Android File System

BCG연구실 IT정보공학과 김아은

INDEX

Android File System

모바일 운영체제 종류

❖ 모바일 운영체제란?

- 스마트폰과 같은 기기를 구성하는 하드웨어 부품(메모리, LCD, CPU 등)을 효율적으로 관리하고 사용자와의 편리한 의사소통을 하기 위해 만들어진 소프트웨어 플랫폼
- 대표적인 모바일 OS : 애플의 iOS, 구글의 Android, MS의 Windows Phone 10

[전 세계]
1위 Android
2위 iOS
3위 Samsung

[대한민국]
1위 Android
2위 iOS
3위 Samsung

모바일 운영체제 종류

❖ 심비안 OS(Symbian)

- LTD에서 개발한 32비트 모바일 기기 운영체제
- 마이크로소프트사의 독점을 방지하기 위해 1998년 8개 업체(사이온, 노키아, 소니, 에릭슨, 지멘스, 삼성전자, 모토로라, 파나소닉)가 공동으로 컨소시엄을 결성하여 설립한 심비안사가 개발한 운영체제
- 아이폰, 안드로이드의 등장 이후 계속된 실적 하향 및 실패로 현재는 사용하지 않음

모바일 운영체제 종류

❖ 블랙베리 OS(BlackBerry)

- 캐나다 림(RIM) 사가 블랙베리 스마트폰을 위해 만든 소프트웨어 플랫폼
- 쿼티(QWERTY) 키보드 장착, 전자메일 실시간 수신 가능한 푸시 메일 기능 덕분에 업무용으로 직장인들이 애용
- 다중 작업 기능, 뛰어난 안정성, 빠른 응답 속도를 제공
- RIM이 채택한 트랙 휠, 트랙볼, 트랙패드(사용자의 손가락 동작을 감지하여 디지털 신호로 변환시키는 장치), 터치스크린과 같은 특화된 입력장치 지원

모바일 운영체제 종류

❖ iOS

- 맥 OS X 기반, 아이폰과 아이팟, 아이패드 등에 사용되는 애플의 전용 운영체제
- 2010년 아이폰 4가 발표되면서 기존 명칭이었던 'iPhone OS'에서 'iOS4'로 변경됨
- 특유의 디자인을 기본으로 실용성이 강조된 터치스크린과 아이콘 중심의 GUI를 제공
- 강화된 멀티터치(Multi Touch) 기능과 자이로 센서(물체의 이동을 감지하는 센서로 게임과 같은 다양한 모션에 활용됨), 멀티태스킹 기능도 제공
- iOS를 자체 기기에만 탑재하는 폐쇄성 → 국내 실정에 적합한 기능을 탑재하기 어렵고 주변 기기들과 호환성이 다소 부족

모바일 운영체제 종류

❖ 안드로이드 OS(Android)

- 모바일 운영체제와 미들웨어, 핵심 애플리케이션을 포함하는 소프트웨어 스택
- iOS와 달리 개방형 → HTC, 삼성전자 등 국내외 많은 제조사들이 경쟁적으로 안드로이드 기반의 다양한 제품 출시 중
- 각 버전의 코드명으로 음식 이름을 사용(버전 10 Queen Cake, 11 RVC(Red Velvet Cake), 12 Snow Cone)
- 빠른 반응 속도와 높은 편의성
- 다양한 애플리케이션들이 지속적으로 개발되고 있으며 주변 기기와의 호환성도 높은 편
- 개방성으로 인해 보안에 취약, 운영체제가 업데이트될 때마다 제품에 즉시 적용되지 못함

모바일 운영체제 종류

❖ 윈도우 폰 OS(Windows Phone)

- 마이크로소프트 사의 모바일 운영 체제인 윈도우 모바일(Windows Mobile)은 스마트폰과 MP3 플레이어 등의 포터블 미디어 기기에 사용됨 → 시장에서 외면 → 후속 버전인 윈도우 모바일 7(윈도우 폰 7) 발표
- 사용자 인터페이스 기능을 대폭 개선해 사용자의 편리성에 맞게 보완
  (ex. SNS인 트위터나 페이스북, MSN 메신저와의 연동 기능을 강화)
- 윈도우10에서 윈도우의 통합을 우선시하여 개발 중, 유니버설 앱에 대하여 윈도우 10과 완벽한 호환성을 갖추고 있음

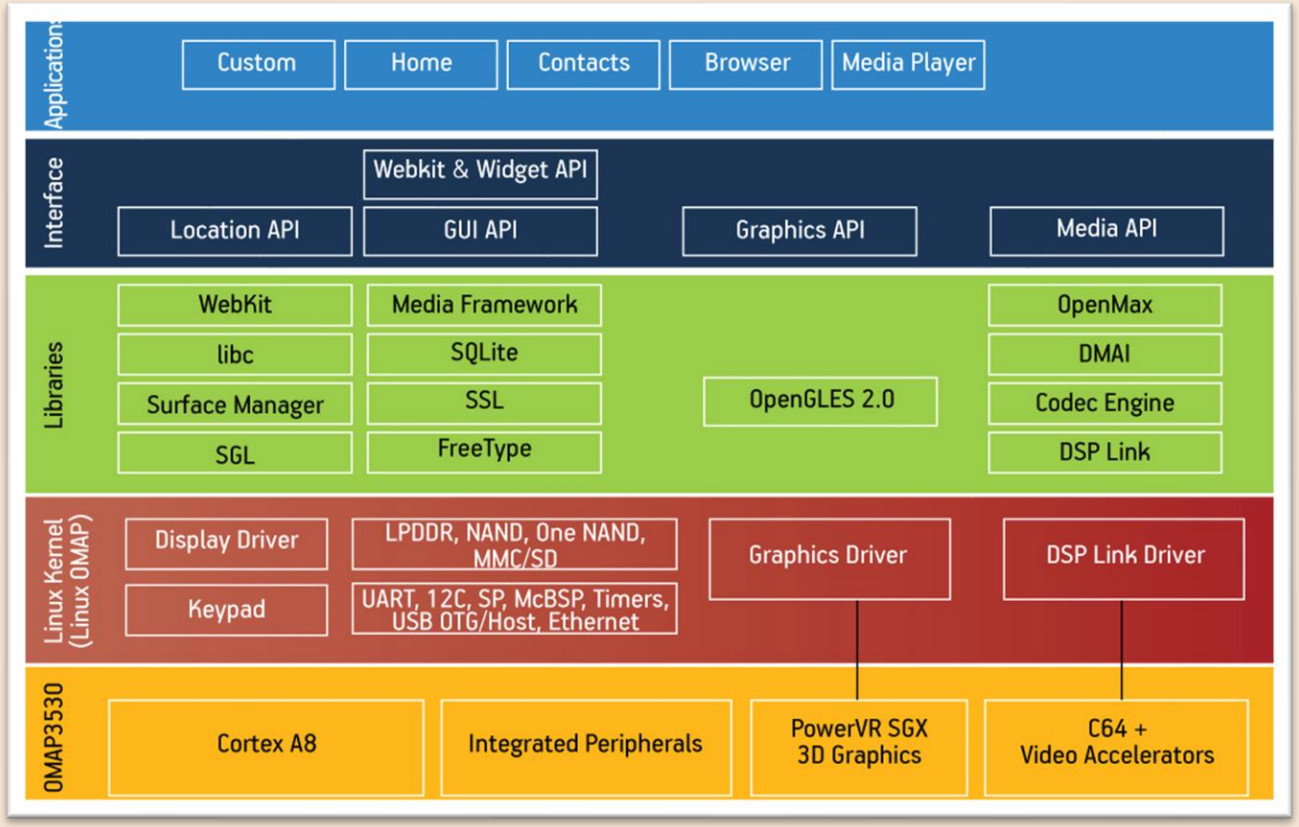
모바일 운영체제 종류

❖ 카이 OS(Kai)

- KaiOS Technologies에서 배포하는 모바일 운영체제
- 개발이 중단된 Firefox OS의 소스 코드를 저사양 피처폰에 맞게 경량화 → 설치 용량 200MB
- 4G와 Wifi를 지원, 매우 저렴한 가격
- 전용 앱 마켓인 'KaiOS Store' 운영, 구글 지도, 유튜브, 트위터, 왓츠앱 등이 등록되어 있음
- 웹브라우저는 출신에 맞게 '파이어폭스' 기반으로 설정됨

안드로이드 아키텍처

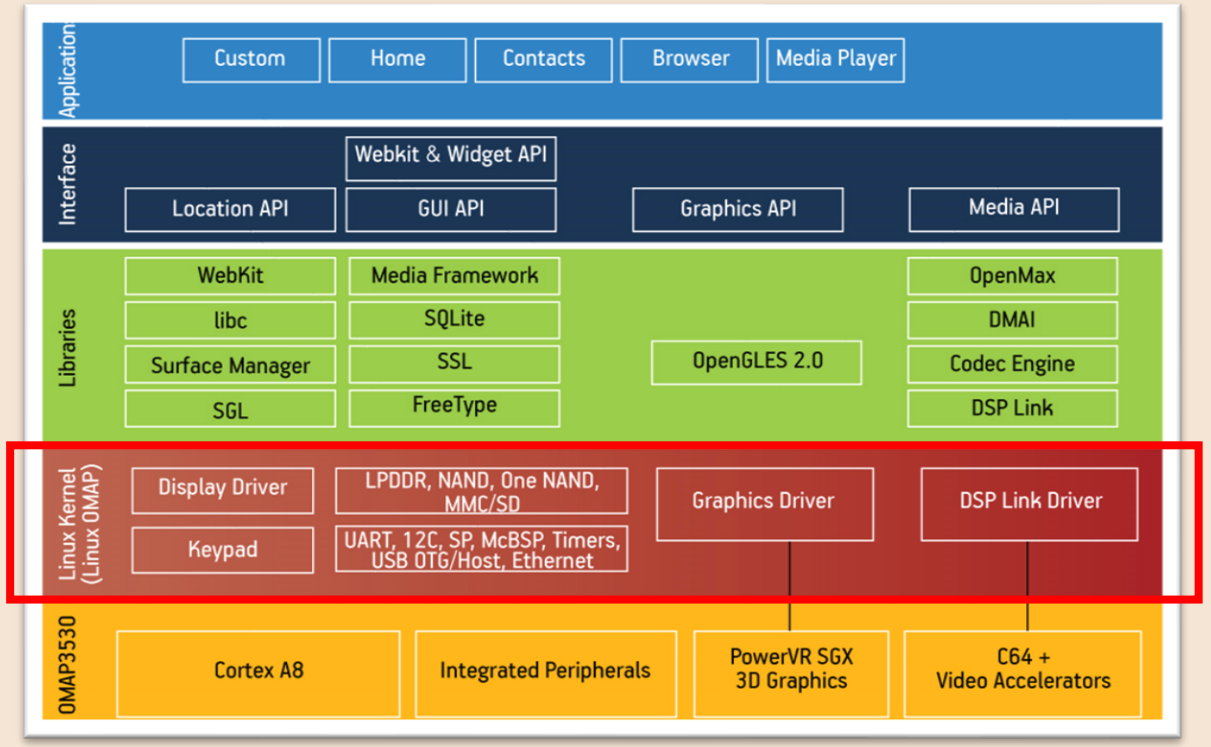
❖ 안드로이드 아키텍처



- 리눅스 기반의 운영체제

안드로이드 아키텍처

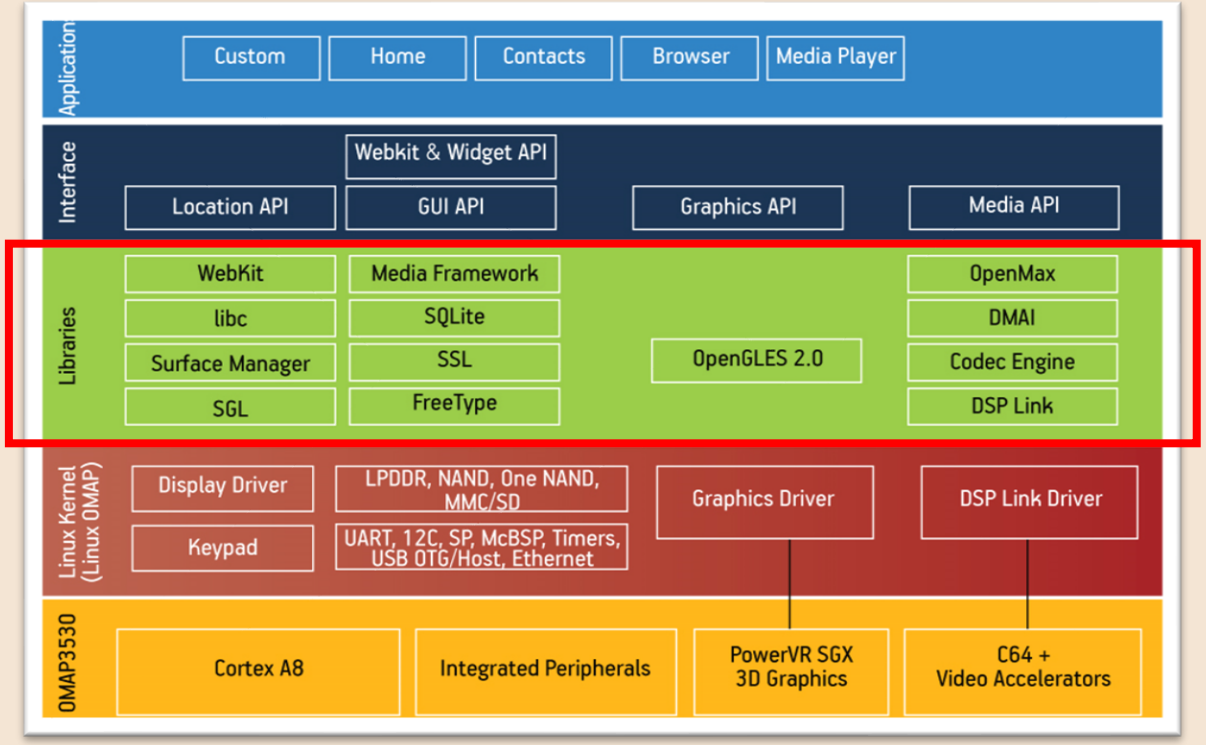
❖ 리눅스 커널



- 최하위 레이어는 리눅스 커널로 구성
- 카메라, 오디오, 무선 와이파이 등 다양한 드라이버로 구성됨
- 보안, 메모리 관리, 프로세스 관리 등 주요 시스템 서비스를 리눅스에 의존함

안드로이드 아키텍처

❖ 라이브러리



- 안드로이드의 네이티브 라이브러리
- C/C++ 언어로 작성됨
- 안드로이드 시스템의 다양한 컴포넌트가 사용됨
- 개발자들에게 안드로이드 애플리케이션 프레임워크를 통해 노출됨
- 해당 라이브러리들은 리눅스 커널 내에서 프로세스로 동작함

안드로이드 아키텍처

❖ 안드로이드 런타임

- apk의 일부인 bytecode를 CPU가 이해할 수 있는 machine code로 컴파일하는 역할

안드로이드 아키텍처

❖ 안드로이드 런타임

JIT(Just in Time) compilation

→ 당시에는 매우 부족했던
RAM 사용을 최적화 함

AOT(Ahead Of Time) compliation

→ 런타임 성능 개선(20배 ↑)

→ 설치 시간 ↑
RAM 사용량 ↑
시스템 업데이트 시간 ↑

Profile guided compilation

→ Hybrid JIT/AOT

→ RAM 사용량 ↓
앱의 주요 부분에 최상의 성능 제공

→ 앱 처음 사용 시 다소 속도 ↓
사전 컴파일 하기 위해
사용자가 실제로 앱을 사용해야 함

Profiles in the cloud

→ 대부분의 사용자가
앱을 매우 유사한 방식으로 사용함

→ Core Profile에 따라 사전 컴파일

안드로이드 파일 시스템 계층 구조

❖ 안드로이드 파일 시스템 구조

- 안드로이드 장치의 파일 시스템 구조는 리눅스 커널 위에 구축된 프레임워크

  → 리눅스와 유사점을 공유하지만, 안드로이드만의 독특한 특징도 가지고 있음
- 여러 branch가 있는 single tree와 같은 **단일 디렉터리의 파티션**을 사용함
- 안드로이드의 기본 6개 파티션
  1) /boot
  2) /system
  3) /recovery
  4) /data
  5) /cache
  6) /misc
- SD카드와 관련된 /sdcard, /sd-ext 파티션도 존재함

안드로이드 파일 시스템 계층 구조

❖ **/boot**

- **전원을 켤 때 안드로이드 장치를 부팅하는데 필요한 파티션**
- 안드로이드 커널 + ramdisk
- 복구(recovery) 시 반드시 필요한 경우에만 해당 파티션을 삭제해야 함
- 삭제 후 재부팅하려면 반드시 새 부팅 파티션을 다시 설치해야 함

❖ **/system**

- **안드로이드 OS 전체를 위해 제공되는 파티션**
- 안드로이드 GUI + 장치에 설치된 시스템 앱
- 장치를 recovery 또는 bootloader 모드로만 설정할 수 있음

안드로이드 파일 시스템 계층 구조

❖ **/recovery**

- **백업용으로 설계된 파티션**
- 장치를 복구(recovery) 모드로 부팅하여 데이터 백업 및 삭제, factory 설정으로 복원, 기타 유지관리 작업 수행

❖ **/data**

- **설정, 연락처, 앱 및 메시지를 포함한 모든 사용자 데이터로 구성되는 파티션**
- 해당 파티션을 삭제하면 장치의 모든 사용자 설정, 앱, 메시지를 제거한 상태로 factory 설정으로 저장됨

안드로이드 파일 시스템 계층 구조

❖ **/cache**

- **자주 접근하는 앱 데이터와 구성요소가 저장되는 파티션**
- 해당 파티션을 삭제하면 빌드된 캐시가 지워지고, 계속 사용하면 장치가 다시 빌드로 돌아감
- 캐시 삭제 시 장치의 일부 공간이 확보되고 때로는 특정 문제 해결할 수도 있음

❖ **/misc**

- **기타 모든 시스템 설정(일반적으로 on/off 스위치)이 포함됨**
- 설정에는 통신사 또는 지역 ID, USB 구성 및 특정 하드웨어 설정이 포함될 수 있음
- 해당 파티션이 손상되거나 누락되면 여러 장치 기능이 오작동할 수 있으므로 매우 중요함 (부팅이 안될 수도 있음!)

OWASP Mobile Top 10 2016

❖ OWASP Mobile Top 10

구분	한글명	영문명	
M1	부적절한 플랫폼 사용	Improper Platform Usage	루팅
M2	안전하지 않은 데이터 저장	Insecure Data Storage	로그인 정보 등
M3	안전하지 않은 통신	Insecure Communication	암호화 통신(계정 정보, 금융 정보)
M4	안전하지 않은 인증	Insecure Authentication	유료 서비스, 다른 사용자, 관리자 권한
M5	불충분한 암호화	Insufficient Cryptography	sha256 이상?
M6	안전하지 않은 권한	Insecure Authorization	앱마다 주어지는 권한 획득
M7	클라이언트 코드 품질	Client Code Quality	
M8	코드 변조	Code Tampering	악성코드, 무결성 검증
M9	역공학(리버스 엔지니어링)	Reverse Engineering	apk, iso 파일을
M10	불필요한 기능	Extraneous Functionality	불필요한 기능 배포(테스트 기능)

OWASP Mobile Top 10 2016

❖ M1: Improper Platform Usage(부적절한 플랫폼 사용)

- 플랫폼 기능들을 잘못 사용하거나 보안 통제를 미적용하여 문제가 발생하는 것 (※ 플랫폼 ≒ OS)
- 안드로이드의 경우, 개발자가 앱을 만들 때 필요한 권한을 명시하도록 하고, 설치할 때 사용자가 권한을 승인하도록 함
- 사용자는 권한을 전부 허가하거나 설치 자체를 거부해야 하기 때문에 취약점 발생 가능
- Threat
  1) 악의적인 앱이 과도한 권한을 가지게 될 경우
  2) (여러 역할을 분담한) 앱 간의 협력으로 공격하는 경우
  3) 여러 앱이 같은 시그니처를 사용하여 사용자 ID 공유하는 경우 → 권한도 함께 공유됨
- Mitigation
  1) 앱 개발 시 과도한 권한을 부여하지 않기
  2) 플랫폼 기능들을 사용할 때는 가이드에 따라 적절하게 사용하며 보안 통제 적용하기
  3) 앱을 설치할 때 불필요한 권한을 요청하는 건 아닌지 확인 후 동의하기

OWASP Mobile Top 10 2016

❖ M2: Insecure Data Storage(안전하지 않은 데이터 저장)

- 안전하지 않은 데이터 저장소에 중요한 데이터를 저장하여 의도하지 않은 데이터 유출이 발생하는 취약점
- 애플리케이션에서 ID/PW 입력 시, 계정 정보를 어딘가에 저장해야 함 → 암호화하지 않는 경우 !
- Threat
  1) 누군가가 모바일 기기를 컴퓨터에 연결 후 모바일 저장소에 접근해서 안전하지 않은 곳에, 평문으로 저장된 데이터를 확인할 경우
  2) 타 앱에서 접근이 가능한 데이터 영역에 저장했을 경우
- Mitigation
  1) 가장 좋은 방법은 중요 정보를 모바일 기기에 저장하지 않는 것
  2) 저장이 필요한 경우 안전한 암호 알고리즘을 기반으로 암호화하여 저장하기
  3) 사용 종료 시 관련 임시파일 삭제되도록 설정하기
  4) 메모리에 민감한 정보가 평문으로 저장되지 않도록 사용 후 관련 메모리 영역을 알 수 없는 값으로 초기화하거나, 민감한 정보가 삭제되도록 하기

OWASP Mobile Top 10 2016

❖ M3: Insecure Communication(안전하지 않은 통신)

- 서버와 서버간의 통신 시 중요 정보가 암호화 되지 않거나, 암호화 되더라도 취약한 SSL 버전을 사용하는 취약성
- Threat
  1) 사용자의 회원가입, 로그인 시도, 결제 시도 등 통신 중에 데이터를 암호화하지 않고 전달하는 경우
- Mitigation
  1) 데이터의 기밀성을 위해 항상 암호화 시켜 통신하기
  2) 구버전의 SSL을 사용해야 한다면 보안 전문가와 상의하기

OWASP Mobile Top 10 2016

❖ M4: Insecure Authentication(안전하지 않은 인증)

- 중요한 정보 제공 시 사용자를 안전하게 식별하지 못하는 취약점
- Threat
  1) 가용성을 높이기 위해 단순한 인증방식을 사용할 경우(4자리 PIN인증 등)
  2) 열악한 인증 체계로 인해 요청 값을 변조하여 세션 토큰을 제거하는 경우
- Mitigation
  1) 비밀번호나 세션을 클라이언트 측에 보관·사용하지 않고, 세션 토큰 사용시 반드시 1회성으로 사용 후 파기하기
  2) 단순한 비밀번호 사용하지 않기(최소한 8자리 이상의 영문+숫자 조합 사용)
  3) 2가지 이상의 인증 채널 활용하여 보안 강화하기(SMS나 이메일 인증코드 확인, CAPTCHA 사용)
  4) 앱 소스코드 난독화를 통해 인증과 관련된 코드에 대한 접근 차단 및 무결성 검사 수행하여 권한 없는 코드 변경 감지하기

OWASP Mobile Top 10 2016

❖ M5: Insufficient Cryptography(불충분한 암호화)

- M2- 불안전한 데이터 저장 : 중요 데이터 저장 시 암호화하지 않아 생기는 취약점
- 암호화를 했지만 취약한 알고리즘 또는 프로세스 결함을 이용하여 암호문을 원래의 데이터로 복호화 함으로써 중요 데이터가 노출되는 취약점
- Threat
  1) 공격자가 읽을 수 있는 디렉터리에 키를 관리하거나 바이너리에 키를 하드코딩한 경우
  2) 사용자 지정 알고리즘 생성 및 사용하는 경우
  3) 취약한 알고리즘을 사용하는 경우
- Mitigation
  1) 모바일 장치에 중요 데이터 저장 시 반드시 암호화 하되,
     향후 10년 동안의 시간 테스트에 견딜 수 있도록 충분한 키 길이를 가지는 검증된 암호화 표준 적용하기
     → KISA의 '암호 알고리즘 및 키 길이 이용 안내서' 참고

OWASP Mobile Top 10 2016

❖ M6: Insecure Authorization(안전하지 않은 권한)

- M4- 인증 : 개인을 식별하는 행위

  M6- 권한 : 인증을 마친 개인이 수행할 수 있는 범위를 확인하는 행위 → 불안전한 권한을 부여하여 발생하는 취약점
- Threat
  1) 클라이언트 측의 권한과 관련된 요청을 일방적으로 신뢰하는 경우
  2) 인증을 마친 후 특정 기능을 수행하는 사용자에 대한 권한 체크 미흡한 경우
  3) 권한 체크 로직의 부재로 인해 권한 없는 사용자 요청에 대해 서버가 응답하거나 앱에서 사용하지 않는 불필요한 테스트용 기능들을 제공하지 않을 경우
- Mitigation
  1) 중요 기능에 대한 사용자 요청 시 반드시 서버 측에서 권한을 체크하며, 사용자의 인증/권한 정보를 저장하여 확인
  2) 개발 시 불필요한 권한을 부여하지 않도록 개발
  3) 테스트용으로 사용한 activity는 개발 완료 후 배포 전에 삭제하여 불필요한 정보 노출 최소화

OWASP Mobile Top 10 2016

❖ M7: Client Code Quality(클라이언트 코드 품질)

- Client 모바일 장치에서 실행되는 코드의 구현 미흡으로 발생하는 취약점
- Threat
  1) 잘못된 API를 사용하는 경우
  2) API를 안전하지 못하게 사용하는 경우
  3) 안전하지 않은 언어 구문을 사용하는 경우
- Mitigation
  1) 조직의 모든 사람이 일관된 코딩 패턴으로 작성하기
  2) 읽기 쉽고 문서화된 코드 작성하기
  3) 정적 분석 도구를 사용하여 버퍼 오버플로우 및 메모리 누수 식별하기

OWASP Mobile Top 10 2016

❖ M8: Code Tampering(코드 변조)

- 코드 변조를 통해 생성된 악성 앱을 설치하여 발생하는 취약점
- 모바일 애플리케이션은 대체로 바이너리 코드와 데이터 리소스가 사용자 장치에 저장되는데
  공격자는 직접 코드를 수정하거나, 메모리 내용을 동적으로 변경하거나, 응용 프로그램이 사용하는 시스템 API를
  변경하여 애플리케이션의 데이터와 리소스를 수정할 수 있음
- Threat
  1) 코드 변조를 통해 생성된 악성 앱을 배포하여 중요 정보 탈취 및 악성코드 실행하는 경우
- Mitigation
  1) 소스코드 난독화 및 코드의 무결성 탐지와 안전한 앱스토어에서 배포되도록 하기
  2) 컴파일 당시 생성된 무결성 코드로 실시간 무결성 위반 감지하기
  3) 앱을 난독화하여 코드 변조 못하도록 하기

OWASP Mobile Top 10 2016

❖ M9: Reverse Engineering(역공학, 리버스 엔지니어링)

- 모바일 애플리케이션은 디컴파일이 가능하여 역공학에 취약함
- 안드로이드 앱의 경우 apk 확장자를 가지고 있는데, apk는 자바 jar의 확장이며 jar 파일은 zip 포맷을 가지고 있음
- AndroidManifest.xml : 앱의 패키지 이름, 컴포넌트, 권한 등 명시
- classes.dex : 안드로이드 환경에서 실행할 수 있는 dex 실행코드
- Threat
    1) 스트링 테이블을 분석하여 백엔드 DB의 인증정보를 얻어내는 경우
    2) 소스코드의 중요한 로직 및 정보 등이 노출되는 경우
    3) 악성코드를 삽입 후 리패키징하여 악용되는 경우
- Mitigation
    1) 바이너리 암호화
    2) 코드 난독화

OWASP Mobile Top 10 2016

❖ M10: Extraneous Functionality(불필요한 기능)

- 애플리케이션에 불필요하거나 관계없는 기능이 남겨져있어 공격자가 악용할 수 있는 취약점
- Threat
  1) 백엔드 시스템 작동 방식의 노출과 허가되지 않은 최고 권한 실행하는 경우
  2) 숨겨진 스위치, 테스트 코드, 주석에 달아 놓은 비밀번호 등
- Mitigation
  1) 앱의 구성 설정 검사
  2) 테스트 코드가 최종 버전의 앱에 포함되어 있지는 않은지 확인
  3) 모든 API 엔드포인트를 검사
  4) 모든 로그를 검사하여 과도한 설명이 되어있지 않은지 확인

감사합니다