자바스크립트 화이트박스 암호와 크롬 라인 메신저의 보안 강화

안상환

LINE Corporation / Security R&D Team

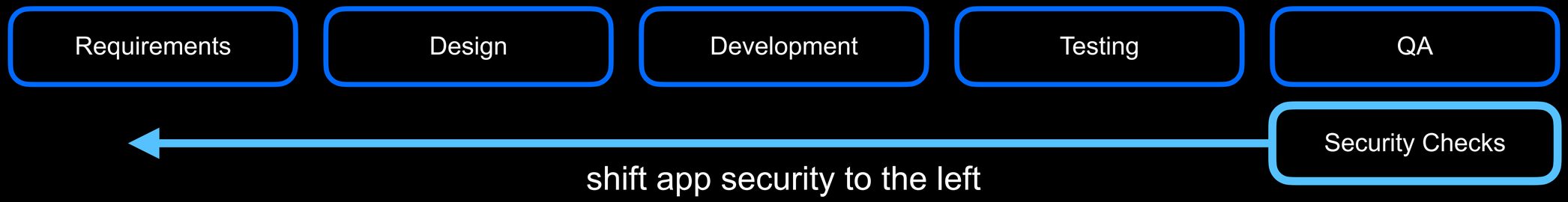
NAVER DEVIEW 2023

CONTENTS

1.보안을 개발 파이프라인에 통합시키는 방법

1.1 보안을 왼쪽으로

NAVER
DEVIEW
2023



- 보통 소프트웨어 개발 생명 주기상에서 마지막 단계에서 보안 테스트를 진행
- 마지막 단계에서 발견된 문제를 수정하는데 드는 비용이 (매우) 비쌈
- 개발 프로세스 전반에 걸쳐 보안적으로 안전한 디자인의 구현은 개발자의 역량에 완전히 의존

1.1 보안을 왼쪽으로

NAVER
DEVIEW
2023

- 보안 결함의 가능성을 최소화 하기 위해 기획/개발 초기 단계부터 보안을 고려하여 설계
- 개발 초기 단계에 보안 결함을 식별할 수 있어 문제를 해결하는 노력과 비용이 크게 감소
- 보안 부서 내에서 일부 개발을 수행하는것이 더 효율적일 수 있음

2.웹 애플리케이션에서의 보안 위협

2.1 자바스크립트

NAVER
DEVIEW
2023

보안 측면에서의 결정적 결함

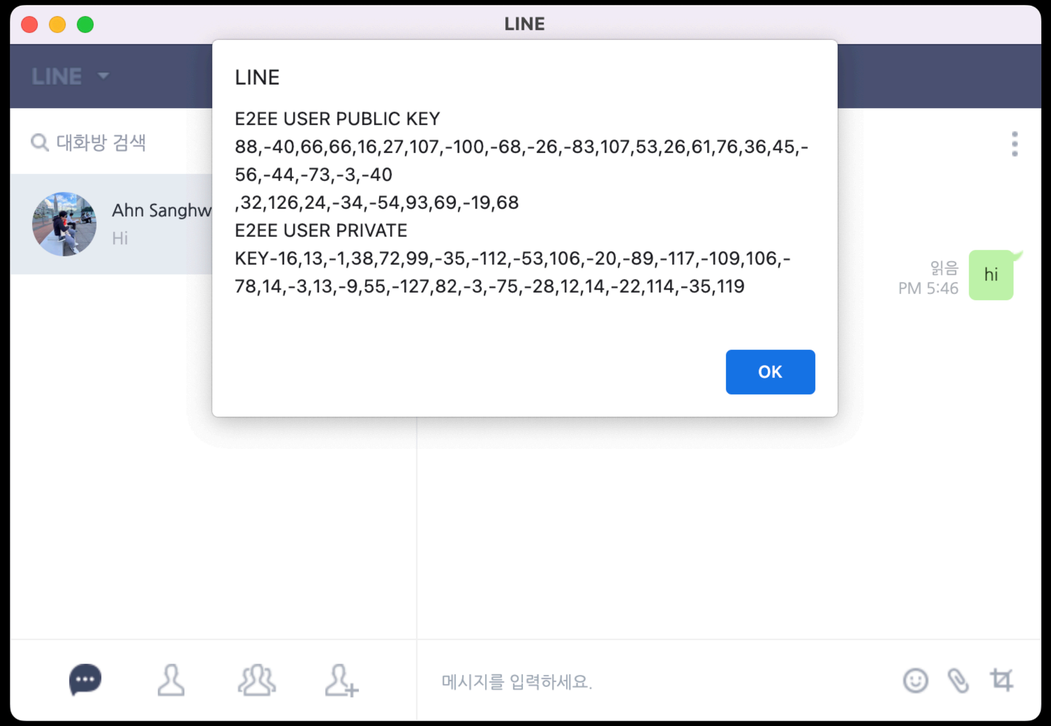
- 인터프리터 언어이므로, 배포 전에 기계어로 컴파일되는 대신 런타임에서 컴파일되고 실행됨
- 네이티브 모바일 코드와 달리, 자바스크립트 코드는 서명되지 않으므로 공격자가 쉽게 코드를 분석하고 수정할 수 있음
- 코드 주입이 네이티브에 비해 비교적 쉬우며, 임의의 코드 실행에 대한 추가적인 완화책(Mitigation)이 없음

2.2 크로스 사이트 스크립팅 취약점

NAVER
DEVIEW
2023

임의의 코드 실행 가능

- 웹 애플리케이션에서 가장 일반적인 유형의 취약점
- 공격자가 애플리케이션 컨텍스트에서 임의의 코드를 실행하도록 허용
- 공격자는 암호, 민감한 정보, 암호키, 자격 증명 정보의 액세스 등을 포함하여 웹 애플리케이션이 할 수 있는 모든 작업을 수행할 수 있음



2.3 브라우저의 시큐어 스토리지의 부재

NAVER
DEVIEW
2023

키 관리 문제

- 브라우저에서 암호키와 같은 민감한 데이터를 안전하게 저장할 수 있는 옵션이 없음
- 필연적으로 하드코딩된 암호 키 문제로 이어짐
- 브라우저에서 제공하는 스토리지에 저장하거나 자바스크립트상에서만 키를 사용하더라도 공격자는 자바스크립트상에서 해당 키에 접근이 가능함

2.4 예상치 못한 액세스 토큰의 유출

NAVER
DEVIEW
2023

액세스 토큰 유출 문제

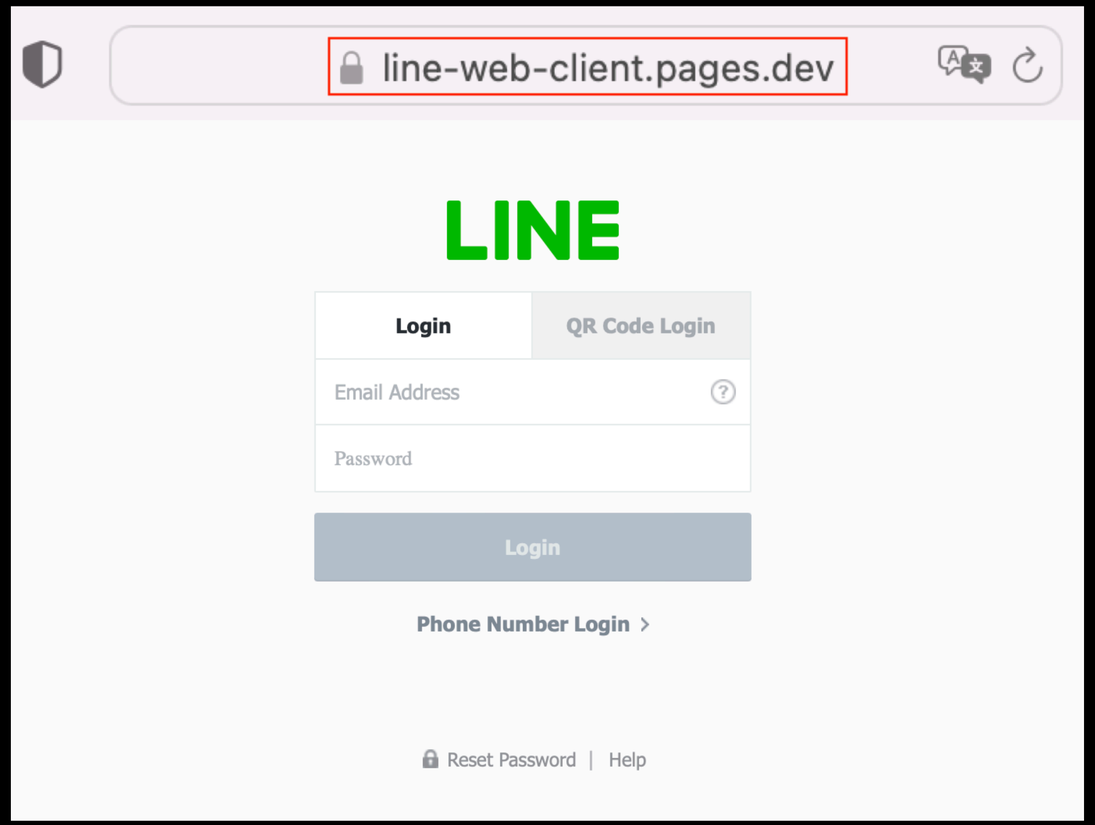
- 도난당한 액세스 토큰이나 악성 봇으로 부터 불법 액세스를 방지하기 위해 토큰 발행 시 토큰을 공개 키에 바인딩하고 클라이언트가 토큰을 사용할 때 개인 키를 소유하고 있음을 증명하도록 요구함으로써 이를 완화할 수 있지만 (예, OAuth2.0 DPoP), 브라우저에서는 바인딩 키를 안전하게 저장할 수 있는 적절한 수단이 없음

2.5 API 어뷰징

NAVER
DEVIEW
2023

악성 봇 문제

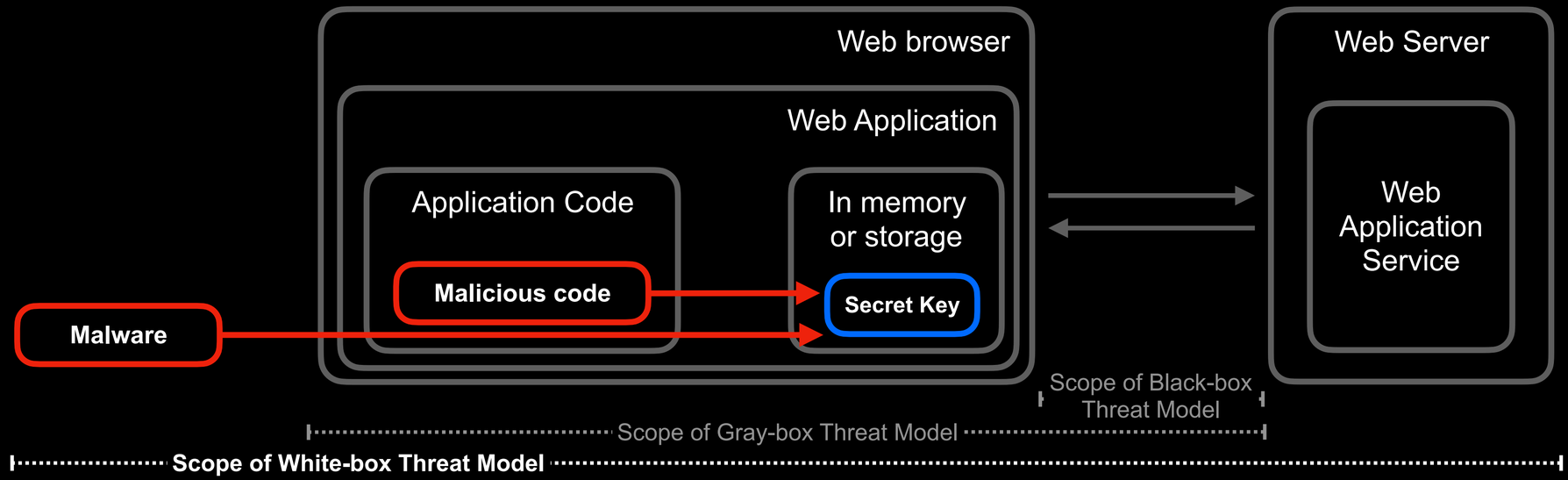
- 공격자는 모바일 또는 웹 애플리케이션을 리버스 엔지니어링하여 작동 방식을 파악한 다음 악성 봇을 프로그래밍하여 비즈니스 API를 스팸, 피싱, DDoS에 악용할 수 있음
- 변조된 웹 애플리케이션을 구축하여 계정을 탈취하는데 악용할 수 있음



3.위협 모델

3.1 화이트박스 위협 모델

NAVER
DEVIEW
2023



- 암호화 프로세스 뿐만 아니라 실행환경을 완전히 제어할 수 있다고 가정
- 공격자는 메모리 또는 브라우저의 스토리지에 저장된 키에 대한 액세스 권한을 얻기위해 시도
- 맬웨어는 파일 시스템에 액세스할 수 있으므로 스크립트 실행 없이도 비밀 키에 접근 가능

4. 모던 웹 기술

4.1 Web Crypto API와 IndexedDB

NAVER
DEVIEW
2023

- 비밀키의 내용은 Javascript 환경에서 액세스할 수 없음
- 키가 추출되지 않도록 추가 보호 계층을 제공하고 합법적인(legitimate) 작업에서만 작동
  - CryptoKey.extractable: true or false
  - CryptoKey.usages: encrypt, decrypt, sign, verify, wrapKey, unwrapKey, deriveKey, deriveBits
- CryptoKey 객체는 IndexedDB에 직접 저장할 수 있음

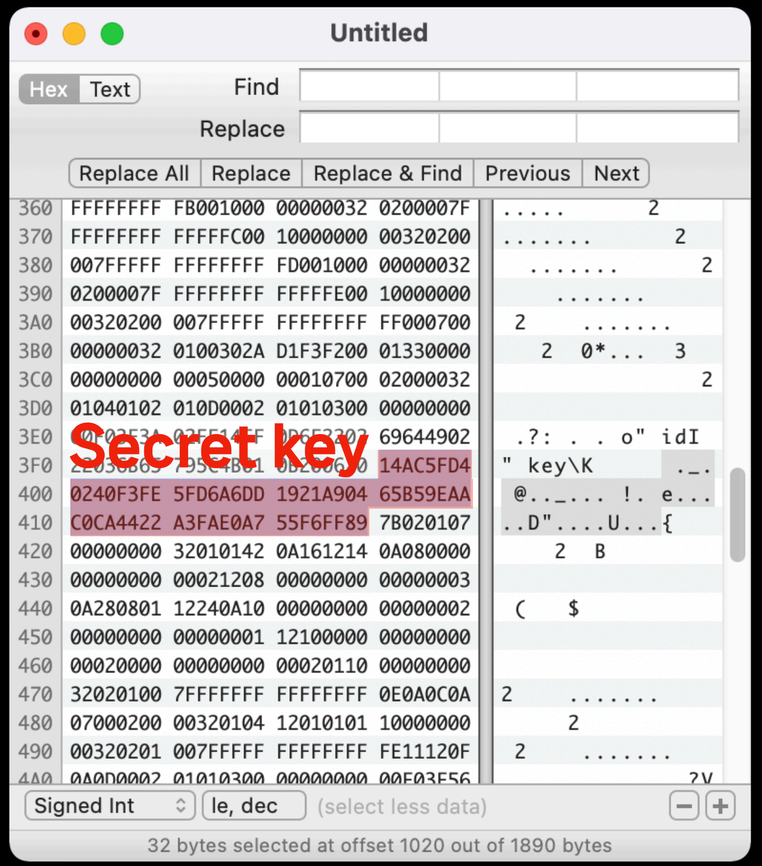
```
> window.crypto.subtle.generateKey({name: "AES-CTR", length: 256}, false, ["encrypt", "decrypt"]).then(function(key){
      console.log(key);
  })
<· ▶ Promise {<pending>}

  ▼ CryptoKey {type: 'secret', extractable: false, algorithm: {…}, usages: Array(2)} i
    ▶ algorithm: {name: 'AES-CTR', length: 256}
      extractable: false
      type: "secret"
    ▶ usages: (2) ['encrypt', 'decrypt']
    ▶ [[Prototype]]: CryptoKey
```

4.2 제약사항

NAVER
DEVIEW
2023

- 파일시스템 수준에서 보호를 보장하지 않음
- Key Unwrapping에서 Unwrapping Key를 어디에 둘 수 있습니까?
- 키 증명(Key Attestation)을 제공하지 않음
- Curve25519를 지원하지 않음

(LINE의 E2EE 프로토콜은 Curve25519 기반)



5.암호 연산중에도 키가 노출되지 않는 화이트박스 암호

5.1 화이트박스 암호

NAVER
DEVIEW
2023

- 알고리즘에 키가 내장되어 있어 본질적으로 추출하기 어려움
- 암호키는 메모리에서도 일반 텍스트 형식으로 노출되지 않음
- Frida, PINTool 및 Valgrind와 같은 DBI(Dynamic Binary Instrumentation) 툴킷과 리버스 엔지니어링대한 향상된 저항을 제공
- 순수 소프트웨어 기반
- 플랫폼 전반에서 커스터마이즈된 암호화 API의 구현이 가능

5.2 화이트박스 암호의 활용

NAVER
DEVIEW
2023

키 객체에 대한 추가적인 보호 계층

- 직접적인 액세스에 대한 추가적인 보호 계층을 제공하여 비밀키의 내용을 알 수 없도록

보안 키 프로비저닝

- 하드코딩된 정적(Static) 키 문제를 해결
- 서버에서 보낸 정적(Static) 키 또는 동적(Dynamic) 키를 보호
- 장치에 키를 바인딩

5.3 화이트박스 암호 공격

NAVER
DEVIEW
2023

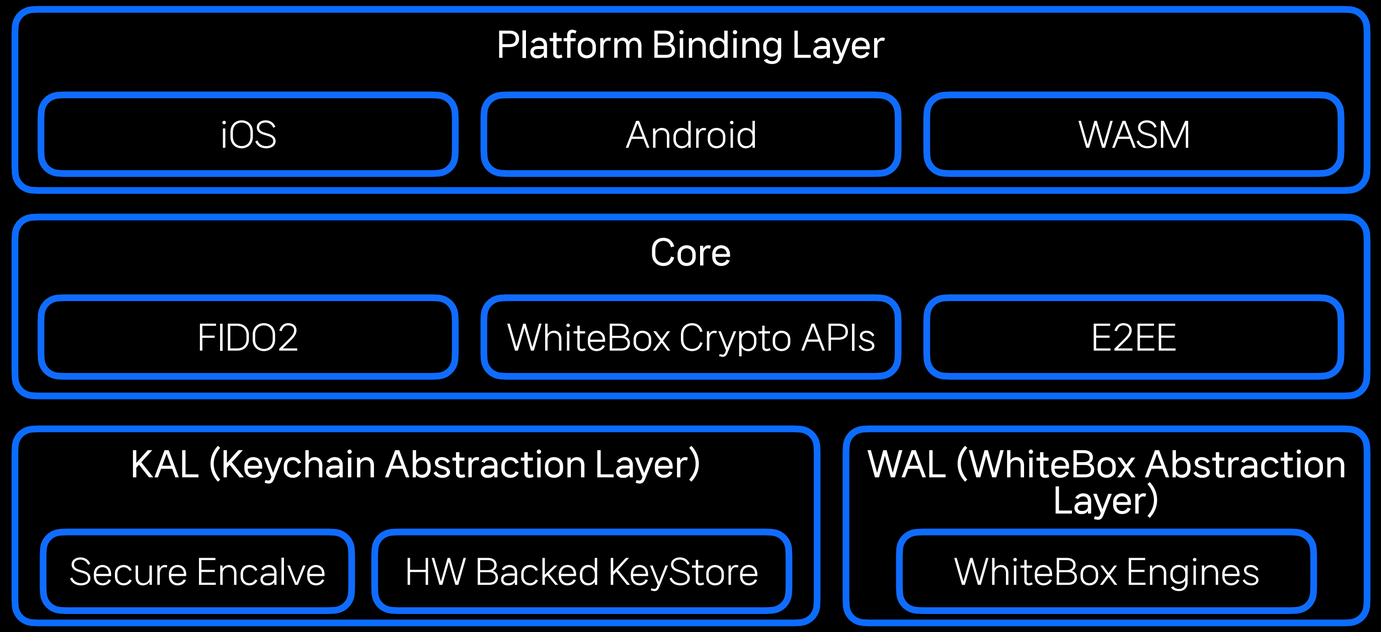
부채널 공격

- 리버스엔지니어링으로는 키를 복구해내기 매우 어려움
- 공개된 화이트박스 스킴의 경우 부채널 공격으로 키를 복구해내는것이 가능 (Unboxing the white-box, Black Hat Europe 2015)
- 상용 화이트박스 암호의 경우 Shuffling, Masking등 부채널공격에 향상된 저항성을 가질 수 있도록 구현
- 상용 화이트박스 또한 개선된 부채널공격에 의해 키 복구 가능 (Key recovery attacks against commercial white-box cryptography implementations, PacSec2017/CodeBlue2017/HitCon2017)

6.LTSM WASM 아키텍쳐

6.1 LTSM(LINE Trusted Security Module)

NAVER DEVIEW 2023



- 화이트박스 기반의 공통 암호화 API를 제공하는것 뿐만아니라, 하드웨어 기반의 보안 기능을 활용한 인증 프로토콜(예, FIDO2, Key Attestation등)을 제공
- 최신 보안 기술 및 각종 보안 기능을 제공할 수 있는 보안 플랫폼
- 라인 메신저, 라인 페이, 라인 뱅크에서 사용중

6.2 웹 어셈블리

NAVER
DEVIEW
2023

브라우저에서 네이티브 구현이 가능

- 거의 네이티브에 가까운 성능으로 C/C++ 및 Rust와 같은 네이티브 코드를 실행할 수 있음
- WebAssembly는 호스트 런타임과 분리된 샌드박스 환경 내에서 실행됨
- Memory Safety, Undefined Behavior Sanitizer, Address Sanitizer, Control-Flow Integrity와 같은 최신의 보안 기능을 제공

6.3 LTSM WASM

NAVER
DEVIEW
2023

- C++ 기반
- Emscripten을 사용하여 코드를 WASM 모듈로 컴파일
- Embind를 사용하여 C++ 함수 및 클래스를 JavaScript에 바인딩
- Chrome 확장 프로그램의 샌드박스 페이지에 LTSM WASM이 로드됨, 샌드박스 페이지는 확장 API에 액세스할 수 없음

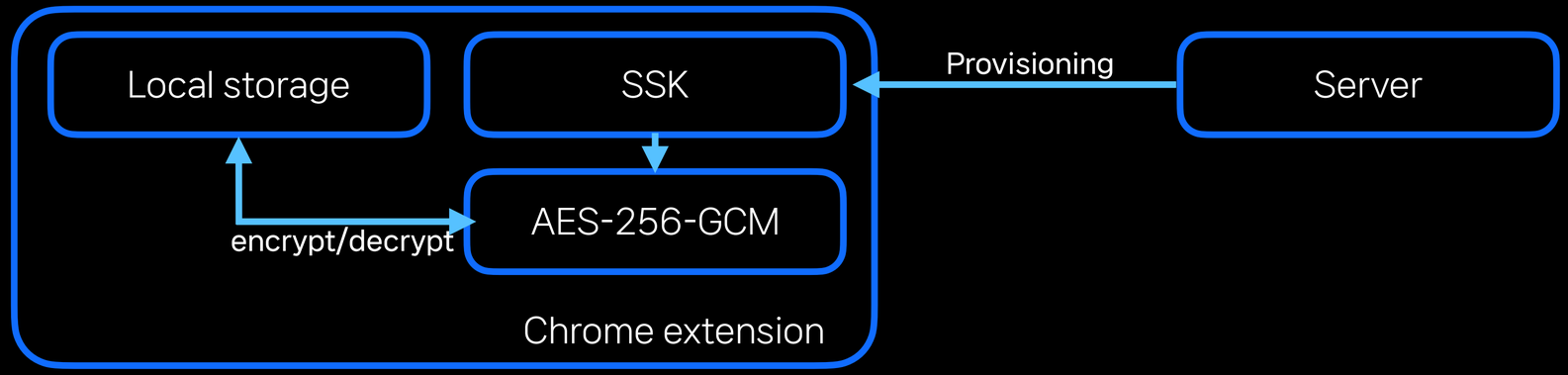
7.유스케이스

7.1 시큐어 스토리지

NAVER
DEVIEW
2023

화이트박스로 보호된 키로 스토리지 데이터를 암호화

- 브라우저의 로컬 스토리지 시스템에 의존
- SSK(Secure Storage Key)는 서버로 부터 프로비저닝됨
- 키 객체는 메모리에서만 유효하며 추출할 수 없음

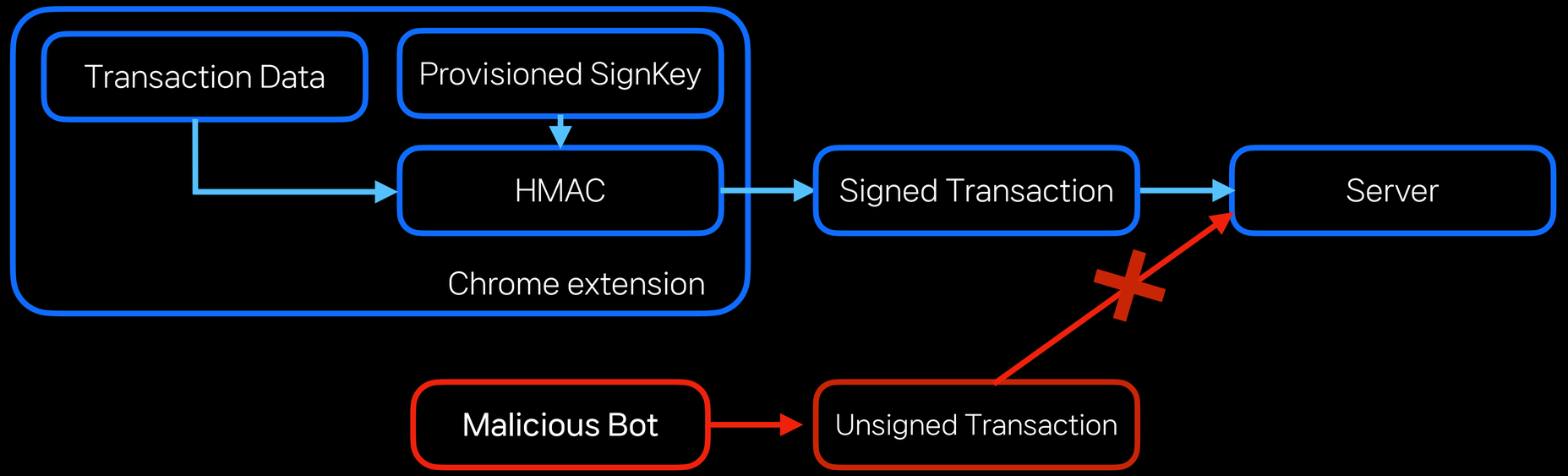


7.2 트랜잭션 서명

NAVER
DEVIEW
2023

프로비저닝된 화이트박스 키로 트랙잭션을 서명

- 서버는 서명된 트랙잰션 외에는 모두 거부, 인가된 클라이언트만 API를 호출할 수 있음
- 서명키는 인가된 클라이언트(환경)에 바인딩되어 다른 환경에서는 로드할 수 없음



7.3 종단간 암호화

NAVER
DEVIEW
2023

Letter Sealing

- LINE의 End to End 암호화 프로토콜
- 각 사용자의 장치에서 개인 키를 소유
- ECDH 키 교환 알고리즘 기반, Curve25519
- 문자 메시지, 위치 메시지, 음성/영상 통화는 전송되기 전에 LINE 클라이언트에서 암호화
- Letter Sealing에 대한 자세한 내용은 이하의 백서에 기술

https://scdn.line-apps.com/stf/linecorp/en/csr/line-encryption-whitepaper-ver2.1.pdf

8.화이트박스 통합 이슈

8.1 화이트박스 구현의 제약사항

NAVER
DEVIEW
2023

화이트박스는 키의 임의 바이트 조작을 허용하지 않음

- 호환되지 않는 패딩 이슈
- Curve25519 개인 키 생성 시 '클램핑'

솔루션 벤더에게 추가 기능을 요청(feature requests)할 수 도 있지만

- 우선순위 문제
- 제한된 일정내에 추가 기능이 구현된 버전을 전달받기 어려움

8.2 Workaround

NAVER
DEVIEW
2023

화이트박스에서 제공하는 도구를 이용

- 화이트박스는 일반 데이터에서 화이트 박스로 보호된 Keyblob을 생성하는 도구(Key Export Tool)를 제공
- 화이트박스는 키 유도 함수(Key Derivation Function)에서 특별한 목적의 알고리즘을 제공
  - Concat: 두 개의 기존 키를 연결하여 새 키를 유도, 단 키는 블록사이즈여야 함
  - Slice: 입력 키에서 바이트의 임의 하위 문자열을 가져와 새 키로 유도

8.2 Workaround

NAVER
DEVIEW
2023

Key Export Tool

- 일반 입력 데이터에서 화이트박스로 보호된 Keyblob을 생성
- 이 작업은 신뢰할 수 있는 환경에서만 수행
- SecureKey.importKey( )로 Keyblob을 로드


```
4A 32 4D 72 39 33 33 6C
61 54 4E 6B 32 4D 4A 30
```

```
49 D8 AD DC 2B AE 89 D1
EE 67 D0 5F CB F3 5C 07
35 2D B4 93 F1 63 D8 51
DC 58 BB DA E0 9A 60 0B
12 15 DA E0 2D 2B AE D8
…
```

8.2 Workaround

NAVER
DEVIEW
2023

사전 계산된 룩업테이블을 이용

- 화이트박스는 임의의 바이트 조작을 허용하지 않으나,
- 특정 바이트셋을 룩업테이블로 만들고 Key Export Tool을 통해 변환된 KeyBlob을 이용하여 원하는 바이트셋을 화이트박스에 주입할 수 있음
- 생성된 룩업테이블과 화이트박스 키 유도함수(Concat/Slice)를 이용하여 임의의 오프셋으로부터 블록단위로 키를 이어 붙이고 제거하는것이 가능
- 간접적으로 화이트박스 키의 임의 바이트 조작이 가능

8.3 호환되지 않는 패딩 이슈

NAVER
DEVIEW
2023

원인

- 키 랩핑시 일치하지 않는 포맷(써드파티와 호환되지 않는 패딩 알고리즘)으로 인한 이슈
- 서로 다른 표준, PKCS#7(RFC, IETF) 패딩 과 XML Encryption(W3C) 패딩의 사용

PKCS#7 padding															
FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF
FF	FF	FF	FF	FF	FF	FF	FF	08	08	08	08	08	08	08	08

XML encryption padding															
FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF
FF	FF	FF	FF	FF	FF	FF	FF	??	??	??	??	??	??	??	08

8.3 호환되지 않는 패딩 이슈

NAVER
DEVIEW
2023

룩업테이블 생성

- PKCS#7 패딩 값들을 룩업테이블로 생성
- 16바이트 패딩 예시

10	10	10	10	10	10	10	10	10	10	10	10	10	10	10	10

- Key Export Tool을 사용하여 생성된 룩업테이블을 화이트박스로 보호된 Keyblob으로 변환
- 이후 SecureKey.importKey( )를 이용하여, Keyblob을 SecureKey로 로드

8.3 호환되지 않는 패딩 이슈

NAVER
DEVIEW
2023

패딩 일치시키기

- 고정 패딩 값으로 구성된 룩업테이블을 키 랩핑 이전에 키 뒤에 이어붙임

Curve25519 private key as SecureKey															
FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF
FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF

+

Fixed padding as SecureKey, loaded from exported keyblob using SecureKey.importKey( )															
10	10	10	10	10	10	10	10	10	10	10	10	10	10	10	10

8.3 호환되지 않는 패딩 이슈

NAVER
DEVIEW
2023

패딩 일치시키기

- 키 랩핑 후 WarppedKey의 마지막 16바이트(기본 패딩)을 제거

Curve25519 private key as SecureKey															
FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF
FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF

Fixed padding as SecureKey, loaded from exported keyblob using SecureKey.importKey( )															
10	10	10	10	10	10	10	10	10	10	10	10	10	10	10	10

삭제

Default padding(XML Encryption) in Key Wrapping)															
??	??	??	??	??	??	??	??	??	??	??	??	??	??	??	10

8.4 Curve25519 키 생성시 '클램핑'

NAVER
DEVIEW
2023

클램핑

- 암호학적으로 안전한 소스로 부터 32바이트 임의의 개인키를 생성
- 개인 키를 사용하기 전에 "클램핑(surjective mapping function)"을 수행하여 적절한 스칼라 값을 도출
- 클램핑 예시

```
secret[0]  &=  248; //최하위 3비트를 제거
secret[31] &= 127; //최상위 비트를 제거
secret[31] |= 64;  //다음 최상위 비트를 설정
```

Ref. Section 5, RFC 7748, Elliptic Curves for Security, https://www.rfc-editor.org/rfc/rfc7748#section-5

8.4 Curve25519 키 생성시 '클램핑'

NAVER
DEVIEW
2023

목표

- Curve25519 개인키가 적절한 스칼라 값을 가질 수 있도록 첫번째, 마지막 바이트를 조작

32bytes Random SecureKey															
FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF
FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF

8.4 Curve25519 키 생성시 '클램핑'

NAVER
DEVIEW
2023

룩업테이블 생성

- 첫번째, 마지막 바이트의 가능한 모든 값에 대해 미리 계산된 룩업테이블 생성
- 룩업테이블 생성 예시

```
for(x = 0; x < 256 ; x++) {
    lut00[256] = x & 248;
    lut31[256] = (x & 127) | 64;
}
```

- Key Export Tool을 사용하여 생성된 룩업테이블들을 화이트박스로 보호된 Keyblob으로 변환
- 이후 SecureKey.importKey( )를 이용하여, Keyblob을 SecureKey로 로드

8.4 Curve25519 키 생성시 '클램핑'

NAVER
DEVIEW
2023

첫번째 바이트 조작

- LUT00에서 임의의 오프셋을 선택
- KDF Slicing을 이용하여 오프셋으로부터 16바이트를 가져옴

lut00

0	30303030	30303030	30303030	30303030	30383038	30383038	30383038	30383038	31303130	31303130	31303130	31303130	31383138	31383138	31383138	31383138
64	32303230	32303230	32303230	32303230	32383238	32383238	32383238	32383238	33303330	33303330	33303330	33303330	33383338	33383338	33383338	33383338
128	34303430	34303430	34303430	34303430	34383438	34383438	34383438	34383438	35303530	35303530	35303530	35303530	35383538	35383538	35383538	35383538
192	36303630	36303630	36303630	36303630	36383638	36383638	36383638	36383638	37303730	37303730	37303730	37303730	37383738	37383738	37383738	37383738
256	38303830	38303830	38303830	38303830	38383838	38383838	38383838	38383838	39303930	39303930	39303930	39303930	39383938	39383938	39383938	39383938
320	41304130	41304130	41304130	41304130	41384138	41384138	41384138	41384138	42304230	42304230	42304230	42304230	42384238	42384238	42384238	42384238
384	43304330	43304330	43304330	43304330	43384338	43384338	43384338	43384338	44304430	44304430	44304430	44304430	44384438	44384438	44384438	44384438
448	45304530	45304530	45304530	45304530	45384538	45384538	45384538	45384538	46304630	46304630	46304630	46304630	46384638	46384638	46384638	46384638
512																

Signed Int | le, dec | (select some data)

443 out of 512 bytes

8.4 Curve25519 키 생성시 '클램핑'

NAVER
DEVIEW
2023

첫번째 바이트 조작

- KDF Concat을 사용하여 LUT00으로부터 가져온 16바이트의 키와 16바이트의 랜덤 SecureKey를 연결

Randomly picked from LUT00															
42	30	42	30	42	30	42	30	42	30	42	30	42	30	42	30

+

16bytes Random SecureKey															
FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF

8.4 Curve25519 키 생성시 '클램핑'

NAVER
DEVIEW
2023

첫번째 바이트 조작

- KDF Slice을 사용하여 15번째 인덱스로부터 16바이트의 데이터를 가져옴으로써, 첫번째 바이트를 조작

Random picked from LUT00															
42	30	42	30	42	30	42	30	42	30	42	30	42	30	42	30

16bytes Random SecureKey															
FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF

8.4 Curve25519 키 생성시 '클램핑'

NAVER
DEVIEW
2023

마지막 바이트 조작

- LUT31에서 임의의 오프셋을 선택
- KDF Slicing을 이용하여 오프셋으로부터 16바이트를 가져옴

lut31

0	34303431	34323433	34343435	34363437	34383439	34413442	34433444	34453446	35303531	35323533	35343535	35363537	35383539	35413542	35433544	35453546
64	36303631	36323633	36343635	36363637	36383639	36413642	36433644	36453646	37303731	37323733	37343735	37363737	37383739	37413742	37433744	37453746
128	34303431	34323433	34343435	34363437	34383439	34413442	34433444	34453446	35303531	35323533	35343535	35363537	35383539	35413542	35433544	35453546
192	36303631	36323633	36343635	36363637	36383639	36413642	36433644	36453646	37303731	37323733	37343735	37363737	37383739	37413742	37433744	37453746
256	34303431	34323433	34343435	34363437	34383439	34413442	34433444	34453446	35303531	35323533	35343535	35363537	35383539	35413542	35433544	35453546
320	36303631	36323633	36343635	36363637	36383639	36413642	36433644	36453646	37303731	37323733	37343735	37363737	37383739	37413742	37433744	37453746
384	34303431	34323433	34343435	34363437	34383439	34413442	34433444	34453446	35303531	35323533	35343535	35363537	35383539	35413542	35433544	35453546
448	36303631	36323633	36343635	36363637	36383639	36413642	36433644	36453646	37303731	37323733	37343735	37363737	37383739	37413742	37433744	37453746
512																

Signed Int | le, dec | (select some data)

352 bytes out of 512 bytes

8.4 Curve25519 키 생성시 '클램핑'

NAVER
DEVIEW
2023

마지막 바이트 조작

- KDF Concat을 사용하여 16바이트의 랜덤 SecureKey와 LUT31으로부터 가져온 16바이트의 키를 연결

16bytes Random SecureKey															
FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF

+

Random picked from LUT31															
34	32	34	33	34	34	34	35	34	36	34	37	34	38	34	39

8.4 Curve25519 키 생성시 '클램핑'

NAVER
DEVIEW
2023

마지막 바이트 조작

- KDF Slice을 사용하여 2번째 인덱스로부터 16바이트의 데이터를 가져옴으로써, 마지막 바이트를 조작

16bytes Random SecureKey															
FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF

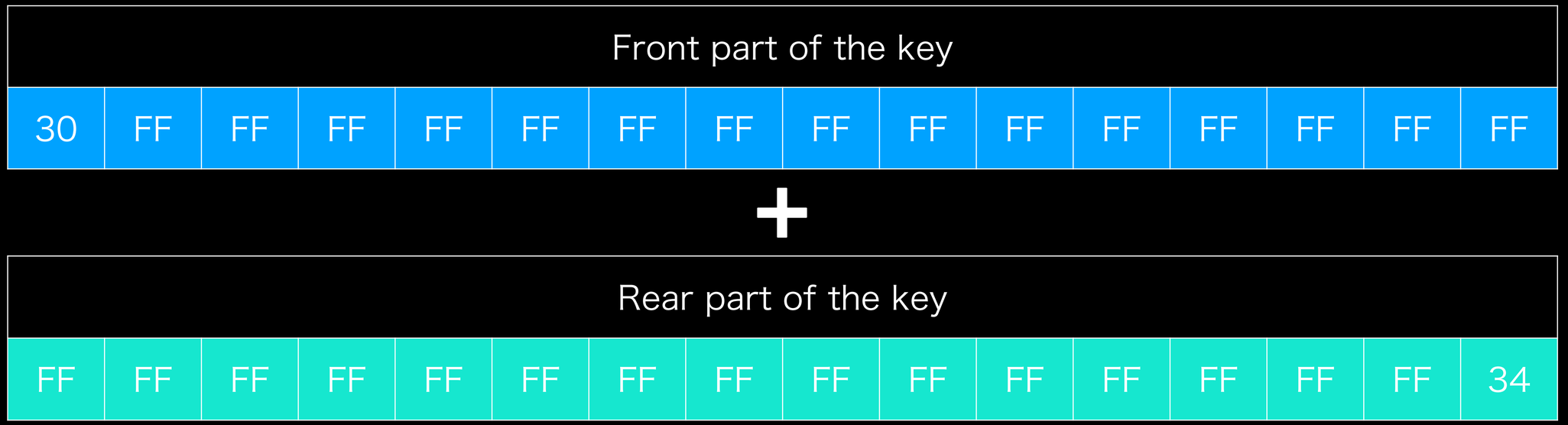
Random picked from LUT31															
34	32	34	33	34	34	34	35	34	36	34	37	34	38	34	39

8.4 Curve25519 키 생성시 '클램핑'

NAVER
DEVIEW
2023

두 키를 연결

- 첫번째 바이트가 조작된 16바이트 키와 마지막 바이트가 조작된 16바이트 키를 연결



8.4 Curve25519 키 생성시 '클램핑'

NAVER
DEVIEW
2023

클램핑 완료

Clamping done 32bytes curve25519 private key															
30	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF
FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	34

9.결론

9.1 구현

NAVER
DEVIEW
2023

- Workaround를 통해 임의의 바이트를 간접적으로 제어할 수 있었지만 구현이 매우 복잡해짐
- WebAssembly를 사용하면 화이트박스 구현을 웹 애플리케이션의 많은 보안 기능에 통합할 수 있음
- 보안 부서 내에서 일부 보안 기능을 개발하여 클라이언트 팀에 제공하는 것이 보안적으로 안전한 설계를 구축하는데 더 효율적
- 디버그 기능을 구현하면 화이트박스를 서비스에 통합하는데 많은 도움이 됨
  - 키의 해시값 출력
  - 고정 키 주입

9.2 안전한 설계

NAVER
DEVIEW
2023

- 화이트 박스 위협 모델 내에서 보안을 달성하는 것은 소프트웨어 구현만으로는 매우 어렵지만, 이것이 바로 화이트박스 암호가 달성하고자 하는 것
- 화이트박스 암호는 만능(All-in-one) 솔루션이 아니며 안전하지 않은 보안 설계를 안전하게 만드는 마법의 총알(Magic Bullet)도 아님
- 별도 디바이스의 하드웨어 보안 기능과 통합함으로써 웹 애플리케이션의 보안수준을 더욱 더 효과적으로 끌어올릴 수 있음
  - 인증: WebAuthn, FIDO2
  - 키 관리: WebAssembly + 화이트박스암호 + Cloud KMS/HSM

Q & A

Thank You