Open Source DRM Solution

오픈소스 문서보안 시스템 솔루션

CONTENTS

01. 팀원 소개

01

PM

백서진

9231386

웹서버 제작

02

박윤지

92313336

정책 모듈 개발

03

정보남

92313623

로컬 에이전트 개발

04

함근희

92313714

암복호화 모듈 개발

02. 주제 선정 배경

- **DRM(Digital Rights Management)이란?**

  디지털 콘텐츠의 불법 복제, 유출, 무단 사용을 방지하기 위한 기술로,

  콘텐츠에 대한 접근을 제어하고 사용 권한을 관리하는 보안 메커니즘이다.

  DRM은 일반적으로 콘텐츠를 암호화하고, 해당 콘텐츠에 접근할 수 있는 사용자나 기기에 대해

  라이선스를 발급하며, 이를 통해 사용자의 권한을 정책적으로 통제한다.

02. 주제 선정 배경 상용 DRM 시스템

국내

Fasoo

국내 대표적 DRM 솔루션
파일 단위 암호화 및 접근권한 관리
사용자, 조직별 보안 정책 적용 가능
로컬/ 클라우드/ 이메일 데이터보호 지원
기업 환경에서 강력한 DRM기능 지원

MarkAny

국내 DRM 및 워터마킹 전문 기업
디지털 콘텐츠 위변조 방지 및 추적 기능
문서, 영상, 오디오 등 다양한 형식
공공기관 및 기업에서 보안 솔루션

국외

WIDEVINE

google 제공 멀티플랫폼
클라이언트에서 Widevine 라이선스를
받아야 하므로, 완전한 오픈소스는 아님

Adobe Digital Editions

전자책 콘텐츠 보호용 DRM 솔루션
EPUB 및 PDF 포맷 기반 콘텐츠 지원
콘텐츠 다운로드 및 복사 방지 기능 포함

02. 주제 선정 배경

- 기존 DRM 구조의 한계

  국내 상용 DRM은 대부분 중앙 서버 기반의 라이선스 관리 구조를 채택하고 있으며,

  콘텐츠 접근 시 정책 판단과 키 분배를 서버 인증을 통해 수행하고 라이선스를 **“임시캐싱”**하고있음.

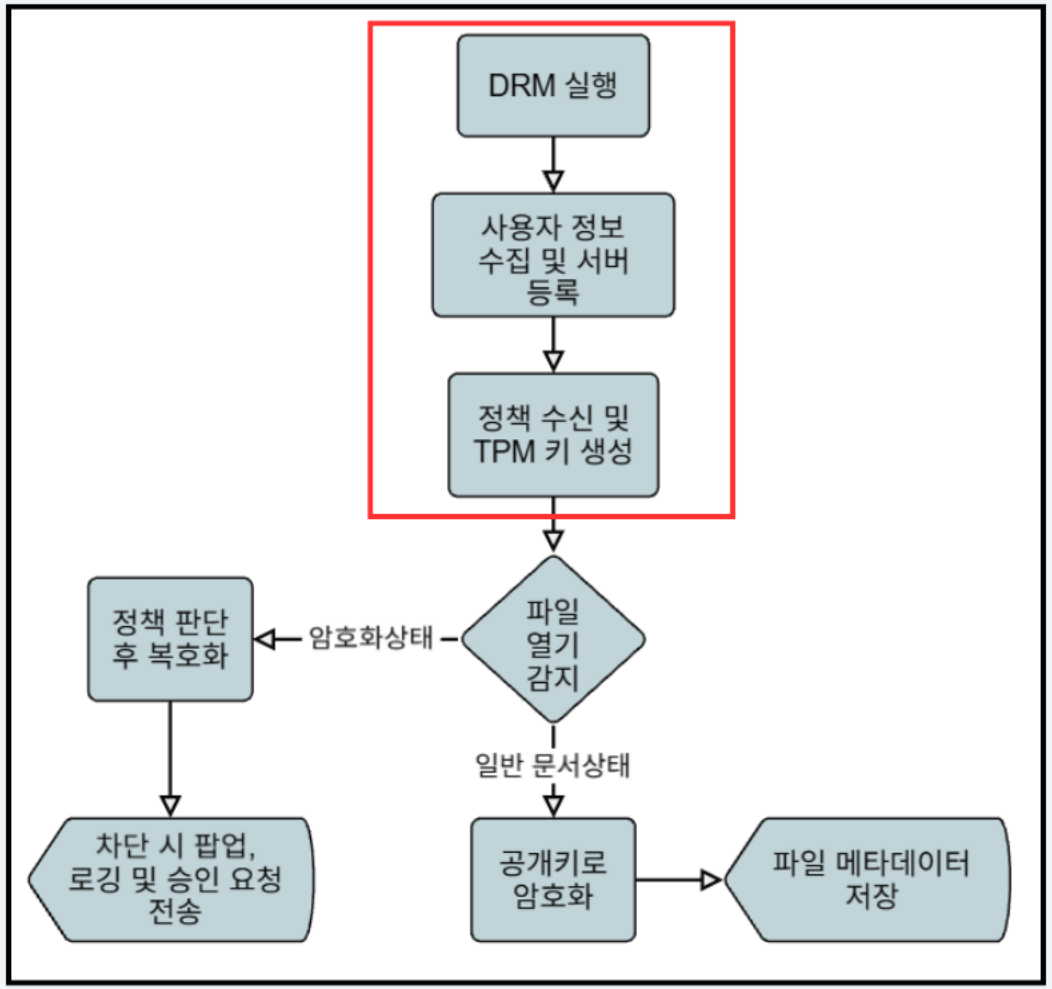
  이 구조는 정상적인 네트워크 환경에서는 효과적이지만,

  - 서버 장애
  - 오프라인 환경

  위와 같은 상황에서는 업무 연속성 측면에서 한계를 초래할 수 있음

→ SQLite 기반 정책 캐시 + TPM 기반 키 캐싱 구조로 동작하는 DRM을 제안

03. 동작 구조 초기 등록 및 스캐너 동작



1. DRM 실행

- DRM 클라이언트가 실행되며 자동으로 초기 스캐너가 동작

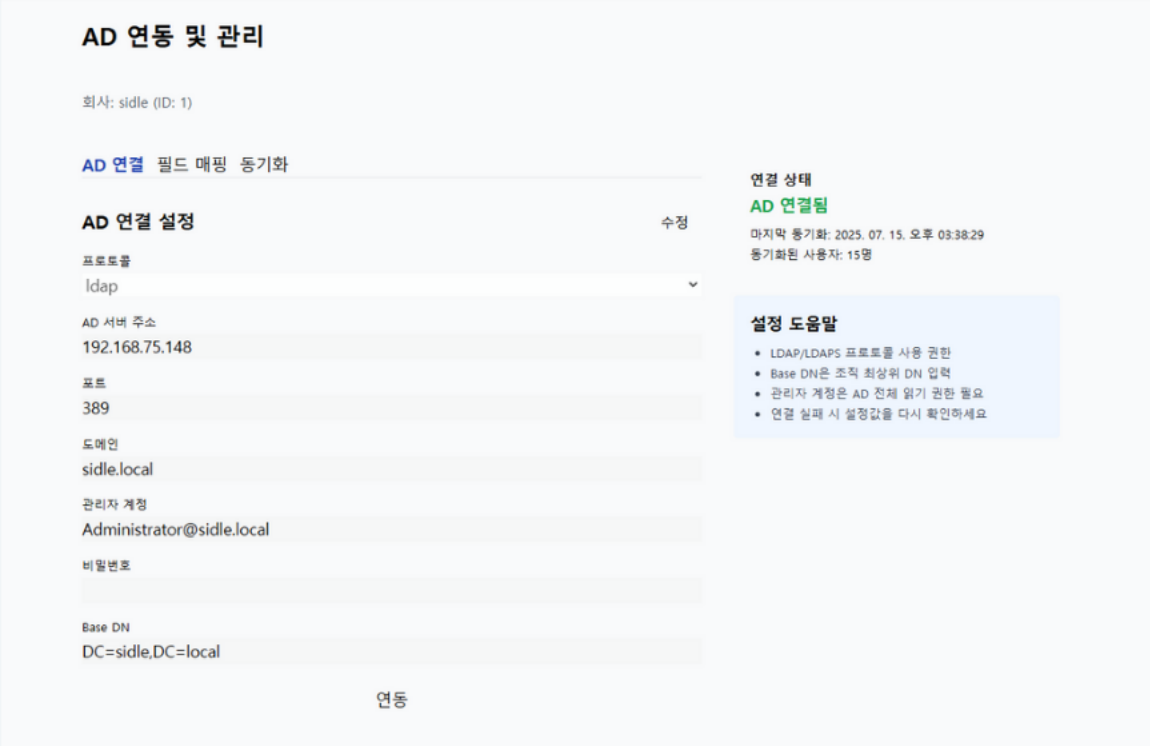
2. 사용자 정보 수집 및 서버 등록

- 계정 정보, 도메인을 수집해 서버에 질의
- 등록된 정보를 바탕으로 사용자에 맞는 정책 적용을 준비

3. 정책 수신 및 TPM 키 생성

- 서버로부터 정책을 수신하고 AD정보 및 직급별 Rank를 로컬 SQLite에 캐싱함
- TPM(Trusted Platform Module)을 이용해 사용자 전용 키 생성
- 이후 DLL 인젝션 및 파일 감지 루틴으로 진입

03. 동작 구조 초기 등록 및 스캐너 동작



1. DRM 실행

- DRM 클라이언트가 실행되며 자동으로 초기 스캐너가 동작

2. 사용자 정보 수집 및 서버 등록

- 계정 정보, 도메인을 수집해 서버에 질의
- 등록된 정보를 바탕으로 사용자에 맞는 정책 적용을 준비

3. 정책 수신 및 TPM 키 생성

- 서버로부터 정책을 수신하고 AD정보 및 직급별 Rank를 로컬 SQLite에 캐싱함
- TPM(Trusted Platform Module)을 이용해 사용자 전용 키 생성
- 이후 DLL 인젝션 및 파일 감지 루틴으로 진입

03. 동작 구조 초기 등록 및 스캐너 동작


```
{
  "user": {
    "user_id": 5,
    "user_name": "bonam",
    "guid": "92e347a1-a6b2-4166-916d-80de3cb6856c",
    "rank": 3,
    "role": "일반사용자",
    "department": "개발팀"
  },
  "default_policy_mode": "rank_only",
  "exception_policies": [
    {
      "file_extension": ".log",
      "department": "개발팀",
      "action": "skip_encryption"
    },
    {
      "role": "총관리자",
      "action": "no_size_limit"
    }
  ],
  "group_policies": {
    "개발팀": ["encrypt","audit"],
    "보안팀": ["audit","alert"]
  }
}
```


1. DRM 실행

- DRM 클라이언트가 실행되며 자동으로 초기 스캐너가 동작

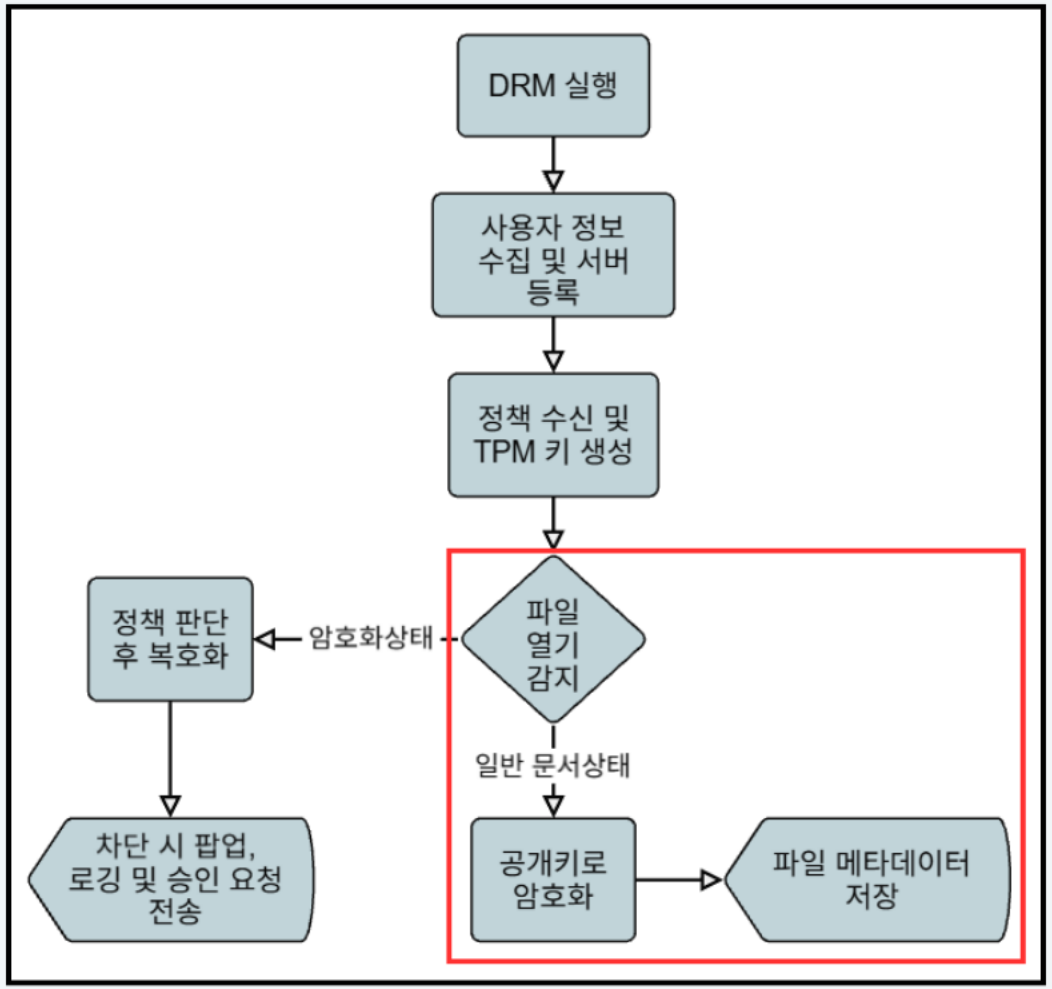
2. 사용자 정보 수집 및 서버 등록

- 계정 정보, 도메인을 수집해 서버에 질의
- 등록된 정보를 바탕으로 사용자에 맞는 정책 적용을 준비

3. 정책 수신 및 TPM 키 생성

- 서버로부터 정책을 수신하고 AD정보 및 직급별 Rank를 로컬 SQLite에 캐싱함
- TPM(Trusted Platform Module)을 이용해 사용자 전용 키 생성
- 이후 DLL 인젝션 및 파일 감지 루틴으로 진입

03. 동작 구조 파일 암호화 동작



1. 파일 생성 감지 및 DLL 인젝션

- explorer.exe에 부모 DLL 인젝션
- 이후 자식 프로세스가 생성되면 CreateFileW 후킹해 파일 생성 이벤트를 실시간으로 감시

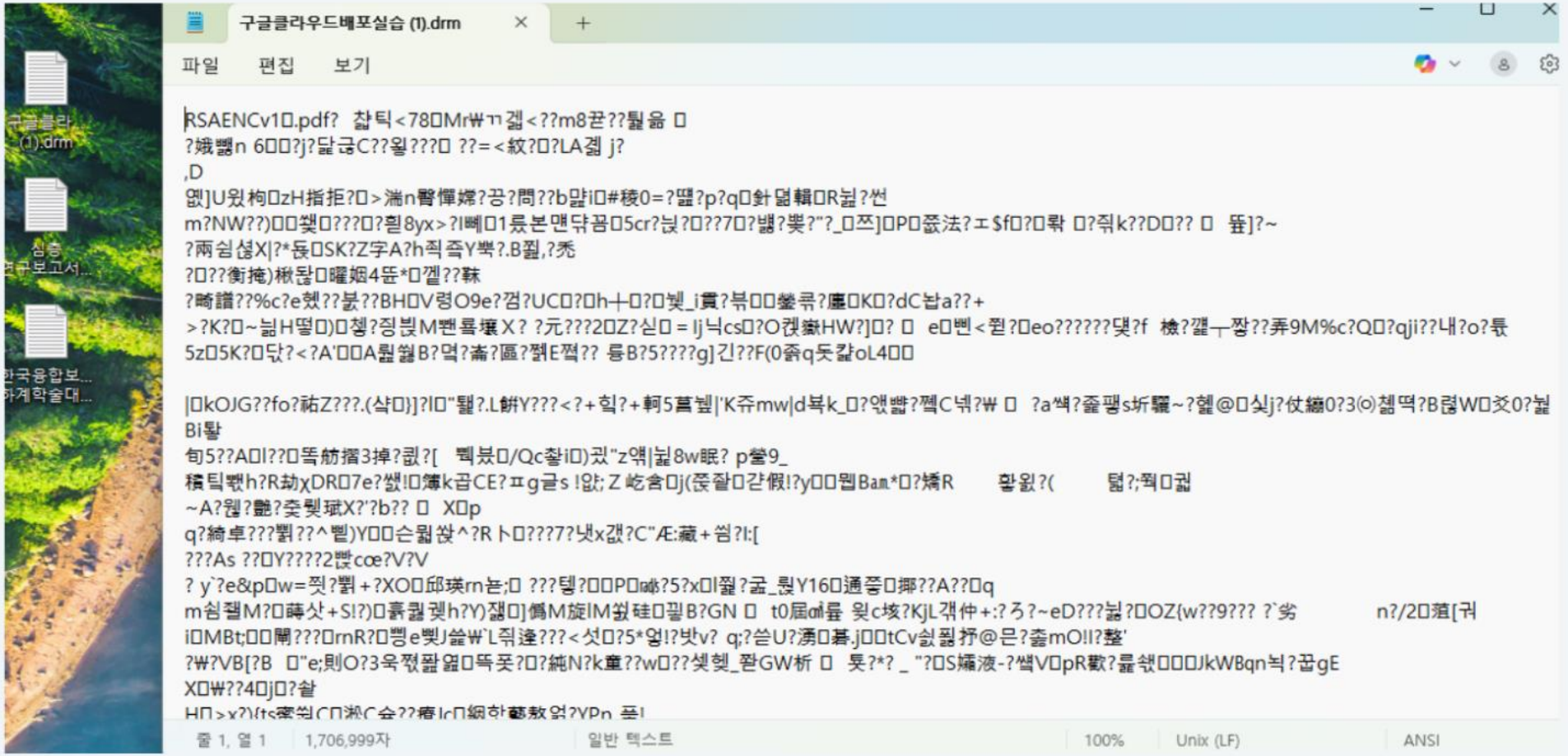
2. 타겟 확장자 필터링 및 암호화 수행

- 타겟 확장자 파일 생성 시 RSA 공개키로 청크 단위 암호화
- 암호화된 파일은 .drm 확장자로 저장됨

3. 파일 헤더 및 메타데이터 처리

- 헤더 구성: 파일 시그니처 | 청크 수 | 파일 해시
- 메타데이터: 사용자 OU, Rank 기준으로 SQLite에 저장됨

03. 동작 구조 파일 암호화 동작



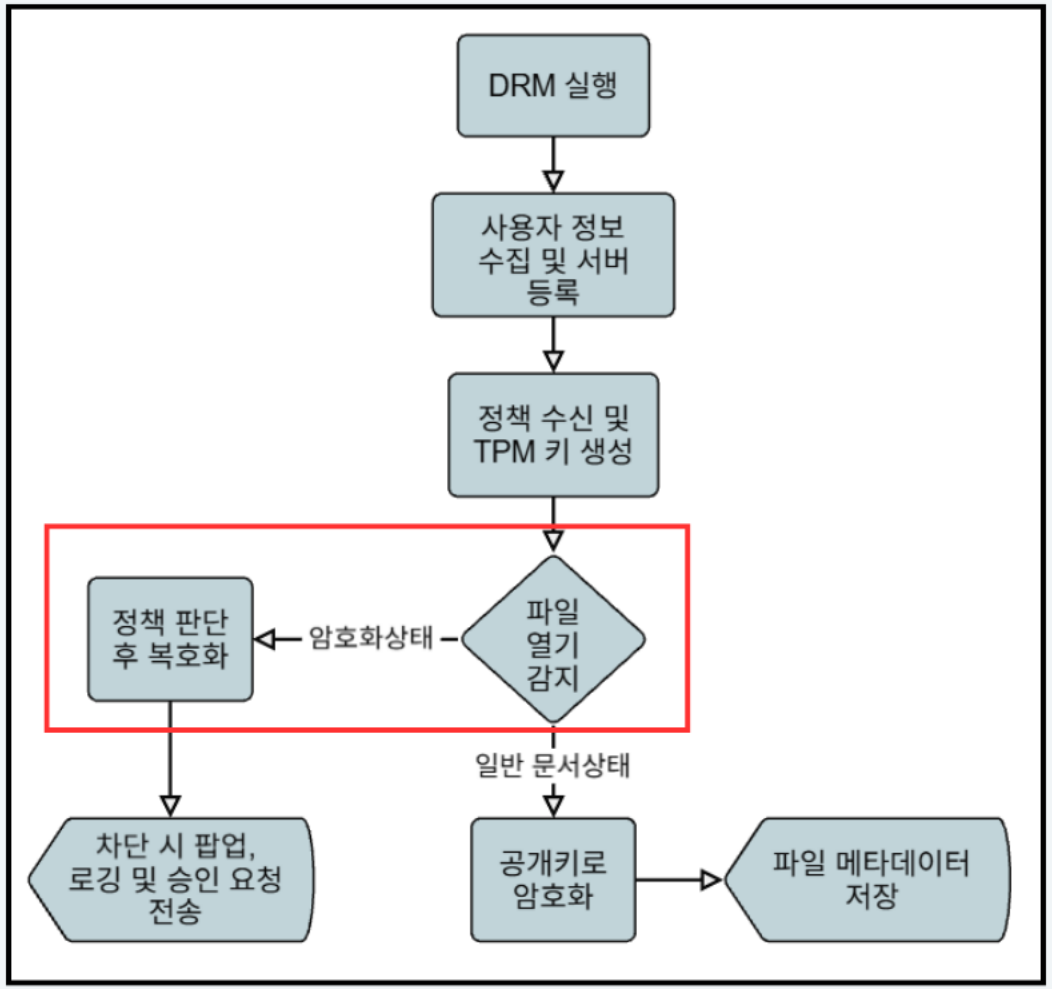
03. 동작 구조 파일 암호화 동작

암호화 후 헤더 구조

오프셋	내용	크기	설명	예시
0x00	"RSAENCv1"	8 bytes	시그니처	고정값
0x08	확장자 길이	1 byte	확장자 길이	4 (".pdf")
0x09	확장자 문자열	ext_len	확장자	"pdf"
0x09+x	청크 수	4 bytes	암호화 단위	
0x0D+x	파일 해시	32 bytes	file_hash	
...	데이터 본문	가변	암호화	

```
sqlite> SELECT * FROM filemetadata;
file_hash                                                        file_ext    user_ou     user_rank
---------------------------------------------------------------  ----------  ----------  ---------
CACA94821612EC998A5A86AE1172D7B06766F768F6539124DA
027878F787B108                                                   .pdf        보안팀      5

C8C53F2443C0FF32BBBB7FAFC9DF25C2B149EE90FAAD36DC48
4AFAC20C67A13B                                                   .hwp        보안팀      5

2CD912942D82FA27BC54AF08E0A0F98657480F64A5633F8B8F
3312945D91604C                                                   .pdf        보안팀      5

DC087222C07E1277DA30DC1202485A6A5220C2C1808FF8A4E0
FA455DA43C1E2E                                                   .hwp        보안팀      5

94F9F30BDA37E4B667E803C96EAE750C9441985D764AE255DB
E6A681AC6B8860                                                   .txt        보안팀      5

9A3038EBC86F587A0D393610920AEC0A111D29657AE630BB98
AC47ED47EFB762                                                   .txt        보안팀      5

F5800F0B471928B0299E9C33BD7452F0E5FE12FF98650F3A5E
AF52C3F11DAEC4                                                   .txt        보안팀      5
```

03. 동작 구조 복호화 흐름



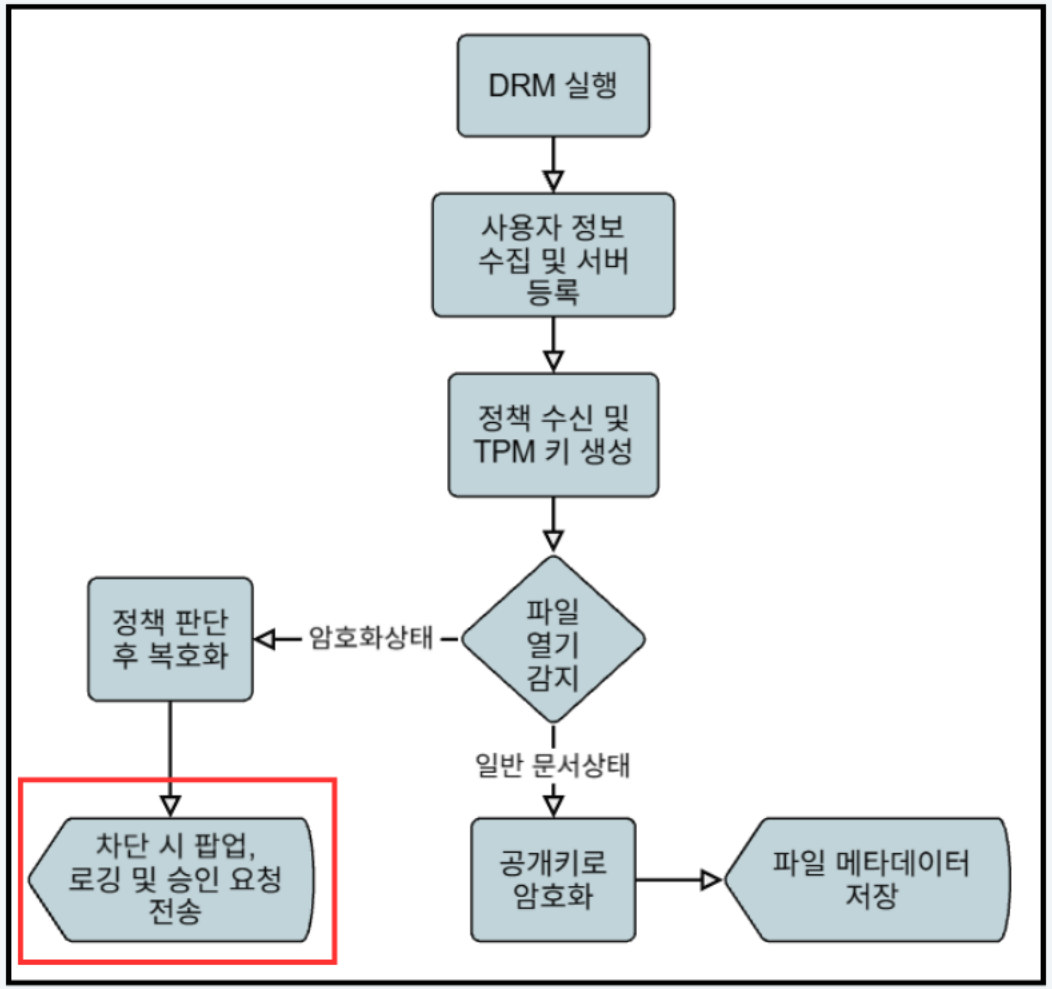
1. 파일 열기 감지 및 정책 판단

- 사용자가 암호화된 파일을 열려고 하면 후킹된 DLL이 CreateFileW를 통해 접근 시도를 감지
- 정책 캐시 확인 (SQLite) → 접근 가능 여부를 판단

2. 임시 복호화 파일 생성

- 접근이 허용되면 TPM을 이용해 임시 파일로 복호화
- 파일을 닫으면 임시 파일 자동 삭제

03. 동작 구조 승인 요청 흐름



1. 차단 시 팝업 및 승인 요청 전송

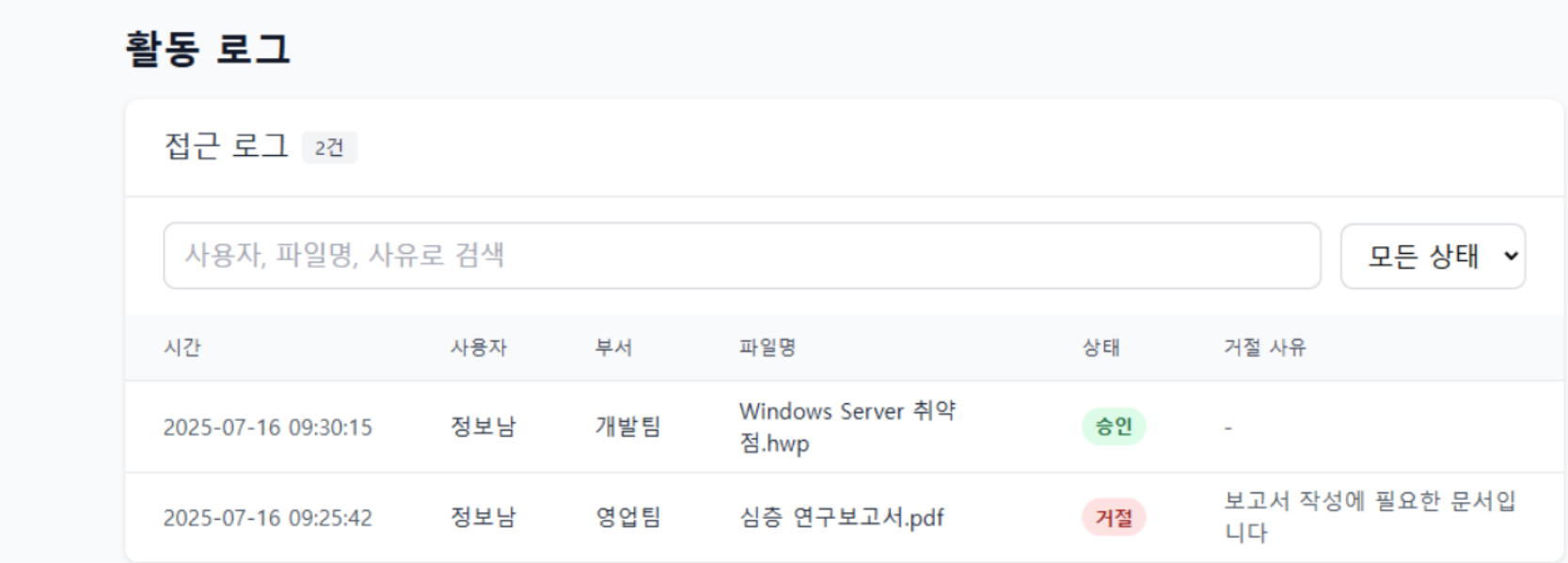
- PyQt 팝업으로 차단 사유 안내
- 사용자가 사유 입력 시 승인 요청 전송

  → 부서 관리자에게 로그 및 요청 전달됨

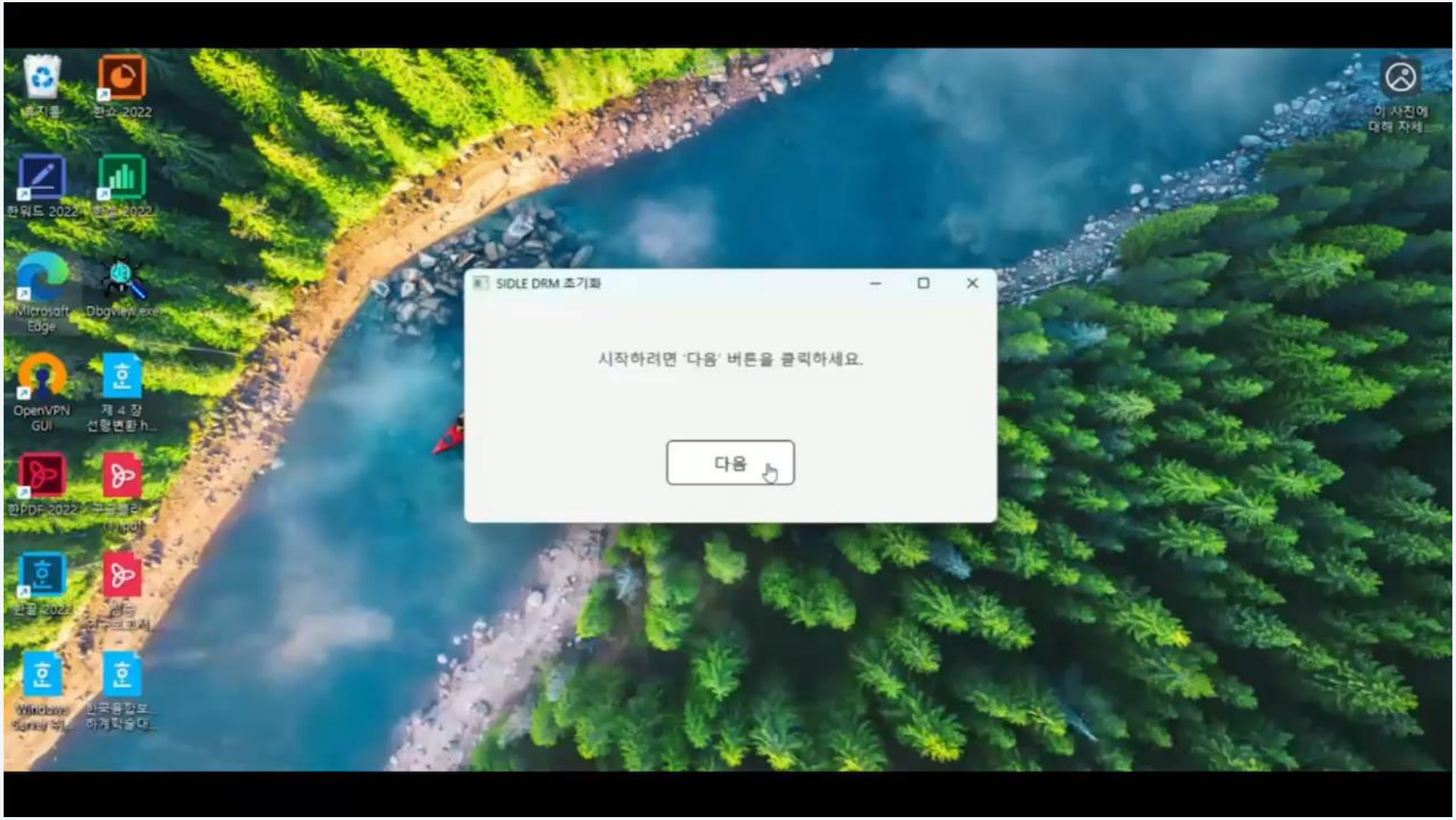
2. 관리자가 사유 확인 후 승인 OR 거절

- 관리자는 사용자가 요청한 내용을 확인 후 승인 OR 거절
- 승인 시 사용자에게 파일 복호화 권한 제공
- 거절 시 거절 사유와 함께 파일 접근 차단

03. 동작 구조 승인 요청 흐름



04. 영상 시연



05. 결론 프로젝트 성과

01

정책, 키 분리 저장

정책은 SQLite에, 키는 TPM에 분리 저장되어 정책만 독립적으로 갱신 가능

02

서버 의존성 감소

클라이언트 단독으로도 DRM 기능 수행 가능, 서버 장애 시에도 지속 사용 가능

03

기대 효과

실제 운영 환경에서 업무 연속성 강화에 기여

04

얻은점

AD 서버와의 통신 기능을 구현하기 위해 직접 서버를 구축하고 연동을 실습하며 LDAP 구조에 대해 이해

05. 결론 아쉬운점 및 향후 계획

01

아쉬운 점

- 마스터키 문제를 해결하지 못함
- IOS 도입 실패
- KMS 도입 실패 (시간 부족)
- 시스템 파일과 일반 파일 구분이 잘 안돼서 일부 부분은 DRM이 제대로 동작하지 않음

02

향후 계획

- 오픈소스 기반으로 핵심 기능은 구현 완료
- 아쉬웠던 부분 구현 계획
- 향후 다양한 기능 확장할 계획

THANK YOU

감사합니다.