvParameterSpec iv;
    // 문자인코딩 방식
    private final String charset = "UTF-8";

    public AESCipher(String aesKey) {
        if(aesKey == null){
            throw new NoSecretKeyException("No SecretKey, Please Set SecretKey
!!!");
        }

        if(aesKey.length() > 16) {
            this.iv = new IvParameterSpec(aesKey.substring(0, 16).getBytes());
        } else {
            this.iv = new IvParameterSpec(aesKey.getBytes());
        }

        this.secretKey = new SecretKeySpec(aesKey.getBytes(), "AES");
    }

    // 암호화
    public String encrypt(String str) throws Exception {
        Cipher c = Cipher.getInstance(algorithm);
        c.init(Cipher.ENCRYPT_MODE, this.secretKey, this.iv);
        return new String(Base64.encodeBase64(c.doFinal(str.getBytes(charset))));
    }

    // 복호화
    public String decrypt(String str) throws Exception {
        Cipher c = Cipher.getInstance(algorithm);
        c.init(Cipher.DECRYPT_MODE, this.secretKey, this.iv);
        return new String(c.doFinal(Base64.decodeBase64(str.getBytes())), charset);
    }
}
class NoSecretKeyException extends RuntimeException {
    public NoSecretKeyException(String msg) {
        super(msg);
    }
}
```

### O 보안 관련 적용사항

1. **White-Box Cryptography 기반 mSafeBox 사용**
   1) PASS인증서 개인키 저장

2. **공통**
   1) 모든 웹페이지/웹서비스 기본적으로 Secure Coding 가이드에 따른 개발
   2) 네트워크 구간 SSL 적용
   3) IPS / Anti-Webshell 적용
   4) 개인정보 DB 암호화

3. **서버 시스템**
   1) 연동 시스템과 전용회선 사용
   2) 연동 시스템 접근 IP 제어(AWS 제공 Security Group 사용)
   3) AWS 콘솔의 구글 OTP 인증을 통한 관리자 접근
   4) DB 접근제어 솔루션 적용

4. **API 호출**
   1) 접근 토큰(Access Token)을 통해 API 호출 권한 제어
   2) API 호출 시스템에 대한 접근 IP 제어 (AWS 제공 Security Group 사용)

# 5. End To End(이용기관 <-> CA 기관) 보안을 위한 CI 전달 방법

CI (Connection Information)은 사용자를 식별할 수 있는 정보로 사용자의 중복가입 방지 등을 차단하기 위해서 사용하는 값입니다.
만약 외부에 노출이 될 경우 보안상 큰 위험이 있기 때문에 CI의 경우 이용기관과 인증기관(CA) 사이에 End To End 암호화를 적용할 수 있습니다.

- 이용기관 데이터 암복호화용 인증서 (KMCert)
    - 이용기관 개인키 생성 : 이용기관만 보유하는 개인키 (CI 복호화에 사용)
      **(※ OpenSSL RSA키 생성.pdf 가이드 참고)**
- CI 암복호화 진행 순서
    - PASS인증서 서비스에서 이용기관에서 생성한 공개키를 전달받아 저장합니다.
    - 이용기관에서 전자서명 요청이 있을 경우에 PASS인증서 서비스는 CA 기관에 전자서명 요청과 함께 이용기관의 공개키를 같이 전달합니다.
    - CA 기관은 전자서명을 생성한 후 전달된 이용기관의 공개키로 전자서명을 요청한 사용자의 CI를 암호화하여 전달합니다.
- **암호화 알고리즘은 “RSA/ECB/PKCS1Padding” 을 사용합니다.**
- 이용기관은 전자서명 결과를 요청하면 다음 값을 응답으로 전달받습니다.
  **전자서명값 / 전자서명 결과 / 암호화된 사용자 정보(AES256 대칭키 암호화) / 암호화된 사용자 CI( RSA2048 비대칭키 암호화)**
- 암호화된 사용자의 CI는 이용기관만 가지고 있는 개인키(kmprikey.pem)을 이용하여야만 복호화가 가능합니다. 즉, 이용기관에서만 복호화를 수행할 수 있습니다.

### CI 암호화 샘플 소스

```java
import javax.crypto.BadPaddingException;
import javax.crypto.Cipher;
import javax.crypto.IllegalBlockSizeException;
import javax.crypto.NoSuchPaddingException;
import java.io.IOException;
import java.nio.charset.Charset;
import java.nio.file.Files;
import java.nio.file.Path;
import java.nio.file.Paths;
import java.security.InvalidKeyException;
import java.security.KeyFactory;
import java.security.NoSuchAlgorithmException;
import java.security.interfaces.RSAPrivateKey;
import java.security.spec.InvalidKeySpecException;
import java.security.spec.PKCS8EncodedKeySpec;
import java.util.Base64;

public class RsaDecryptSample {
    public static void main(String[] args) throws IOException,
NoSuchAlgorithmException, InvalidKeySpecException, NoSuchPaddingException,
InvalidKeyException, IllegalBlockSizeException, BadPaddingException {
        //이용기관의 공개키로 암호화한 CI 값 (CA 기관에서 전달)
        String encryptedCiByPublicKey = "{암호화된 CI}";

        //이용기관 등록시 발급한 개인키 파일 로드
        String fileName = "{개인키 파일 절대경로}";
        Path filePath = Paths.get(fileName);

        String getPrivate = new String(Files.readAllBytes(filePath),
Charset.defaultCharset());
        String privateKey = getPrivate.replace("-----BEGIN PRIVATE KEY-----",
"").replaceAll(System.lineSeparator(), "").replace("-----END PRIVATE KEY-----",
"");

        byte[] key = Base64.getDecoder().decode(privateKey);
        PKCS8EncodedKeySpec keySpec = new PKCS8EncodedKeySpec(key);
        KeyFactory kf = KeyFactory.getInstance("RSA");

        RSAPrivateKey pk = (RSAPrivateKey) kf.generatePrivate(keySpec);

        //알고리즘 및 패딩 설정
        Cipher cipher = Cipher.getInstance("RSA/ECB/PKCS1Padding");
        cipher.init(Cipher.DECRYPT_MODE, pk);

        byte[] decBytes =
Base64.getDecoder().decode(encryptedCiByPublicKey.getBytes());

        //복호화 수행
        String decData = new String(cipher.doFinal(decBytes));

        //복호화된 개인키
        System.out.println("복호화된 CI = " + decData);
    }
}
```

# 6. App TO App 연동

## Android

### AndroidMenifest.xml 에 APP Scheme 등록

- AndroidMenifest.xml 의 호출받을 Activity 에 Scheme 등록Example

```
<activity android:name=".ui.MainActivity">
    <intent-filter>
        <action android:name="android.intent.action.VIEW" />
        <category android:name="android.intent.category.DEFAULT" />
        <category android:name="android.intent.category.BROWSABLE" />
        <data
            android:scheme="app2appdummy" />
    </intent-filter>
</activity>
```

### PASS 앱 전자서명 요청 (이용기관 앱-> PASS 인증 앱)

- URLScheme 호출

| 통신사 | URLScheme |
|---|---|
| SKT | {scheme}://{host}?reqTxId={reqTxId}&telcoTxId={telcoTxId}&certTxId={certTxId}&callbackScheme=app2appdummy&packageName=com.atoncorp.app2appdummy |
| KT | {scheme}://{host}?reqTxId={reqTxId}&telcoTxId={telcoTxId}&certTxId={certTxId}&callbackScheme=app2appdummy&packageName=com.atoncorp.app2appdummy |
| LGU | {scheme}://{host}?reqTxId={reqTxId}&telcoTxId={telcoTxId}&certTxId={certTxId}&sc=004&callbackScheme=app2appdummy&packageName=com.atoncorp.app2appdummy |

#### PARAMETER 정의

- SKT

| PARAMETER | DESCRIPTION |
|---|---|
| reqTxId | 이용기관의 Transaction ID |
| certTxId | 패스 플랫폼 서버의 Transaction ID |
| telcoTxId | 이동통신사의 암호화된 Transaction ID |
| callbackScheme | PASS앱에서 인증완료이후 이용기관앱을 호출하기 위한 Scheme(앞에서 등록한 scheme) |
| packageName | PASS앱에서 인증완료이후 이용기관앱을 호출하기 위한 PackageName |

- KT

| PARAMETER | DESCRIPTION |
|---|---|
| reqTxId | 이용기관의 Transaction ID |
| certTxId | 패스 플랫폼 서버의 Transaction ID |
| telcoTxId | 이동통신사의 암호화된 Transaction ID |
| callbackScheme | PASS앱에서 인증완료이후 이용기관앱을 호출하기 위한 Scheme(앞에서 등록한 scheme) |
| packageName | PASS앱에서 인증완료이후 이용기관앱을 호출하기 위한 PackageName |

- LGU+

| PARAMETER | DESCRIPTION |
|---|---|
| reqTxId | 이용기관의 Transaction ID |
| certTxId | 패스 플랫폼 서버의 Transaction ID |
| telcoTxId | 이동통신사의 암호화된 Transaction ID |
| sc | 서비스 코드(004) |
| callbackScheme | PASS앱에서 인증완료이후 이용기관앱을 호출하기 위한 Scheme(앞에서 등록한 scheme) |
| packageName | PASS앱에서 인증완료이후 이용기관앱을 호출하기 위한 PackageName |

- 통신사 PASS URL Scheme, HOST , packageName

| 통신사 | Scheme | HOST | packageName |
|---|---|---|---|
| **SKT** | tauthlink | certauth | com.sktelecom.tauth |
| **KT(개발,검증)** | ktpasslink20 | requestSignCert | kt.fido.sw.asm.api |
| **KT(상용)** | ktpasslink20 | requestSignCert | com.kt.ktauth |
| **LGU+** | upluscorporation | | com.lguplus.smartotp |

- 이용기관 앱에서 PASS 앱 호출

```
public void startPassAuth(Context context){
        //통신사 PASS 패키지명
        //LGU+
        String telPackageName = “com.lguplus.smartotp”;
        //KT(개발/검증)
        String telPackageName = “kt.fido.sw.asm.api”;
        //KT(상용)
        String telPackageName = “com.kt.ktauth”;
        //SKT
        String telPackageName = “com.sktelecom.tauth”;

       //URL Scheme
       //LGU+
        String schemeUrl = "upluscorporation://?reqTxId={reqTxId}&telcoTxId=
{telcoTxId}&certTxId=
{certTxId}&sc=004&callbackScheme=app2appdummy&packageName=com.atoncorp.app2appdummy";


        //KT
         String schemeUrl = “ktpasslink20://requestSignCert?reqTxId=
{reqTxId}&telcoTxId={telcoTxId}&certTxId=
{certTxId}&callbackScheme=app2appdummy&packageName=com.atoncorp.app2appdummy”;

       //SKT
         String schemeUrl = “tauthlink://certauth?reqTxId={reqTxId}&telcoTxId=
{telcoTxId}&certTxId=
{certTxId}&callbackScheme=app2appdummy&packageName=com.atoncorp.app2appdummy";

        Intent intent;
        String marketUrl = "market://details?id=" + telPackageName;
        Intent marketIntent =
context.getPackageManager().getLeanbackLaunchIntentForPackage(telPackageName);

        if (marketIntent == null){
            //PASS 앱이 설치되어 있지 않은 경우
            intent = new Intent(Intent.ACTION_VIEW, Uri.parse(marketUrl));
        }else{

            intent = new Intent(Intent.ACTION_VIEW, Uri.parse(schemeUrl));
        }

        try {
            context.startActivity(intent);
        }catch (Exception e){
            e.printStackTrace();
        }
    }
```

### callback scheme을 이용한 이용기관 앱 실행 (PASS 앱 -> 이용기관 앱)

- 이용기관 앱 Scheme 호출

```
app2appdummy://?code={code}
```

- Example

```
public void authCallback() {
    StringBuilder uri = new StringBuilder(callbackScheme);
    uri.append(“://?code=”);
    uri.append(code);

    Intent intent = new Intent(Intent.ACTION_VIEW, Uri.parse(uri.toString()));
    intent.addFlags(Intent.FLAG_ACTIVITY_SINGLE_TOP);
    intent.setPackage(callbackPackageName);
    startActivity(intent);
    finish();
}
```

- 이용기관 앱을 실행한 이후 PASS 앱 종료

| PARAMETER | DESCRIPTION |
|---|---|
| callbackScheme | 2.1의 PASS 인증 요청 시에 이용기관 앱으로부터 전달받은 이용기관 앱으로 되 돌아가기 위한 scheme |
| callbackPackageName | 2.1의 PASS 인증 요청 시에 이용기관 앱으로부터 전달받은 이용기관 앱으로 되 돌아가기 위한 패키지명 |
| code | 아래 오류코드 정의 참조 |

## IOS

### 실행환경

IOS Version 9.0 이상

### PASS 앱 전자서명 요청 (이용기관 앱 -> PASS 앱)

- Universal Link 호출

| 통신사 | Universal Link |
|---|---|
| SKT | {universal_link}?reqTxId={reqTxId}&certTxId={certTxId}&telcoTxId={telcoTxId}&callbackScheme=App2AppDummy |
| KT | {universal_link}?reqTxId={reqTxId}&certTxId={certTxId}&telcoTxId={telcoTxId}&callbackScheme=App2AppDummy&callBackUrl={callBackUrl} |
| LGU | {universal_link}?reqTxId={reqTxId}&certTxId={certTxId}&telcoTxId={telcoTxId}&callbackScheme=App2AppDummy&sc=004&callBackUrl={callBackUrl} |

- 통신사별 universal_link 정의
  - SKT
    - 개발
    - 검수 : https://passhome-qa.minwise.co.kr/applink/certauth
    - 상용 : https://www.sktpass.com/applink/certauth

- KT
  - 개발 : https://kaf.dayside.co.kr/requestSignCert
  - 검수 : https://tb.fido.kt.com/requestSignCert
  - 상용 : https://fido.kt.com/requestSignCert
- LGU+
  - 개발 : https://fidotest.uplus.co.kr/rp/lgauthPass
  - 검수 : https://fidotest.uplus.co.kr/rp/lgauthPass
  - 상용 : https://fido.uplus.co.kr/rp/lgauthPass

**PARAMETER 정의**

- SKT

| PARAMETER | DESCRIPTION |
|---|---|
| reqTxId | 이용기관의 Transaction ID |
| certTxId | 패스 플랫폼 서버의 Transaction ID |
| telcoTxId | 이동통신사의 암호화된 Transaction ID |
| callbackScheme | PASS앱에서 인증완료이후 이용기관앱을 호출하기 위한 Scheme(2.1에서 등록한 scheme) |

- KT

| PARAMETER | DESCRIPTION |
|---|---|
| reqTxId | 이용기관의 Transaction ID |
| certTxId | 패스 플랫폼 서버의 Transaction ID |
| telcoTxId | 이동통신사의 암호화된 Transaction ID |
| callbackScheme | PASS앱에서 인증완료이후 이용기관앱을 호출하기 위한 Scheme(2.1에서 등록한 scheme) |

- LGU+

| PARAMETER | DESCRIPTION |
|---|---|
| reqTxId | 이용기관의 Transaction ID |
| certTxId | 패스 플랫폼 서버의 Transaction ID |
| telcoTxId | 이동통신사의 암호화된 Transaction ID |
| sc | 서비스 코드(004) |
| callbackScheme | PASS앱에서 인증완료이후 이용기관앱을 호출하기 위한 Scheme(2.1에서 등록한 scheme) |

- 이용기관 앱에서 PASS 앱 호출

```
- (void)requestAuthToPassApp
{
    // SKT
    NSString *customURL = @"{ universal_link }?reqTxId={reqTxId}&certTxId=
{certTxId}&telcoTxId={telcoTxId}&callbackScheme=App2AppDummy";
    // KT
    NSString *customURL = @"{ universal_link }?reqTxId={reqTxId}&certTxId=
{certTxId}&telcoTxId={telcoTxId}&callbackScheme=App2AppDummy";
    // LGU+
    NSString *customURL = @"{ universal_link }?reqTxId={reqTxId}&certTxId=
{certTxId}&telcoTxId={telcoTxId}&callbackScheme=App2AppDummy&sc=004&callBackUrl=
{callBackUrl}";

    UIApplication*application = [UIApplication sharedApplication];
    NSURL*URL = [NSURL URLWithString:[customURL
stringByAddingPercentEscapesUsingEncoding:NSUTF8StringEncoding]];

    [[UIApplication sharedApplication] openURL:URL
              options:@{UIApplicationOpenURLOptionUniversalLinksOnly: @YES}
    completionHandler:^(BOOLsuccess){
        if(!success) {
            //fail
        }
    }];
}
```

### callback scheme을 이용한 이용기관 앱 실행 (PASS 앱 -> 이용기관 앱)

- 이용기관 앱 Scheme 호출

```
{callbackScheme}://?code={errorCode}
```

- Example

```
- (IBAction)actionBtnOpenApp:(id)sender {
    NSString *urlScheme = [dicParam objectForKey:@"callbackScheme"];
    NSString *customURL = [NSString stringWithFormat:@"%@://?code=%@", urlScheme,
@”E0000”];
    UIApplication *application = [UIApplication sharedApplication];
    NSURL *URL = [NSURL URLWithString:customURL];
    [application openURL:URL];
}
```

- 이용기관 앱을 실행한 이후 PASS 앱 종료

| PARAMETER | DESCRIPTION |
|---|---|
| callbackScheme | 2.1의 PASS 인증 요청 시에 이용기관 앱으로부터 전달받은 이용기관 앱으로 되 돌아가기 위한 scheme |
| code | 아래 오류코드 정의 참조 |

## 오류코드 정의

App To App에서 발생하는 오류는 이용기관의 앱에서 Pass 앱을 호출한 후 다음과 같은 케이스에서 발생한다.

- PASS 앱 미설치
  : 이용기관에 각 이통사의 PASS 앱 스토어로 이동할 수 있도록 샘플 및 가이드 제공
- PASS 서비스 미가입
  : 서비스 가입을 안내 팝업 후 이용기관 앱의 callback scheme으로 에러코드를 리턴 후 앱 종료
- PASS 인증서 미발급
  : 인증서 발급 안내 후 이용기관 앱의 callback scheme으로 에러코드를 리턴 후 앱 종료

| CODE | DESCRIPTION |
|---|---|
| E0000 | 성공 |
| E3001 | 패스 서비스 미가입자 |
| E3002 | 패스 인증서 미발급자 |
| E3003 | 사용자 인증 시 취소 또는 서명 과정 중 취소 |
| E3004 | 사용자 인증 시 인증 실패 (생체 인증 실패 등) |
| E3005 | 단말 네트워크 환경 등으로 인한 통신 실패 |
| E3006 | 통신사 서버에서 에러 발생 |
| E3007 | 인증서 Library 서명에서 에러 발생 |
| E3999 | 기타 오류 |