(19) 대한민국특허청(KR)
(12) 공개특허공보(A)

(11) 공개번호 10-2016-0082016
(43) 공개일자 2016년07월08일

(51) 국제특허분류(Int. Cl.)
G06F 21/62 (2013.01) *G06F 21/50* (2013.01)
(21) 출원번호 10-2014-0193609
(22) 출원일자 2014년12월30일
심사청구일자 2014년12월30일

(71) 출원인
고려대학교 산학협력단
서울특별시 성북구 안암로 145, 고려대학교 (안암동5가)
(72) 발명자
최진영
서울특별시 노원구 동일로248길 16 506-1003(상계동, 수락파크빌아파트)
강성욱
경기도 용인시 수지구 죽전로 152, 단대원룸 205호 (죽전동)
이환택
인천광역시 남동구 동암남로20번길 34-2, B01호 (간석동)
(74) 대리인
특허법인충현

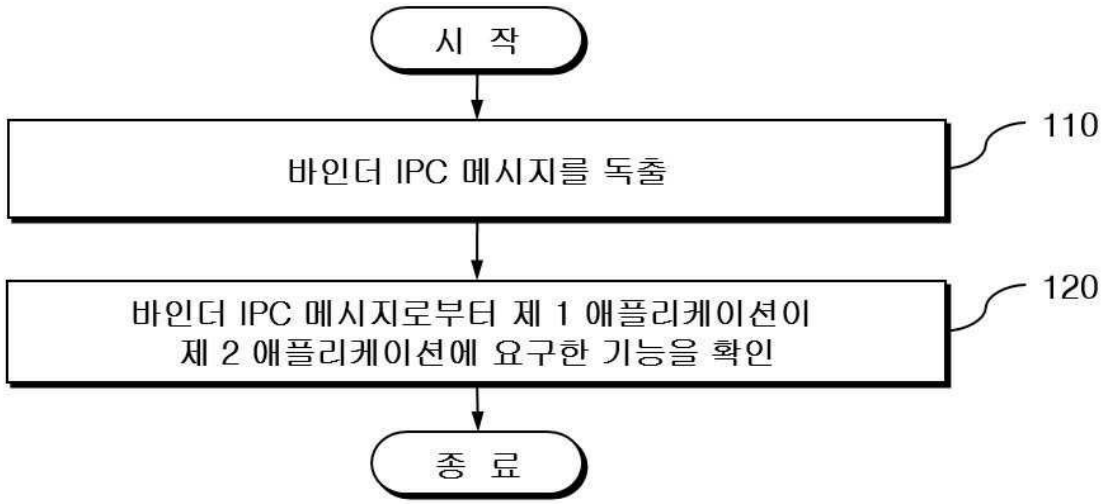
전체 청구항 수 : 총 6 항

(54) 발명의 명칭 **안드로이드 프로세스간 통신 모니터링을 이용한 개인정보 유출 탐지 기법**

(57) 요 약

본 발명은 프로세스간 통신 모니터링 방법에 관한 것으로, 바인더 IPC 메시지를 독출하는 단계, 및 상기 바인더 IPC 메시지로부터 제 1 애플리케이션이 제 2 애플리케이션에 요구한 기능을 확인하는 단계를 포함하고, 상기 바인더 IPC 메시지는 커널 영역에서 독출되는 것을 특징으로 함으로써, 임의의 앱으로부터 개인정보가 유출되는 것을 탐지하고, 이를 경고할 수 있다.

대 표 도 - 도1



이 발명을 지원한 국가연구개발사업
과제고유번호 1711015820
부처명 미래창조과학부
연구관리전문기관 정보통신산업진흥원
연구사업명 대학 IT 연구센터육성 지원사업
연구과제명 고품질 융합 소프트웨어 개발지원 도구 연구
기 여 율 1/1
주관기관 고려대학교 산학협력단
연구기간 2013.09.01 ~ 2016.12.31

명 세 서

청구범위

청구항 1

프로세스간 통신 모니터링 방법에 있어서,

바인더 IPC 메시지를 독출하는 단계; 및

상기 바인더 IPC 메시지로부터 제 1 애플리케이션이 제 2 애플리케이션에 요구한 기능을 확인하는 단계;

상기 바인더 IPC 메시지는 커널 영역에서 독출되는 것을 특징으로 하는 방법.

청구항 2

제 1 항에 있어서,

상기 바인더 IPC 메시지를 독출하는 단계는,

커널 프로브를 이용하여 상기 제 1 애플리케이션의 바인더 드라이버에 대한 시스템 콜을 독출함으로써 수행되는 것을 특징으로 하는 방법.

청구항 3

제 1 항에 있어서,

상기 바인더 IPC 메시지를 독출하는 단계는,

상기 제 1 애플리케이션의 시스템 콜을 분석하여 바인더 IPC와 관련된 시스템 콜을 분류하여 독출하는 것을 특징으로 하는 방법.

청구항 4

제 1 항에 있어서,

상기 제 1 애플리케이션이 제 2 애플리케이션에 요구한 기능을 확인하는 단계는,

상기 바인더 IPC 메시지에 포함된 서비스 클라이언트에 연결된 서비스, 해당 서비스에 있는 기능, 바인더 IPC를 송신한 프로세스, 및 해당 서비스를 사용하기 위한 매개 변수를 분석하여 수행되는 것을 특징으로 하는 방법.

청구항 5

제 1 항에 있어서,

상기 요구한 기능에 개인정보가 포함된 경우, 상기 제 1 애플리케이션이 외부 네트워크와 소켓을 여는 시스템 콜을 독출하여, 상기 제 1 애플리케이션이 외부 네트워크에 상기 개인정보를 유출하는 지를 확인하는 단계를 더 포함하는 방법.

청구항 6

제 1 항에 있어서,

상기 확인된 기능이 상기 제 1 애플리케이션으로부터 선언된 기능인지를 확인하는 단계를 더 포함하고,

상기 제 1 애플리케이션이 요구하는 기능들이 상기 제 1 애플리케이션으로부터 선언된 기능에 포함되는 경우, 상기 제 1 애플리케이션이 상기 제 1 애플리케이션으로부터 선언된 기능을 모두 요구하지 않더라도 상기 제 1 애플리케이션은 악성 애플리케이션이 아닌 것으로 확인하는 것을 특징으로 하는 방법.

발명의 설명

기 술 분 야

[0001] 본 발명은 프로세스간 통신 모니터링 방법에 관한 것으로서, 바인더 IPC 메시지에 대한 모니터링을 이용하여 개인정보 유출을 탐지하는 방법에 관한 것이다.

배 경 기 술

[0002] 프로세스간 통신을 모니터링하기 위한 연구가 활발히 진행중에 있다. 이와 관련된 종래의 기술은 다음과 같다.

[0003] 첫 번째는 'Quire' 제목의 논문으로 안드로이드 보안모델의 취약점을 이용하여 일종의 권한 상승 공격과 유사한 결과를 발생시키는 'Confused Deputy Attack'을 방지하기 위한 기술이다. 개인 정보를 요청하는 앱 및 프로세스의 권한을 체인 형태로 관리하여 개인 정보를 제공하는 앱에서 요청하는 앱 및 프로세스들의 권한이 존재하는 체인을 확인한 후, 요청 권한이 없을 경우, 정보 제공을 금지시키는 방법이다. 그러나 이 기법은 Confused Defuty 공격에 대해서만 유효할 뿐, 여러 종류의 악성코드의 행위 분석 및 모니터링은 불가능하다.

[0004] 두 번째는 'Aurasium' 제목의 논문으로 유저 레벨에서 일종의 샌드박스 형태로 동작하게 하여 앱의 각종 행위를 모니터링 하고 내부 정책으로 악성 행위를 방어하는 기법이다. 그러나 이 기법이 동작하려면 결국 기존 앱을 재패키징 해야 하는데, 현재 기술력 상 재 패키징 확률이 100%가 아니기 때문에 기법 적용을 못 할 수 있다.

[0005] 세 번째는 'TaintDroid' 제목의 논문으로 개인 정보 유출을 모니터링 하기 위한 기법이다. 플랫폼의 다양한 부분을 변경하여 개인정보의 근원인 Taint Source부터 개인정보 유출지인 Taint Sync 까지의 흐름을 추적하는 기술이다. 그러나 이를 적용하기 위해서는 플랫폼 전반과 커널의 소스코드를 수정하고, 다시 빌드를 해야 하므로, 실제 적용하기는 힘들다.

[0006] 본 발명과 관련된 선행기술로는 '무선 액세스 포인트 장치 및 그를 이용한 네트워크 트래픽침입탐지 및 차단방법(한국공개특허: 10-2007-0054067)' 등이 있다.

발명의 내용

해결하려는 과제

[0007] 본 발명이 해결하고자 하는 과제는 바인더 IPC 메시지에 대한 모니터링을 이용하여 개인정보 유출을 탐지하는 방법을 제공하는 것이다.

과제의 해결 수단

[0008] 본 발명은 상기 과제를 해결하기 위하여, 프로세스간 통신 모니터링 방법에 있어서, 바인더 IPC 메시지를 독출하는 단계; 및 상기 바인더 IPC 메시지로부터 제 1 애플리케이션이 제 2 애플리케이션에 요구한 기능을 확인하는 단계; 상기 바인더 IPC 메시지는 커널 영역에서 독출되는 것을 특징으로 하는 방법을 제공한다.

[0009] 본 발명의 다른 실시예에 의하면, 상기 바인더 IPC 메시지를 독출하는 단계는, 커널 프로브를 이용하여 상기 제 1 애플리케이션의 바인더 드라이버에 대한 시스템 콜을 독출함으로써 수행되는 것을 특징으로 하는 방법일 수 있다.

[0010] 본 발명의 다른 실시예에 의하면, 상기 바인더 IPC 메시지를 독출하는 단계는, 상기 제 1 애플리케이션의 시스템 콜을 분석하여 바인더 IPC와 관련된 시스템 콜을 분류하여 독출하는 것을 특징으로 하는 방법일 수 있다.

[0011] 본 발명의 다른 실시예에 의하면, 상기 제 1 애플리케이션이 제 2 애플리케이션에 요구한 기능을 확인하는 단계는, 상기 바인더 IPC 메시지에 포함된 서비스 클라이언트에 연결된 서비스, 해당 서비스에 있는 기능, 바인더 IPC를 송신한 프로세스, 및 해당 서비스를 사용하기 위한 매개 변수를 분석하여 수행되는 것을 특징으로 하는 방법일 수 있다.

[0012] 본 발명의 다른 실시예에 의하면, 상기 요구한 기능에 개인정보가 포함된 경우, 상기 제 1 애플리케이션이 외부 네트워크와 소켓을 여는 시스템 콜을 독출하여, 상기 제 1 애플리케이션이 외부 네트워크에 상기 개인정보를 유출하는 지를 확인하는 단계를 더 포함하는 방법일 수 있다.

[0013] 본 발명의 다른 실시예에 의하면, 상기 확인된 기능이 상기 제 1 애플리케이션으로부터 선언된 기능인지를 확인하는 단계를 더 포함하고, 상기 제 1 애플리케이션이 요구하는 기능들이 상기 제 1 애플리케이션으로부터 선언된 기능에 포함되는 경우, 상기 제 1 애플리케이션이 상기 제 1 애플리케이션으로부터 선언된 기능을 모두 요구

하지 않더라도 상기 제 1 애플리케이션은 악성 애플리케이션이 아닌 것으로 확인하는 것을 특징으로 하는 방법일 수 있다.

발명의 효과

[0014] 본 발명에 따르면, 커널 기반의 모니터링 기법으로, 임의의 앱으로부터 개인정보가 유출되는 것을 탐지하고, 이를 경고할 수 있다. 또한, 악성 앱의 개인정보 유출을 탐지하고 알림으로써, 사용자들의 안드로이드에 대한 신뢰 회복할 수 있다.

도면의 간단한 설명

[0015] 도 1은 본 발명의 일 실시예에 따른 프로세스간 통신 모니터링 방법의 흐름도이다.

도 2 내지 3은 본 발명의 다른 실시예에 따른 프로세스간 통신 모니터링 방법의 흐름도이다.

도 4 내지 5는 프로세스간 바인더 IPC를 수행하는 과정을 도시한 것이다.

도 6은 개인정보 유출여부를 확인하는데 이용되는 인자들을 나타낸 것이다.

도 7 내지 8은 현재 기기의 위치를 유출하려는 것을 탐지하는 예를 나타낸 것이다.

발명을 실시하기 위한 구체적인 내용

[0016] 본 발명에 관한 구체적인 내용의 설명에 앞서 이해의 편의를 위해 본 발명이 해결하고자 하는 과제의 해결 방안의 개요 또는 기술적 사상의 핵심을 우선 제시한다.

[0017] 본 발명의 일 실시예에 따른 프로세스간 통신 모니터링 방법은, 바인더 IPC 메시지를 독출하는 단계, 및 상기 바인더 IPC 메시지로부터 제 1 애플리케이션이 제 2 애플리케이션에 요구한 기능을 확인하는 단계를 포함하고, 상기 바인더 IPC 메시지는 커널 영역에서 독출되는 것을 특징으로 한다.

[0018] 이하 첨부된 도면을 참조하여 본 발명이 속하는 기술 분야에서 통상의 지식을 가진 자가 본 발명을 용이하게 실시할 수 있는 실시 예를 상세히 설명한다. 그러나 이들 실시예는 본 발명을 보다 구체적으로 설명하기 위한 것으로, 본 발명의 범위가 이에 의하여 제한되지 않는다는 것은 당업계의 통상의 지식을 가진 자에게 자명할 것이다.

[0019] 본 발명이 해결하고자 하는 과제의 해결 방안을 명확하게 하기 위한 발명의 구성을 본 발명의 바람직한 실시예에 근거하여 첨부 도면을 참조하여 상세히 설명하되, 당해 도면에 대한 설명시 필요한 경우 다른 도면의 구성요소를 인용할 수 있음을 미리 밝혀둔다. 아울러 본 발명의 바람직한 실시 예에 대한 동작 원리를 상세하게 설명함에 있어 본 발명과 관련된 공지 기능 또는 구성에 대한 구체적인 설명 그리고 그 이외의 제반 사항이 본 발명의 요지를 불필요하게 흐릴 수 있다고 판단되는 경우, 그 상세한 설명을 생략한다.

[0020] 도 1은 본 발명의 일 실시예에 따른 프로세스간 통신 모니터링 방법의 흐름도이다.

[0021] 본 발명의 일 실시예에 따른 프로세스간 통신 모니터링 방법을 구현하기 위하여, 바인더 IPC의 메시지를 모니터링하여 특정 애플리케이션이 어떤 서비스 및 기능을 요구하는지를 확인한다.

[0022] 본 발명의 실시예에 따른 프로세스간 통신 모니터링 방법이 구현되는 환경은 도 4와 같다. 사용자 영역(440)의 제 1 애플리케이션(410)과 제 2 애플리케이션(420)이 커널영역(450)의 바인더 드라이버(430)를 통해 바인더 IPC 통신을 수행하는 환경으로, 제 1 애플리케이션(410)의 시스템 콜(460)을 분석하여 제 1 애플리케이션(410)이 제 2 애플리케이션(420)에 요구하는 기능을 모니터링할 수 있다.

[0023] 프로세스간 통신 중 바인더(Binder) IPC(Inter-Process Communication)는 모든 프로세스가 공유하는 커널 메모리에 요청할 내용과 응답받을 내용을 기록하고, 각 프로세스가 그 메모리 주소를 참조하는 형식으로 메모리 복사 오버헤드를 최소화하는 통신 방법이다. 안드로이드에서 프로세스간 통신에 사용되는 기법이다.

[0024] 안드로이드에서 바인더(Binder)는 각각 독립된 프로세서들을 연결해 주는 역할을 한다. 리눅스에서 시스템의 기능을 이용하기 위해서 시스템 콜(System Call)을 사용하여 시스템에서 제공하는 프로세서, 파일 시스템 기능을 이용하도록 제공하고 있다. 하지만 안드로이드에서는 각 독립적으로 운영되는 프로세스, 특히 서비스의 기능을

이용할 수 있도록 제공하는 것이 바인더의 핵심이다.

[0025] 바인더 통신이 이루어지기 위해서는, 각 프로세스가 매핑한 메모리 주소와 커널에서 매핑한 메모리 주소를 변환시켜 줄 수 있어야 있어야하므로, 이를 위하여 커널 영역에 바인더 드라이버(Binder Driver)가 존재한다. 상기 바인더 드라이버가 커널 영역에 존재하기 때문에 사용자 모드에서 동작하는 애플리케이션은 리눅스에서 제공하는 시스템 콜을 호출하여 통신을 수행할 수 있다.

[0026] 110 단계는 바인더 IPC 메시지를 독출하는 단계이다.

[0027] 보다 구체적으로, 필요한 정보를 확인하기 위하여, 바인더 IPC 메시지를 독출한다. 상기 바인더 IPC 메시지를 커널 영역에서 독출함으로써 사용자 영역에 영향을 주지 않으면서 필요한 정보를 독출한다. 상기 바인더 IPC 메시지는 제 1 애플리케이션이 제 2 애플리케이션과 통신을 수행하기 위해 이용하는 메시지로, 상기 바인더 IPC 메시지를 모니터링함으로써 제 1 애플리케이션이 제 2 애플리케이션에 요구하는 기능을 알 수 있다.

[0028] 상기 바인더 IPC 메시지를 독출하기 위하여, 커널 프로브를 이용하여 제 1 애플리케이션의 바인더 드라이버에 대한 시스템 콜을 독출할 수 있다. 커널은 다른 애플리케이션과 달리 디버깅하기 쉽지 않다. 커널을 수정하고 다시 빌드하여 사용하는 경우, 빌드하는데 엄청난 시간이 소요된다. 이를 해결하기 위하여, 동적 프로브(Dynamic probe)를 이용한다. 동적 프로브는 커널을 다시 빌드하지 않고 동적으로 디버깅하는 방법으로, 커널 루틴에 원하는 코드를 디버깅하거나 패치하는 작업도 가능하다. 어떤 특정 함수가 실행될 때 특정로그를 남기거나 함수를 수정할 수도 있다. 동적 프로브에는 커널프로브(Kprobe), 점퍼프로브(Jprobe), 또는 리턴프로브(Kreprobe) 등이 있다. 커널 프로브(Kprobes(Kernel Dynamic Probes))는 동적 프로브로 커널에 특정 가상의 인스트럭션을 삽입한다. 커널 프로브는 리눅스 커널의 디버깅 메커니즘으로 리눅스 커널 내부의 이벤트 등을 모니터링 할 수 있다. 주로 커널 내부에서 수행되는 함수를 후킹(hooking) 하는데 사용되며, 커널 프로브를 이용하여 제 1 애플리케이션의 바인더 드라이버에 대한 시스템 콜을 독출할 수 있다. 제 1 애플리케이션은 리눅스에서 제공하는 ioctl() 시스템 콜을 호출할 수 있다. 시스템 콜들은 기존 커널 소스코드와 함께 컴파일되어, 커널 심볼이 존재하는 함수이기 때문에 '/proc/kallsyms'에서 심볼 정보를 추출할 수 있고, 추출한 심볼 정보를 기반으로 커널 프로브(Kprobes)로 시스템 콜을 모니터링하는 모듈을 작성할 수 있다.

[0029] 상기 바인더 IPC 메시지를 독출하는 단계는 상기 제 1 애플리케이션의 시스템 콜을 분석하여 바인더 IPC와 관련된 시스템 콜을 분류하여 독출할 수 있다. 안드로이드 플랫폼에서 ioctl()은 바인더 외에도 다른 용도로도 호출되기 때문에, 분석량을 줄이기 위해선 분석에 필요한 호출만 뽑아내야한다. 우선, 바인더 ioctl()의 인자를 분석한 후 이를 이용하여 바인더 IPC와 관련된 시스템 콜을 분류하여 독출한다. 바인더 ioctl() 메시지 구조는 하기와 같다.

[0030] ioctl("/dev/binder" , BINDER_WRITE_READ, struct binder_write_read)

[0031] 바인더의 ioctl()은 "/dev/binder"를 첫 번째 인자로, BINDER_WRITE_READ를 두 번째 인자로 받도록 되어있다. 이를 이용하여 바인더 IPC를 위한 ioctl()을 걸러 낼 수 있다. 바인더에서 세 번째 인자 'arg'는 binder_write_read 구조체로, 바인더 프로토콜에 따라서 내용이 변한다. 바인더 프로토콜은 데이터 전송 방향에 따라 도 5와 같이 사용된다. 도 5를 참조하면, 바인더 IPC 프로토콜 BC_TRANSACTION 프로토콜이 호출될 경우만 확인해도 프로세스 A가 프로세스 B에게 어떤 기능을 요구했는지 확인할 수 있으므로, 분석할 내용을 더 줄일 수 있다.

[0032] 120 단계는 상기 바인더 IPC 메시지로부터 제 1 애플리케이션이 제 2 애플리케이션에 요구한 기능을 확인하는 단계이다.

[0033] 보다 구체적으로, 바인더 IPC 메시지에 포함된 정보를 분석하여 제 1 애플리케이션이 제 2 애플리케이션에 어떠한 기능을 요구하는지를 확인한다. 제 1 애플리케이션의 제 2 애플리케이션에 대해 요구한 기능을 확인하기 위하여, 상기 바인더 IPC 메시지에 포함된 서비스 클라이언트에 연결된 서비스, 해당 서비스에 있는 기능, 바인더 IPC를 송신한 프로세스, 및 해당 서비스를 사용하기 위한 매개 변수를 분석할 수 있다.

[0034] 110 단계에서 BC_TRANSACTION 프로토콜을 호출하는 ioctl()이 추출되고, 추출된 ioctl()의 3번째 인자인 binder_write_read 구조체는 도 6과 같다. binder_transaction_data 구조체에서 handle, code, sender_pid,

data 이 4개의 인자(410)를 분석한다. 상기 각 인자의 기능은 다음 표 1과 같다.

표 1

[0035]

변수명	기능
handle	서비스 클라이언트에 연결된 서비스 구별
code	해당 서비스에 있는 기능 구별
sender_pid	바인더 IPC를 송신한 프로세스 확인
data	해당 서비스를 사용하기 위한 매개 변수

[0036] 상기 4개의 인자를 분석함으로써 제 1 애플리케이션이 제 2 애플리케이션에 요구하는 기능을 확인할 수 있다.

[0037] 도 2 내지 3은 본 발명의 다른 실시예에 따른 프로세스간 통신 모니터링 방법의 흐름도이다.

[0038] 210 단계는 120 단계에서 확인된 제 1 애플리케이션이 요구한 기능에 개인정보가 포함된 경우, 상기 제 1 애플리케이션이 외부 네트워크와 소켓을 여는 시스템 콜을 독출하여, 상기 제 1 애플리케이션이 외부 네트워크에 상기 개인정보를 유출하는 지를 확인하는 단계이다.

[0039] 보다 구체적으로, 제 1 애플리케이션이 제 2 애플리케이션에 요구한 기능에 개인정보가 포함되는지 여부를 통해 개인정보가 유출되는지를 모니터링할 수 있다. 제 1 애플리케이션이 정당한 권한을 가지고 개인정보를 요구하였는지 아니면 개인정보를 외부로 유출하기 위해 개인정보를 요구하였는지를 확인하기 위하여, 제 1 애플리케이션이 외부 네트워크와 소켓을 여는 시스템 콜을 하였는지를 확인할 수 있다. 개인정보를 요구하고 외부 네트워크와 소켓을 여는 경우 개인정보를 유출하려는 것으로 판단할 수 있다.

[0040] 바인더 IPC를 통하여 개인정보를 수집한 악성 애플리케이션이 외부 네트워크로 수집한 정보를 유출시키기 위해선 Connect() 시스템 콜을 활용하여 외부 네트워크와 소켓을 열어야 한다. 그러나 Conncet() 시스템 콜은 외부 네트워크 통신(TCP/IP) 뿐만 아니라 내부 프로세스끼리의 통신(IPC), 커널 수준과 사용자 수준의 통신(NETLINK)에도 쓰일 수 있기 때문에 이 역시 필터링을 할 필요가 있다. Connect() 시스템 콜의 매개변수인 sockaddr 구조체는 수많은 소켓 주소 체계를 호환하여 사용하기 위한 범용 구조체로, 각 용도에 따라 포함되는 데이터나 크기가 달라진다. 외부와의 연결에 필요한 sockaddr 구조체는 sockaddr_in 구조체로 16 Byte이기 때문에 이 크기를 활용하여 외부와 연결하는 Connect() 만을 모니터링 할 수 있다.

[0041] 이하, 개인 정보(위치 정보) 유출 탐지하는 예로써 어떤 애플리케이션 com.example.sms_puller가 LocationManager를 이용해 현재 기기의 위치를 유출하려는 것을 탐지하는 것을 이용하여 자세히 설명하도록 한다. 도 7은 하나의 예시로 작성한 프로그램의 소스코드이다. 위치 정보를 현재 위치로 갱신하기 위하여 requestLocationUpdate API를 호출한다. 본 발명의 실시예에 따른 프로세스간 통신 모니터링 방법이 적용된 기기에서는 이러한 개인정보에 관련된 연산이 일어날 경우, 바인더 IPC 모니터링에 의하여 도 8과 같은 로그를 출력한다. LoacationManager 인터페이스의 RPC Code 1은 requestLoacationUpdates 이므로, 특정 애플리케이션 com.example.sms_puller가 기기 위치 연산을 호출한 것을 탐지했다는 결과를 알 수 있다. 특정 애플리케이션이 개인 정보에 접근한 것만으로는 개인정보가 외부로 유출된다고 보기는 어렵다. 그러나 Binder IPC 모니터링에 특정 애플리케이션이 개인정보에 접근한 상태에서 Connect() 모니터링에서 외부 네트워크와 소켓 통신을 확인할 경우, 개인 정보가 외부 네트워크로 유출될 가능성이 높다고 판단한다. 기기 위치 정보뿐만 아니라, 주소록, IMEI, SMS 등의 민감한 정보의 유출 탐지에도 활용이 가능하다.

[0042] 310 단계는 상기 확인된 기능이 상기 제 1 애플리케이션으로부터 선언된 기능인지를 확인하는 단계이다.

[0043] 보다 구체적으로, 110 단계 및 120 단계를 통해 제 1 애플리케이션이 요구하는 기능들을 모니터링함으로써 제 1 애플리케이션이 악성 애플리케이션인지를 확인할 수 있다. 악성 애플리케이션이 아닌 경우에도, 애플리케이션이 선언한 기능과 다르게 기능을 요구하는 경우, 악성 애플리케이션으로 판단될 수 있다. 이러한 문제점을 해결하기 위하여, 제 1 애플리케이션이 요구하는 기능들을 모니터링하고, 모니터링되는 제 1 애플리케이션의 요구 기능을 확인하여, 제 1 애플리케이션이 악성 애플리케이션인지를 확인할 수 있다.

[0044] 상기 제 1 애플리케이션이 요구하는 기능들이 상기 제 1 애플리케이션으로부터 선언된 기능에 포함되는 경우, 상기 제 1 애플리케이션이 상기 제 1 애플리케이션으로부터 선언된 기능을 모두 요구하지 않더라도 상기 제 1 애플리케이션은 악성 애플리케이션이 아닌 것으로 확인할 수 있다.

[0045] 본 발명의 실시예에 따른 프로세스간 통신 모니터링 방법은 프로세스간 통신 모니터링 장치에 의해 구현될 수 있다. 프로세스간 통신 모니터링 장치는 메시지 등을 송수신하는 하나 이상의 통신부, 정보를 독출하고 처리하는 하나 이상의 처리부, 및 통신부에서 수신하거나 처리부에서 처리된 정보를 저장하는 하나 이상의 저장부로 구성될 수 있다. 프로세스간 통신 모니터링 장치는 사용자 단말이거나 별도의 장치일 수 있다. 프로세스간 통신 모니터링 장치에서 수행되는 과정에 대한 상세한 설명은 도 1 내지 도 3의 프로세스간 통신 모니터링 방법에 대한 상세한 설명에 대응하는바, 도 1 내지 도 3의 프로세스간 통신 모니터링 방법에 대한 상세한 설명으로 대신한다.

[0046]

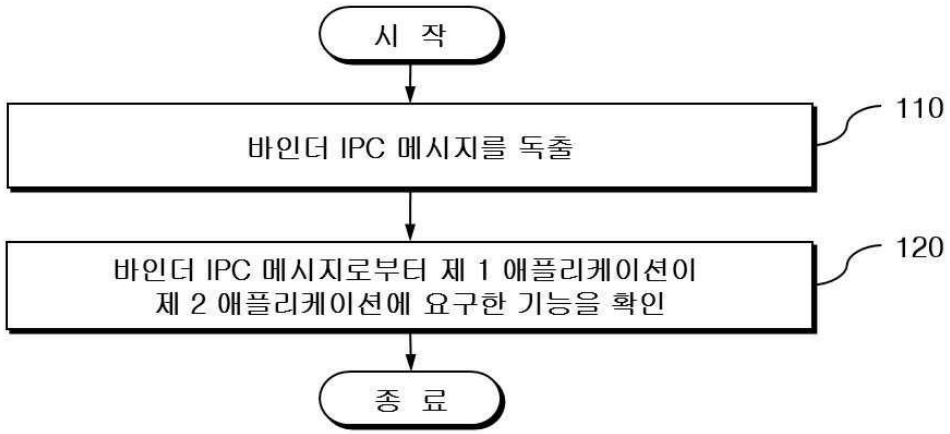
[0047] 본 발명의 실시예들은 다양한 컴퓨터 수단을 통하여 수행될 수 있는 프로그램 명령 형태로 구현되어 컴퓨터 판독 가능 매체에 기록될 수 있다. 상기 컴퓨터 판독 가능 매체는 프로그램 명령, 데이터 파일, 데이터 구조 등을 단독으로 또는 조합하여 포함할 수 있다. 상기 매체에 기록되는 프로그램 명령은 본 발명을 위하여 특별히 설계되고 구성된 것들이거나 컴퓨터 소프트웨어 당업자에게 공지되어 사용 가능한 것일 수도 있다. 컴퓨터 판독 가능 기록 매체의 예에는 하드 디스크, 플로피 디스크 및 자기 테이프와 같은 자기 매체(magnetic media), CD-ROM, DVD와 같은 광기록 매체(optical media), 플롭티컬 디스크(floptical disk)와 같은 자기-광 매체(magneto-optical media), 및 롬(ROM), 램(RAM), 플래시 메모리 등과 같은 프로그램 명령을 저장하고 수행하도록 특별히 구성된 하드웨어 장치가 포함된다. 프로그램 명령의 예에는 컴파일러에 의해 만들어지는 것과 같은 기계어 코드뿐만 아니라 인터프리터 등을 사용해서 컴퓨터에 의해서 실행될 수 있는 고급 언어 코드를 포함한다. 상기된 하드웨어 장치는 본 발명의 동작을 수행하기 위해 하나 이상의 소프트웨어 모듈로서 작동하도록 구성될 수 있으며, 그 역도 마찬가지이다.

[0048] 이상과 같이 본 발명에서는 구체적인 구성 요소 등과 같은 특정 사항들과 한정된 실시예 및 도면에 의해 설명되었으나 이는 본 발명의 보다 전반적인 이해를 돕기 위해서 제공된 것일 뿐, 본 발명은 상기의 실시예에 한정되는 것은 아니며, 본 발명이 속하는 분야에서 통상적인 지식을 가진 자라면 이러한 기재로부터 다양한 수정 및 변형이 가능하다.

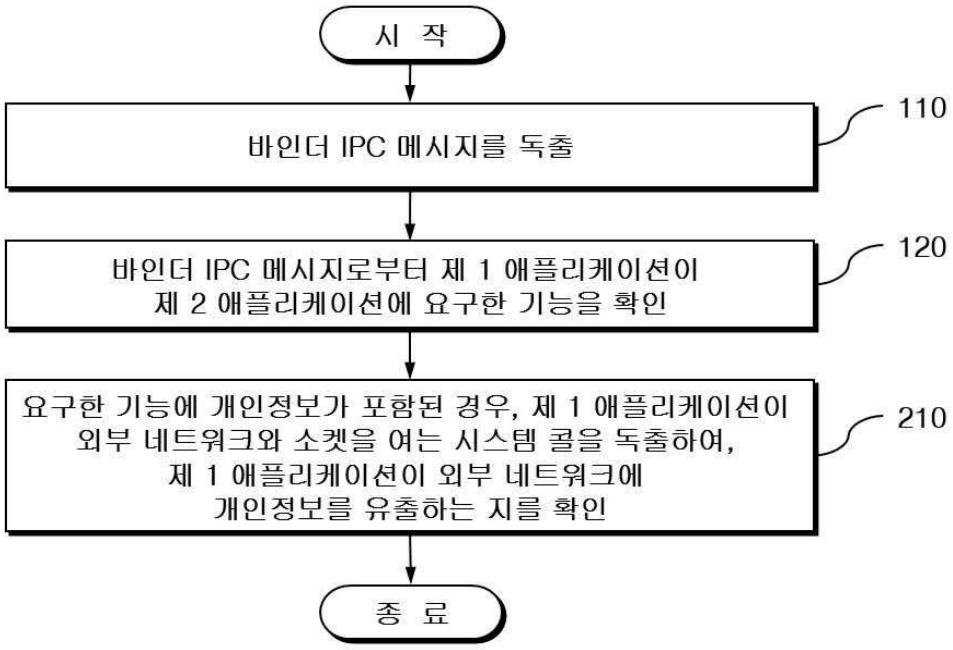
[0049] 따라서, 본 발명의 사상은 설명된 실시예에 국한되어 정해져서는 아니되며, 후술하는 특허청구범위뿐 아니라 이 특허청구범위와 균등하거나 등가적 변형이 있는 모든 것들은 본 발명 사상의 범주에 속한다고 할 것이다.

도면

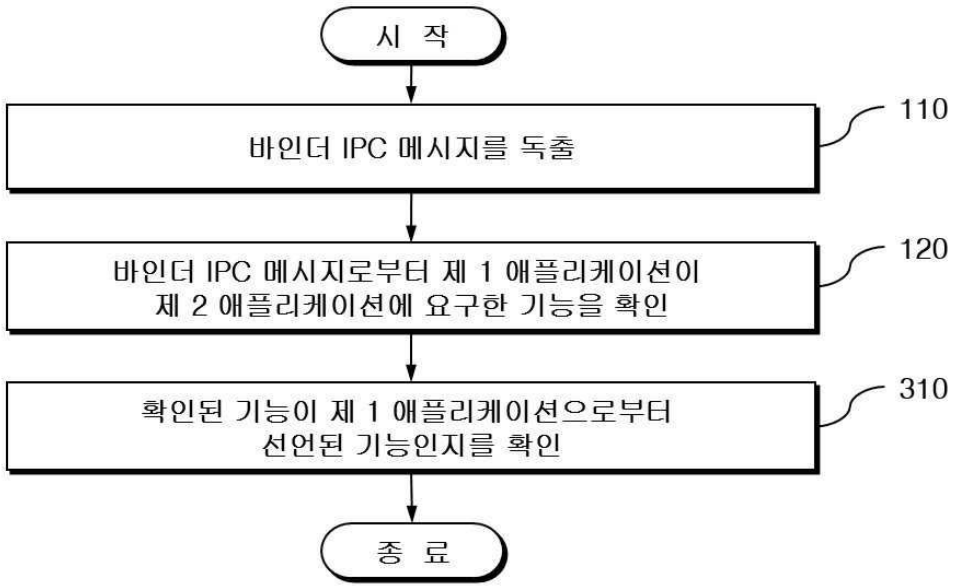
도면1



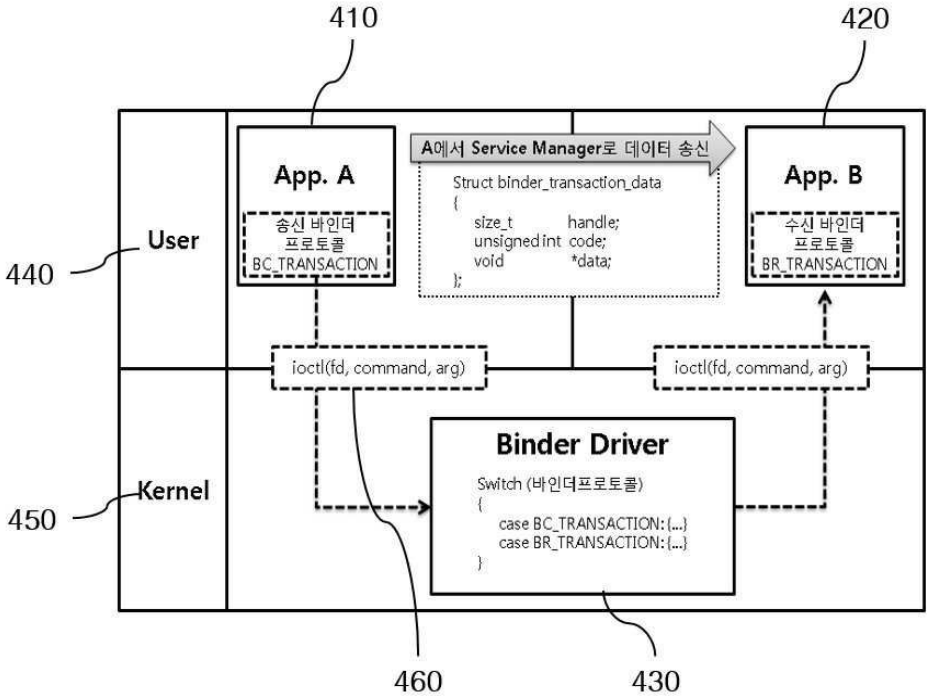
도면2


시 작
바인더 IPC 메시지를 독출
110
바인더 IPC 메시지로부터 제 1 애플리케이션이
제 2 애플리케이션에 요구한 기능을 확인
120
요구한 기능에 개인정보가 포함된 경우, 제 1 애플리케이션이
외부 네트워크와 소켓을 여는 시스템 콜을 독출하여,
제 1 애플리케이션이 외부 네트워크에
개인정보를 유출하는 지를 확인
210
종 료

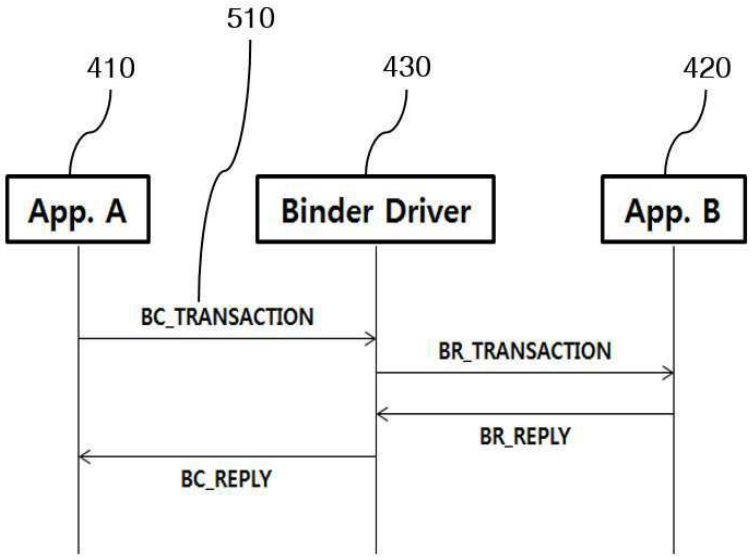
도면3


시 작
바인더 IPC 메시지를 독출
110
바인더 IPC 메시지로부터 제 1 애플리케이션이
제 2 애플리케이션에 요구한 기능을 확인
120
확인된 기능이 제 1 애플리케이션으로부터
선언된 기능인지를 확인
310
종 료

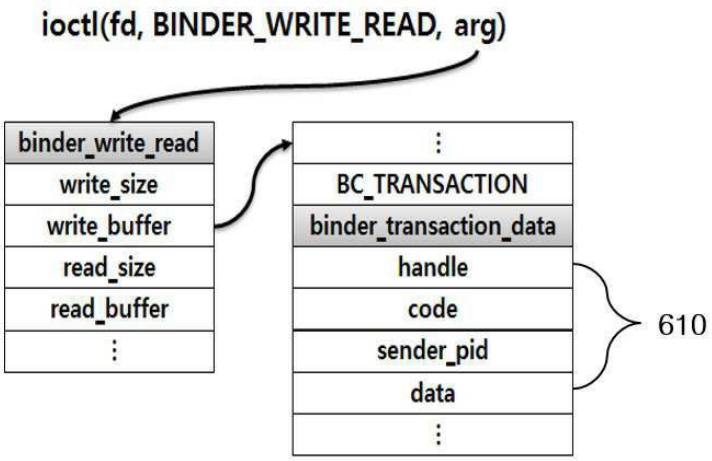
도면4


410
420
App. A
송신 바인더
프로토콜
BC_TRANSACTION
A에서 Service Manager로 데이터 송신
Struct binder_transaction_data
{
size_t handle;
unsigned int code;
void *data;
};
App. B
수신 바인더
프로토콜
BR_TRANSACTION
User
440
ioctl(fd, command, arg)
ioctl(fd, command, arg)
Binder Driver
Switch (바인더프로토콜)
{
case BC_TRANSACTION: {...}
case BR_TRANSACTION: {...}
}
Kernel
450
460
430

도면5


510
410
430
420
App. A
Binder Driver
App. B
BC_TRANSACTION
BR_TRANSACTION
BR_REPLY
BC_REPLY

도면6



도면7

```
private void get_location() {
    LocationManager locationManager = (LocationManager)getSystemService(Context.LOCATION_SERVICE);

    LocationListener listener = new LocationListener() {
        @Override
        public void onLocationChanged(Location location) {
            Log.d("get_location","LOCATION:"+location.getLatitude());
        }
        public void onStatusChanged(String provider, int status,
        public void onProviderEnabled(String provider) {
        public void onProviderDisabled(String provider) {
    };
    locationManager.requestLocationUpdates(LocationManager.NETWORK_PROVIDER, 0, 0, listener);

}
```

도면8

```
----- RC_TRANSACTION -----
-SENDER-          com.example.sms_puller
-RECEIVER-        system_server

----- BUFFER -----
Interface = android.location.ILocationManager
RPC Code = 1

BYTE | 00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F
-----+-------------------------------------------------
0000 | 04 03 00 00 21 00 00 00 61 00 6e 00 64 00 72 00   ....!...a.n.d.r.
0010 | 6f 00 69 00 64 00 2e 00 6c 00 6f 00 63 00 61 00   o.i.d...l.o.c.a.
0020 | 74 00 69 00 6f 00 6e 00 2e 00 49 00 4c 00 6f 00   t.i.o.n...I.L.o.
0030 | 63 00 61 00 74 00 69 00 6f 00 6e 00 4d 00 61 00   c.a.t.i.o.n.M.a.
0040 | 6e 00 61 00 67 00 65 00 72 00 00 00 01 00 00 00   n.a.g.e.r.......
0050 | c9 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................
0060 | 00 00 00 00 ff ff ff ff ff ff ff 7f ff ff ff 7f   ................
0070 | 00 00 00 00 00 00 00 00 07 00 00 00 6e 00 65 00   ............n.e.
0080 | 74 00 77 00 6f 00 72 00 6b 00 00 00 ff ff ff ff   t.w.o.r.k.......
0090 | 85 2a 62 73 7f 01 00 00 78 77 56 76 58 77 56 76   .*bs....xwVvXwVv
00A0 | 00 00 00 00 16 00 00 00 63 00 6f 00 6d 00 2e 00   ........c.o.m...
00B0 | 65 00 78 00 61 00 6d 00 70 00 6c 00 65 00 2e 00   e.x.a.m.p.l.e...
00C0 | 73 00 6d 00 73 00 5f 00 70 00 75 00 6c 00 6c 00   s.m.s._.p.u.l.l.
00D0 | 65 00 72 00 00 00 00 00                           e.r.....
00E0 |
```