PETRA
CIPHER V3.1

DB SECURITY SOLUTION

[파일 암호화 옵션]

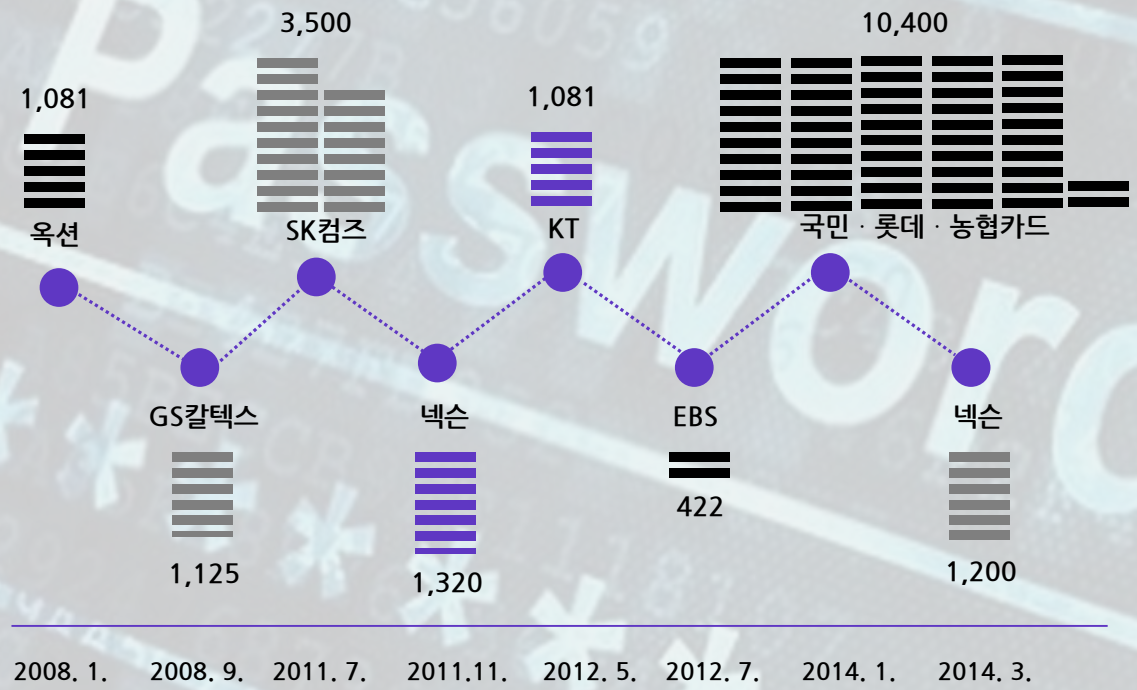
Table of Contents

암호화와 법률준수

개인정보보호법은 공공, 금융, 의료 등 각 분야별로 적용되던 규범을 일원화하였고,
이에 따라 개인정보를 취급하는 모든 기업 및 단체는 정보유출에 대한 기술적인 안전조치가 필요

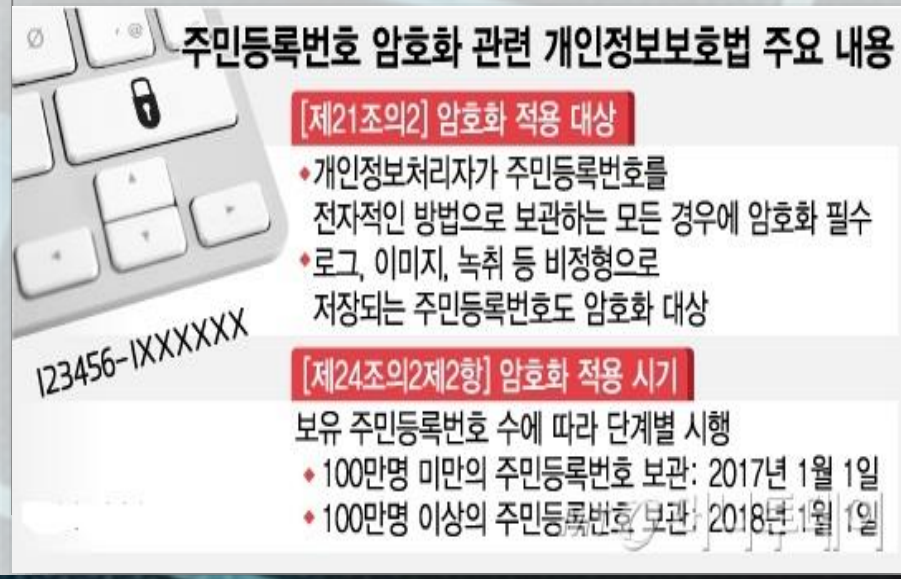
개인정보 유출 규모

[단위: 만 건]



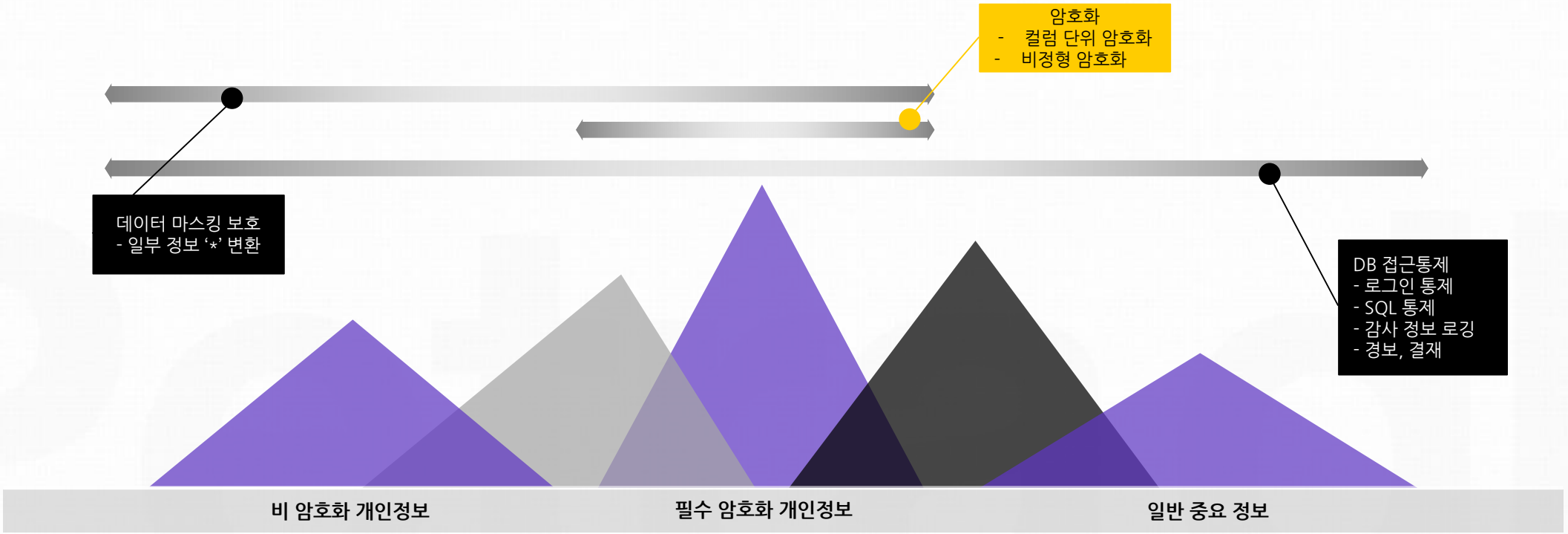
개인정보 보호법

제24조의 2항에 의거 주민등록번호에 대한
암호화 조치 의무화



정보에 따른 솔루션 배치

개인정보보호법, 정보통신망법, 위치정보보호법 등에서 규정한 필수 암호화 컬럼에 대한 암호화 조치 필수
데이터 유출을 통한 기업의 경제적 손실과 이미지 실추를 미연에 방지



2 이름, 주소, 핸드폰번호, 이메일, 병력, 학력, 가족관계, 종교 정보 등

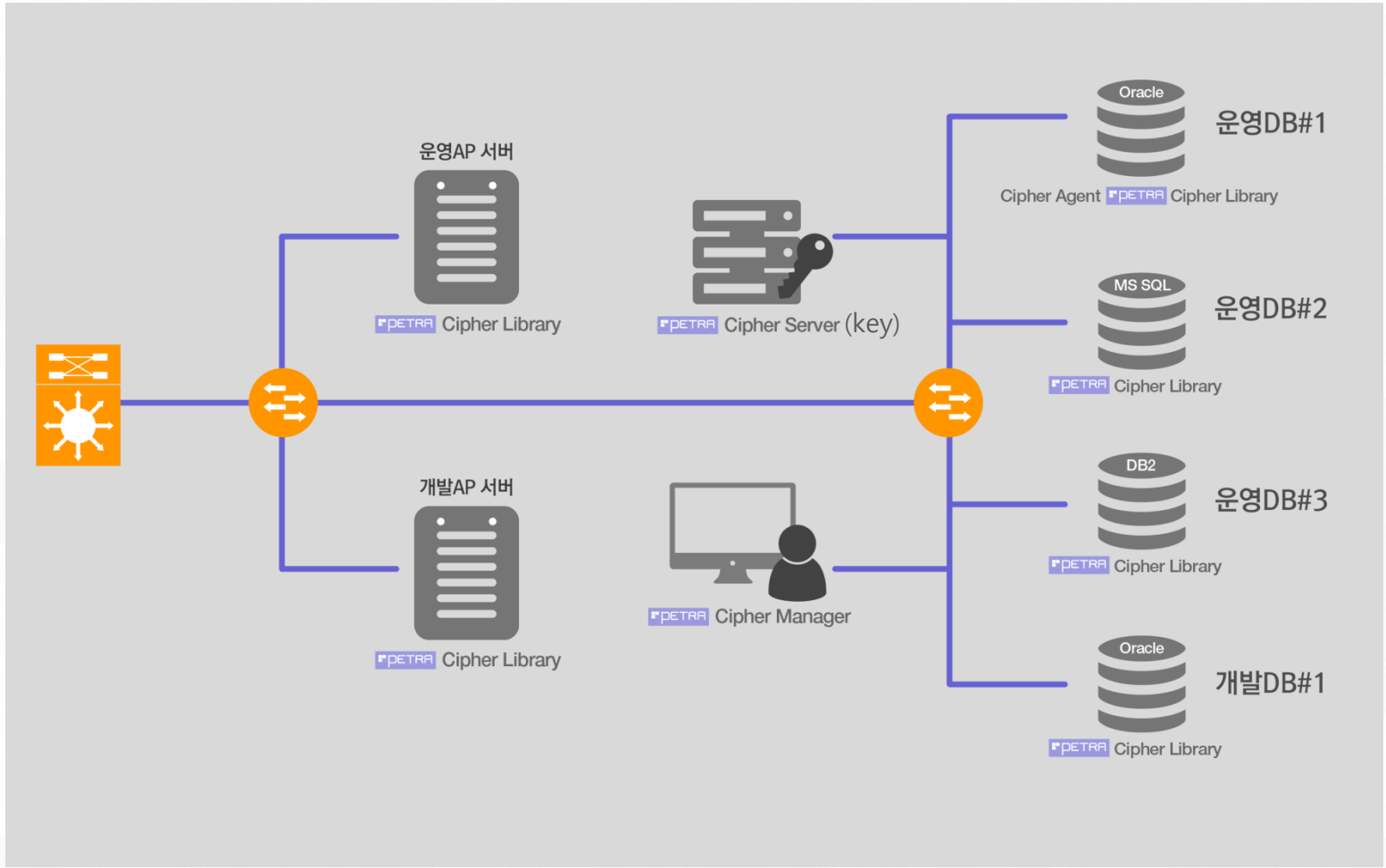
1 고유식별정보(주민번호 · 운전면허번호 · 여권번호 · 외국인등록번호), 비밀번호, 카드번호, 계좌번호, 바이오 정보

3 구매, 급여, 고과, 매출, 경영평가, 회계, 재무, 생산통계 정보 등

Table of Contents

국정원 요건에 충족하는 알고리즘

고성능 아키텍처: GUI 기반 관리콘솔을 통해 키 정보 및 규칙 정보를 효율적으로 관리, 성능을 극대화 할 수 있는 구조
키 기밀성 보장: DB 서버의 Disk에 키를 저장하지 않고, 개발자에게 데이터 키가 누출되지 않아 국정원 요건 100% 충족



Cipher Manager

- 키 생성, 초기데이터 암호화 및 컬럼에 대한 권한 관리
- 로그 조회 및 보고서 생성

Cipher Server (Key)

- 키 정보, 권한 정보 저장
- 암호화 컬럼에 대한 조회 로그 저장
- 이중화 구성되어 안정적인 운영

Cipher Agent

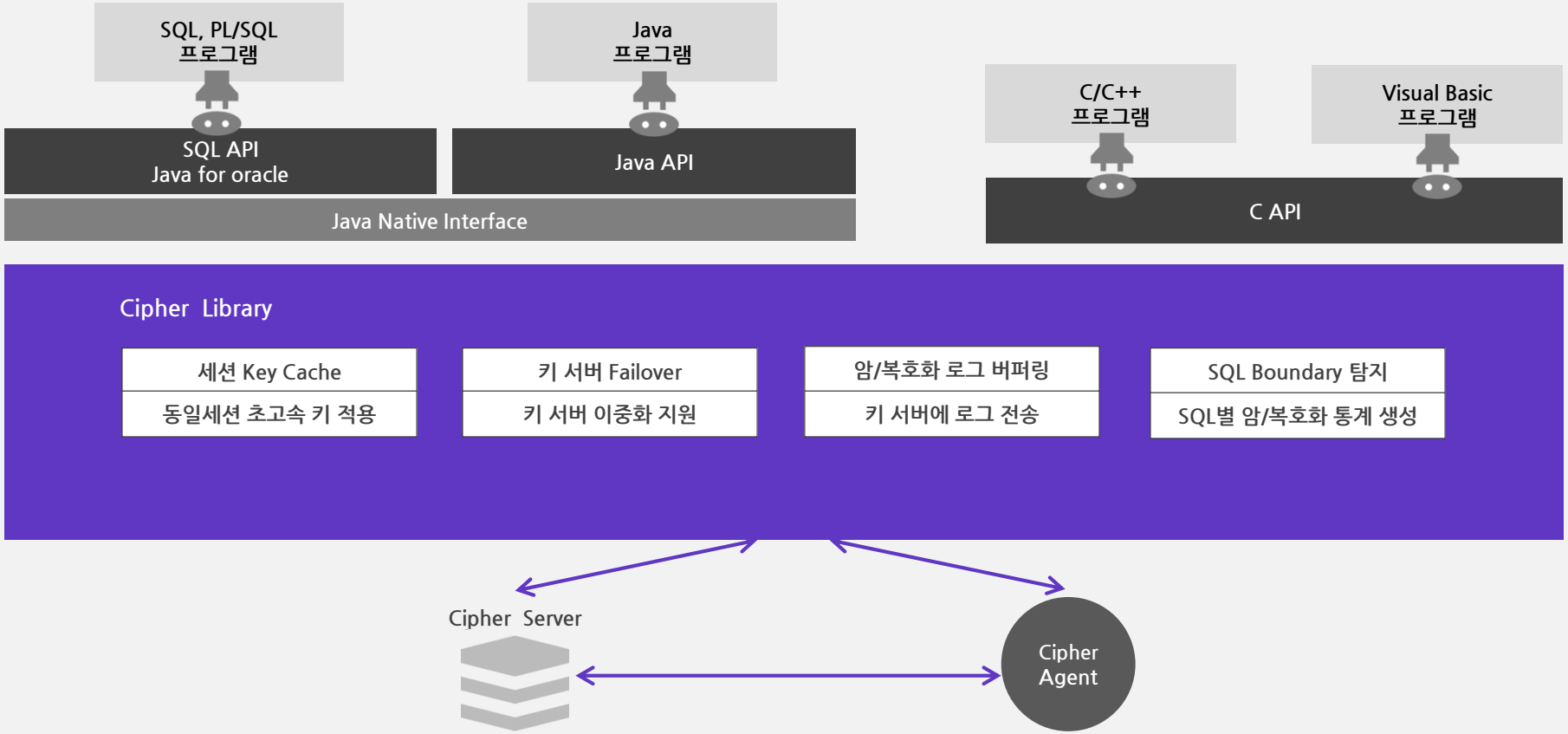
- Cipher Lib로 부터 키 요청 처리
- 키와 정책에 대한 Cache 기능 제공
- 로그 수집 기능

Cipher Library

- 암/복호화 기능 수행
- App Server를 위한 API 제공
- 국정원 암호모듈검증모듈 및 키 기밀성 요구사항 충족

고성능, 고기능 API 제공

암/복호화를 위한 API 외에도 세션 설정, 로그 저장 등을 위한 API를 별도로 제공하여 암호화 후 효율적인 관리가 가능



지원 알고리즘

양방향 알고리즘

보안강도	블록암호 알고리즘	KISA 권고 여부	지원 여부
80비트 미만	DES	권고하지 않음	X
80 비트	2TDEA		
112 비트	3TDEA		
128 비트	SEED, ARIA-128, AES-128	권고함	O
192 비트	ARIA-192, AES-192		
256 비트	ARIA-256, AES-256		

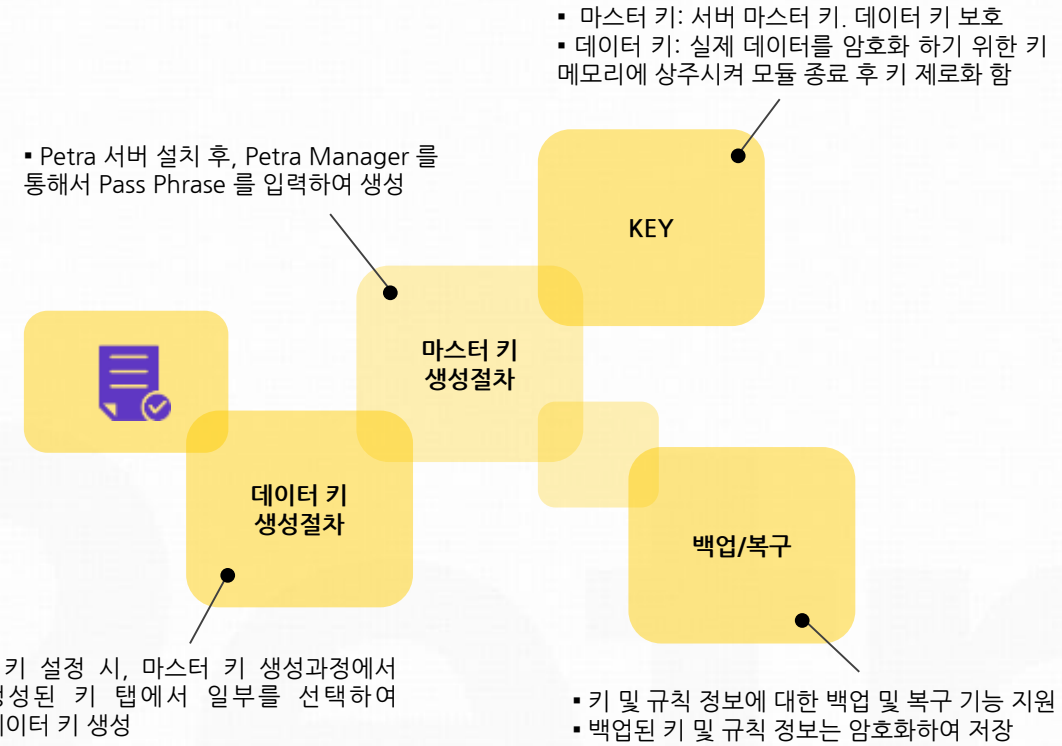
해시(Hash) 알고리즘

보안강도	블록암호 알고리즘	KISA 권고 여부	지원 여부
80비트 미만	MD5, SHA-1	권고하지 않음	X
80 비트	HAS-160		
112 비트	SHA-224	권고함	O
128 비트	SHA-256		
192 비트	SHA-384		
256 비트	SHA-512		

국정원 "DB암호화 제품 보안요구사항"

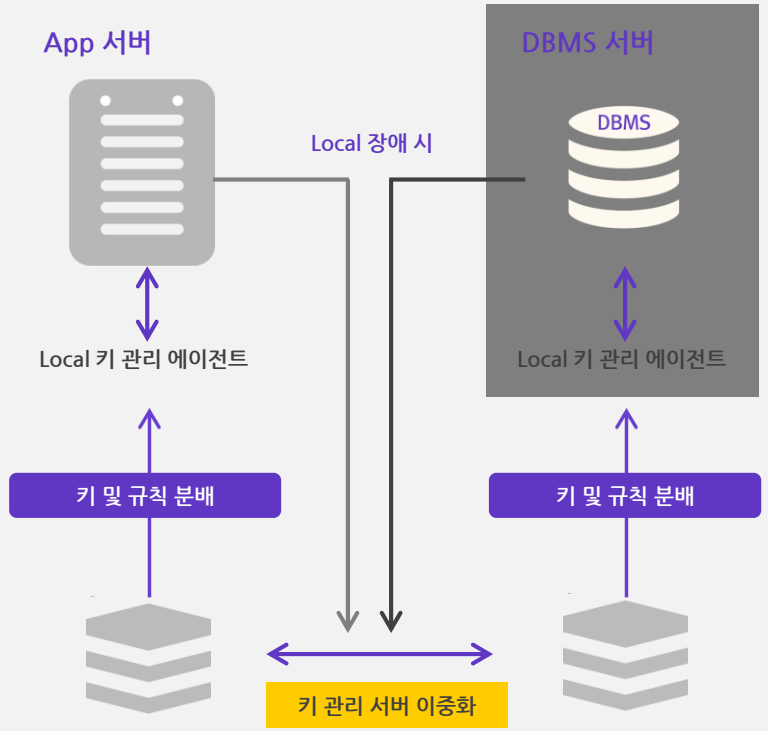
구분	요구 사항	충족여부	구현 방안
암호 지원	안전성이 검증된 암호모듈·알고리즘 등 사용	O	▪ 대칭키: 128/192/256의 ARIA/AES/SEED(128) 지원 ▪ 해쉬: SHA-256/384/512 지원
암호키 관리	암호키 생성·접근·갱신·파기 등의 안전성 확보	O	▪ NIST SP 800-132 PBKDF(Password Based Key Derivation Function)를 사용하여 생성함 ▪ 리부팅 시 암호키와 매개변수는 제로화 되어 대상 서버 상에는 더 이상 존재하지 않음
DB데이터 암/복호화	암호문·등 중요 데이터의 안전성 확보	O	▪ 국정원 검증 암호모듈을 사용하여 안전성이 확인된 암호모듈을 사용하여 데이터의 암·복호화를 수행
접근통제	암호키·암호문 등에 대한 비인가자의 접근 차단	O	▪ 사용자 ID(인증기능 사용시), 사용자 IP, 접속툴, 접속 DB, 접속 계정 별로 암호화 컬럼별 복호화 권한을 통제
암호통신	전송 데이터의 기밀성·무결성 유지	O	▪ SSL의 후속버전인 TLS 기반의 암호화 통신을 제공하여 데이터 전송 시 기밀성과 무결성 제공
식별 및 인증	인가된 사용자의 신원 확인 및 검증	O	▪ 인증데이터 재 사용시 인증 불가하여 공격 방지
보안감사	제품 관련 중요 이벤트에 대한 감사 기록	O	▪ 사용자의 IP, OS계정, 사용한 암복호화 키 등에 대한 상세한 정보를 기록
보안관리	보안정책·감사기록 등의 효율적인 관리	O	▪ Petra Cipher Manger를 통해 암호키 및 보안정책에 대한 백업 및 복구 기능 제공

안전한 키 관리



통제기술 -1. 안전한 키 생성 및 관리

- 안전한 방식으로 초기 키 생성
- 생성된 데이터 키는 마스터 키에 의해 암호화 하여 저장되어 유출 방지
- 단일 키를 이용하여 다양한 DBMS에 적용 가능
- 마스터 키의 패스워드는 주기적으로 변경하여 보안성 강화
- 데이터 키에 대한 직접적인 접근은 근본적으로 불가능하여 유출 불가



통제기술 -2. 키 관리 서버 이중화 및 로컬 관리

- 키 서버 이중화로, 한 개의 서버 장애 시에도 나머지 다른 서버를 통해 서비스 유지
- DBMS서버, APP 서버 내부에 로컬 키 에이전트를 구성할 수 있도록 하여 안정성 강화 (키 정보 및 규칙 정도 동기화 지원)
- 세션 메모리 내부에 키 정보 및 규칙 정보를 저장하여 모든 키 서버 장애 시에도 서비스 유지

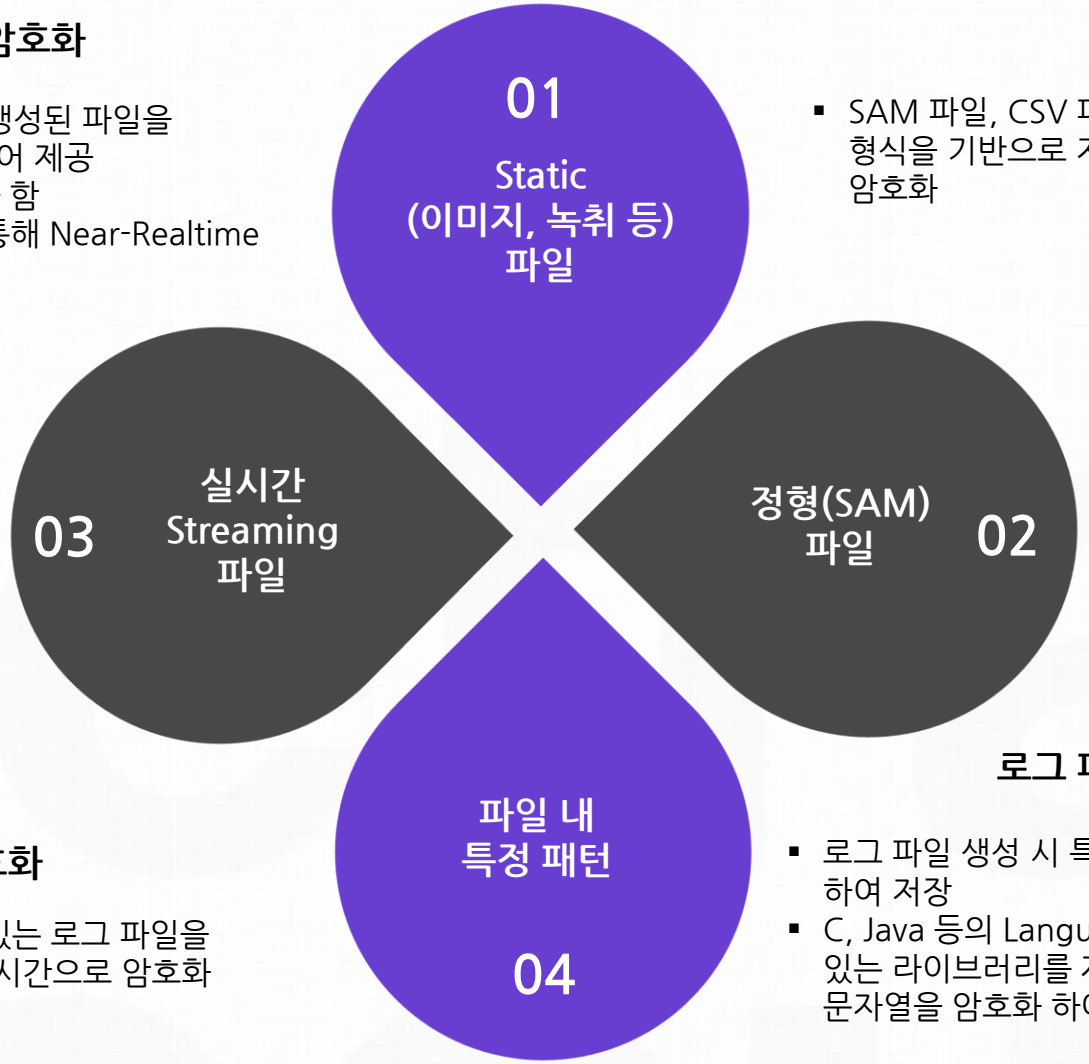
Table of Contents

주요 기능 및 특징

파일 단위 암호화

이미지, 녹취 파일 등 암호화

- 이미지, 녹취 파일 등의 생성된 파일을 암호화 할 수 있는 OS 명령어 제공
- 파일 전체 내용을 암호화 함
- 서버 내 Agent 설치를 통해 Near-Realtime 암호화



SAM 파일 암호화

- SAM 파일, CSV 파일 등 파일 내 일정한 형식을 기반으로 저장된 파일의 특정 부위 암호화

실시간 Streaming 암호화

- WAS 등에서 사용되고 있는 로그 파일을 Streaming 설정하여 실시간으로 암호화

로그 파일 특정패턴 암호화

- 로그 파일 생성 시 특정 패턴 정보만 암호화 하여 저장
- C, Java 등의 Language에서 사용할 수 있는 라이브러리를 제공하여 개인정보 관련 문자열을 암호화 하여 저장할 수 있도록 함

주요 특징

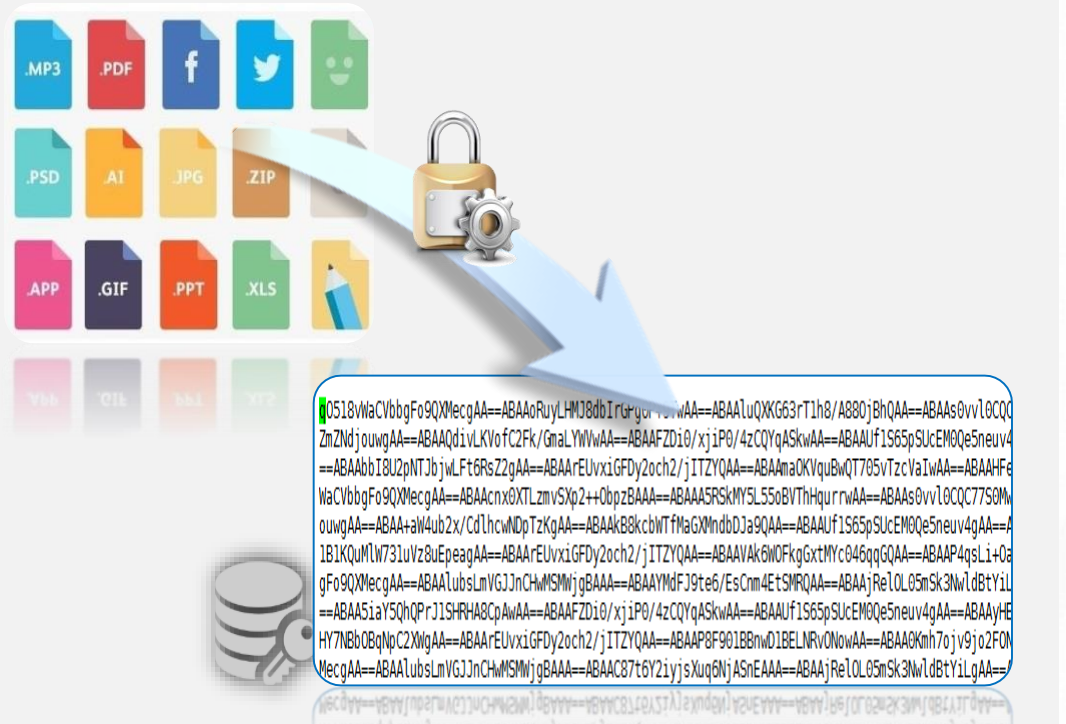
DB암호화와 파일암호화의 완벽한 통합

- **손쉬운 설치와 유지 관리 (OS 커널 등에 대한 조작 없음)**
- **API, OS Command, Agent 등 다양한 방식 제공**
- **암호화 된 파일에 대해 관리자에 대한 통제 가능 (파일 유출 시에도 안정성 확보)**
- **Multi-Thread 아키텍처 기반의 고성능 암/복호화 기능**
- **암호화 결과를 Binary, Base 64 등 으로 저장**

파일 암호화 대상 및 방식

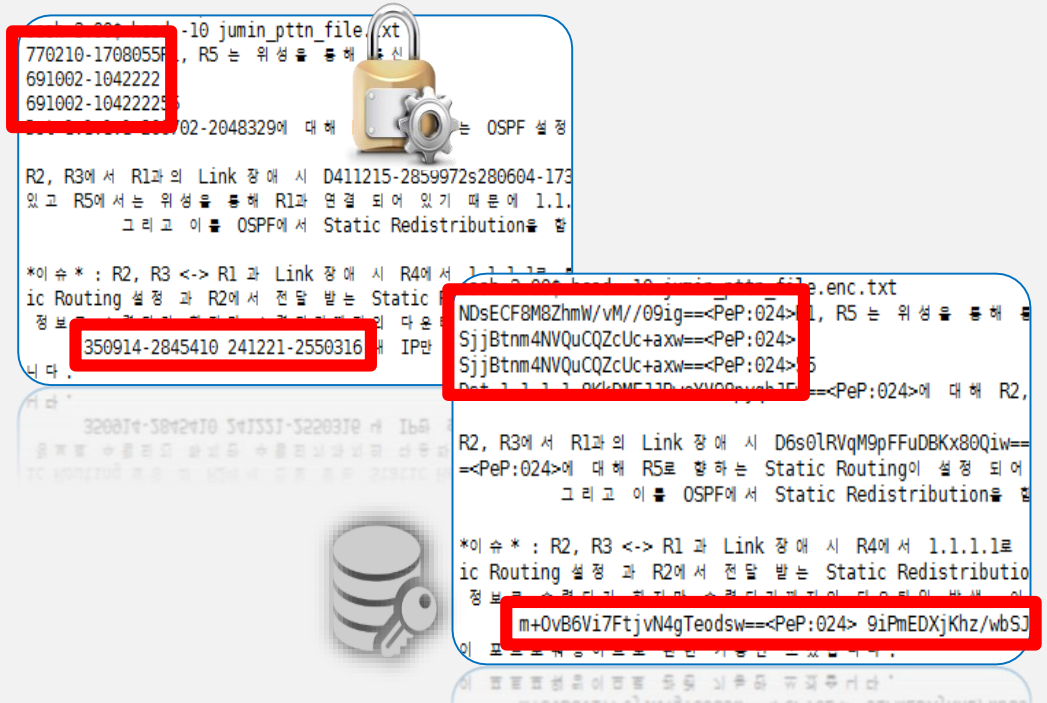
구분	설명
파일 종류	• 문서 파일(워드, 한글, PPT, PDF 등) • 이미지 파일(지문, 여권, 신분증 스캔 파일 등) • 동영상, 음성 파일(녹취, 인터뷰 등) • 텍스트 파일(시스템 로그(Log), XML, HTML 등)
파일 상태	• 고정(Static) : 현재 변경 중이지 않은 파일, 저장된 문서 또는 이미지 같이 생성을 완료한 상태 • 유동(Dynamic) : 현재 내용이 추가되고 있는 파일, 로그 파일과 같이 실시간 데이터를 기록 중인 상태
파일 내 암호화 범위	• 전체 암호화 : 파일 단위로 내용 전체를 암호화(모든 종류의 파일 지원) • 부분(패턴) 암호화 : 파일 내 특정 패턴의 데이터를 인식하여 암호화(텍스트 파일 지원)
인터페이스	• API(파일 또는 문자열 단위 암호화) • Command(명령어 기반 파일 암호화) • Stream(실시간 로그 암호화)

파일 내 암호화 범위



전체 암호화 방식 설명

- 파일 전체 암호화
- 파일 내용 확인 시 전체 복호화
- 암호화 결과는 binary 또는 Base64 Encoding 형태로 저장

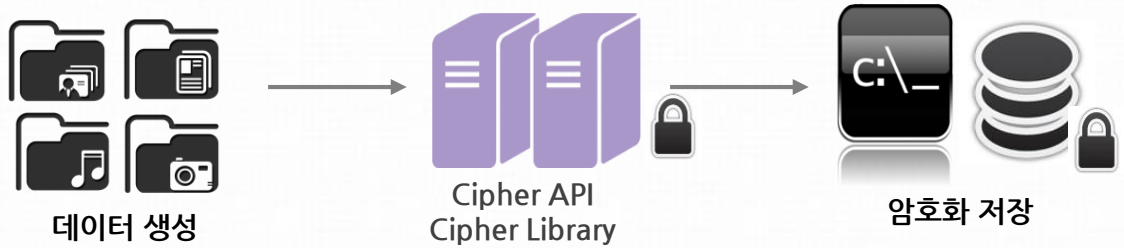


부분 암호화 방식 설명

- 파일 내 특정 패턴(예 : 주민등록번호)을 데이터를 인식하여 암호화
- 파일 내용 확인 시 암호화 한 데이터만 복호화
- 암호화 결과는 Base64 Encoding 형태로 저장
- Dynamic상태(실시간 로그 파일) 파일에도 적용

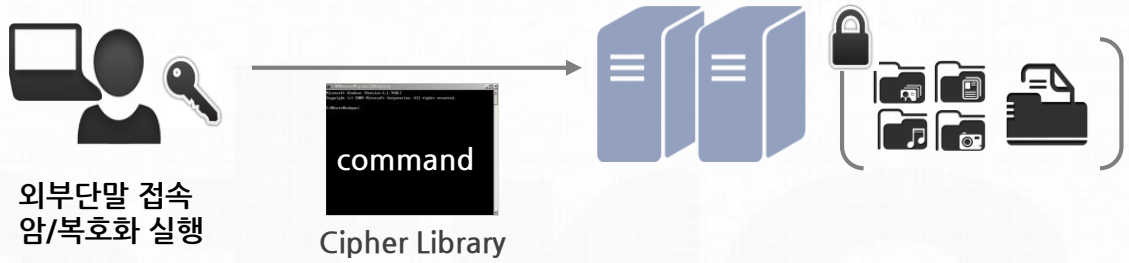
암호화 인터페이스

API 방식



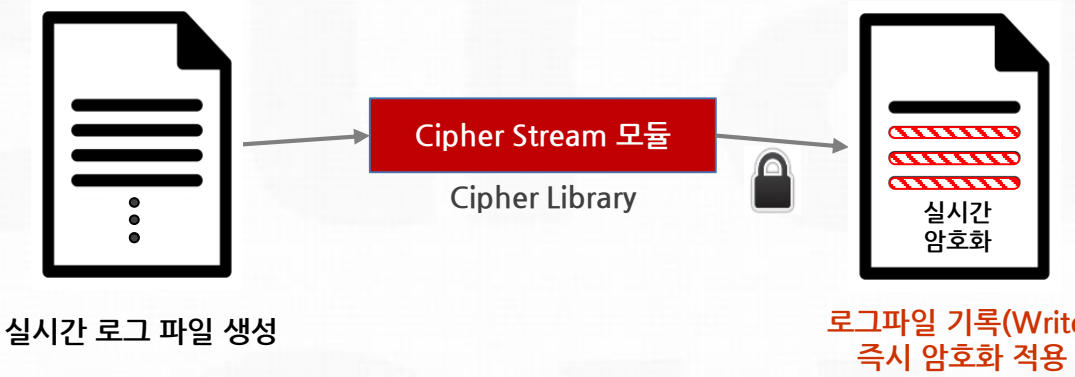
C, Java 등 프로그램 내
파일 암/복호화 API 제공

Command 방식



명령어 수행을 통한 Static
상태의 파일 암/복호화

Streaming 방식



Dynamic 상태 로그파일에
기록 중인 데이터를 16byte
단위로 실시간 암호화
(부분 암호화 지원)

파일 단위 암호화

- 파일의 종류, 내용에 상관없이 파일 단위로 암호화 제공
- 암호화 결과는 Binary 타입, Base64 인코딩 타입으로 선택하여 저장

[평문 파일]

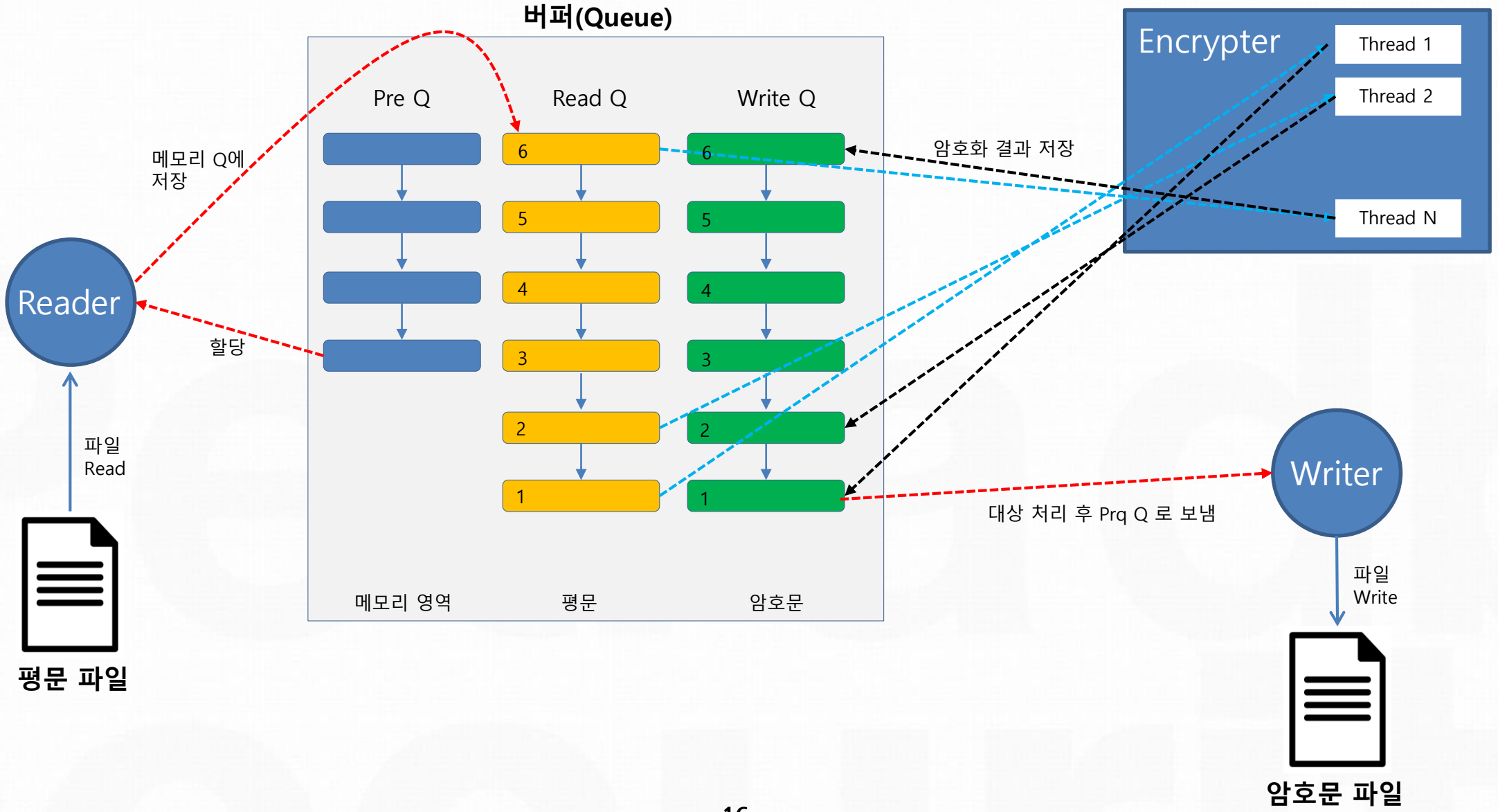
```
bash-3.00$ head -10 jumin_pttn_file.txt
770210-1708055R1, R5 는 위성을 통해 통신
691002-1042222
691002-104222255
Dst 1.1.1.1 200702-2048329에 대해 R2, R3 에서는 OSPF 설정에 따라 R1을 통해 통신 함

R2, R3에서 R1과의 Link 장애 시 D411215-2859972s280604-1733351t 1.1.1.1 670105-1077372에 대해 R5로 향하는 Static Routing이 설정 되어
있고 R5에서는 위성을 통해 R1과 연결 되어 있기 때문에 1.1.1.1로 통신 가능,
          그리고 이를 OSPF에서 Static Redistribution을 함

*이슈* : R2, R3 <-> R1 과 Link 장애 시 R4에서 1.1.1.1로 통신 하기 위해 Next-hop 인 R3로 패킷을 전달, R3에서는 1.1.1.1 로 향하는 Stat
ic Routing 설정 과 R2에서 전달 받는 Static Redistribution 정보가 Ping Pong 되는 이슈가 발생 약간의 시간이 지난 후에는 Static Routing
 정보로 수렴되긴 하지만 수렴되기까지의 다운타임 발생, 이러한 이슈에 대한 원인 및 해결방안을 위한 테스트
      350914-2845410 241221-2550316 내 IP만 ssh 접속가능하게 - 같은 기본세팅을 쓸까 했는데 글이 포트포워딩이므로 관련 기능만 쓰겠습
니다.
```

Pcp_file_crypt_encrypt enc_key
Jumin_pttn_file_txt
Jumin_pttn_file_enc.txt

```
bash-3.00$ cat jumin_pttn_file.enc.txt
uR4FzdxIxn/o9c2QSDzpHODRq4bjYIxJBVkdVTG61tC6ph6a+IVY/w3yLW82zImRs8p6C36z82/PUh7adIfsQHYT4toHSBK4o3njhrlDA5IWRjW9KTZT5Nvq8A+Qdhf74xl4
8gk51DmOI8K4gVHi7ts5ikTxG09kCrlShchMQey0N8G2KSEuKlEUXa/iBY+8jSx2HBJfKYbwTA8kTwEEpQJI1QiTz/luJx3r4D5N+hJhRmxi+Ii4yAb9Mh6Ng48d2eqOOQuJ
e4uCYbGQa9WoRlN3XK7n/a2tJDh9OGnBOqVMJE6Sp2On8YRG8Dpw2tc5Ooq3XNKK3gIs6KK6xX0PdzLVJmuz1dsc6XTDNzOM9bAwdluHsnboCHy3+D8LTq4ZMb+dsaBBlQ1r
l0/v1w27DznA+D1knguQD+D7RxArMfiEBWu7QED+svkh/Oos0ABJMbf5cQmfrrRlX9EKd2Q1cq4NcFIq/SmrBh34BYeVGnmGUzSNRd0H2A0SAlJp+EOWJrUy+mxV1jSeIcrB
ISM4XZ5LDIQ1+FviEP0r6FdGans9etQxvce/C18kw9//ftphO/X60mM7BOT+vsHTw5cS+NkYJ60qDgHPfnE8DNxXG8LR28IzP0A2mXQZCwSG2Ndtf0r+DzP5xmr1KkMlcolO
EXuR0FDKWGPIxp7CxYv5HXh8eX5f/2FuVEly0khYkUTrN6wyU343mY7dDETwpGBmqKihGRTkykQ8eEs6xgQ2xuMxP9mz+KGid2nZ5+2ZyftmsIKAUz/Jv1eoqX7IHfeSit/p
J1egP2+E0x2vdjpjCylgKGtCEp1xAu2IIlWNRfHYT8WhrehiJVWWOVVXx+qc3x9qgm3cFnSU6zA8jiZzF2wxTNN4mjQ/bWBBTKxiq507HnvK+mKg5iUhRM/SYjk4YycRxP2U
M7LWW0cF7tPnDdDLPEkeV5DZCNq2UtllBFFGPhv9t/WpeexbG1WmmSVaxyEJ2IJsw8o1lP3b1cmmpZGxv25EX+u9ndJClARiVD3Oxz4ADeaatm05BE9ruUWXW0EmsNMhANjR
LoOsSy46ku0+aExlHkW8Ruq7KqXsHSgMtpvq06I0QRKPtmJRGsliTfJ2lE7lsDzXNlE+R2xyEgratoIBZfeaQJFoseoRPXEPd6A1KdLhauV+TZ8HfTSXw1IDVxkou2Gl3Q67
P4JwgyYa61+i5AoNh5pN05OPtapwkd2MXqC56Oez9EbEiCSDd8vP28/NndRDy3aEvFFeUJbN2mzW7A2dURYM7d6Xmhp82rPxN/nGSlRIeMlIc1jqqWy79k5HWh1poYx/xtRG
0Rjx+wt9ZvYoNJ5wM+w80S5ZMlJolYTzthL0SKMsRupuKqlhKft6FpWf3zvkz3xEYIXZMstBm1oh0ShGQFWjCXN6HBlIZuJ1VO74Aovw96+tcuFu+FwwtlZgHWFNqP+ii1ZS
k236Bh91M2xN2ePuExrixwWBNa7ppuryOal+9fTx/hYSEg43lErn15SyXBn1DiR7A7Bg85/Ysdoxr/H8tPFEfq6xHpzAPGPbq7kjZbdn1NvQkRUZLbZmstS2IJik/3UEeb8r
```

[암호문 파일]

고성능 아키텍처 기반의 대용량 초기데이터 암호화



고성능 아키텍처 기반의 대용량 초기데이터 암호화

➢ LINUX 환경 성능 테스트

H/W : 2.30 GHz/64 core
File 크기 : 1.8 GB
OS : CentOS 6.5
파일 복사 시 소요 시간 : 2.6초

Parallel 정도	1	4	8	64
소요 시간	80.1초	20.5초	11.5초	5.2초

➢ UNIX 환경 성능 테스트

H/W : .53GHz /8 core
File 크기 : 1.8 GB
OS : HPUX IA
파일 복사 시 소요 시간 : 3.5초

Parallel 정도	1	2	4	8
소요 시간	40.7초	20.4초	17.0초	10.7초

SAM 파일 암호화

- SAM 파일 등 형식이 있는 파일 내의 특정 컬럼을 암/복호화
- 컬럼에 대한 위치는 설정 파일에서 정의하며, 콤마로 분리된 형식이거나 Byte 단위 위치 기반

[Configuration File]

```
(pfe=
  (trace_level = 3)
  (mode = encrypt)
  (log_file = ./fencrypt.log)
  (num_data_chunks = 100)
  (rows_per_chunk = 3)
  (in=
    (file_path = ./ori.dat)
    (delimiter = ",")
    (format = delimiter)
    (num_columns = 4)
    (column=
      (3 = (ecid=1))
    )
  )
  (out=
    (file_dir = ./)
    (file_name = enc.dat)
    (format = delimiter)
    (delimiter = ",")
    (rows_per_file = 1000)
    (num_columns = 4)
  )
)
```

Trace_leve : 암호화 간 로그를 출력하기 위한 레벨 (범위 0 ~ 10)
Mode : 암호화 / 복호화를 선택할 수 있는 모드
Log_file = trace_level 설정 시 출력되는 로그파일 경로
Num_data_chunks : 암/복호화 수행 시 동시에 수행할 DATA CHUNK 수
Rows_per_chunk : 암/복호화 수행 시 동시에 수행할 ROW 수

In : 입력 파일에 대한 설정 부분
File_path :암/복호화 대상 파일 경로 및 파일 명
Delimiter : 파일 내 컬럼을 구분하기 위한 구분자 포맷
Format : 데이터를 구분할 기준 (기본값 delimiter)
Num_columns : 암/복호화 대상 파일의 컬럼 수
Column : 암/복호화 대상 컬럼 및 암호화 키 설정
  형식 : column posision = (ecid=encrypt key id)

In : 출력 파일에 대한 설정 부분
file_dir :암/복호화 출력 파일 경로
File_name : 암/복호화 출력 파일 명
Delimiter : 파일 내 컬럼을 구분하기 위한 구분자 포맷
Format : 데이터를 구분할 기준 (기본값 delimiter)
Row_per_file : 파일의 최대 출력 라인 수
Num_columns : 암/복호화 대상 파일의 컬럼 수

[평문 파일]

```
(key_svr):/home/dhkim/fencrypt $ cat ori.dat
0001,Nikita,5704041920384,remark
0002,Dana,5704041920384,remark
0003,Daisy,5704041920384,remark
0004,Raye,5704041920384,remark
0005,Rebecca,5704041920384,remark
0006,Rachel,5704041920384,remark
0007,Daria,5704041920384,remark
0008,Mikhaila,5704041920384,remark
0009,Melissa,5704041920384,remark
0010,Marcia,5704041920384,remark
```

```
(key_svr):/home/dhkim/fencrypt $ cat enc.dat
0001,Nikita,RFlTpn3RlCg0lQR6m5G/1AAA==AA0A,remark
0002,Dana,RFlTpn3RlCg0lQR6m5G/1AAA==AA0A,remark
0003,Daisy,RFlTpn3RlCg0lQR6m5G/1AAA==AA0A,remark
0004,Raye,RFlTpn3RlCg0lQR6m5G/1AAA==AA0A,remark
0005,Rebecca,RFlTpn3RlCg0lQR6m5G/1AAA==AA0A,remark
0006,Rachel,RFlTpn3RlCg0lQR6m5G/1AAA==AA0A,remark
0007,Daria,RFlTpn3RlCg0lQR6m5G/1AAA==AA0A,remark
0008,Mikhaila,RFlTpn3RlCg0lQR6m5G/1AAA==AA0A,remark
0009,Melissa,RFlTpn3RlCg0lQR6m5G/1AAA==AA0A,remark
0010,Marcia,RFlTpn3RlCg0lQR6m5G/1AAA==AA0A,remark
```

[암호문 파일]

실시간 시스템 로그 암호화

- 실시간으로 저장되고 있는 로그 파일에 대한 암호화 (패턴 암호화/전체 암호화 중 택일)
- 암호화 된 결과에 대해 복호화 처리를 해야만 로그 파일을 볼 수 있음
  (관리자 권한이 탈취된 상황에서도 **복호화 미 처리 시 로그 확인 불가 - 유출방지**)

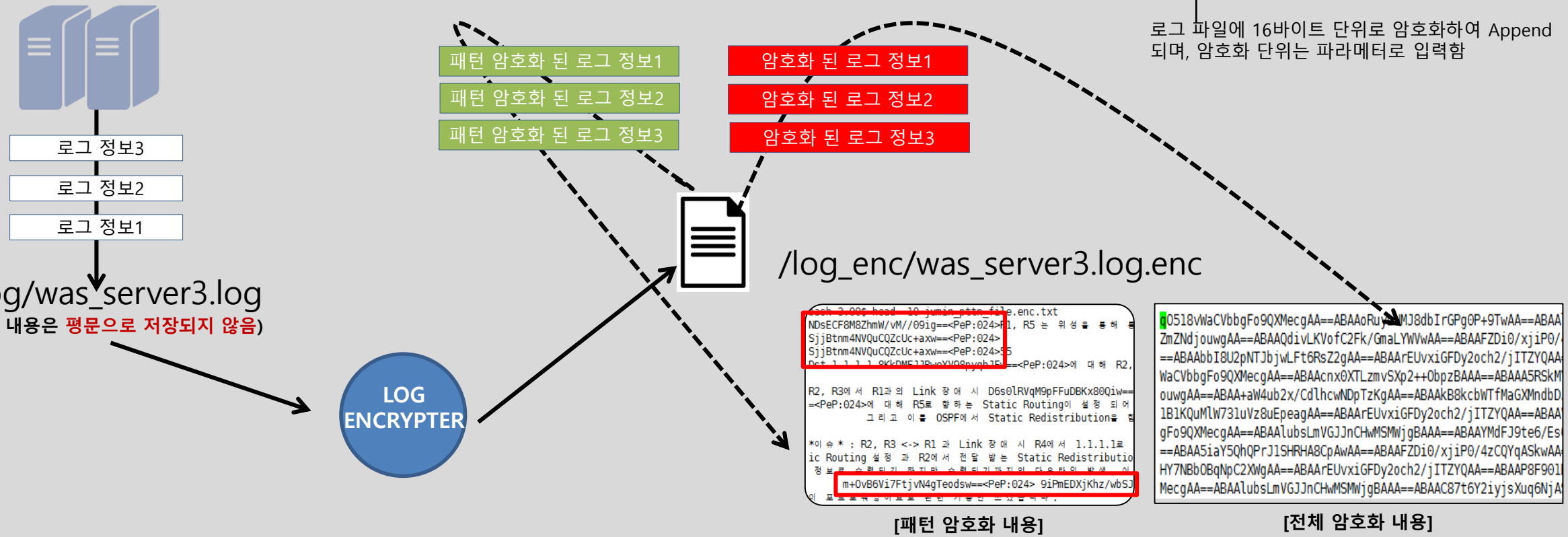
파일 내 특정패턴(주민번호) 암호화

```
Log_pttr_encrypter /log/was_server3.log
/log_enc/was_server3.log.enc enc_key  jumin_pttn.expr
```

파일 전체 암호화

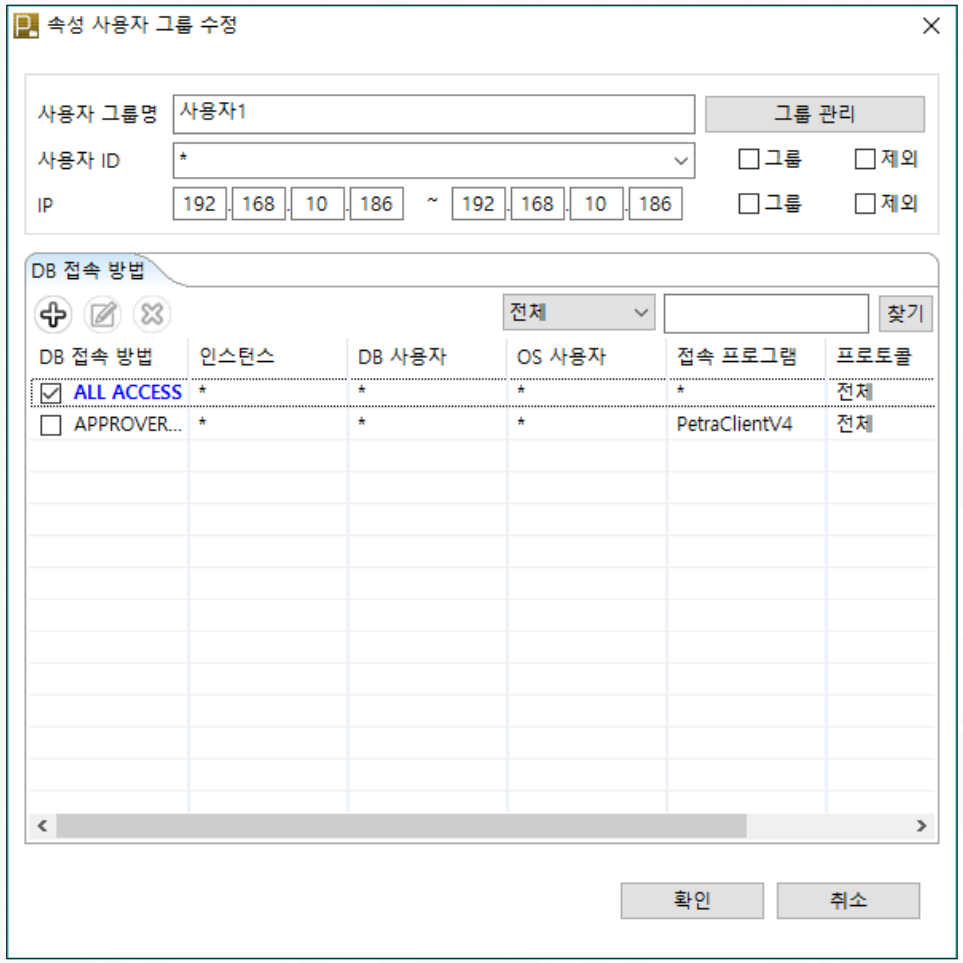
```
log_encrypter /log/was_server3.log
/log_enc/was_server3.log.enc enc_key  16
```

로그 파일에 16바이트 단위로 암호화하여 Append 되며, 암호화 단위는 파라메터로 입력함

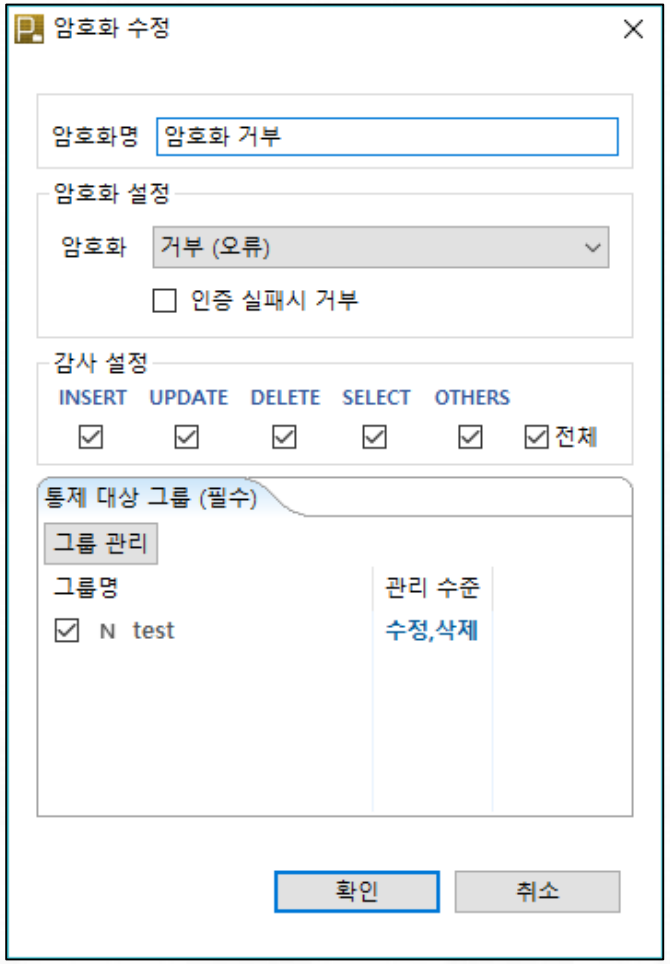


암/복호화 수행 시 권한 통제

- 암/복호화 수행 시, 수행 IP (실제 Client IP), OS 계정 기반의 권한 통제
- OS 명령어에서 프로그램 명을 파라메터로 입력 받아 프로그램 별 통제 기능 제공



[통제 대상 IP 설정]



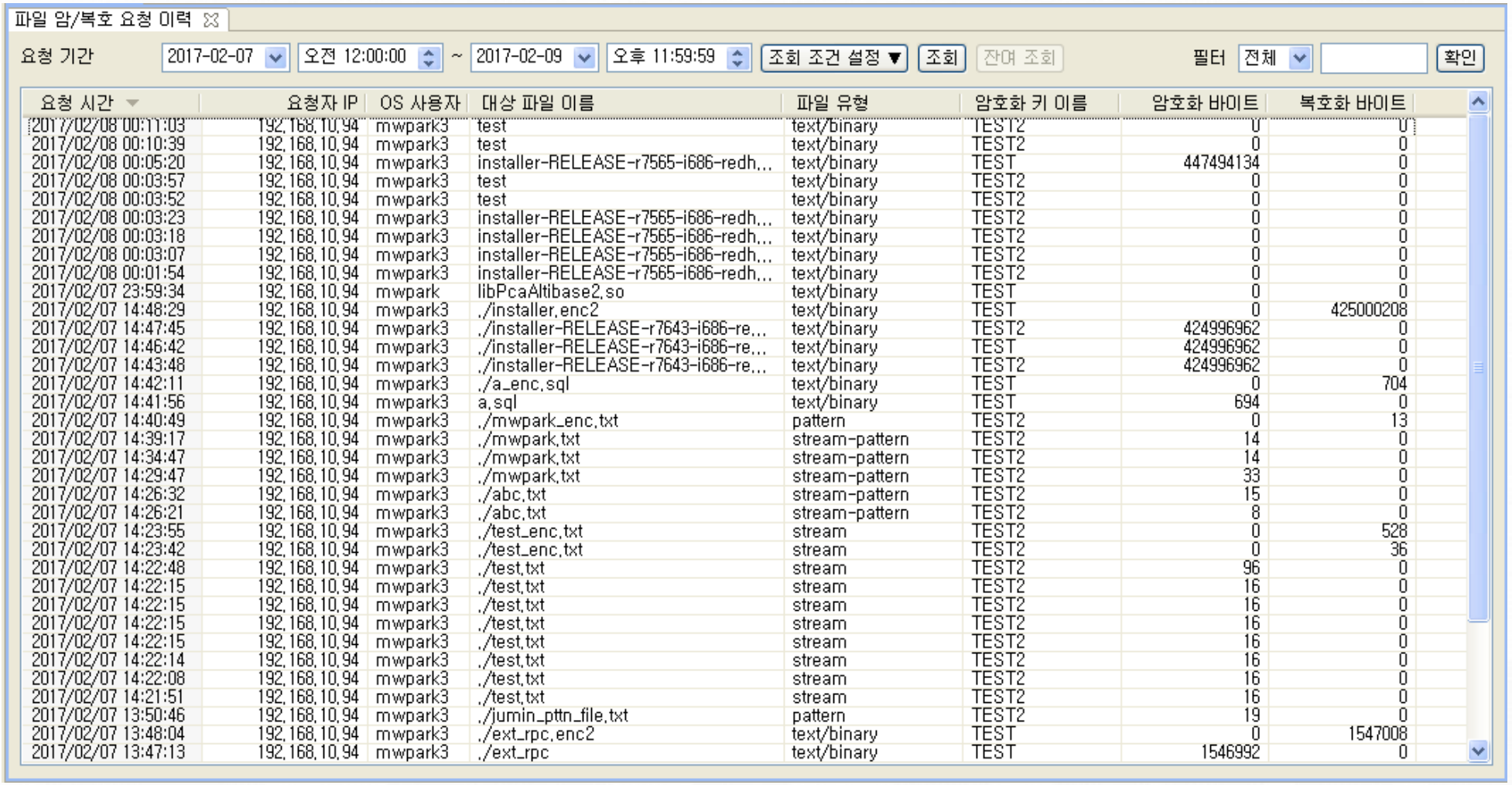
[권한 설정]

```
$ ./pcp_file_crypt encrypt TEST ./test.txt ./test.txt.enc
getSession Failed[-30107]
file encrypt failed
```

[암호화 시도 시 에러 발생]

파일 암호화 로그 관리

- 암/복호화 수행 이력에 대한 로그 관리 및 조회 화면 제공
- 요청시간, 사용자 IP, OS 사용자, 파일이름, 유형, 암호화 키 이름, 암/복호화 바이트 등 정보 제공



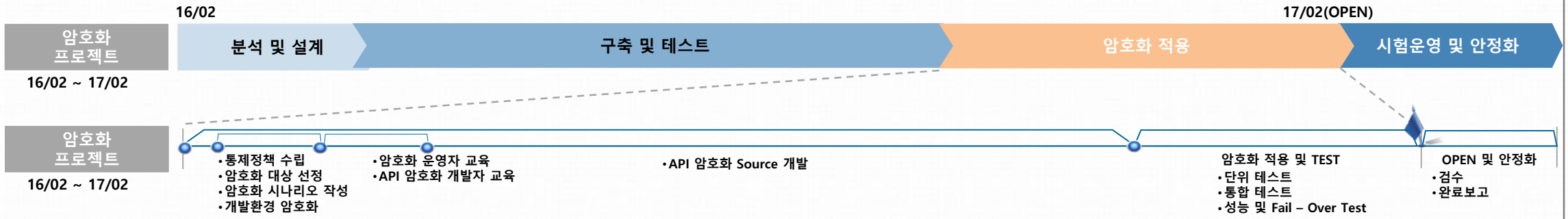
요청 시간	요청자 IP	OS 사용자	대상 파일 이름	파일 유형	암호화 키 이름	암호화 바이트	복호화 바이트
2017/02/08 00:11:03	192,168,10,94	mwpark3	test	text/binary	TEST2	0	0
2017/02/08 00:10:39	192,168,10,94	mwpark3	test	text/binary	TEST2	0	0
2017/02/08 00:05:20	192,168,10,94	mwpark3	installer-RELEASE-r7565-i686-redh,..	text/binary	TEST	447494134	0
2017/02/08 00:03:57	192,168,10,94	mwpark3	test	text/binary	TEST2	0	0
2017/02/08 00:03:52	192,168,10,94	mwpark3	test	text/binary	TEST2	0	0
2017/02/08 00:03:23	192,168,10,94	mwpark3	installer-RELEASE-r7565-i686-redh,..	text/binary	TEST2	0	0
2017/02/08 00:03:18	192,168,10,94	mwpark3	installer-RELEASE-r7565-i686-redh,..	text/binary	TEST2	0	0
2017/02/08 00:03:07	192,168,10,94	mwpark3	installer-RELEASE-r7565-i686-redh,..	text/binary	TEST2	0	0
2017/02/08 00:01:54	192,168,10,94	mwpark3	installer-RELEASE-r7565-i686-redh,..	text/binary	TEST2	0	0
2017/02/07 23:59:34	192,168,10,94	mwpark	libPcaAltibase2,so	text/binary	TEST	0	0
2017/02/07 14:48:29	192,168,10,94	mwpark3	./installer,enc2	text/binary	TEST	0	425000208
2017/02/07 14:47:45	192,168,10,94	mwpark3	./installer-RELEASE-r7643-i686-re,..	text/binary	TEST2	424996962	0
2017/02/07 14:46:42	192,168,10,94	mwpark3	./installer-RELEASE-r7643-i686-re,..	text/binary	TEST	424996962	0
2017/02/07 14:43:48	192,168,10,94	mwpark3	./installer-RELEASE-r7643-i686-re,..	text/binary	TEST2	424996962	0
2017/02/07 14:42:11	192,168,10,94	mwpark3	./a_enc,sql	text/binary	TEST	0	704
2017/02/07 14:41:56	192,168,10,94	mwpark3	a,sql	text/binary	TEST	694	0
2017/02/07 14:40:49	192,168,10,94	mwpark3	./mwpark_enc,txt	pattern	TEST2	0	13
2017/02/07 14:39:17	192,168,10,94	mwpark3	./mwpark,txt	stream-pattern	TEST2	14	0
2017/02/07 14:34:47	192,168,10,94	mwpark3	./mwpark,txt	stream-pattern	TEST2	14	0
2017/02/07 14:29:47	192,168,10,94	mwpark3	./mwpark,txt	stream-pattern	TEST2	33	0
2017/02/07 14:26:32	192,168,10,94	mwpark3	./abc,txt	stream-pattern	TEST2	15	0
2017/02/07 14:26:21	192,168,10,94	mwpark3	./abc,txt	stream-pattern	TEST2	8	0
2017/02/07 14:23:55	192,168,10,94	mwpark3	./test_enc,txt	stream	TEST2	0	528
2017/02/07 14:23:42	192,168,10,94	mwpark3	./test_enc,txt	stream	TEST2	0	36
2017/02/07 14:22:48	192,168,10,94	mwpark3	./test,txt	stream	TEST2	96	0
2017/02/07 14:22:15	192,168,10,94	mwpark3	./test,txt	stream	TEST2	16	0
2017/02/07 14:22:15	192,168,10,94	mwpark3	./test,txt	stream	TEST2	16	0
2017/02/07 14:22:15	192,168,10,94	mwpark3	./test,txt	stream	TEST2	16	0
2017/02/07 14:22:15	192,168,10,94	mwpark3	./test,txt	stream	TEST2	16	0
2017/02/07 14:22:14	192,168,10,94	mwpark3	./test,txt	stream	TEST2	16	0
2017/02/07 14:22:08	192,168,10,94	mwpark3	./test,txt	stream	TEST2	16	0
2017/02/07 14:21:51	192,168,10,94	mwpark3	./test,txt	stream	TEST2	16	0
2017/02/07 13:50:46	192,168,10,94	mwpark3	./jumin_pttn_file,txt	pattern	TEST2	19	0
2017/02/07 13:48:04	192,168,10,94	mwpark3	./ext_rpc,enc2	text/binary	TEST	0	1547008
2017/02/07 13:47:13	192,168,10,94	mwpark3	./ext_rpc	text/binary	TEST	1546992	0

[암/복호화 로그 조회 UI]

Table of Contents

구축사례

NH농협은행- (공통)고유식별 정보 암호화를 위한 DB암호화 솔루션 도입



대상 개요

1 암호화 필수대상
- 주민번호
- 비밀번호(단방향)

암호화 방식

- API(C,JAVA) 방식 권고 (WAS 자원 사용)
- 성능, 차세대 개발 및 인프라 보안 정책 등으로, API(C, JAVA) 방식으로 선택
- 개인정보 포함된 전문/로그 파일, SAM 파일은 해당 일부 정보만 암호화 하여 기록
- 프레임워크에 적용 : PROFRAME(티맥스) , DATA STAGE 적용

암호화 알고리즘 및 주요 확장 기능

- 암호화 알고리즘 선택 - 양방향 : SEED-128, 256(기존 암호화 사용 컬럼), ARIA-256, 단방향 :SHA-256
- 주요 확장 기능
  - 외부 키, 외부 IV를 입력할 수 있는 기능 제공
  - SAM 파일 일부 정보 암호화 기능 제공
  - 파일 (전체) 암호화 기능 제공 : 전문, xml 파일 암호화
  - getKey 인터페이스 제공(Key 만 제공)
  - 암호화 데이터에 키 식별 Trailer 첨부하여 저장 기능 제공 (은행과 중앙회 암호화 키 별도로 적용할 수 있도록)
  - 시스템 부하 없이 대량 로그 생성/저장 기능 제공 (로그를 UDP로 송신하여 기존 어플리케이션 성능 영향 최소화)

암호화

- FRAMEWORK 시스템에 암호화 적용(PROFRAME, DATA STAGE)
- 운영계, 계정계, 채널계 260개
- 검증계, 개발계 129개
- 인터넷 뱅킹 업무 포함

개발

- API 암호화 메커니즘 및 개발 가이드 교육
  - API(C, JAVA)
- 개발기간 : '16/02~'17/02(12개월)
- 개발자 인원 : 약 320명 이상 (농협중앙회와 농협은행 분리 작업 수행 인원이 병행 함)

정책

- 암호화 키 정책 (키 서버 3중화) (Local Key, Server #1, Key Server#2)
  - 운영, DR에 키 서버 구축
- 통제 및 감사 정책
  - IP(AP서버) 통제 적용
  - 암호화 키 호출 이력 및 암/복호화 이력(AP IP, 호출 건수 등)

테스트 및 결과

- 성능테스트
- 삼중화 테스트 완료

구축사례

NH농협은행- (공통)고유식별 정보 암호화를 위한 DB암호화 솔루션 도입

암호화 키
- 정형 암호화와 동일한 키 서버를 사용
- 정형과 통합되어 있으므로 별도의 키 관리 메커니즘이 필요 없음

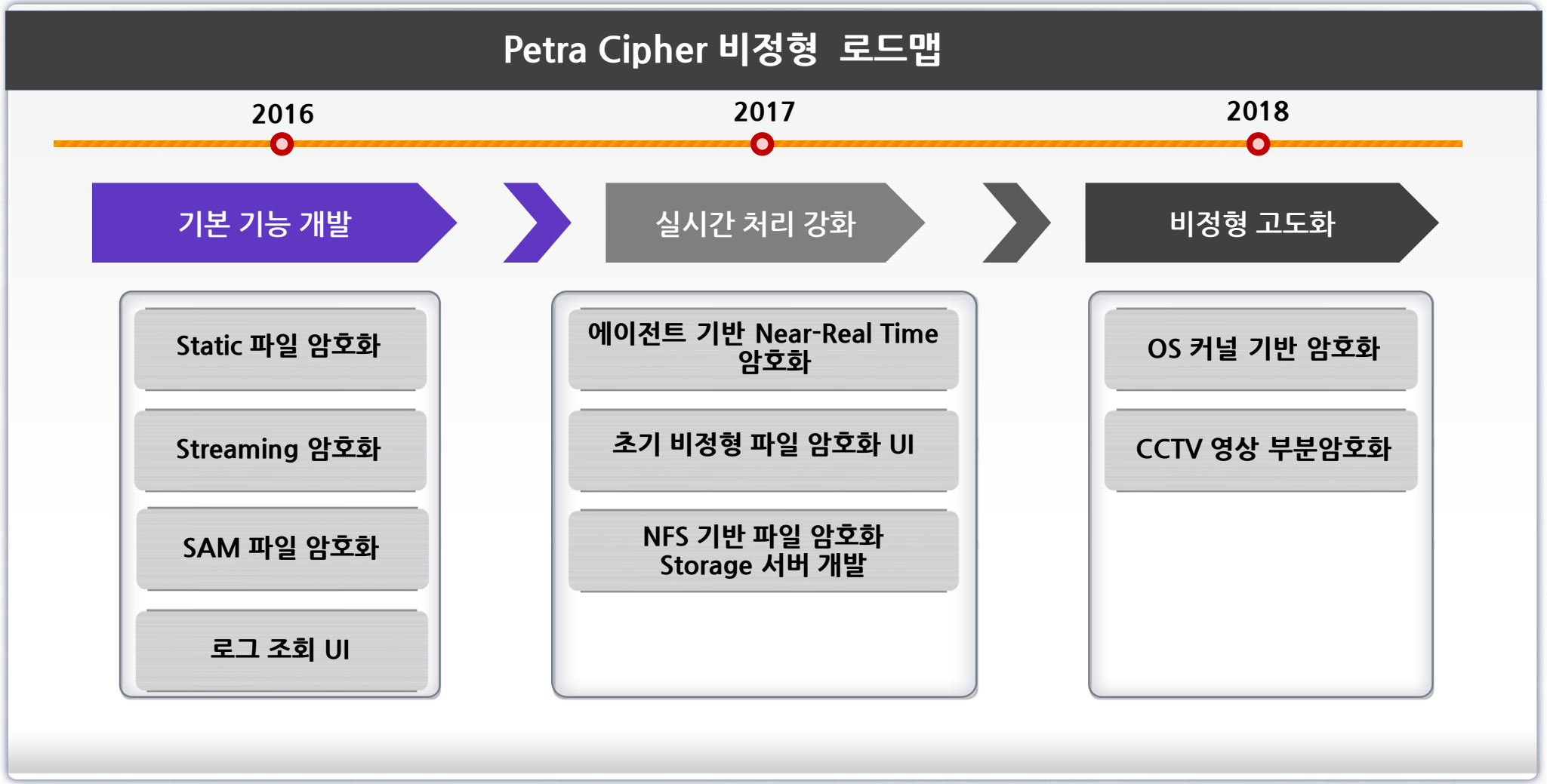
알고리즘
- 국정원 알고리즘 ARIA256을 사용하여 암호화
- 국정원 인증 모듈 사용

암호화 대상
- 전문, 로그 등에 대한 파일 단위 암호화
- SAM 파일의 경우 특정 위치에 있는 개인정보 암호화

인터페이스
- 암/복호화 OS 명령어를 이용하여 Shell 프로그램 작성
- 파일 생성 후 Shell 프로그램을 호출

Table of Contents

제품 로드맵


Petra Cipher 비정형 로드맵
2016
2017
2018
기본 기능 개발
실시간 처리 강화
비정형 고도화
Static 파일 암호화
Streaming 암호화
SAM 파일 암호화
로그 조회 UI
에이전트 기반 Near-Real Time 암호화
초기 비정형 파일 암호화 UI
NFS 기반 파일 암호화 Storage 서버 개발
OS 커널 기반 암호화
CCTV 영상 부분암호화

Thank You.

Bell I&S
Bell Information&Solutions

Tel) 02-6925-1131 / Fax) 02-2664-1575

[본사]서울시 서대문구 충정로 8 종근당빌딩 8층
[솔루션사업부] 서울시 서대문구 충정로 13 삼창빌딩 3층
E-Mail : shlee@bellins.net