



(19) 대한민국특허청(KR)
(12) 공개특허공보(A)

(11) 공개번호 10-2013-0118335
(43) 공개일자 2013년10월29일

- (51) 국제특허분류(Int. Cl.)
G06F 21/00 (2006.01) G06F 11/30 (2006.01)
G06F 11/22 (2006.01)
- (21) 출원번호 10-2013-7014101
- (22) 출원일자(국제) 2011년11월03일
심사청구일자 없음
- (85) 번역문제출일자 2013년05월31일
- (86) 국제출원번호 PCT/US2011/059244
- (87) 국제공개번호 WO 2012/061663
국제공개일자 2012년05월10일
- (30) 우선권주장
61/409,670 2010년11월03일 미국(US)
61/475,713 2011년04월15일 미국(US)
- (71) 출원인
버지니아 테크 인터렉추얼 프라퍼티스, 인크.
미국, 버지니아 24060, 블랙스버그, 스위트
1050, 크라프트 드라이브 2200
- (72) 발명자
리드, 제프리, 에이치.
미국 24060 버지니아주 블랙스버그 이하트
에스티. 1105
아구아요 곤잘레스, 카를로스, 알.
미국 24060 버지니아주 블랙스버그 아파트먼트 하
이즈 디알. 케이-4 308
- (74) 대리인
백만기, 양영준, 정은진

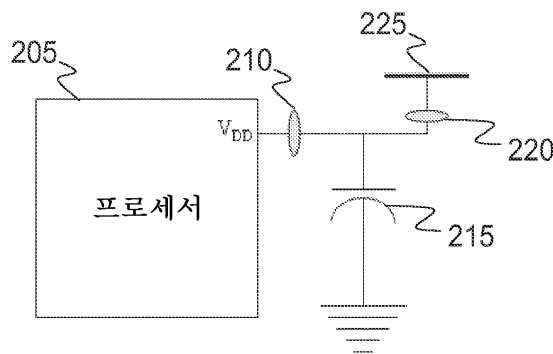
전체 청구항 수 : 총 13 항

(54) 발명의 명칭 컴퓨터 기반 시스템의 무결성을 모니터하고 보안을 강화하기 위한 전력 핑거프린팅(PFP)의 이용

(57) 요약

전력 핑거프린팅(PFP)을 포괄적인 침입방호 보안의 검출 단계에 통합하고; PFP 사용가능한 노드들의 네트워크를 설치하고; 특정한 상태 시퀀스와 특정한 소프트웨어 실행을 강제하는 미리 정의된 입력으로 비신뢰 장치들을 실행하고; 모듈 식별 정보를 동기화 시그널링에 임베드하고; 다양한 보드 요소들로부터의 신호들을 결합하고; PFP 성능을 향상시키기 위해 멀웨어 서명들을 이용하고; 자동 특성화 및 서명 추출을 실행하고; 안전한 서명 갱신을 제공하고; 사이드-채널 공격으로부터 보호하고; 이들의 동적 전력 소비를 모니터하고 이를 신뢰 코드로부터의 서명에 비교함으로써 내장된 플랫폼에서 실시간 무결성 평가를 실행하고; 플랫폼의 전력 소비를 미리 특성화하고 내부 실행 상황에 대한 가장 많은 정보를 전달하는 트레이스 섹션들에 집중하고; 디지털 프로세서에서 소프트웨어의 인증된 실행으로부터의 편차를 검출하기 위해 비트 트랜지션들의 시퀀스로부터의 PFP를 이용함으로써, PFP에 의해 판정된 타겟 시스템 실행 무결성을 향상시키기 위한 절차가 설명된다.

대표도 - 도2



특허청구의 범위

청구항 1

컴퓨터 처리 플랫폼에서 루틴의 실행의 실시간 무결성 평가를 수행하기 위한 방법으로서,

상기 루틴의 실행 동안에 샘플링에 의해 프로세서의 전력 소비를 추적(tracing)함으로써 상기 루틴의 실행을 모니터링하는 단계;

플랫폼 특성화 기술을 이용하는 단계 - 상기 플랫폼 특성화 기술을 이용하는 단계는 상기 프로세서에서 상태 트랜지션들(state transitions)에 대한 가장 큰 의존을 나타내는 트레이스들의 섹션들을 검출하는 단계와, 상기 섹션들을 이용하여 가장 많은 정보를 전달하는 특징들을 선택하는 단계를 더 포함함 -;

상기 섹션들에 포함된 상기 루틴의 선택된 특징들의 특성화로부터 상기 루틴의 신뢰 전력 핑거프린트들(trusted power fingerprints)의 세트를 획득하는 단계;

상기 신뢰 핑거프린트들로 구성된 서명으로부터의 거리의 확률 분포에 기초하여 특정 거짓 경보 레이트(false alarm rate)에 대한 임계치를 설정하는 단계;

상기 신뢰 핑거프린트들의 라이브러리와 비신뢰 코드의 실행으로부터의 트레이스들로부터 추출된 특징들을 비교하는 단계;

상기 핑거프린트들과 상기 추출된 특징들 간의 거리를 결정하는 단계; 및

상기 거리가 상기 임계치를 초과하는 경우, 예외(exception)를 보고하는 단계를 포함하는 방법.

청구항 2

제1항에 있어서, 모듈 식별 정보를 상기 루틴에 임베드함으로써 상기 루틴의 실행과 상기 추적을 동기화시키는 단계를 더 포함하는 방법.

청구항 3

제2항에 있어서, 상기 모듈 식별 정보는 상기 루틴의 실행 전에 IO 레지스터에 기록된 이진 식별 코드인 방법.

청구항 4

제2항에 있어서, 상기 모듈 식별 정보는 구별되는 전력 소비 패턴을 산출하는 명령어들의 시퀀스인 방법.

청구항 5

제1항에 있어서, 상기 전력 소비의 추적은 복수의 프로세서 회로로부터의 신호들을 결합하는 방법.

청구항 6

제1항에 있어서, 상기 라이브러리의 핑거프린트에 공지된 멀웨어의 서명들을 부가함으로써 상기 예외의 보고의 품질을 향상시키는 단계를 더 포함하는 방법.

청구항 7

컴퓨터 처리 플랫폼에서 루틴의 실행의 실시간 무결성 평가를 수행하기 위한 시스템으로서,

상기 루틴의 실행 동안에 샘플링에 의해 프로세서의 전력 소비를 추적함으로써 상기 루틴의 실행을 모니터링하는 수단;

플랫폼 특성화 기술을 이용하는 수단 - 상기 플랫폼 특성화 기술을 이용하는 수단은 상기 프로세서에서 상태 트랜지션들에 대한 가장 큰 의존을 나타내는 트레이스들의 섹션들을 검출하는 수단과, 상기 섹션들을 이용하여 가장 많은 정보를 전달하는 특징들을 선택하는 수단을 더 포함함 -;

상기 섹션들에 포함된 상기 선택된 특징들의 특성화로부터 상기 루틴의 신뢰 전력 핑거프린트들의 세트를 획득하는 수단;

상기 신뢰 핑거프린트들로 구성된 서명으로부터의 거리의 확률 분포에 기초하여 특정 거짓 경보 레이트에 대한 임계치를 설정하는 수단;

상기 신뢰 핑거프린트들의 라이브러리와 비신뢰 코드의 실행으로부터의 트레이스들로부터 추출된 특징들을 비교하는 수단;

상기 핑거프린트들과 상기 추출된 특징들 간의 거리를 결정하는 수단; 및

상기 거리가 상기 임계치를 초과하는 경우, 예외를 보고하는 수단

을 포함하는 시스템.

청구항 8

제7항에 있어서, 상기 루틴에 임베드된 모듈 식별 정보를 더 포함하는 시스템.

청구항 9

제8항에 있어서, 상기 모듈 식별 정보는 상기 루틴의 실행 전에 IO 레지스터에 기록된 이진 식별 코드인 시스템.

청구항 10

제8항에 있어서, 상기 모듈 식별 정보는 구별되는 전력 소비 패턴을 산출하는 명령어들의 시퀀스인 시스템.

청구항 11

제7항에 있어서, 상기 전력 소비를 추적하는 수단은 복수의 프로세서 회로로부터의 신호들을 결합하는 시스템.

청구항 12

제7항에 있어서, 상기 라이브러리의 핑거프린트에 공지된 멀웨어의 서명들을 부가함으로써 상기 예외의 보고의 품질을 향상시키는 수단을 더 포함하는 시스템.

청구항 13

디지털 프로세서에서 소프트웨어의 인증된 실행으로부터의 편차를 검출하기 위한 방법으로서,

공지된 설정에서 상기 소프트웨어의 적어도 하나의 루틴의 실행 동안에 상기 프로세서의 전력 소비를 관측하는 단계;

상기 적어도 하나의 루틴의 전력 소비를 특성화하는 비트 트랜지션들의 시퀀스를 결정하는 단계; 및

편차를 결정하기 위해 상기 소프트웨어의 관측된 실행과의 비교를 위한 핑거프린트로서 상기 전력 소비의 특성화를 이용하는 단계

를 포함하는 방법.

명세서

기술분야

[0001] 본 발명은 일반적으로 전력 핑거프린팅(power fingerprinting)에 관한 것이며, 특히 컴퓨터-기반 시스템에 이용된 컴퓨터 프로세싱 및 소프트웨어의 무결성을 모니터하고 보안을 강화하기 위한 기술로서 전력 핑거프린팅의 이용에 관한 것이다.

배경기술

[0002] CMOS 디지털 회로에서는, 매 비트 트랜지션(transition)마다, 게이트에서의 짧은 단락 회로와 이 회로의 출력에서 기생 용량의 충전 및 방전으로 인해 생기는 과도 전류 드레인(transient current drain)이 있다. 프로세서

에서, 이들 과도 전류의 세기, 따라서 특정한 클록 사이클에서 소비되는 전체 전력은 해당 사이클에서 나타나는 전체 비트 트랜지션의 수에 의해 결정된다. 비트 트랜지션들의 수는 실행되는 특정한 명령 시퀀스(specific instruction sequence)는 물론이고 그들의 어드레스 및 파라미터에 의해 결정된다. 전력 핑거프린팅은 이상 검출을 위해서 프로세서의 전력 소비에 대해 세립 측정(fine-grained measurement)을 한 다음 이들을 신뢰 서명(trusted signature; 실행 동안 특정 비트 트랜지션들의 시퀀스의 결과로 나타나는 패턴)에 비교하는 것을 기초로 한 중요한 사이버 시스템을 위한 무결성 평가 및 침입 검출 솔루션이다. 전력 핑거프린팅의 배후가 되는 기본적인 접근법은 동일한 코드가 실행되고 있는지 여부를 판단하기 위하여 신뢰 소프트웨어의 실행을 특성화시킨 다음 그의 전력 서명(power signature)들을 추출하고, 이들을 테스트 트레이스(test trace)들을 비교하기 위한 기준(reference)으로 이용하는 것이다.

[0003] 전력 핑거프린팅(PFP) 모니터는 도 1에 도시된 바와 같이 모든 패턴 인식 시스템에 공통인 3개의 메인 요소: 감지(110), 특징(feature) 추출(120) 및 검출/분류(130)로 이루어진다. 감지는 디지털 하드웨어의 순간 전류 드레인을 측정하는 것을 포함하는데, 이는 상업용의 전류 프로브 및 고성능 오실로스코프를 이용하여 성취될 수 있다. 특징 추출은 PFP를 위한 중요한 요소이며 주어진 소프트웨어 루틴의 실행을 고유하게 식별하는 전력 소비의 통계적 및 일시적 속성들의 식별을 포함한다. 이는 프로세서의 아키텍처와 소프트웨어의 구조에 대한 깊은 이해를 요구하는 도전적인 과제이지만, 이는 소프트웨어 자체를 서명을 향상시키고 판단을 개선하는 특정한 특성들로 구축하면 가능해질 수 있다. 이상적으로는, 서명은 코드에 있는 모든 실행 경로로부터 추출된다. 이것이 가능하지 않은 경우에는, OS 커널 모듈 및 코어 애플리케이션들과 같이 단지 소수의 중요 섹션들만이 특성화되어 모니터링된다.

[0004] 일반적인 전력 핑거프린팅 접근법에서, 센서(110)는 가능한 한 전원 핀들에 가깝게 프로세서의 보드에 배치된다. 센서는 프로세서의 순간 전류 드레인을 캡처(capture)한다. 센서는 상업적인 전류 프로브, 분류기(shunt resistor), 또는 전류 미러일 수 있다. 센서로부터의 신호는 프로세서의 메인 클록 속도보다 높은 속도로 디지털화되어야만 한다. 프로세서가 동작 주파수를 증가시키기 위하여 내부 위상-고정 루프를 갖추고 있다면, 이는 유효 클록 주파수가 된다. 3.5X유효 클록 주파수를 이용하여 만족스러운 결과가 얻어졌지만, 이는 하계(lower bound)를 나타내지 않는다. 샘플링 요구조건을 줄이기 위해 몇몇 메커니즘을 이용할 수 있다.

[0005] 순간 전류 드레인이 전력 트레이스로 디지털화된 후, 트레이스들로부터 차별적인 특징을 추출하기 위해 다양한 신호 처리 기술이 적용된다. 특징들이 추출된 후에, 이들은 신뢰 소프트웨어로부터의 트레이스들(140)을 이용하여 이미 훈련된 감시 분류기(supervised classifier) 또는 검출기(130)를 통과한다. 이 검출기는 궁극적으로 소프트웨어 실행이 인증된 소프트웨어에 대응하는지 여부를 판정한다. 종래 기술의 일반적인 전력 핑거프린팅 접근법에 대한 도식적인 묘사를 도 1에 나타내었다.

[0006] 특정한 전력 트레이스로부터의 특징들이 인증된 실행에 대응하는지 여부에 대한 판정은 세심하게 설계된 검출기에 의해 실행되며, 이 검출기는 들어오는 전력 트레이스들을 인증된 코드로부터의 모든 저장된 서명들(140)에 비교한다. 관측된 트레이스들이, 합리적인 허용치 안에서, 저장된 서명들 중 어떤 것에도 매치(match)될 수 없다면, 침입이 발생한 것으로 판정된다. 각각의 특징에 대한 차이가 작을 수 있을지라도, 특징들의 수가 많기 때문에 침입 판정의 신뢰성은 매우 높을 수 있고 임의로 설정될 수 있다.

[0007] 그러나, 현재의 기술과 절차는 컴퓨터 시스템의 실행 무결성을 위해 전력 핑거프린팅에 의존하는 보호장치를 극복하거나 무력화하려는 자들에 의해 개발되고 이용되는 기술과 관행을 따라잡기 위해서는 향상되고 개선되어야만 한다.

발명의 내용

해결하려는 과제

[0008] 그러므로 본 발명의 목적은 전력 핑거프린팅(PFP)을 포괄적인 심층방호 보안의 검출 단계에 통합하고; PFP 사용 가능한 노드들의 네트워크를 설치하고; 특정한 상태 시퀀스와 특정한 소프트웨어 실행을 강제하는 미리 정의된 입력으로 비신뢰 장치들을 실행하고; 모듈 식별 정보를 동기화 시그널링에 임베드(embed)하고; 다양한 보드 요소들로부터의 신호들을 결합하고; PFP 성능을 향상시키기 위해 멀웨어 서명들을 이용하고; 자동 특성화 및 서명 추출을 실행하고; 안전한 서명 갱신을 제공하고; 사이드-채널 공격으로부터 보호하고; 이들의 동적 전력 소비를 모니터링하고 이를 신뢰 코드로부터의 서명에 비교함으로써 내장된 플랫폼에서 실시간 무결성 평가를 실행하고; 플랫폼의 전력 소비를 미리 특성화하고 내부 실행 상황에 대한 가장 많은 정보를 전달하는 트레이스 섹션들에 집중하고; 상용의 내장된 플랫폼에서 인증된 실행으로부터의 편차를 검출하는 PFP의 성능을 개선함으로써 전력

핑거프린팅(PFP)에 의해 판정된 타겟 시스템 실행 무결성 평가를 향상시키기 위한 절차를 제공하는 것이다.

과제의 해결 수단

[0009] 본 발명의 한 양태는 컴퓨터 처리 플랫폼에서 루틴의 실행에 대한 실시간 무결성 평가를 실행하기 위한 방법이다. 이는 루틴의 실행 동안에 프로세서를 샘플링해서 프로세서의 전력 소비를 추적하여 루틴 실행을 모니터링함으로써 성취된다. 트레이스들의 섹션들, 즉 프로세서에서 상태 트랜지션들에 가장 큰 의존을 나타내는 섹션들을 검출하는 플랫폼 특성화 기술이 이용된다. 이들 섹션은 가장 많은 정보를 전달하는 특징들을 선택하는데 이용된다. 이러한 플랫폼 특성화는 플랫폼에 적용되고 플랫폼에서 수행되는 모든 루틴에 이용될 수 있다. 다음 단계는 플랫폼 특성화에서 식별된 섹션들에 포함되어 있는 바와 같은 루틴의 선택된 특징들의 특성화로부터 루틴의 신뢰 전력 핑거프린트들의 세트를 얻는 것이다. 다음에는, 신뢰 핑거프린트들로 구성된 서명으로부터의 거리 확률 분포를 기반으로 특정한 거짓 경보 비율에 대한 임계치가 설정된다. 신뢰 핑거프린트들의 라이브러리는 이후 비신뢰 코드의 실행으로부터의 트레이스들로부터 추출된 특징들에 비교되고, 이후 라이브러리에 있는 핑거프린트들과 비신뢰 코드의 실행으로부터 추출된 특징들 간의 거리가 결정된다. 이 거리가 임계치를 초과하면 예외(exception)가 보고된다.

[0010] 전력 핑거프린팅(PFP)를 기반으로 무결성 평가 및 침입 검출 시스템의 운영, 유효성, 유용성 및 성능을 향상시키기 위한 다양한 절차가 설명된다. 다양한 절차는 다음을 포함한다:

- [0011] · 모듈 식별 정보를 동기화 시그널링에 임베드
- [0012] · 다양한 보드 요소로부터의 신호들을 결합하여 PFP 모니터링을 개선
- [0013] · 기존의 배터리 모니터링 기술에 일반화되는 PFP 성능을 개선하기 위해 멀웨어 서명을 이용
- [0014] · 자동 특성화 및 서명 추출
- [0015] · 안전한 서명 갱신
- [0016] · 무결성 침해 및 계층 보안에 대응
- [0017] · 사이드-채널 공격으로부터 보호

[0018] 또한 다음을 위한 방법 및 장치가 설명된다:

- [0019] · 멀웨어 역학 및 행동을 모니터링하기 위해 배포된 PFP 모니터 네트워크
- [0020] · PFP를 공급-사슬 신뢰 분석에 적용
- [0021] · 디지털 권리 관리 및 실행 제한 리스
- [0022] · PFP를 기반으로 한 고장 예측

도면의 간단한 설명

[0023] 앞서 언급한 것과 이외의 다른 목적, 양태 및 장점들은 첨부 도면을 참조한 본 발명의 양호한 실시예들에 대한 다음의 상세한 설명으로부터 잘 이해될 것이다.

도 1은 전력 핑거프린팅의 전반적인 블록도이다.

도 2는 PFP 모니터를 위한 이상적인 센서 위치를 보여주는 도면이다.

도 3은 멀티-프로세서 보드를 위한 이상적인 센서 위치를 보여주는 도면이다.

도 4는 물리적인 신호로 트리거하는 예를 보여주는 도면이다.

도 5는 동기화 및 트리거링을 위한 PFP 전략상 명령 삽입을 보여주는 개략적인 도면이다.

도 6은 리눅스 장치 드라이버 패러다임에서 물리적인 자원들에 대한 간접 액세스를 보여주는 개략적인 도면이다.

도 7은 신뢰 코드 특성화 프로세스를 보여주는 흐름도이다.

도 8은 트레이스들의 전력 스펙트럼 밀도를 계산함으로써 트레이스들에 대한 전처리의 예를 보여주는 그래프이

다.

도 9는 시간 도메인에서 트레이스들에 대한 샘플 전처리를 보여주는 그래프이다.

도 10은 저장된 서명과 대비한 테스트 트레이스들의 실행으로부터의 PSD 차를 보여주는 그래프이다.

도 11은 검출기 설계의 프로세스를 보여주는 흐름도이다.

도 12는 검출기 설계 및 임계치 선택을 위해 이용된 신뢰 코드 실행으로부터의 샘플 확률 분포를 보여주는 그래프이다.

도 13은 PFP 무결성 평가 동작 프로세스를 보여주는 흐름도이다.

도 14는 안드로이드 플랫폼을 위한 샘플 구현 셋업을 보여주는 개략도이다.

도 15는 탬퍼링(tamper)되지 않은 오리지널 루틴의 실행 결과인 샘플 분포를 나타내는 그래프이다.

도 16은 탬퍼링된 루틴의 실행 결과인 샘플 분포를 나타내는 그래프이다.

도 17은 다양한 레벨의 차별 정보를 포함하는 트레이스들의 다양한 섹션을 보여주는 샘플 트레이스 세부에 대한 그래프이다.

도 18은 가장 유익한 정보를 주는 관점에서 선형 프로젝션을 이용하는 플랫폼 특성화를 나타내는 개략도이다.

도 19는 PFP를 이용하는 플랫폼 전력 소비 특성화 및 모니터링을 위한 기준 측정 셋업을 보여주는 개략도이다.

도 20은 최소 전력 소비 변화를 검출할 수 있는 능력을 평가하기 위해 베이스라인 코드 실행으로부터의 샘플 트레이스를 보여주는 그래프이다.

도 21은 베이스라인 코드 실행으로부터의 평균 서명을 나타내는 그래프이며, 각 점은 n-차원의 유클리디언 공간에 대한 프로젝션을 나타낸다.

도 22는 베이스라인 코드의 실행으로부터 추출된 평균 서명으로부터의 유클리디언 거리의 샘플 분포를 보여주는 그래프이다.

도 23은 PCA를 이용하여 얻은 변환된 공간에서 베이스라인 서명으로부터의 유클리디언 거리의 샘플 분포를 보여주는 그래프이다.

도 24는 LDA를 위한 프로파일링 명령으로부터의 트레이스들의 중심(centroid)를 보여주는 그래프이다.

도 25는 LDA를 이용하여 얻은 변환된 공간에서 베이스라인 서명으로부터의 유클리디언 거리의 샘플 분포를 보여주는 그래프이다.

도 26은 인증된 소프트웨어 실행으로부터의 편차를 검출하기 위한 전형적인 타겟 플랫폼의 개략 블록도이다.

도 27은 사이버 보안에 대한 심층방호 접근법에서 다양한 층을 보여주는 개략도이다.

도 28은 계층 심층방호 보안 솔루션 내의 PFP 모니터링 스코프를 보여주는 개략도이다.

도 29는 모니터링 및 정보 수집을 위한 PFP 허니팟(Honeypot) 노드를 보여주는 개략도이다.

도 30은 PFP 허니팟 네트워크를 보여주는 개략도이다.

도 31은 PFP를 이용하는 공급-사슬 신뢰 분석의 흐름도이다.

도 32는 PFP를 이용하는 공급-사슬 신뢰 분석을 위한 기준 서명들의 잠재적 소스를 보여주는 개략도이다.

도 33은 동기화 및 식별 시그널링을 PFP 모니터에 제공하기 위한 IO 레지스터의 이용을 보여주는 도면이다.

도 34는 PFP 동기화 및 식별 시그널링을 전력 트레이스들에 임베드하는 것을 보여주는 도면이다.

도 35는 PFP 무결성 평가를 위해 다수의 신호를 결합시키기 위한 샘플 셋업을 보여주는 도면이다.

도 36은 PFP를 위해 장치 고장을 배제하는 특징 추출 프로세스를 보여주는 흐름도이다.

도 37은 자동 특성화 및 특징 추출을 위해 상호작용하는 다양한 시스템 요소들 간의 관계를 보여주는 개략도이다.

도 38은 내장된 PFP 모니터를 부당하게 이용하는 사이드-채널 공격을 방지하기 위한 구조를 보여주는 도면이다.

도 39는 외부 PFP 모니터를 부당하게 이용하는 사이드-채널 공격을 방지하기 위한 구조를 보여주는 도면이다.

발명을 실시하기 위한 구체적인 내용

- [0024] 전력 핑거프린팅(PFP)은 외부의 모니터가 사이버 시스템의 실행 무결성을 평가할 수 있게 해주는 기술이다. PFP는 프로세서의 동적 전력 소비에 의해 전달되는 실행 상태 정보에 의존한다. 이 정보와 함께, 신뢰 기준(trusted references)으로부터의 사전 특성화된 서명을 이용하여, PFP는 타겟 시스템에서의 실행 무결성을 판정할 수 있다. PFP의 실제 응용을 위해서는, 효과적인 모니터링 솔루션을 제공하기 위해 특정한 장치를 구현하고 특정한 절차를 따르는 것이 필요하다. 이러한 작업에 있어서, PFP 모니터링 솔루션의 운영, 유효성, 유용성 및 성능을 개선하기 위한 다양한 절차를 설명한다.
- [0025] 스마트폰 및 다른 내장된 장치에서의 소프트웨어 수정을 검출하기 위해 PFP를 적용
- [0026] 사이버 보안은 국가의 보안을 위한 중요한 요소가 되었다. 마이크로프로세서는 현대 생활의 거의 모든 면에서 어디에나 존재한다. 정보 기술 영역에서의 기술 개발은 이들을 보호하는데 필요한 보안 솔루션보다 더 빠르게 진행되고 있다. 사이버 공격의 위협은 중요한 기반시설 및 국가의 보안에 잠재적인 엄청난 결과를 초래하는 것으로 항상 존재한다. 사이버 공간은 작금에 정부-후원 적들을 포함해서 모든 종류의 위협으로부터 보호될 필요가 있는 국가 보안을 위해 중요한 새로운 전투 영역 및 요소로 고려될 정도로 사이버 기반시설은 중요해졌다.
- [0027] 스마트폰들과 다른 내장된 플랫폼들의 동적인 전력 소비를 모니터링하고 이를 신뢰 코드로부터의 서명들에 비교함으로써 스마트폰들과 다른 내장된 플랫폼들에서의 실시간 무결성 평가를 수행하는 기술을 설명한다. 설명된 방법과 기술은 전력 핑거프린팅의 일반적인 개념으로 이루어지며 복잡한 상업용의 장치에도 일반적으로 적용될 수 있는 개선을 제공한다. 기준 및 예로서 이용될 수 있는 일반적인 기술의 양호한 구현들에 대한 예를 제시한다. 그러나, 이 기술은 일반적인 것이라 어떤 사이버 플랫폼에도 적용될 수 있다.
- [0028] 접근법의 일부로서, 우리는 또한 프로세서의 내부 실행 상태에 대한 가장 많은 정보를 전달하는 트레이스들의 섹션들에 분류 노력을 집중함으로써 이 접근법의 성능을 개선하고 성능에 해를 줄 수 있는 리턴던트 또는 극단적인 노이즈 특성을 무시하기 위해 특정한 플랫폼 및 프로세서가 전력을 소비하는 방식을 사전 특성화하는 방법을 설명한다.
- [0029] 이 목적은 스마트폰, 내장 시스템, 및 일반적인 정보 시스템에서 인증되지 않은 소프트웨어 수정을 검출하기 위한 신뢰할 수 있는 기술을 정의하는 일반적인 전력 핑거프린팅(PFP)을 향상시키는 것이다. 일반적인 종래 기술의 접근법은 도 1에 도시되어 있다.
- [0030] 일반적인 PFP 방법은 신뢰 코드의 실행 동안 전력 소비로부터 세립 측정치를 수집하는 것으로 시작된다. 센서(110)는 프로세서의 동적 전력 소비 또는 순간 전류 드레인의 직접 또는 간접 메트릭 표현(metric representation)을 수집해야 한다. 센서(110)는 상업적인 전류 프로브, 홀 이펙트 센서, 압전/자기변형, 합성 자기장 센서, Rogowski 코일, 고-대역폭 전류 미러, 또는 간단한 저-저항 정밀 분류기로 구현될 수 있다. 센서들은 선택된 특정한 특징 추출 기술이 정한 필요조건에 부합해야 한다는 것은 주목하자.
- [0031] 센서의 물리적인 위치는 이 접근법의 성공을 위한 중요한 요소이다. 이상적인 위치(210)는 도 2에 프로세서(205)의 V_{DD} 신호에 있는 것으로 도시되어 있다. 이 위치가 실현 가능하지 않거나 또는 과도한 전력 노이즈를 유입한다면, 차선의 위치(220)가 도시되어 있다. 센서(220)가 차선의 위치에 배치되었다면, 기생용량과 유도용량을 갖는 구리 트레이스들(copper traces)은 디커플링 커패시터(215)와 함께 전류 트레이스들에 영향을 주는 로우-패스(LP) RLC 필터를 형성한다. PFP의 경우, 상업적인 네트워크 분석기 또는 다른 시스템 식별 기술을 이용하여 LP 필터의 전달 함수 H 를 식별함으로써 이 하드웨어 효과를 사전 특성화하는 것이 유리하다. 고유한 LP 필터의 효과는 트레이스들을 역 전달 함수 H_{inv} 를 갖는 다른 필터를 통해서 통과시킴으로써 최소화될 수 있다. 역필터를 디지털식으로 구현하는 것이 권장된다. H 의 직접 역전(direct inversion)은 필터를 불안정하게 할 수 있기 때문에, H_{inv} 의 가장 가까운 안정 근사치를 선택하는 것이 필요하다.
- [0032] 도 2에서, V_{DD_core} (225)는 다양한 소스들에 의해 제공될 수 있다. 간단한 프로세서의 경우, 이는 전압 조정기로부터 직접 제공된다. 좀 더 정교한 플랫폼의 경우에는, 이는 복잡한 회로인 전력 및 주변 관리 시스템으로부터 제공될 수 있고, 이 시스템은 요구되는 다양한 전압 레벨을 전달하는 것과, 리셋 및 인터럽트 핸들링과, 다른 주변 관리를 포함해서 넓은 범위의 서비스를 제공한다. 전력 매니저는 PFP 관점에서 다양한 신호들을 합병하고

간섭을 부가하며 전력 서명을 숨기는 경향이 있는 복잡한 시스템이다. 전력 관리 회로가 있는 시스템의 경우, 추가 간섭을 피하고 서명 추출이 용이하게 되도록 전류 센서를 전력 관리 시스템 뒤에 배치하는데 필요한 설비를 갖춘 시스템 보드를 설계하는 것이 권장된다. 최적의 시나리오 경우에 있어서, PFP의 통합이 가능해지도록, 전력 센서는 다른 서비스가 제공되어 있듯이 전력 관리 시스템 안에 포함될 것이다.

[0033] 보드 안에 다수의 프로세서가 있는 경우에, 도 3에 도시되어 있는 바와 같이, 동일한 원리가 각각의 프로세서에 반복될 수 있고, 도 3에서, n번째 프로세서(206)는 바람직하게는 211에서 모니터되고, 아니면 디커플링 커패시터(216) 뒤에 있는 차선의 위치(221)에서 모니터된다. 이 경우에, 검출기는 양쪽 센서로부터의 트레이스들을 결합하여 고려하도록 설계되어야만 한다. 같은 패키지 안에 있는 멀티-코어 프로세서들의 경우, 멀티-프로세서 예에서와 같이 동일한 원리가 적용되지만, 위치 및 실행 가능성은 프로세서의 아키텍처, 각 레일에 의해 전력을 공급받는 코어들의 수, 및 디커플링 필요조건에 좌우될 것이다.

[0034] 센서가 적소에 배치되면, 다음 단계는 신뢰 코드를 특성화하는 것이다. 이 프로세스는 제어 환경에서 타겟 신뢰 코드를 반복 실행함으로써(타겟 소프트웨어의 분리, 실행 동안 이용되는 입력 설정, 및 트레이스들을 동기화하는데 도움을 주는 특정 마커들의 삽입을 포함해서) 성취된다. 마커들은 다양한 성질을 가질 수 있고 트리거링 및 동기화에 도움을 준다. 잠재적인 마커들은 물리적인 신호(핀의 전압 레벨을 변화시키는 것으로서), 또는 공지된 전력 소비 시퀀스를 생성하는 명령의 특정 시퀀스를 포함한다. 물리적인 트리거 신호(410)의 예는 도 4에 도시되어 있다. 트리거링을 위한 명령 삽입의 개념은 도 5에 도시되어 있다. 이 경우에, 추가 어셈블리 명령(515)은 트레이스들에서 공지된 패턴(510), 통상은 특정 코드(510)가 실행될 때를 나타내는데 도움을 주는 짧은 기간 동안 전류 드레인에서의 강한 변화(variation)를 산출하도록 선택된다.

[0035] 도 6에 도시된 바와 같이, 타겟 애플리케이션(610)이, 리눅스 장치 드라이버 패러다임을 구현하는 플랫폼에서 또는 물리적인 신호에 간접 액세스(indirect access)할 수 있는 임의의 다른 운영 시스템의 유저 스페이스(User Space)에서 실행되고 있을 때, 간접 액세스로 야기되는 실행 및 타이밍에 있어서의 내재적인 불확실성을 고려할 필요가 있다. 이 경우에, 트리거 명령(515)은, 물리적인 메모리(640)에 직접 액세스하지 못하고, 단지 커널 스페이스(620)에 위치한 장치 드라이버(631)에 의해서 물리적인 신호(650)를 생성하는데 필요한 레지스터(632)에만 액세스할 수 있는 유저 스페이스(610)에서 실행될 것이다. 실행 및 타이밍에서의 불확실성이 존재하는 이유는, 파일 액세스로 인해 프로세스는 적절한 동기화 시그널링을 대기(블록 실행)해야 하고 그동안 커널(620)이 실행될 다른 프로세스들을 스케줄링하기 때문이다.

[0036] 마커(630)들이 마지막 코드에 남아있을 필요가 없을지라도, 이들이 적소에 남아있다면 실행시간 평가의 프로세스가 가능해진다. 마커들이 설치된 버전에 남아있는 경우에, 마커를 위해 이용되는 설비나 서비스가 설치된 플랫폼에 여전히 남아있도록(예를 들어, 마커가 LED(640)를 켜기로 되어 있다면, 그 LED(640)는 설치된 플랫폼에 존재해야만 한다)를 확실히 할 필요가 있다.

[0037] 특성화 동안에 설치될 정확한 코드가 이용되어야 한다는 점을 주목하는 것은 중요하다. 이는 최적화 등의 동일 레벨로, 소프트웨어를 구축하기 위해 정확히 동일한 툴(tool)을 이용하는 것을 포함한다.

[0038] 좀 더 나은 성능을 위해서는, 특성화는 반복적이고 상호의존적인 프로세스이어야 하고, 이 프로세스 동안, 다양한 실행 사례들에 걸쳐서 가장 작은 변화를 갖는 가장 강력한 서명이 산출되도록 각각의 마커와 함께 소스 코드 구조가 공동-개발된다.

[0039] 신뢰 코드의 실행으로부터의 몇몇 트레이스들은, 이들을 평균하고 임의의 물리적인 시스템에 고유한 랜덤 노이즈의 영향을 줄이기 위하여 수집될 필요가 있을 수 있다. 특성화 프로세스는 도 7에 도시되어 있다. 마커들을 코드에 삽입한(710) 후에, 신뢰 소프트웨어가 실행되고 그 결과로 나온 전력 트레이스들이 캡처된다(720). 이는 필요하다면 미리 정의된 입력(735)을 이용하여, 모든 유의미한 실행 경로에 대해서 실행된다(730). 랜덤 파라미터에 기인한 변화들은 PCA(principal component analysis)을 이용하여 제거한다(740). 차별적인 특징들이 추출되고(750), 통계적인 분석과 평균 내기 및 클러스터링(clustering)(760)이 실행되어 인증된 서명들의 세트가 생성된다(770).

[0040] 서명들은 다양한 신호 도메인들로부터 추출되고 다차원일 수 있다. 더욱이, 다수의 서명들이 단 하나의 코드를 식별하는데 이용될 수 있다.

[0041] 트레이스 프로세싱 및 특징 추출

[0042] 저장된 서명에 비교될 테스트 트레이스들을 준비하는 프로세스는 전처리 및 특징 추출이라고 알려져 있다. 트레이스 전처리는 선택된 차별적인 특징을 추출하기 위해 트레이스들을 컨디셔닝하는(condition), 예를 들어, 트

레이스들을 적절한 도메인으로 변환하거나 또는 트레이스들을 특정 마커에 관련하여 정렬하는 일반적인 작업을 포함한다. 트레이스 전처리의 예는 도 8에 도시되어 있는데, 여기서 OMAP3 프로세서가 탑재된 BeagleBoard에서 테스트 소프트웨어의 실행으로부터의 시간 도메인 트레이스들은 그들의 전력 스펙트럼 밀도를 계산함으로써 먼저 주파수 도메인으로 변환된다.

[0043] 기본적인 전처리의 또 다른 예는 상관관계 검출기로 전달되기 이전에 도 9에 도시된 기본 실행 및 대체(-1 비트 트랜지션) 트레이스들의 정렬과 같이 시간 도메인 트레이스들을 정렬하는 것이다. 이 예에서, N 샘플들의 각 트레이스는 다차원 유클리디언 공간에서 점으로 간주된다.

[0044] 특징 추출은 검출기들에 전달되어 무결성을 판정하는데 이용되는 최종 테스트 통계(새로운 트레이스들로부터의)를 계산하는 프로세스이다. 이 프로세스는 각각의 선택된 특징에 고유한 것이다. 예를 들어, 기본적인 시간 도메인 상관관계 분석에 있어서, 전처리는 특정한 플랫폼 전력 소비 패턴에 대한 대략적인 동기화 및 보상을 포함할 수 있는 한편, 특징 추출은 상관관계 팩터 또는 유클리디언 거리를 계산함으로써 저장된 서명에 비교하는 것을 포함한다. 특징 추출의 예가 도 10에 도시되어 있고, 도 10은 도 8의 PSD 예를 따르는 BeagleBoard의 OMAP3 프로세서에서의 신뢰 코드와 템퍼링된 코드의 실행에 대응하는 테스트 트레이스들의 PSD 오차(dB 단위)를 보여주고 있다. 이러한 차 벡터를 이용하여, 검출기에 전달된 최종 테스트 통계 또는 차별적인 특징은 평균 제곱 오차 또는 임의 다른 거리 또는 오차 메트릭으로 표현될 수 있다.

[0045] **검출기 설계**

[0046] 일단 서명들이 추출되고 차별적인 특징들이 선택되었다면, PFP 프로세스에서 다음 단계는 최종 무결성 평가를 실행하기 위한 최적의 검출기를 설계하는 것이다. 이들 검출기는 모니터링 작업 동안 테스트 트레이스를 침입으로 간주해야 하는지 여부에 대한 최종 판정을 한다. 검출기 설계의 프로세스와 정상적인 모니터링 작업은 매우 유사하다. 검출기 설계에 있어서, 신뢰 소프트웨어의 실행으로부터의 테스트 트레이스들은 캡처되어 선택된 차별적인 특징들이 추출되도록 처리된 다음 저장된 서명들에 비교된다. 몇몇 트레이스들은 모아져서 처리되고, 그들의 통계적 샘플 분포는 예측된 성능 타겟을 산출하는 임계치를 식별하는데 이용된다. 검출기 설계의 프로세스는 도 11에 도시되어 있다. 랜덤한 또는 미리 정의된 입력(1110)이 신뢰 소프트웨어(1120)에 제공되고, 그의 실행으로부터 새로운 테스트 트레이스들이 캡처된다. 그 결과들은 정렬되어 동기화되고(1130), 트레이스들은 전처리 및 컨디셔닝된다(1140). 비교를 위한 인증 서명들(770)을 이용하여, 선택된 차별적인 특징들이 추출되고 거리 메트릭이 생성된다(1150). 이후 통계적 분석 및 분포 피팅(distribution fitting)이 최종 메트릭에 대해서 실행된다(1160). 마지막으로, 네이만 피어슨(Neyman-Pearson) 기준이 적용되어(1170), 예측된 성능 타겟들에 부합하는 임계치가 결정된다.

[0047] 최적의 검출기들을 생성하기 위한 공통적인 접근법은 주어진 거짓 경보의 확률에 대한 검출 확률을 최대화하기 위해 네이만 피어슨 기준을 적용하는 것을 포함한다. 기본적인 가설 테스트 이론에서 나온 이러한 기준을 간단히 상기해 보면, 거짓 경보의 타겟 확률은 최종 판정에서 실수를 하는 것의 허용 오차 및 추정 비용을 기반으로 설정된다. 신뢰 코드로부터의 차별적인 서명들의 확률 분포의 추정을 이용하여, 올바른 검출의 확률을 최대화하면서 거짓 경보의 예측 확률을 산출하는 거리 임계치가 계산된다. 이 프로세스의 예는 도 12에 도시되어 있는데, 여기서 거짓 경보(1230)의 예측 확률을 산출하는 확률 분포(1210)에 대한 거리 임계치(1220)가 계산된다.

[0048] 그러나, 선택된 차별적인 특징들의 성질에 따라서 개선된 결과를 산출할 수 있는 다양한 기술이 있다는 점을 주목하는 것은 중요하다. 검출기 설계 및 기계 트레이닝을 위한 다른 기술은 신경망(Neural Networks), 서포트 벡터 머신(Support Vector Machines), 및 은닉 마코프 모델(Hidden Markov Models)을 포함한다.

[0049] **모니터링 작업**

[0050] 신뢰 코드의 실행으로부터 서명들이 추출되었고, 차별적인 특징들이 선택되었고, 최적의 검출기가 설계되었다면, PFP 모니터링은 테스트 소프트웨어의 무결성을 평가할 준비가 되어 있다. 앞서 언급한 바와 같이, 정상적인 무결성 평가 프로세스는 검출기 설계 프로세스와 매우 유사하다. 정상적인 동작 동안, 모니터링 또한 필요한 전처리 후에 전력 트레이스들로부터 선택된 차별적인 특징들을 추출하지만, 검출기 설계를 위해 실행된 바와 같이 몇몇 트레이스들로부터 통계들을 수집하는 대신에, 이들은 각각의 임계치들에 비교되어 테스트 코드 실행의 무결성 상태가 판정되도록 적절한 검출기를 통과한다. 검출기는 테스트 트레이스들을 모든 공지된 서명들에 비교하고, 인증된 코드가 실행되었다고 판정하기에 충분한 테스트 통계가 하나도 없다면 침입이 있는 것으로 보고된다. 이 프로세스는 도 13에 다이어그램으로 묘사하였다. 타겟 소프트웨어는 정상적인 동작 동안에 또는 미리 정한 입력을 이용하여 실행되어(1310), 테스트 트레이스들이 캡처되고(1320), 이후 이들은 정렬되고

동기화되고(1330), 그 후 전처리되고 컨디셔닝된다(1340). 이후 검출기는 추출된 특징들을 모든 공지된 서명들(1370)에 비교하여 거리(distance)를 판정하고(1350), 미리 정의한 임계치(1220)를 이용하여 무결성 평가 판정을 실행한다(1360).

[0051] 샘플 결과

[0052] 스마트폰 및 다른 내장된 플랫폼에서의 PFP 프로세스를 예시하기 위하여, 안드로이드 플랫폼을 실행하는 ARM 프로세서(OMAP3@720 MHz)를 갖춘 BeagleBoard 리비전 C4를 이용하여 이 기술의 기준 구현을 설명하기로 한다. BeagleBoard(1410)는 전류 프로브(1430)를 연결하기 위해서 코어 전력 레일에 전력을 제공하는 메인 트레이스(1420)를 절단함으로써 약간 수정된다. 캡처 시스템은 상용의 실시간 오실로스코프(1440) 및 전류 프로브(1430)를 이용하여 구현된다. 오실로스코프는 2.5 GSps의 샘플링 속도로 설정되어 있고, 트리거(1450)에 의해 개시되는 매 트레이스마다 전체 30K 샘플이 수집된다. 구성은 도 14에 도시되어 있다.

[0053] 프로세스를 입증하고 실현가능성을 보여주기 위한 기본적인 테스트 애플리케이션이 개발되었다. 이 기본적인 애플리케이션은 장치 스크린에 증가하는 정수를 표시하는 간단한 카운터로 구성된다. 이 애플리케이션의 동작은 리스팅 1에 설명되어 있고 텍스트 박스를 보여주기 위한 스크린을 준비하고 카운터로서 이용된 정수 변수를 설정하는 초기화 루틴을 갖는 전형적인 안드로이드 자바 애플리케이션으로 구성된다. 여기에는 또한 카운터의 값을 증가시키고 이를 스크린에 보여주는 역할을 맡고 있는 디스플레이 카운터(DisplayCounter)라 불리는 루틴이 있다. 이 루틴은 매 초 호출되는 반복 작업(recurrent task)으로 구성된다.

[0054] 리스팅 1. 안드로이드 테스트 앱의 의사-코드

LISTING 1. Pseudo-code of Android Test App

```
Initialize
DisplayCounter()
{
    counter = IncrementValue(counter);
    Display data
}
Sleep for one sec
DisplayCounter at wake up
```

[0055]

[0056] 중요한 IncrementValue 루틴은 원시 C 코드로 구현되었고, 전통적인 자바 구현 대신에 안드로이드의 NDK 툴셋에 의해 외부 라이브러리(external library)로서 포함되었다. 이 중요한 섹션 이전에, 물리적인 트리거(1450)는 전력 트레이스들의 수집을 시작하도록 캡처 시스템에 신호(1460)를 주도록 설정된다.

[0057] 우리는 단지 리스팅 2에 보여준 중요한 IncrementValue 루틴만을 특성화하고 모니터한다.

[0058] 리스팅 2. C로 작성된 모니터된 원시 루틴의 의사-코드

LISTING 2. Pseudo-code of monitored native routine in C

```
/*Critical native routine*/
int incrementValue(int Val)
{
    /*trigger LED usr1*/
    Open device driver control file
    Write 1 into file

    /*Increment Val*/
    Val++;

    /*General extra processing*/
    i = 1000;
    while(i)--;

    /*Reset LED usr1 Trigger*/
    Write 0 into file
    Close driver control file
    return Val;
}
```

[0059]

[0060] 서명 추출은 주파수 도메인에서 신뢰 코드 실행으로부터의 몇몇 트레이스들의 PSD를 간단히 평균 냄으로써 실행된다. 트레이스의 위상 정보는 무시된다. 200개 트레이스의 PSD는 함께 평균화되어 서명이 산출된다.

[0061] 차별적인 특징들도 역시 주파수 도메인에서 서명과 테스트 트레이스들의 PSD 간의 평균 제곱 오차(dB 단위)에 의해 추출된다. MSE를 계산하기 전에 최근의 3개의 테스트 트레이스들의 PSD가 함께 평균화된다. PSD의 첫 번째 200 MHz만이 MSE 계산에 이용된다.

[0062] 서명 추출을 위한 이 프로세스는 단-차원의 차별적인 특징을 산출한다.

- [0063] 검출기 설계는 1%의 거짓 경보의 타겟 확률 P_{FA} 를 이용하는 앞서 설명한 Neyman-Pearson 기준을 이용하여 실행되었다. 트레이스의 샘플 통계들은 신뢰 코드의 실행으로부터의 200개 트레이스의 샘플로부터 추출된다.
- [0064] 샘플 분포는 평균 및 분산이 트레이닝 샘플 분포 평균 및 분산과 같은 Rayleigh 분포에 맞추어졌다. 이러한 분포를 이용하여, 타겟 1% P_{FA} 를 산출하는 임계치를 찾기 위해 역확률 분포(inverse probability distribution)가 계산된다.
- [0065] 신뢰 코드로부터 실행 편차(deviations)를 검출할 수 있는 능력을 테스트하기 위하여, 우리는 애플리케이션의 약간 탬퍼링된 버전을 이용하여 이전에 설계된 모니터를 테스트한다. 리스팅 3에 도시된 탬퍼링된 애플리케이션은 특정한 조건이 충족될 때까지는 침입이 비활동으로 남아있는 은밀한 공격(covert attack)을 에뮬레이트(emulate)하도록 설계되어 있다. 이 침입은 카운터의 값이 특정한 값(조건)에 도달할 때만 파일이 기록되는 매우 간단한 수정으로 이루어진다.
- [0066] 리스팅 3. 탬퍼링된 중요한 네이티브 루틴의 의사 코드

LISTING 3. Pseudo code of tampered critical native routine

```

/*Critical native routine*/
Int incrementValue(int Val)
{
    /*trigger LED usrl*/
    Open device driver control file
    Write 1 into file
    /* Tamper */
    if(Val == 1){
        /*open temporary file
        //Write Val into file
        //Close file
        */
        /*Increment Val*/
        Val++;

        /*General extra processing*/
        i = 1000;
        while(i)--;

        /*Reset LED usrl Trigger*/
        Write 0 into file
        Close driver control file
        return Val;
    }
}

```

- [0067]
- [0068] 탬퍼링에서 발생하는 파일 기록은 실행 동안에 한 번만(즉, 카운터가 1일 때) 나타난다는 점을 주목하는 것은 중요하다. 루틴이 호출되는 시간 외에는 조건이 충족되지 않으며 추가의 파일이 기록되지 않는다. 그러므로, 루틴이 호출되는 시간의 대부분 동안, 논리적 관점에서 본 유일한 수정은 주어진 조건의 추가의 평가이다.
- [0069] **작업 결과**
- [0070] 루틴의 탬퍼링되지 않은 오리지널 버전이 실행될 때 모니터를 실행한 결과는 도 15에 도시되어 있다.
- [0071] 우리는 테스트를 하는 기간 동안, 설계된 거짓 경보의 확률에 일치하는 임계치(1510)를 넘어선 단지 몇몇 사례를 가졌음을 알 수 있다.
- [0072] 애플리케이션의 탬퍼링된 버전으로 모니터를 실행한 결과는 도 16에 도시되어 있다. 인증된 실행으로 잘못 분류된 사례가 없으며 탬퍼링된 애플리케이션의 실행은 단 하나라도 임계치(1610)를 넘어서는 침입으로 표시(flag)될 것임에 주목하자. 또한 탬퍼링의 조건부 실행에 기인해서, 이들 결과에 이용된 실행 사례들 동안에 단지 한 번만 파일이 실제로 기록되었다는 점을 주목하는 것도 중요하다. 나머지 시간 동안 단지 한 조건이 검사되었고 이 조건이 충족되지 않았을 때는 정상적인 실행이 재개되었다.
- [0073] **플랫폼 특성화 및 최소 민감도의 평가**
- [0074] 전력 소비의 세립 측정은 매우 작은 차별적인 정보를 부가하지만, 서명에 상당한 노이즈 및 불확실성을 부가할 수 있는 리턴던트 정보로 이어질 수 있다. 시간 도메인에서 이는 도 17처럼 보인다. 이 경우에, 우리는 두 개의 실행 간에 작은 분산을 보여주는 섹션, 예를 들어, 1720과는 대조적으로, 두 실행 간에 가장 큰 분산(1710)을 갖고 있는 트레이스들(디멘전)의 섹션에 우리의 관심을 집중하고 싶다. 한편, 랜덤한 파라미터를 취하는 특정 소프트웨어 루틴을 특성화할 때, 이들 랜덤한 파라미터의 영향은 서명에 노이즈를 유입하는 것이며, 이로 인해 성능이 감소하고 거짓 경보의 확률이 증가한다. 이 경우에, 우리는 노이즈를 부가하는 것들은 무시하면서 타겟 소프트웨어의 실행 동안 일정하게 남아있는 디멘전(예를 들어, 1720)에 우리의 관심을 집중하고 싶다. 이 경우에, 우리는 큰 분산(예를 들어, 1710)을 표시하는 디멘전은 무시하고 싶다.

[0075] PFP의 성능을 개선하기 위해서, 가장 많은 정보를 전달하는 것들에만 집중함으로써 분석된 특징들의 수를 감소시킬 필요가 있다. 이는 트레이닝의 일부로서 주어진 플랫폼에 대한 가장 많은 정보를 전달하는 특징들을 사전 특성화하고, 트레이스들을 검출기에 전달하기 전에 전처리 동안 리턴던트 정보를 제거함으로써 성취된다.

[0076] 기술 배경

[0077] 전통적인 패턴 인식 시스템에서, 특정한 기준(specific criterion)을 최대화하는 특징들의 서브세트를 선택하는 프로세스는(PFP의 경우에 우리는 차별적인 정보 PFP를 최대화하기를 원한다) 최적의 특징 선택으로 알려져 있다. 클러스팅 시스템에서, 이는 보통은 선형 변환에 의해서 가장 유용한(또는 유용한 정보를 주는) 관점에서 볼 때 보다 소수의 디멘전을 갖는 변환된 공간에 트레이스들 x 를 투사함으로써 성취된다.

[0078] 이 변환은 다음 식으로 묘사된다:

$$y = Wx$$

[0080] 여기서 W 는 테스트 트레이스들에 적용될 때 특정 기준을 최대화하는 보다 낮은 차원수(dimensionality)를 갖는 변환된 트레이스를 산출하는 신중하게 설계된 선형 변환 매트릭스이다. 최적의 변환을 식별하기 위한 다양한 기준이 있다. 우리는 차별적인 정보의 관점에서 특징 선택을 최적화하고자 하므로, 정보 이론 접근법을 따르는 것은 당연하다. 이러한 최적화는 이전부터 실행되어 왔고, 패턴 인식 문헌에 있는 몇몇 소스, 예를 들어, J. T. Tou 및 R. C. Gonzalez "Pattern Recognition Principles" Addison-Wesley Publishing Company, 1974에서 찾아볼 수 있다.

[0081] 주성분 분석

[0082] 트레이스들에서 엔트로피(또는 정보)를 최적화하는 적절한 W 를 결정하기 위한 잘 알려져 있는 접근법은 주성분 분석(PCA, Principal Component Analysis)으로 알려져 있다. 우리는 다양한 클래스의 공분산 매트릭스들 C_i 는 정상적으로 분포되어 있고 동일하다($C_i = C$)고 가정한다. 그러므로, 고유벡터는 고려되고 있는 트레이스들에 대한 정보 전달자(bearer)로 생각할 수 있다. 이들 벡터 중에서 어떤 것은 성능에 많은 불이익을 주지 않고도 안전하게 제거될 수 있는 다른 것들보다 분류 면에서 더 많은 차별적인 정보를 전달한다. 최적의 특징 벡터들은 이들 고유벡터에 묶여있으며 대응하는 고유값에 따라서 내림차순으로 고유벡터들을 집합시켜 변환 매트릭스 W 를 생성하는데 이용된다는 것은 뜻밖의 일이 아니다. PFP에 있어서 우리는 단지 클록 사이클당 하나의 포인트만 필요하므로, 변환 매트릭스 W 는 가장 큰 고유값에 연관된 공분산 매트릭스의 고유벡터에 의해서 주어진다.

[0083] 선형 변환은 가장 유용한 정보를 주는 관점에서 보다 낮은 차원수의 변환 공간으로의 테스트 트레이스의 투사(projection)로서 해석될 수 있다. PCA는 특정 목적에 따라 다양한 방식으로 적용될 수 있다. 클러스팅의 관점에서, 가장 작은 고유값들에 관련된 고유벡터들을 이용하여 W 를 구성하는 것이 바람직한데, 왜냐하면, 이는 변환 공간에 보다 조밀한 클러스터를 산출하기 때문이다. 한편, 다양한 실행들로부터의 트레이스들이 이용될 때 가장 큰 고유값에 연관된 고유벡터들을 이용하는 것도 가능하다. 이 방법으로 적용될 때, PCA는 클래스들 간에 가장 큰 분산을 나타내는 특징들을 선택할 것이다. 공분산 매트릭스들이 동일하다고 가정하면, 이들 고유벡터는 PCA에 사용된 특정 트레이스들 간의 최대 차별 정보를 포함하는 특징들을 나타낼 것이다.

[0084] 선형 판별 분석(LDA, Linear Discriminant Analysis)

[0085] PCA는 트레이스 엔트로피를 최적화하기 위해서 분산의 관점에서 오름차순 또는 내림차순으로 특징 서브세트를 선택한다. 그러나, 분포들 간의 거리를 최대화하는 최적의 특징들의 세트를 선택하기 위해 클래스들 간의 특징 차이를 고려하지는 않는다. 선형 판별 분석(LDA)은 확률 분포들 간의 거리의 척도인 분포들 간의 발산(divergence)을 최대화한다. 발산은 정보 이론에서 상대 엔트로피의 개념에 밀접하게 관련되어 있다.

[0086] 다양한 클래스들로부터의 특정 정보와 최적화 기준으로서의 발산을 이용하여, LDA는 트레이스들 간의 최대 분리를 산출하는 고유 관점(unique perspective)에서 트레이스들을 투사할 최적의 변환 매트릭스를 식별한다. 이는 변환 벡터 W 가 두 개의 분포 간의 최적의 차별적 초평면(optimal discriminating hyper-plane)에 수직이기 때문이다.

[0087] 트레이스들이 정상적으로 분포되어 있다는 가정에 따라, 발산 극값(divergence extremum)을 산출하는 변환 매트릭스는 비-제로 고유값에 연관된 단지 $C^{-1} \delta \delta^T$ 의 고유벡터에 의해서 주어진다는 것을 알 수 있다. 이 벡터는 다음 식에 의해 주어진다:

$$W_0 = C^{-1}(\mu_1 - \mu_0)$$

- [0088]
- [0089] 여기서 W_0 는 두 클래스를 분리하기 위한 최적의 투사를 제공하며, μ_0 및 μ_1 은 두 개의 트레이닝 클래스에 대한 각각의 중심(centroid)이다. LDA는 M 차별적 클래스들까지 확장될 수 있다. 이 경우에, 비-제로 고유값들에 관련된 $M-1$ 고유벡터들이 있을 것이다.
- [0090] **플랫폼 전력 소비 특성화**
- [0091] 이전에 언급한 바와 같이, 실행 편차가 발생하였는지 여부를 판정하는데 있어서 테스트 트레이스의 모든 샘플들이 동등하게 중요한 것은 아니다. 큰 오버샘플링 비율(oversampling ratio) 및 전력 트레이스들의 성질 때문에, 다른 것들보다 더 차별적인 정보를 전달하는 트레이스들의 특정 섹션들이 있다. PFP 경우에, 리턴던시를 제거하면서 가장 많은 정보를 전달하는 디멘전을 강조함으로써 트레이스들의 차원수를 감소시키는 선형 변환을 식별하는 것이 목표다.
- [0092] 이 아이디어는 최적의 변환 매트릭스를 이용하는 트레이스들의 선형 투사를 이용함으로써 디멘전을 줄이기 위해 차별 특징들을 변환하는 것이다. 시간 도메인에서, 폴 클록 사이클(1810)에 대응하는 트레이스 섹션들은 도 18에 도시된 바와 같이 변환 공간에서 단일 점(1820)으로 줄어든다. 분류기는 또한 정상적인 모니터링 작동 동안에 고려되어야하는 디멘전의 수를 줄이면서, 변환된 공간에서 동작하도록 설계되어야만 한다.
- [0093] 특성화는 랩에서 통제 조건하에서 실행되고 플랫폼마다 1회만 요구된다. 이전 장에서 설명한 바와 같이, 최적의 변환 매트릭스를 식별하기 위한 두 가지의 일반적인 접근법: PCA 및 LDA가 있다.
- [0094] **PCA를 이용하는 플랫폼 특성화**
- [0095] PCA를 이용하여 변환 매트릭스를 생성하기 위해서는, 랜덤 클록 사이클 동안 프로세서의 전력 소비를 관찰해야 할 필요가 있다. 트레이스들은 프로세서 실행의 동적 동작에 의해서 가장 영향을 받는 트레이스들의 섹션들을 명료하게 보여주도록 매 클록 사이클마다 정렬된다. 일단 트레이스들이 정렬되면, PCA는 트레이스들에서 최대의 분산을 나타내는 분산 벡터를 식별하는데 이용된다.
- [0096] PCA를 이용하여 플랫폼 특성화를 실행하는 것은 구현하기가 비교적 용이하고 파이프라인에서 콘텐츠를 제어하기가 매우 어려운 복잡한 플랫폼에도 잘 맞는다.
- [0097] **LDA를 이용하는 플랫폼 특성화**
- [0098] LDA를 이용하여 플랫폼 전력 소비 특성화를 실행하기 위해서는 두 개의 주의 깊게 맞추어진 루틴의 개발이 필요하다. 이들 루틴은 특정 클록 사이클 동안 미리 정한 차이를 보여주는 두 세트의 트레이스를 생성하기 위해 올바른 시퀀스로 특정 어드레스 및 파라미터를 갖는 특정 명령을 실행해야만 한다. 두 루틴의 실행으로부터의 트레이닝 트레이스들은 LDA가 최적의 차별적인 초평면을 찾게 될 두 개의 클래스를 제공하며, 이 초평면은 최적의 변환 벡터가 될 것이다.
- [0099] 이 특수한 특성화 루틴의 목적은 각 실행 단계(페칭, 래칭, 실행 등) 동안 특정 클록 사이클에서 공지된 변화가 있도록 파이프라인을 적절하게 로딩하기 위해 주의 깊게 만들어진 명령 시퀀스를 실행하는 것이다. 이러한 변화는, 바람직하게는 각 레지스터에 변화 비트가 소수이기 때문에, 비교적 작아야만 한다. 특성화 루틴은 고유한 것은 아니지만, 특성화되는 플랫폼의 아키텍처, 명령 세트 등에 의존하기 때문에 플랫폼 종속적이다. 상이한 프로세서들은 아마도 상이한 시퀀스를 필요로 할 것이다.
- [0100] 두 시퀀스의 실행으로부터의 트레이스들이 캡처되어 동기화되면, LDA는 최적의 변환 벡터 W 를 찾는데 이용된다. LDA를 이용하는 플랫폼 특성화는 두 개의 공지된 클래스의 이용가능성을 고려해 볼 때 최고의 성능을 제공할 것으로 예상되지만, 그의 구현은 PCA보다 복잡하다.
- [0101] **플랫폼 전력 소비 특성화 기준 구현 결과**
- [0102] 이러한 기준 구현을 위해, 우리는 IEEE 802.15.4의 평가 및 개발 플랫폼으로 의도된 PICDEM Z 데모 키트에 이용된 것들과 유사한 Microchip Technology Inc.의 PIC18LF4620 8-비트 마이크로컨트롤러를 갖춘 마더보드를 이용한다. 이는 메모리 관리 유닛이 없는 대중적인 내장 마이크로컨트롤러이다.
- [0103] 프로세서 마더보드는 전력 소비 특성을 강화하기 위해 약간 수정된다. 이 보드로부터, 전체 누적 6 microF인 전체 6개의 디커플링 커패시터가 제거된다. 이들 커패시터의 기능은 디지털 프로세서에 의해 발생한 강한 전류 피크에 의해서 전원에 가해지는 스트레스를 경감시키는 것이다. 전류 센서가 프로세서 전력 핀에 더 가깝게 배

치되거나 또는 최종 LP 필터의 효과를 신호 처리를 이용하여 상쇄시킬 수 있다면 디커플링 커패시터를 제거할 필요가 없을 것이라는 점은 주목해야 한다.

[0104] 트레이스 수집은 Tektronix TDS 649C 실시간 오실로스코프(1910) 및 Tektronix TC-6 전류 프로브(1920)를 이용하여 실행된다. 프로브는 마더보드에 있는 전압 조정기를 바로 지나서 연결된다. 오실로스코프는 500 MS/s 및 10 mV로 설정된다. 트리거는 LED1(1930)에 의해 구동되고 폴링-에지에서 40mV 레벨로 설정되고, 트리거 이전(pre-trigger) 샘플은 유지되지 않는다. 모든 트리거 발생 후에 전체 L=30,000 샘플이 수집된다. 측정 셋업은 도 19에 도시되어 있다. 트레이스들은 캡처되어 나중 분석을 위해 GPIB를 이용하여 호스트 컴퓨터에 전달된다.

[0105] 샘플 루틴은 1) 플랫폼 특성화를 실행하기 위해 트레이닝 루틴을 제공하고 2) 접근법의 성능을 측정하기 위해 기본 변화를 제공하는 두 가지의 목적으로 이 실험을 위해 개발되었다. 우리는 먼저 루틴의 평가 이용에 대해 설명하고 비교를 위한 베이스라인 성능을 제공한다. 테스트 루틴은 리스팅 4에 도시되어 있고 무한 루프로 실행된다. 이 루틴에서, 레지스터 W의 콘텐츠는 다양한 명령을 이용하여 00으로부터 0f까지 토글(toggle)된다. 이 루틴에 있는 실제 로직은 전력 핑거프린팅에 영향을 주지 않는다는 점에 주목하자. 이 루틴은 나타나는 비트 트랜지션의 수를 제어하기가 용이하기 때문에 선택되었다. 그러나, 그 결과는 실행되는 특정 소프트웨어에 의존하지 않는다. 그러므로, 이 루틴은 대표적인 예를 제공한다.

[0106] 리스팅 4

LISTING 4.

```

BYTE i; //addr 00
BYTE j; //addr 01
BYTE k; //addr 10
BYTE l; //addr 11

// Initialize the system
BoardInit();

// Initialize data variables
_asm
movlw 0x07
movwf i, 0 //addr 0x00
movlw 0x0f
movwf j, 0 //addr 0x01
movlw 0x0f //Set for minimum change
movwf k, 0 //addr 0x10
movlw 0x1f
movwf l, 0 //addr 0x11
movlw 0x00
_endasm

//Target code infinite loop
while(1){
    TMR0H = 0x00; //Restart TIM0
    TMR0L = 0x00;
    LED_2 = 1; //Trigger
    LED_2 = 0;

    _asm
        nop
        iorwf j, 0, 0 //w = 0f
        andlw 0x00 //w = 00
        movf j, 0, 0 //w = 0f
        andlw 0x00 //w = 00
        movf k, 0, 0 //w = 0f Change in k (one bit)
        movlw 0x00 //w = 00
        xorwf j, 0, 0 //w = 0f
        movlw 0x00 //w = 00
        iorwf j, 0, 0 //w = 0f

```

[0107]

```

        xorlw 0x00 //w = 00
        nop
        ... x 10
        nop
    _endasm
}

```

[0108]

[0109] 루틴은 리스팅 4에 도시된 바와 같이 기본 실행을 나타낸다. 타겟 코드 이전에 우리는 보드에 있는 LED를 이용하여 트리거를 생성한다. 이 트리거는 오실로스코프와 트레이스 캡처를 동기화시키는데 이용된다. 트리거와 타겟 코드 간의 "NOP" 명령은 타겟 트레이스들을 트리거로부터의 임의 잔여 효과로부터 분리하기 위한 완충 장치로서 포함되어 있다. 메인 루프에서 레지스터 W는 00으로부터 0f로 토글되어 명령마다 이 레지스터에 4개의 비트 트랜지션을 생성한다. 대안적인 또는 수정된 코드는 하나 적은 수의 비트 트랜지션을 갖는다. 라인 15에서, 우리는 변수 j의 콘텐츠를 0f에서 07로 변경한다. 이러한 식으로, 타겟 코드가 실행될 때, 라인 35에서, 파라미터 k는 00으로부터 07로 나아가는 레지스터 W에 로드되고 이 명령을 위해 레지스터에는 단지 3비트가 트랜지션된다. 수정된 코드와 레지스터 W에 0f를 로드하는 기본 실행 간에는 단지 1비트 차이만 있으며, 이 실행에 있어서의 그 밖의 모든 것은 명령, 파라미터, 및 어드레스를 포함해서 동일하게 유지된다는 것에 주목하자.

이 명령에 하나 적은 트랜지션이 들어오고 이 명령으로부터 하나 적은 트랜지션이 나가므로 이러한 1-비트 트랜지션은 실제로는 2 클록 사이클에 영향을 준다는 점에 주목하자. 타겟 코드를 추적하면 루프가 반복되기 전에 "NOP" 명령의 스트림이 있다.

[0110] 전형적인 트레이스에 대한 상세한 사항은 도 20에 도시되어 있다. 이 도면에서, 우리는 타겟 코드의 하나의 전체 실행 사이클을 캡처한다. 전력 트레이스에 대한 트리거 효과는 두 개의 사각형 스텝(2010 및 2020)으로 명료하게 볼 수 있다. 개개의 명령 사이클도 또한 뚜렷하다. 이들은 125개의 샘플마다 반복하는 4개의 스파이크 그룹으로 식별될 수 있다. 프로세서의 문서에 있는 타이밍 정보를 이용하여, 타겟 코드의 실행에 대응하는 트레이스의 섹션을 판정할 수 있다. 도 20에서, 이 섹션은 10개의 명령 사이클 기간에 걸쳐서 실선(2030)으로 강조되어 있다. 이는 각각이 실행을 위해 하나의 버스 사이클을 취하는 10개의 어셈블리 명령으로 구성되는 실제의 코드와 일치한다.

[0111] 몇몇 트레이스가 기본 실행 및 대체 실행으로부터 각각 캡처되고 각각의 실행으로부터의 트레이스들은 함께 평균이 내어져서 하나 적은 비트 트랜지션의 전체 효과를 보여주는 양쪽 실행의 명료한 화상이 제공된다. 평균화된 트레이스들은 도 21에 도시되어 있다. 이 그림에는, 기본 코드의 실행에 대응하는 10 클록 사이클이 도시되어 있고, 각 실행으로부터의 트레이스들이 정렬되어 있는 것으로 보인다. 그러나, 샘플 인덱스(650) 주위에는 두 개의 트레이스들 간에 작은 차이가 있음을 알 수 있다. 차이(2110에서의)는 좀 더 자세히 살펴볼 수 있게 해주는 도 21의 상단에서 더 현저하다. 양쪽 시나리오로부터의 중심들의 접근과 함께, 오버 샘플링에 기인해서 트레이스들이 상당히 연관성이 있으며 또한 트레이스들의 특정 섹션들만이 유용한 차별 정보를 전달하고 있다는 것도 분명하다.

[0112] 비교의 목적으로, 우리는 플랫폼의 사전 특성화 없이 시간 도메인에서의 나이브(naive) 분류 접근법의 결과를 제공한다. 우리는 기본적인 최소-거리 분류기를 이용한다. 이 접근법에서, 길이 $L = 1250$ (타겟 코드의 길이)의 각 캡처된 트레이스는 L -차원 유클리디언 공간에서 점을 나타낸다. 유클리디언 거리는 기본 실행 중심으로부터 각각의 인입되는 테스트 트레이스까지 취해진다. 분류 목적을 위해, 기본 중심과 테스트 트레이스는 1250 디멘전을 갖는 다차원 유클리디언 공간에서 단일 점 또는 벡터를 나타낸다. 테스트 트레이스들은 기본 중심을 얻기 위해 이용된 트레이닝 트레이스들과는 다르다. 이는 서로 다른 시나리오들을 정확히 식별하기 위한 최소-거리 분류기의 평가시 바이어스(bias)를 피하기 위함이다.

[0113] 양쪽 루틴의 실행으로부터의 테스트 트레이스들은 도 22에 도시된 유클리디언 거리 분포를 갖는다. 이러한 나이브 예에서, 전력 핑거프린팅의 성능은 실질적으로 오버랩되는 분포들 간에 거의 어떤 차이도 없으므로, 고무적이지 않다. 이와 같이 열악한 성능은 기본 실행과 대체 실행 간의 전력 비교시에 작은 차이를 고려하면 예상된다.

[0114] 플랫폼 특성화에 대한 첫 번째 결과는 PCA를 적용함으로써 얻어진다. 이 프로세스의 경우, 우리는 리스팅 4에 도시된 루틴에서 우리의 타겟 코드의 실행에 대응하는 모든 클록 사이클을 이용한다. 트레이스의 전체 실행에 대응하는 트레이스는 단일 클록 사이클 실행에 대응하는 다양한 섹션들로 분할된다. 이후 서브섹션이 정렬되고 최고의 분산을 나타내는 고유벡터에 대응하는 변환 벡터 W 를 찾기 위해 PCA가 이용된다. 이 경우에, 앞서 설명한 바와 같이, 우리는 하나의 클록 사이클에 대해 오버샘플된 트레이스를 취하고 이를 한 점으로 줄인다.

[0115] PCA를 이용하여 플랫폼 특성화를 실행한 후에, 평가 루틴으로부터의 테스트 트레이스들은 플랫폼 사전 특성화의 성능 개선을 입증하기 위해 다시 처리된다. 변환된 테스트 트레이스들로부터 새로운 PCA 변환 공간의 서명까지의 최소 거리 분포는 도 23에 도시되어 있다.

[0116] 대부분의 분포들 간에 명료한 분리를 볼 수 있으며, 이는 도 22에 도시된 나이브 분류 성능에 비해서 명확한 개선을 나타낸다.

[0117] LDA를 이용하는 플랫폼 특성화의 결과

[0118] LDA를 적용하는데 필요한 트레이닝 트레이스들을 얻기 위하여, 우리는 기본 루틴과 약간 수정된 버전을 실행한다. 우리는 리스팅 4의 코드인 기본 실행과 리스팅 5에 도시된 것의 약간 수정된 버전으로부터의 두 세트의 트레이스들을 비교함으로써 특정 플랫폼 특성화 트레이스들을 얻는다. 실행에 있어서의 변화들은 기본 실행에 비교해서 각 실행 단계에서 하나 적은 비트 트랜지션이 일어나도록 주의 깊게 선택된다. 이러한 수정된 버전에서, 라인 36에 있는 명령은 opcode 0001 10da를 갖는 xorwf로부터 opcode 0001 00da를 갖는 iorwf로 변경된다(선택적 인수(optional argument) d 및 a는 각각 목적지와 RAM 액세스 비트를 제어하고 두 경우에 동일한 값으로 유지된다). 실행 동안에, opcode에 있어서의 차이는 명령어를 래칭(latching)할 때 하나 적은 비트 트

랜지션을 야기할 것이다. 명령에 있는 파라미터는 액세스 RAM에서 어드레스 0x01에 위치한 j로부터 어드레스 0x00에 있는 i로 변경되었다. 한번 더, 이 변경은 실행될 때 하나 적은 비트 트랜지션을 야기할 것이다. 더욱이, j 및 i의 콘텐츠는 또한 1 비트 차이가 있음을 주목하자. 이는 또한 파라미터를 파싱(parsing)할 때, 명령을 실행할 때 그리고 그 결과를 기록할 때 하나 적은 비트 트랜지션이 될 것이다.

[0119] 리스팅 5. 플랫폼 특성화를 위한 수정된 루틴

LISTING 5. Modified routine for platform characterization

```

35      ...
36      movlw 0x00      //w = 00
37      iorwf i, 0, 0   //w = 07
38      movlw 0x00      //w = 00
39      ...

```

[0120]

[0121] 플랫폼 특성화를 위해 우리는 리스팅 5의 라인 36의 실행에 대응하는 트레이스들만을 이용한다. 이들 트레이스의 평균(각 실행에 대한, 기본 실행 및 하나 적은 비트 트랜지션)은 도 24에 도시되어 있다.

[0122] 이들 트레이스를 이용하여, 우리는 가장 유익한 정보를 주는 관점에서 우리의 트레이스들을 투사하는 최적의 차별 초평면 및 선형 변환을 식별하기 위해 LDA를 이용한다. 평가 루틴으로부터의 테스트 트레이스들은 플랫폼 사전 특성화의 성능 개선을 입증하기 위해 다시 처리된다. 새로운 LDA 변환된 공간에서 변환된 테스트 트레이스로부터 서명까지의 최소 거리 분포들은 도 25에 도시되어 있다.

[0123] 소프트웨어-제어 라디오 플랫폼 및 다른 내장 시스템에서 인증된 소프트웨어 실행과의 편차를 검출

[0124] 프로세서의 동적 전력 소비는 그것이 예상된 실행에 대응하는지 또는 편차가 발생하였는지를 판단하기 위해 모니터링될 수 있다.

[0125] 플랫폼 설명

[0126] 전력 핑거프린팅(PFP)의 이용을 예시하기 위한 전형적인 타겟 플랫폼은 라디오 동작의 특정 구성이 소프트웨어에 의해 제어되는 소프트웨어-제어 라디오이다. 도 26에는 의도된 플랫폼의 전반적인 블록도가 도시되어 있다.

[0127] 이 플랫폼에서, RF 트랜시버(2610)의 동작 및 구성은 프로세서(2620)에 의해 제어된다. 애플리케이션(2626)은 가장 높은 계층을 나타내고 프로세서를 위한 의도된 기능을 구현한다. RF 트랜시버(2610)와 효과적으로 상호작용하기 위하여, 메인 애플리케이션에 대해 하드웨어와의 상호작용의 복잡함을 추상화하는 한 세트의 애플리케이션 프로그램 인터페이스(API)(2624)가 있다. 이들 API는 요구되는 드라이버 및 프로토콜 스택 구현(2622)과 함께 특정 트랜시버를 위한 보드 지원 패키지를 제공한다. 프로토콜 스택(2622)은 의도된 수신기가 정보를 추출할 수 있도록 필요한 헤더를 추가하고 페이로드를 준비해서, 미리 정한 패턴으로 전송될 수 있게 데이터를 배열한다. 이는 또한 원격 장치로부터 수신된 정보를 추출하여 이를 애플리케이션 층(2626)에 제공하는 역할을 맡고 있다. 암호 모듈(2612)은 RF 트랜시버에 또는 프로토콜 스택의 일부로서 소프트웨어로 구현될 수 있다. 도 26은 그것을 트랜시버의 일부로서 보여주고 있다. 암호 모듈(2612)의 위치는 접근법에서 실질적인 차이를 나타내지 않는다. RF 트랜지스터(2610)의 MAC(2614) 및 PHY(2616) 층은 매체 접근 및 정보의 물리적인 전송 및 수신을 담당한다.

[0128] 설명한 접근법은 애플리케이션 소프트웨어(2626)의 실행, 특히 암호 모듈(2612)의 동작에 영향을 주는 API 콜(2624)의 실행을 특성화한다. 이 접근법에서, API 콜의 결과로서 실행된 특정 코드는 암호가 사용되었는지 어떤 유형의 암호가 사용되었는지를 판정하는데 이용된다. 예를 들어, 애플리케이션이 특정 유형의 암호 전송을 요구한다면, 이 접근법은 암호를 호출하는 코드의 실행을 어서트(assert)한다. 악성 또는 사고 탬퍼링의 경우에, 이 접근법은 수정에 대한 신뢰할 수 있는 인디케이터를 제공한다.

[0129] 서명 추출

[0130] 인증된 코드 a의 i번째 실행 동안 캡처된 길이 L의 트레이스들은 다음 식으로 표현된다:

$$r_a^{(i)}[n], \quad n = 0, \dots, L-1$$

[0131]

[0132] 다른 보드 구성요소들로부터의 잠재적인 저-주파수 간섭을 피하기 위하여, 트레이스 샘플들 간의 차를 계산함으로써 기본적인 승수 없는 고역 필터가 도입된다:

$$d_a^{(i)}[n] = r_a^{(i)}[n] - r_a^{(i)}[n-1]$$

[0133]

- [0134] 인증된 코드의 실행으로부터 몇몇 캡처된 트레이스들은 서명, 우리의 타겟 핑거프린트를 산출하는데 이용된다. 타겟 서명을 형성하고 개별 트레이스에서 랜덤한 노이즈의 효과를 줄이기 위해서 N 트레이스들은 평균화된다:

$$s_a[n] = \frac{1}{N} \sum_{i=0}^{N-1} d_a^{(i)}[n] \quad n = 0, \dots, L-1$$

[0135]

[0136] 특징 추출

- [0137] 차별 특징들을 추출하는 프로세스는 타겟 서명에 대한 간단한 시간 도메인 상관관계로 구성된다. 그러나 상관관계는 서명과 트레이스의 $j > 0$ 부분 섹션에 대해 실행되고, 각각의 섹션은 길이 $w = \text{floor}\{L\}$ 을 갖는다. 이러한 부분 상관관계는 전체-트레이스 상관관계에 걸쳐서 전력 트레이스들에서의 전위차의 확산(spreading potential differences)을 피하기 위하여 실행된다.

- [0138] 트레이스들의 섹션 j 의 다양한 샘플 래그(lags), $0 \leq k \leq w$ 에 대한 교차 상관관계는 다음 식으로 주어진다:

$$\rho_{s_a d_b^{(i)}}(j, k) = \frac{1}{(w-1)\sigma_{s_a}\sigma_d} \sum_{n=(j-1)w}^{jw} s_a[n] d_b^{(i)}[k+n] - w \bar{s} \bar{d}$$

[0139]

- [0140] 여기서, $\bar{s}$ 및 σ_{s_a} 는 s_a 에 있는 대응 섹션의 샘플 평균 및 표준 편차이며 $\bar{d}$ 및 σ_d 는 $d_b^{(i)}$ 에 있는 대응 섹션의 샘플 평균 및 표준 편차이다.

- [0141] 임의의 클록 드리프트(drift)를 보상하기 위하여, 우리는 다양한 래그에 대해서 최대의 상관관계 값들을 유지한다. 이러한 액션은 우리의 트레이스들의 차원수를 모든 트레이스에 대해서 j 피크 상관관계 값들의 시퀀스로 줄여준다:

$$\hat{\rho}_{s_a d_b^{(i)}}(j) = \max_k \{ \rho_{s_a d_b^{(i)}}(j, k) \}$$

[0142]

- [0143] 이상적인 조건하에서 $b \equiv a$ 일 때, 모든 섹션 j 에 대해서 $\hat{\rho}_{s_a d_b^{(i)}}(j) = 1$ 이다. 전력 소비 특성들로부터의 임의의 편차는 축소된 상관관계 인자(reduced correlation factor)에 의해서 반영될 것이다. 트레이스들을 평가하기 위해 이 작업에 이용된 실제의 차별적인 특징들 또는 테스트 통계들은 그 특정 트레이스에 대한 최대 피크 상관관계 값이다:

$$x_b^{(i)} = \min_j \{ \hat{\rho}_{s_a d_b^{(i)}}(j) \}$$

$$X_b = x_b^{(i)}; i = 0, \dots, N-1$$

[0144]

- [0145] 랜덤한 변수 $x_b^{(i)}$ 는 코드 b 의 인스턴스 i 의 서명으로부터의 최대 편차를 나타낸다. X_b 를 이용하여, 우리는 시스템으로부터 선택적으로 모을 수 있는 통계 정보에 따라서 다양한 기준을 이용하여 적절한 검출기를 설계할 수 있다.

[0146] 무결성 침입 및 계층 보안에 대한 대응

- [0147] PFP는 사이버 시스템에서 실행 편차를 검출하는데 매우 효과적인 접근법이다. 그러나, 완전한 솔루션을 얻기 위해서는 PFP 모니터가 예상 실행과의 편차를 검출할 때 무결성 침입을 다루는 구조 정책(structured policy)을 가질 필요가 있다.

- [0148] 컴퓨터 보안에는 명확히 정의된 3개의 단계가 있다:

- [0149] • 예방. 공격자가 시스템을 파괴하고 정보를 공개하는 등의 공격을 계속하지 못하게 단념시키고, 좌절시키고 및 방지하기 위한 활동적인 메커니즘을 포함한다.

- [0150] • 검출. 완벽하고 절대적인 예방은 가능하지 않기 때문에, 시스템의 무결성을 끊임없이 모니터링하는 것이 필요하다.

- [0151] • 대응. 성공적인 공격에 반응하기 위해 설정된 정책 집합.

- [0152] PFP를 포괄적인 심층방호 보안 접근법에 통합하기 위한 아키텍처를 지금 설명하기로 하다. 이 접근법에서 PFP는 잠재적인 공격을 예방 및 단념시키기 위한 다수의 다양한 기술을 보완할 수 있는 "검출" 단계에 대한 강력한 솔루션을 제공한다. 다양한 성공적인 공격에 대한 적절한 반응은 "대응" 단계에 정의되어 있고 이하 설명되는 보안 정책에 따라서 설명된다.
- [0153] 시스템 보안을 성취하려면 고립된 메커니즘 또는 기술만이 아니라 프로세스가 필요하지만, 우리는 무결성 평가 및 침입 판정을 위한 연속적인 또는 간헐적인 모니터링을 제공하기 위해 PFP가 전통적인 보안 메커니즘을 보완할 수 있는 영역을 설명하는데 집중하고자 한다. PFP의 역할을 설명하기 전에 보안 프로세스가 다음의 몇몇 단계를 포함한다는 것을 언급하는 것이 중요하다.
- [0154] · 설계. 정상적인 설계 접근법에 따라서, 보안 시행을 용이하게 하고, 취약성을 줄이고, 접근 제어를 시행하는 등을 위해 시스템을 설계한다. 전형적인 예는 보안 기능들이 나머지 기능과 분리되고 접근 제어 특징들이 내재적으로 시행되도록 시스템을 설계하는 것이다.
- [0155] · 개발. 취약성이 줄어든 유지가능 제품을 생산하기 위해 최고의 개발 관례를 따른다.
- [0156] · 설치. 인증된 모듈들만이 설치되도록 한다. 이는 강력한 인증 및 부인 방지(non-repudiation) 접근법들을 필요로 한다.
- [0157] · 운영. 강력한 접근 제어 및 다른 보안 정책을 시행함으로써 보안 환경을 유지한다.
- [0158] · 모니터링. 시스템의 무결성을 끊임없이 평가한다. PFP, 안티바이러스 및 네트워크 침입 검출 시스템.
- [0159] · 대응. 공격이 성공적일 때 뒤따르는 정책 및 절차를 정의한다. 정책은 시스템의 위험 상태(criticality)를 고려하여 개발되어야만 하고 엄격히 시행되어야만 한다.
- [0160] 이 장은 한 계층의 취약성이 다음의 것에 의해 커버되는 상호보완 보안 메커니즘을 포함하는 포괄적인 보안 솔루션에 PFP 모니터를 통합하기 위한 아키텍처를 설명한다. 다양한 계층에 포함되어 있는 접근법 및 기술은 다음을 포함한다: 보관중 데이터(data at rest)의 암호화, 강력한 인증, 접근 제어, 탬퍼링 방지(tamper-resistance), 방화벽, 샌드박스, 가상화(virtualization), 및 물리적인 보호. 이 아키텍처는 또한 PFP에 의해 검출된 무결성 침입에 반응하도록 보안 정책을 정의하고 시행하기 위한 메커니즘을 제공한다.
- [0161] 이 아키텍처는 PFP 모니터가 침입자가 어떻게든 해서 모든 다른 방어 메커니즘을 통과하는 때를 검출함으로써 마지막 방어선을 제공하는 계층 보안 솔루션을 제공한다. 도 27은 심층방호 보안에 있어서의 다양한 층(2700)을 보여주고 있다. 다양한 층들은 적대자를 지체시켜서 의식하지 못하게 방어 층을 파괴하는 것을 점차 더 어렵게 만들도록 되어 있다. 바깥 층들에는 인터넷 방화벽, 장비 및 패스워드의 물리적인 보호, 및 보안 정책(즉, 사회 공학 기법 공격 방어용)과 같은 외부 방어 메커니즘이 있다. 내부 층들은 호스트(2750) 안에 상주하는 다양한 방어에 해당한다. 이들은 접근 제어(2740) 및 보관중 데이터의 암호화로 시작한다. 이들은 애플리케이션(2760) 및 운영 시스템(2770)을 보호하도록 의도된 다양한 보안 메커니즘으로 계속된다. 코어(2780)에는, 가장 기본적인 커널 및 안전 운영을 위한 제어들이 있다.
- [0162] PFP는 다양한 계층의 무결성을 효과적으로 모니터한다. 코어 레벨(2780)에서, PFP는 모든 다른 메커니즘이 의존하는 커널 및 안전 운영의 무결성을 평가할 수 있다. 이는 또한 중요한 유저-레벨 애플리케이션(2760)의 무결성은 물론이고 운영 시스템(2770)에 있는 코어 애플리케이션의 무결성을 모니터하는 데까지 확장될 수 있다. PFP가 도 28에 도시된 바와 같이, 안티바이러스 모듈 및 암호화 모듈을 포함해서 프로세서 범위 내에 상주하는 모든 모듈의 무결성을 모니터할 수 있음은 주목하자.
- [0163] 사이버 보안을 위한 심층방호 보안에 PFP를 통합하면, 잠재적인 사건들이 그들의 목적을 달성해서 손상을 입히기 전에 이들을 좀 더 신속하게 처리할 수 있다.
- [0164] 암호화 및 안티바이러스와 같은, 다른 보안 모듈의 실행으로부터의 전력 서명은 실행 시간에 추출되고 평가된다. PFP 관점에서 보면, 커널 모듈 및 안티바이러스 프로그램으로부터의 서명들은 동일 기술을 이용하여 동일 방식으로 추출된다.
- [0165] PFP가 실행 무결성을 모니터하기 위한 임의 다른 장치에까지 확장될 수 있음에 주목하는 것은 중요하다. 이는 방화벽, 디지털 보안 락(lock) 등과 같은 다양한 보안 계층을 구현하는데 이용될 수 있는 장치를 포함한다.
- [0166] PFP에 의한 사이버 방어의 마지막 단계는 모니터링 및 평가 작업에 의해 검출된 다양한 이상을 처리하기 위한 적절한 정책을 정의하는 것이다. 성공적인 침입에 대한 적절한 반응은 각각의 플랫폼 및 애플리케이션에 고유

한 몇몇 팩터에 의존하기 때문에, 우리는 PFP 모니터에 의해 검출된 다양한 침입에 대한 대응을 일반화할 수가 없다. 그러므로, 기본적인 운영 원칙 및 구조를 유지하고 재이용하면서, 다양한 시스템에 맞추어질 수 있는 다양한 보안 정책 정의를 수용하고 시행하는 아키텍처를 따를 필요가 있다.

[0167] 멀웨어 역학 및 행동을 모니터링하기 위해 분포된 PFP 모니터

[0168] 이 장은 멀웨어의 확산을 모니터링하고, 타겟 공격을 검출하고, 악의 있는 적의 잠재적 의도를 찾아내기 위해 다양한 지정학적 또는 논리적 영역에 걸쳐서 설치되어 있는 PFP 능력을 갖춘 넓은 노드 네트워크의 운영을 설명한다. 이 접근법은 특정한 논리적 또는 지정학적 영역의 은밀한 원격 공격을 찾아내는데 적용될 수 있다.

[0169] 이 응용을 위해 PFP를 이용하는 주요한 장점들 중 하나는 그의 스텔스(stealth)에 있고, 이 스텔스는 적들이 모니터링 활동 그 자체를 탐지하지 못하게 해서 그들에게 은밀함에 대한 잘못된 인식을 심어주고(그들이 탐지되지 않았다고 믿게 하는 것), 그들이 그들의 활동을 계속하도록 속여서 그들의 의도 및 능력을 찾아내는 것이다. 이러한 PFP의 응용은 정보 수집을 위한 매우 강력한 도구이다.

[0170] 운영

[0171] 스텔스 모니터링은 PFP의 낮은 풋프린트(low footprint), 및 메모리에 대한 무시 가능한 영향 및 타겟 시스템에 대한 레이턴시 덕분에 성취된다. PFP 노드의 분산된 네트워크는 다음 단계를 이용하여 구현될 수 있다:

[0172] 1. PFP를 갖춘 대표적인 노드들을 이용가능하게 한다(이들에 PFP 모니터를 설치하고 이들의 타겟 구성요소로부터 신뢰 서명을 추출한다). 모니터는 타겟 노드들이 단지 허니 팟(honey pot)으로 작동할 때 랙 마운트(rack mounted)될 수 있고 풍부해질 수 있다(resourceful).

[0173] 2. 당해 타겟 지정학적 또는 논리적 영역에 PFP 이용가능 노드들의 네트워크를 설치한다.

[0174] 3. 도 29에 도시된 바와 같이 무결성 침해 및 침입에 대해 각 노드를 개별적으로 모니터링한다.

[0175] 4. 무결성 결과를 로깅(logging) 및 분석을 위해 중심 위치에 주기적으로 제출한다.

[0176] 5. 무결성 침해의 이벤트시에, 보고서는 다음을 포함해야 한다:

[0177] (a) 침해당한 전력 트레이스의 사본

[0178] (b) 침해 전에 실행된 탐퍼링되지 않은 모듈 실행의 정연한 시퀀스

[0179] (c) 침해 후에 실행하는 모듈의 정연한 시퀀스.

[0180] 이러한 PFP의 응용은 도 30에 도시되어 있다. 이 도면은 다양한 지정학적 네트워크에 있는 PFP 허니 팟을 보여주고 있다. 그러나 네트워크 분리가 동일한 네트워크의 다양한 섹션들에서와 같이 논리적일 수 있거나 또는 다양한 정부 기관 또는 기업 분할을 위한 네트워크에서와 같이 사회적/정치적일 수 있다는 것에 주목하는 것이 중요하다.

[0181] 도 30에 점선으로 나타낸 허니 팟들과 중앙 분석 로케이션 간의 링크는 분리 네트워크(예를 들어 전용 무선 링크)로 구현될 수 있거나 또는 공공 교환 전화 네트워크(PSTN) 또는 인터넷과 같은 이용가능한 광역 네트워크를 이용하여 만들어질 수 있다. 어떤 경우에도, 트레이스들의 근원 및 무결성의 입증((높은 보장 인증)을 제공하는 역할을 맡는 강력한 부인 방지 메커니즘은 전체로서 시스템에 신뢰성을 유지할 수 있게 배치되어야만 한다.

[0182] 공급-사슬 신뢰 분석에 PFP의 응용

[0183] 외국의 신뢰할 수 없는 공장과 제조업자에게 위탁된 장치 제조 및 생산은 잠재적인 보안 위반 및 탐퍼링에 기회를 준다. 신뢰할 수 있는 제공자라 하더라도, 이질적이거나 불만이 있는 자가 중요한 시스템의 운영 및 기능을 방해할 수 있는 가능성이 존재한다.

[0184] PFP는 시스템의 수명 전반에 걸쳐서 공급 사슬에서 신뢰할 수 없는 링크에 의해 도입된 소프트웨어, 펌웨어 및 하드웨어에 인증되지 않는 수정 및 다른 탐퍼링을 탐지하기 위한 메커니즘을 제공한다. PFP를 이용하는 새로운 수송 물품 및 신뢰할 수 없는 장치에 대한 무결성 평가는 도 31에 도시된 단계들을 필요로 한다. 입력 벡터 발생기(3110)는 제어 환경(3120)에 있는 타겟 장치들의 실행을 위해 필요한 입력을 제공하는데 이용되고, 타겟 장치들의 실행 동안 전력 트레이스들이 수집된다(3130). 전력 소비의 개별적인 특성에 대한 파라미터는 특징 추출(3150)을 실행하기 전에 보상된다(3140). 최종 특징들이 저장된 기준 서명(3170)에 비교되고(3160), 이 비교로부터 최종 평가 결과가 얻어진다.

- [0185] PFP를 이용하는 디지털 장치의 무결성 검출은 파괴적인 프로세스가 아니며 단지 평가되는 장치로부터의 최소한의 협력(collaboration)만을 요구한다. 더욱이, 전력 소비의 세립 측정은 장치의 내부적인 실행 상태에 상당한 가시성을 제공하며, 이는 눈에 띄지 않게 수정하는 것을 극히 어렵게 만든다. 예를 들어, PFP는 상태를 검사하는 동안에 추가 기능 또는 실행 플로우의 부분적인 활동으로 인해 단지 특정 조건 하에서 활동하는 탬퍼링(로직 및 시한 폭탄이라고도 알려져 있음)을 검출할 수 있다. 추가 또는 누락된 기능을 검출하기 위한 PFP의 능력은 삽입 그 자체의 목적이나 의도에 의존하지 않는다.
- [0186] PFP의 다른 장점은 악의적인 활동이 유발되지 않더라도 PFP로 검증된 특정 실행 경로는 신뢰받을 수 있다는 것이다. 다른 말로, PFP가 서명으로부터 상당한 편차를 발견하지 못하면, 이는 탬퍼링이나 또는 추가 기능이 이 특정 실행 경로에 발생하지 않았다는 것을 의미한다.
- [0187] PFP에 의한 공급 사슬 신뢰 분석을 실행하는데 있어서의 키 요소는 제어 환경(3120) 하에서 신뢰할 수 없는 장치를 실행하는 것이다. 이 제어 환경은 특정한 상태 시퀀스를 강제하는 미리 정의된 입력(3110) 및, 프로그램 가능 장치를 위해 실행되는 특정 소프트웨어를 포함한다. 어떤 시스템의 경우, 특정 구성요소의 실행을 제어하고 분리시키기 위하여 지원 스캐폴딩(scaffolding)을 개발해야할 필요가 있을 수 있다. 특정 입력 벡터는 장치 또는 소프트웨어 모듈의 기능에 의존하고, 이들은 장치의 운영을 위한 중요한 실행 경로를 엑서사이즈(exercise)하는 것으로 예측된다. 서명을 추출하는데 이용된 동일한 입력 벡터는 신뢰할 수 없는 장치의 무결성을 평가하는데 이용될 필요가 있다.
- [0188] 제조 동안 약간의 프로세스 편차 때문에, 다양한 장치들은 다양한 전력 소비 특성을 보여줄 것이다. 전력 소비에 있어서의 이들 편차는 잘못된 평가를 피하기 위해 특징 추출(3150) 전에 보상될(3140) 필요가 있다. 이러한 보상은 그의 탭(tap)들이 전력 소비 트레이스들의 특정한 특성에 부합되게 동적으로 수정되는 적응 필터에 의해서 실행된다. 이 적응 필터는 PFP 모니터가 실행 동안에 장치 레지스터에서의 비트 트랜지션의 결과로 나오는 전력 소비에 집중하고 제조 변화에 기인한 트레이스들의 차이를 제거하게 해준다.
- [0189] PFP를 이용하는 효과적인 공급 사슬 신뢰 분석을 위한 가장 중요한 면은 기준 서명(3170)의 이용가능성이다. 도 32에 도시된 바와 같이, 그러한 서명을 위한 다양한 잠재적인 소스가 있다. 최고의 기준은 동일한 신뢰 구현(골드 표준)(3230)에 의해 제공되는 것이다. 그러나, 많은 경우에, 그러한 신뢰할 수 있는 기준은 이용가능하지 않다. 이들 경우에, 기준 서명은 오차 및 신뢰성의 정도가 가변하는 대안적인 방법들에 의해서 추출될 수 있다. 예를 들어, 두 개의 비교적 간단한 대안 기준 소스들은 장치의 이전의 구현(시간 테스트된 것)(3250) 또는 대안적인 공급자의 다른 구현(3260)을 포함한다. 이들 경우에, 서명들은 대안적인 구현들로부터 추출되므로, 다른 공급자들에 의한 두 번의 동일한 수정 기회가 줄어든다. 이전 접근법으로부터의 서명은 이전 버전에 제공된 확인되지 않은 수정을 놓칠 수 있다. 후자의 접근법에서, 공격자는 검출되지 않기 위해서 상이한 공급자들로부터의 양쪽 버전에 동일한 수정을 만들 수 있을 것이다.
- [0190] 서명을 얻기 위해 CAD 모델(3240)을 이용하는 것은 더 많은 노력을 필요로 하지만, 이는 외국의 공장에 의지함이 없이도 실내에서 실행될 수 있다. CAD 모델을 이용하여 서명을 추출하기 위해서는, 결정성 입력 벡터를 이용하여 장치의 실행을 시뮬레이트하는 것이 필요하다. 시뮬레이터는 레지스터 전송 레벨에 대한 전력 소비의 면에서 정확해야할 필요가 있다.
- [0191] **디지털 권리 관리(DRM) 및 실행 제한 리스**
- [0192] PFP에 대한 다른 신규의 응용은 실행 횟수를 기반으로 라이선싱을 가능하게 해주는 디지털 권리의 시행과 실행 사례-제한 임대(execution instance-limited leasing)의 출현이다.
- [0193] 이 접근법은 보호되는 소프트웨어의 실행으로부터 서명을 추출함으로써 구현되고 인증된 모듈의 단독 실행을 할 수 있게 해주는 전력 핑거프린트를 실행시에 모니터링한다. 예를 들어, 소프트웨어 시스템은 상위 라이선스 계층(higher license tier)을 위해 비축된 모듈들의 서브세트와 단지 한 세트의 기능 모듈을 포함하게 라이선스될 수 있다. 모든 모듈로부터의 핑거프린트는 배포(release)되기 전에 추출된다. 실행 시간에 PFP 모니터는 다양한 모듈의 실행을 인증 라이선스에 매칭(matching)시킨다. 도난 패스워드 또는 보안 위반의 결과로, 라이선스되지 않은 모듈이 실행될 때, PFP 모니터는 침해를 발급 대리점(issuing agency)에게 알려줄 수 있다. 더욱이, 보호되는 소프트웨어를 위해 신뢰 실행-사례-제한 라이선스 접근법을 실행하는 것도 가능하다. 이 경우에, PFP 모니터는 라이선스 받은 소프트웨어가 실행된 횟수의 카운트를 유지하고 임대가 종료될 때 발급 대리점에게 통보한다.
- [0194] 유사한 접근법이 라이선스를 받은 미디어 콘텐츠에도 적용될 수 있다. PFP 모니터를 이용하면, 공지된 미디어

플레이어로 특정 파일을 재생하는 것을 PFP를 이용하여 검출하는 것도 가능하다. 이 경우에, 보호 미디어 데이터는 PFP 특성화 동안에 미리 정한 입력을 대신한다. 동일한 미디어가 특정한 플레이어로 재생되면, 전력 서명들이 일치하게 된다. 그러므로, PFP는 인증되지 않은 라이선스를 받은 미디어의 재생을 검출하는데 이용될 수 있다.

[0195] PFP를 기반으로 한 고장 예측

[0196] 하드웨어 구성요소는 피할 수 없는 에이징(aging) 과정을 겪는데, 이러한 에이징은 가혹한 환경에서의 작동에 의해 가속되고 또는 시스템이 연속적인 환경 스트레스의 영향하에서 작동할 때 가속된다. 이러한 에이징은 플랫폼의 전력 소비 특성에도 반영된다. PFP는 소프트웨어의 올바른 실행뿐만 아니라 하드웨어 플랫폼의 무결성을 모니터링하는데 이용될 수 있다. PFP 모니터링은 하드웨어의 전력 소비 특성을 계속해서 추적할 수 있고 실제로 고장이 나기 전에 고장을 예측하여 특정 시스템 또는 요소가 교체되어야만 하는 때를 지시할 수 있다.

[0197] PFP에서 전력 소비 특성을 추적하는 것은 적응 필터를 이용하여 구현된다. 서명이 추출되는 전력 소비에 있어서의 차 또는 환경 조건에 기인한 차를 보상해야할 필요가 있다. 동일한 추적 메커니즘은 또한 하드웨어의 상태를 모니터링하고 전력 소비 특성을 장치의 실험실 검사에서 캡처한 미리 정한 패턴에 비교하는데 이용될 수 있다. 고장 특징을 식별하기 위한 프로세스는 도 36에 도시되어 있다. 이 프로세스에서, 타겟 장치를 급격한 온도 변화에 노출시켜서 가속된 에이징(3610)을 얻을 수 있다. 특성화 프로세스는 테스트 루틴의 실행 동안 트레이스 캡처(3620)가 뒤따르는 1라운드의 간격으로 진행된다. 이들 트레이스는 사후 분석을 위해 수집되고 이 프로세스는 장치가 고장이 날 때까지 반복된다. 일단 장치가 고장이 나면 고장(3630) 이전에 나타난 특정한 특징들을 판정하기 위해 트레이스 세트가 검사된다. 이들 특징은 통계 다양성을 제공하고 포괄적인 특징(3640)을 분리하기 위해서 다른 유사한 장치들로부터 추출된다.

[0198] 모듈 식별 정보를 동기화 시그널링에 임베드

[0199] PFP는 정확한 평가를 제공하기 위하여 실행되는 소프트웨어와의 적절한 동기화를 필요로 한다. PFP의 동기화에는 다음과 같은 두 가지의 레벨이 있다: 클록 사이클 레벨 및 루틴 레벨. 전자는 클록 사이클 속도로 나타나는 전력 소비에 있어서의 구별되는 사이클들을 추적함으로써, 또는 간단한 플랫폼의 경우에는 클록 신호 그 자체를 검출함으로써 쉽게 성취될 수 있다. 후자의 동기화는 성취하기가 좀 더 어렵고 이 프로세스는 특정 루틴의 실행을 PFP 모니터에 통보해 주는 트리거 또는 식별자를 루틴 자체에 임베드함으로써 가능해진다.

[0200] 이 장에서, 우리는 실행중인 노드의 식별을 트리거링 및 시그널링 메커니즘에 임베드하기 위한 메커니즘을 제시한다. 이 메커니즘은 어느 특정 루틴이 실행될 것인지를 PFP 모니터에 통보해 주는 것뿐만 아니라 좀 더 정확한 이상 검출 및 행동 서명 추출을 위해 강력한 동기화 시그널링을 제공하는데 도움을 준다.

[0201] 궁극적인 목적은 PFP를 위한 동기화 및 트리거링 아티팩트(artifacts)에 삽입되는 식별 코드를 특성화될 다양한 모듈에 제공하는 것이다. PFP를 위해 동기화 및 식별 시그널링을 제공하는 데는 두 가지의 주 접근법이 있다: 1) 도 33에 도시된 바와 같이 인접 물리 신호를 생성하는 것, 및 2) 도 34에 도시된 바와 같이 신호를 소비 전력 그 자체에 임베드하는 것. 전자의 경우, 루틴(3322)의 실행 전에 이진 식별 코드가 물리적인 IO 레지스터(3324)에 기록된다. 이후 레지스터는 PFP 모니터(3340)에 전송되고(3335), 이 모니터는 병렬 또는 직렬 방식으로 전력 트레이스들(3315)을 센서(3310)로부터 추출한다. 코드 및 레지스터의 길이는 모니터링 필요에 있는 루틴의 수에 의존한다. 가장 단순하게는, 타겟 루틴의 실행을 신호로 알려주기 위해 LED와 같은 싱글-비트 레지스터를 이용할 수 있다. 개별 물리적인 시그널링의 경우에, 도 33에 도시된 바와 같이, 트리거는 시그널링 레지스터에 이진 수로서 인코딩된다.

[0202] 두 번째 접근법은 구별되는 전력 소비 패턴을 산출하는 정교하게 만들어진 명령의 시퀀스(3422)를 삽입함으로써 전력 소비 그 자체에 임베드될 동기화 시그널링을 필요로 한다. 이 접근법은 도 34에 도시되어 있다. 동기화 루틴들에 대한 명령은 그들의 코드워드, 어드레스 및 파라미터에서의 비트 트랜지션이 궁극적으로 전력 소비를 구동하는 특정 수의 비트 트랜지션을 산출하고 센서로부터 나오는 올바른 트레이스들의 세트(3415)를 캡처하기 위해 특정 시퀀스가 실행된다는 것을 PFP 모니터(3340)에 신호로 알려주도록 선택된다. 비트 트랜지션이 많을수록 전류 드레인이 높아진다. 시퀀스를 정할 때는 파이프라인의 길이와 특성이 고려되어야 한다. 이전의 접근법과 유사하게, 명령(코드)의 시퀀스의 길이는 식별될 필요가 있는 중요한 루틴의 수에 의존한다. 다양한 구별되는 전력 소비 패턴을 생성함으로써, 다양한 모듈을 식별하는데 이용되는 다양한 서명 코드를 생산하도록 시퀀스 그 자체가 선택된다.

[0203] 동기화 시그널링은 가장 중요한 사항인 코드의 섹션들에 대한 평가 노력을 집중할 수 있게 해주기 때문에, 동기

화 시그널링이 효과적인 PFP를 위해 필요한 요소라는 점을 주목하는 것은 중요하다. 식별 코드를 시그널링에 임베드하는 것은 평가 프로세스를 가능하게 해주지만 필요한 필요조건은 아니다. 이는 싱글 트리거를 이용해도 PFP 모니터가 올바른 트레이스들의 세트를 캡처할 수 있으며, 어느 특정 루틴이 실행되었는지 또는 설정될 수 있는 신뢰가능한 매칭이 없는지(이상) 여부를 판정하는데 신호 분류 기술이 이용될 수 있기 때문이다.

[0204] 다양한 보드 요소로부터의 신호들을 결합해서 PFP 모니터링을 개선

[0205] 시스템의 다양한 요소로부터의 신호들은 PFP 모니터에 의해 이용될 수 있고 개선된 성능과 신뢰성을 제공하도록 합쳐질 수 있다. 다수 신호의 소스들은 다수의 프로세서, 코-프로세서, 주변기기, 또는 PFP를 강화할 목적으로만 도입된 다른 특정-목적의 요소(예를 들어, 트리거링을 위해 사용되는 IO 레지스터)를 포함한다.

[0206] PFP에서 다양한 소스들로부터의 신호들을 결합하는 방식은 다양하다. 주요한 접근법들 중 한 접근법은 멀티프로세서 및 멀티코어 보드에 대한 무결성 평가를 실행하기 위해 다양한 프로세서 또는 다른 디지털 회로로부터 전력 트레이스들을 캡처하는 것을 포함한다. 다른 접근법은 무결성 평가 동안에 이용될 추가 맥락 정보를 모으기 위해 시스템의 다른 요소(전력 소비 또는 다른 사이드 및 다이렉트 채널)을 모니터링하는 것이다. 추가 맥락 정보는 동기화를 개선하고 행동 특성을 용이하게 하는데 이용될 수 있다. 맥락 정보는 정상적인 시스템 동작의 산물로서 생성될 수 있거나 또는 설계할 때(예를 들어, 트리거링을 위해 이용되는 IO 레지스터) 의도적으로 도입될 수 있다. 다수의 신호들을 결합하는 PFP 모니터의 표본적인 셋업은 도 35에 도시되어 있다.

[0207] 직접 지원 IO 레지스터로부터, 다양한 요소의 전력 소비로부터, 또는 전자기 방사선과 같은 다른 사이드-채널들로부터 부가적인 신호가 캡처될 수 있다. 다양한 소스로부터의 신호들을 결합하려면 다양한 특징을 지원할 수 있는 특별히 제작된 검출기가 필요하다. 특정한 결합 메커니즘은 시스템 기능 및 지원 플랫폼에 의존한다. 예를 들어, 멀티-코어 프로세서의 경우, 각 코어로부터의 전력 소비는 타겟 루틴에 대응하는 트레이스들을 찾기 위해 스캔될 수 있다. 다른 예로, 소프트웨어-정의 무선(software-defined radio)의 경우, 전력 증폭기(PA)의 활동은 전력 소비를 모니터링함으로써 검출될 수 있고 이는 무선 송신이 발생할 때 나타난다. PA의 활동은 전송될 데이터를 준비하는데 관여된 루틴을 위한 트리거 메커니즘으로서 이용될 수 있다(이 경우에, 이들 루틴은 트리거가 발생하기 전에 실행한다는 점을 주목하자).

[0208] PFP 성능을 강화하기 위해 멀웨어 서명을 이용

[0209] PFP의 주 응용이 이상 검출일지라도, 평가 성능을 향상시키기 위해 공지된 멀웨어로부터의 유용한 정보를 이용하는 것은 중요한 실익이 있다. 멀웨어의 새로운 트렌드가 식별될 때, 그의 PFP 서명을 추출해서 그것을 공지된 서명의 라이브러리에 추가하는 것이 가능하다. 이들 멀웨어 서명은 전통적인 안티바이러스 소프트웨어와 유사한, 설치된 멀웨어의 전통적인 서명-기반 검출을 제공함으로써 PFP 무결성 평가의 성능을 개선하는데 이용될 수 있다. 모니터는 올바르게 않는 평가를 피하기 위하여 각 서명의 개별 성격(화이트 리스트 및 블랙 리스트)을 알도록 만들어질 필요가 있을 것이다. 멀웨어 서명은 또한 실행시의 행동 패턴으로부터 추출될 수 있다. 예를 들어, 고갈 공격(exhaustion attacks)과 같은 어떤 멀웨어 유형은 PFP를 이용하여 용이하게 식별될 수 있는 매우 구별하기 쉬운 실행 패턴을 갖는다.

[0210] 멀웨어로부터 서명을 추출하는 프로세스는, 제어 환경에서 타겟 모듈들이 반복적으로 실행되고 베스트 차별 특성을 갖는 특징들을 선택하기 위해 다양한 신호 처리 기술이 최종 전력 트레이스들에 적용되는 신뢰 소프트웨어로부터 서명을 추출하는 프로세스와 유사하다. 멀웨어 특성화는 멀웨어가 제어 환경에서 식별되고 분리되어 실행되었을 때 가능하다는 점을 주목하는 것은 중요하다.

[0211] 자동 특성화 및 서명 추출

[0212] 새로운 소프트웨어 시스템 또는 기존 시스템의 새로운 버전을 효과적으로 특성화하기 위해서는, 신뢰 기준을 자동으로 특성화하고 특정 소프트웨어의 실행을 고유하게 식별하는 PFP 서명을 추출하기 위한 도구가 필요하다. 어떤 면에서, 이 프로세스는 통제 조건하에서 특정 모듈의 실행을 요구하기 때문에 자동 테스팅과 유사하다. 그러나 자동 테스팅과는 다르게, PFP 특성화는 단지 다양한 모듈의 몇몇 실행 사례를 "관측"하는데만 관련되어 있지 어떤 필요조건이나 속성을 평가하지 않는다.

[0213] 이 장의 목적은 복잡한 시스템 및 소프트웨어 아키텍처의 특성화를 가능하게 하고 현실적으로 복잡한 사이버 시스템의 실제 구현으로부터 서명을 추출할 수 있게 해주는 접근법을 설명하는 것이다. 이러한 자동 접근법이 없으면, 전력 핑거프린팅에 이용될 복잡한 시스템(즉, 상업용 시스템)으로부터 고유한 서명을 특성화하여 추출하는데 긴 시간이 걸릴 것이다.

- [0214] 주요한 목적은 핑거프린트를 형성하는 최적의 차별 특징을 식별하기 위해 다양한 통계 분석과 신호 처리를 이용함은 물론이고 소프트웨어 테스트에 일반적으로 이용되는 것과 유사한 스캐폴딩을 이용함으로써 다양한 모듈에 대한 특성화의 프로세스를 자동화하는 것이다. 이 프로세스는 새로운 소프트웨어 스택이 특성화될 필요가 있을 때 시작한다. 이 프로세스를 위한 도구는 다음을 포함한다: 크리티컬 모듈 디스크립터, 특징 추출을 위한 신호 처리 도구, 검출기 설계 도구, 모듈 실행을 위한 스캐폴딩(테스팅 스캐폴딩과 유사함), 입력 벡터 생성기, 보고서 생성, 및 서명 패키징. 이 접근법의 이해를 도모하기 위해서, 우리는 다양한 서브시스템들 간의 세부사항과 상호관련성을 설명하는 프로세스의 하이-레벨 뷰를 제공한다. 이 관계는 도 37에 도시되어 있다.
- [0215] · 디스크립터는 고유 식별자, 의존성, 입력 분석(다양한 입력 클래스의 브레이크다운), 실행 모드(동적으로 링크된, 우선권, 커널 모듈 등)를 포함해서, 크리티컬 모듈에 대한 필요 정보를 포함한다.
- [0216] · 디스크립터로부터의 정보는 타겟 모듈의 분리 실행을 제어하기 위해 스캐폴드를 구현하는데 이용된다. 스캐폴드는 시스템이 모듈의 실행을 제어하기 위해 결정성 값들을 입력으로 엔터할 수 있게 해준다.
- [0217] · 기능과 다양한 입력 유형에 대한 디스크립터에 있는 정보는 적합한 입력 벡터의 세트를 결정하는데 이용된다.
- [0218] · 커버리지 분석은 시스템을 위한 보호 레벨의 메트릭을 산출하는 엑서사이즈된(exercised) 실행 경로들을 식별하기 위해 실행된다.
- [0219] · 일단 시스템이 로딩되면, 오퍼레이터(이는 자동 시스템일 수 있다)는 스캐폴더로부터 지원을 받고 적절한 입력 벡터를 제공하는 다양한 모듈을 실행한다. 모듈들이 실행되고 있는 동안, PFP 모니터는 전력 소비의 측정치를 캡처한다.
- [0220] · 모니터에 의해 캡처된 전력 트레이스들은 이후 차별 특징들을 추출하기 위한 다양한 신호 처리 기술을 이용하여 처리된다. 다양한 도메인에 있는 각 구성요소에 대해서 다양한 기술을 이용하여 추출될 미리 정의된 특징 세트가 있다.
- [0221] · 몇몇 트레이스가 캡처되고 각각의 고장이 분석된 후에, 응용을 위한 특정 필요조건을 기반으로 정상적인 활동과 이상을 구별할 수 있는 최적의 검출기를 설계하기 위해 실행된다.
- [0222] · 서명 및 검출기는 이후 타겟 시스템의 무결성 평가를 하는 모니터와 함께 설치되도록 함께 패키징된다.
- [0223] 다음 장들은 앞서 설명한 접근법을 성공적으로 구현하는데 필요한 상세한 설명을 포함한다.
- [0224] **디스크립터**
- [0225] 디스크립터는 특성화된 특정 모듈에 대한 메타-정보를 포함한다. 이들은 각 모듈의 실행을 분리하고 다양한 실행 경로를 엑서사이즈하기 위해 제어 환경을 제공하기 위한 스캐폴딩 아티팩트(scaffolding artifacts)를 개발하는데 이용된다.
- [0226] 디스크립터는 사람과 기계가 쉽게 판독할 수 있는, 확장성 마크업 언어(XML)와 같은 마크업 언어로 제공되는 것으로 예상되지만, 콘텐츠, 언어, 및 구조는 특성화 처리를 자동화하는데 이용되는 특정 도구에 의존할 것이고 소유권이 등록되어 있을 수도 있다.
- [0227] PFP 특성화를 위한 모듈 디스크립터에 요구되는 최소한의 정보는 다음을 포함한다:
- [0228] · 각 모듈을 설명하기 위한 고유 식별자. 고유 식별자는 사람이 판독할 수 있어야 하고 당해 모듈을 고유하게 찾는데 필요한 정보를 제공해야 한다. 사람이 판독할 수 있는 부분에 있는 요소들은 회사, 제품, 분류, 모듈 및 버전을 포함한다.
- [0229] · 의존. 모듈의 필수 소프트웨어 및 하드웨어 의존.
- [0230] · 상태 의존. 모듈의 행동에 영향을 주며 일관되고 결정적인 실행을 제공하도록 제어될 필요가 있는 내부 상태의 요소들.
- [0231] · 인터페이스 분석. 다양한 실행 경로를 엑서사이즈하기 위해 다양한 입력 클래스 및 필수 입력 클래스를 제공.
- [0232] · 실행 모드. 모듈이 설치될 때 모듈이 어떤 모드에서 실행될 것인지를 기술한다(즉 통계적으로 링크된 모듈에 대해서는 정적; 동적으로 링크된 모듈에 대해서는 동적; 프로세서가 모듈을 실행할 때 채택하는 운영 모

드에 대해서는 커널 또는 보호 모드; 및 우선권 레벨).

[0233] **입력 벡터 생성기**

[0234] 입력 벡터 생성기들의 기능은 다양한 실행 경로를 포함하는 구성요소를 특정 상태 시퀀스가 되게 강제하기 위해 적절한 입력을 제공한다는 점에서 소프트웨어 테스트에 있어서의 그들의 상대와 유사하다. 그러나 소프트웨어 테스트와 차이점은, PFP 입력 벡터에 대한 목적이 구현 에러를 찾는 것이 아니라 단지 다양한 실행 경로를 역서사이즈한다는 것이다.

[0235] 타겟 시스템의 성질에 따라서, 때로는 입력 벡터들을 저장하고, 평가(즉, 무결성 감사) 동안에 이용될 서명과 함께 이들을 배포해야 할 필요가 있을 것이다. 입력 벡터를 유지해야 하는지 여부에 대한 결정은 선택된 특징의 성격과 랜덤한 입력에 기인한 트래이스들이 제거될 수 있는지 여부에 의존한다.

[0236] 입력 벡터들은 서치-기반(랜덤한 서치, 힐-클라이밍, 유전자 알고리즘 등), 부분 스캔, 선형 프로그래밍, 랜덤 및 의사-랜덤 접근법을 포함해서 다양한 기술을 이용하여 생성될 수 있다.

[0237] 그러나, 유효 테스트 벡터들의 실제 식별에는, 모듈 구조에 대한 이용가능한 정보는 물론이고 타겟 모듈의 특정 기능 및 그의 입력 도메인에 주로 의존하는 휴리스틱 접근법(heuristic approach)이 이용된다. 합리적인 시간 안에 유의미한 입력 벡터를 찾기 위해서 모듈의 실행 구조(존재하는 실행 경로 및 이들을 실행시키는데 필요한 상태 시퀀스)에 대한 특정 지식이 필요한 경우들이 있을 것이다. 더욱이, 때로는 유의미한 유효 입력 벡터를 식별하여 생성하기 위해서 자동 도구에 가이드스를 제공하도록 전문가 분석의 직접 입력이 요구될 수 있다.

[0238] PFP를 위한 테스트 벡터 생성의 중요한 하나의 요소는, 목적이 장치가 설치될 때 나타나는 것으로 예측되는 다양한 경로를 실행하는 것이지 에러를 찾는 것이 아니라는 점이다. 이는 비교적 위험한 접근법인데, 왜냐하면, 유효 실행 상태가 특성화되지 않아서 이상인 것으로 표시(flag)될 수 있기 때문이다. 장점은 서치 공간을 단지 몇몇 상태로 줄일 수 있다는 것이다. 대부분의 중요한 시스템의 경우에, 실행 공간은 비교적 작으며 예측 실행 상태는 서브세트이다.

[0239] **커버리지 보고서**

[0240] 입력 벡터 생성기로부터의 정보를 이용하면 특정 입력 벡터들에 의해 트래버스(traverse)된 실행 경로를 기반으로 커버리지 보고서를 생성할 수 있다. 타겟 모듈로부터의 구조 정보를 이용하면, PFP 커버리지 매트릭을 모듈 내의 기존 경로들과 생성된 입력 벡터들을 이용하여 트래버스된 경로들의 백분율로 계산할 수 있다. 이 보고서는 단지 PFP에 대한 예측 커버리지를 나타내는 표시이다. 이 보고서는 허용가능한 PFP 서명을 실제로 산출한 실행 경로의 수를 식별하여 완성될 필요가 남아 있다.

[0241] 이 보고서는 결국 PFP를 이용하여 모니터링될 수 있는 특정 모듈에 대한 정보를 이용자들에게 제공하도록 제공된다.

[0242] **스캐폴딩**

[0243] 디스크립터 및 소프트웨어 스택으로, 중요한 모듈들 및 이들의 다양한 부분들의 실행을 분리하기 위한 스캐폴딩 프로세스가 실행된다. 이는 자동 테스트를 위한 스캐폴딩의 프로세스와 유사하다. 스캐폴딩의 목적은 시스템의 실행 동안에 전력 트래이스들을 수집하기 위하여 전체 시스템이 설치될 때 발견되는 것과 유사한 제어 환경에서 타겟 모듈들을 실행하는 것이다. 모듈들은 입력에 의존하는 다양한 실행 경로를 갖는 것으로 예측되기 때문에, 스캐폴드는 다양한 입력의 이용을 가능하게 해주어야 한다.

[0244] 물리적인 입력이 필요한 경우, 스캐폴드는 필요한 입력을 제공하기 위해서 적절한 물리적인 인터페이스를 제공해야 한다.

[0245] 이는 부분적인 매뉴얼 프로세스이고 타겟 모듈들의 특성에 의존한다. 다행히도, 스캐폴드를 위해 필요한 요소들의 대부분은 기능이 자동 테스트(예를 들어, 유닛, 통합 및 시스템 테스트)를 위한 전통적인 스캐폴딩과 오버랩되므로, 아주 작은 추가 작업만이 부가된다.

[0246] 비-소프트웨어 구현의 경우, 최종 구현이 다를지라도 스캐폴딩은 유사한 필요조건을 갖는다는 점을 주목하는 것은 중요하다. 이들 경우에, 모듈들은 독립적으로 엑서사이즈될 수 있는 섹션들에 의해 제한될 것이다. 고집적 시스템의 경우, 이는 도전을 의미할 수 있다.

[0247] **신호 처리 및 특징 추출**

- [0248] 전력/순간 전류 센서를 이용하여 캡처한 다양한 모듈의 실행 및 이들의 실행 경로에 해당하는 전력 트레이스들로, 타겟 모듈의 실행을 고유하게 식별하는 차별 특징이 추출되어야 한다. 유용한 서명을 식별하는데 필요한 정확한 기술 및 신호 분석 세트는 타겟 모듈의 특정한 특성에 의존한다.
- [0249] 우리는 이제 다수의 다양한 특징 추출의 병렬 실행과 타겟 모듈을 특성화하는데 요구되는 전체 시간을 줄이기 위한 신호 처리 기술을 설명한다.
- [0250] 주어진 문제에 대한 최적의 차별 특징을 결정하기 위한 효과적인 절차는 공지되어 있지 않다. 그러나, 평가될 수 있는 몇몇 기술이 있으며, 그로부터 최적의 차별 특징들이 선택된다. 추출되는 차별 특징들의 세트는 휴리스틱 접근법과 경험의 결합을 이용하여 결정된다. 이들 특징 중에서 다음이 포함되어 있다: 시간 도메인 상관관계, 유클리디언 거리, 사이클로스테이션너리 분석(cyclostationary analysis), 주파수 분석 등. 최적의 차별 특징을 선택하기 위한 프로세스는 이 세트에 있는 모든 다양한 특징들을 병렬로 계산하고 이들을 인트라-클래스 변화를 기반으로 순위를 정하는 것(rank)을 포함한다. 마할라노비스 거리(Mahalanobis distance)는 그러한 특징들의 평가를 위한 샘플 메트릭이다.
- [0251] 이하 설명되는 특징 추출 프로세스와 검출기 설계의 프로세스는, 특징 추출 결과의 통계 속성이 최적의 검출 임계치를 결정하기 위해 필요한 분석을 결정하기 때문에, 밀접하게 연관되어 있다.
- [0252] **통계 분석 및 검출기 설계**
- [0253] 통계 분석은 타겟 모듈의 독립적인 실행 사례 동안 캡처된 전력 트레이스들로부터 얻은 다양한 특징에 대해서 실행된다. 통계 분석의 목적은 최적의 차별 품질을 갖는 특징들을 선택하고 관측된 특징들의 세트가 타겟 모듈(검출기)에 의해서 생성되는 것으로 생각되는 임계 레벨 또는 영역들을 판정하는 것이다.
- [0254] 이상 검출 접근법인 PFP에서, 거짓 경보의 확률(PFA)은 시스템의 성능을 결정하는 중요한 연산 메트릭이다. PFA는 타겟 모듈의 정상 실행 사례가 허용가능 영역을 벗어나면 이상으로 분류되는 확률로 정의된다. PFP 검출기는 타겟 모듈을 올바르게 식별하는 확률을 최대화하면서 PFA를 최소화하도록 설계되어야 한다. 이는 고전적인 가설 테스트 문제이며, 네이만-피어슨 기준은 임계치를 결정하는데 적용될 수 있다. 그러나, 적용될 수 있는 다른 접근법도 몇몇 있다.
- [0255] 충분한 샘플이 주어지면, 임의 PFA가 PFP에서 성취될 수 있다. 그러나, 실제의 시스템에서는 이는 실현 가능하지 않으며 유한한 유용한 PFA 레벨이 결정되어야만 한다. 용인될 수 있는 PFA는 특정 모듈 및 운영되는 것으로 예상되는 애플리케이션의 성격에 의존한다.
- [0256] 이상적으로는, 동일한 실행으로 인한 다양한 사례로부터의 서명들은 플랫폼 특성화 동안 계산된 최소 민감도를 위한 거리 내에 있어야 한다. 이와 같이 원하는 특징이 성취되지 않는 경우에, PFP가 정확한 평가를 내리게 하는 몇몇 방법이 있다. 하나의 간단한 접근법은 어떤 노이즈를 제거하기 위해 몇몇 트레이스를 평균하는 것이다.
- [0257] **서명 패키징 및 암호화**
- [0258] 타겟 모듈이 특성화되었으면, 최종 서명, 특징 추출 기술, 및 임계치들은 장치들과 함께 설치되도록 패키징된다. 패키징 및 전달 메커니즘은 장치와 애플리케이션의 특성에 의존한다. 선택된 특징을 이용하여 추출된 완전한 서명은 저장되고 모니터에 전달되어야 한다. 예를 들어, 간단한 시간 도메인 상관관계의 경우에, 완전한 벡터는 저장되어야 한다.
- [0259] 보관중인 또는 전송 동안 서명을 보호하기 위해서는, 모니터가 찾고 있는 서명의 정확한 기준이 잠재적 공격자에게 넘어가지 않도록 이들을 암호화해야 한다. 이 암호화는 개인 또는 공개 키 암호화를 위한 다양한 메커니즘을 이용하여 실행될 수 있다. 그러나, 잠재적 공격자가 서명을 취득하더라도 악의적인 행동으로 서명들을 완벽하게 매칭시키는 것은 아직은 매우 어려울 것이다 라는 점을 주목하는 것은 중요하다.
- [0260] **보안 서명 갱신**
- [0261] PFP를 이용하여 모니터되고 있는 설치된 시스템이 갱신될 때, 효과적인 무결성 평가를 유지하기 위해서는 신뢰할 수 있고 안전한 방식으로 PFP 서명을 갱신하는 것도 필요하다. 이는 평가의 신뢰성이 적절한 서명 관리에 의존하기 때문에 매우 중요한 단계이다. 이와 같이 안전한 보안 갱신을 위해서는, 서명의 무결성과 인증을 확인하는 것이 필요하다. 이 장에서 우리는 안전한 PFP 서명 갱신을 제공하는데 필요한 메커니즘을 설명한다.
- [0262] 광범위하게 설치된 PFP 모니터들을 위해서는, 서명은 다른 소프트웨어 갱신과 함께 배포되어야만 한다. 집중된

PPF 모니터들을 위해서는, 갱신은 소프트웨어 갱신과는 별개로 전달될 수 있다. PPF 서명 갱신에 있어서 주요 도전은 인증(즉, 전송자가 인증된 엔티티이고 그리고 서명 그 자체가 올바르고 어떤 방식으로든 탬퍼링되지 않았거나 변경되지 않았음을 확인하는 것)이다. 이 도전은, 서명이 물리적인 미디어 또는 신뢰 네트워크를 이용하여 배포될 수 있고 파라미터가 민감한 서명의 전송을 예측하여 만들어질 수 있는 집중된 PPF 모니터들의 경우에는 어려운 일은 아니다.

[0263] 물리적인 수단 또는 대안적인 신뢰 네트워크를 이용하여 서명 교체가 이루어질 수 없는 광범위하게 설치된 PPF 모니터들의 경우에, 서명 갱신은 실제의 소프트웨어 갱신과 함께 실행되어야 한다. 이 경우에, PPF 시스템에 대한 충분한 지식이 있는 공격자에 의해서 활용될 수 있는 몇몇 취약점이 있다. 예를 들어, 적절하게 인증되지 않았다면, 갱신 프로세스는 중간자 공격(man-in-the-middle-attack)에 의해서 방해받을 수 있다.

[0264] 안전한 서명 갱신 프로세스 및 운영

[0265] 공중과 프로그래밍에 공통으로 이용되는 안전한 콘텐츠 배포 및 소프트웨어 갱신의 배포를 위한 공지된 접근법은 PPF 서명 갱신에 적용될 수 있다. 안전한 서명 갱신은 두 개의 다른 관점: 인증 서명 오리지네이터 및 PPF 모니터의 관점에서 볼 수 있다. 서명 생성 측면에서는 서명과 함께 효과적인 인증 정보를 제공하고 로테이팅 키 방식으로 서명 부분들을 암호화해야 할 필요가 있다.

[0266] 안전한 서명 갱신을 하기 위해 적용될 수 있는 다른 기술은 다음을 포함한다:

[0267] · 샘플 버퍼와 서명 요소들을 스캔블

[0268] · 공개 또는 대칭-키 암호화

[0269] · 각각의 서명 갱신 후에 그의 인덱스를 갱신하는 공지된 시퀀스(PN 시퀀스)에 따라서 서명과 트레이스들을 디스스캔블링하기 위해 이용되는 암호화 키를 변경.

[0270] 사이드-채널 공격에 대한 보호

[0271] PPF는 악의적인 사이드-채널 공격을 위해 부당하게 활용되는 무결성 평가를 위해 동일한 원리를 이용한다. 그러므로, 잠재적 적이 사이드 채널 공격을 실행하기 위해서 PPF 기반을 활용하는 것을 방지하기 위해서는, 센서가 제공한 트레이스들에 대한 접근을 제한시켜서 이들 트레이스를 보호하는 것이 필요하다. 이는 전력 트레이스들이 무선 연결을 이용하여 전송될 때 특히 중요하다. 이 장은 사이드-채널 공격에 남용될 수 있는 전력 트레이스들에 대한 인증되지 않은 접근을 방지하기 위한 메커니즘을 설명한다.

[0272] 운영

[0273] 트레이스 접근에 대한 방지는 PPF 센서와 PPF 모니터 간에 공유되는 키를 이용하여 트레이스들을 암호화하거나 스캔블링함으로써 성취될 수 있다. 이러한 관점에서, PPF에 대한 기본적인 두 가지의 운영 모드가 있다: 내장 모니터(센서 및 디지털타이저) 및 외부 모니터.

[0274] 내장 운영의 경우에, 트레이스들은 강력한 개인 키로 암호화 또는 스캔블된다(대칭적 키 암호화). 이러한 암호화 단계를 실행하는 것은 전력 트레이스들이 외부 처리를 위해서 무선으로 전송될 때 특히 중요하다. 암호화 프로세스는 도 38에 도시되어 있다. 프로세서(3810)의 아날로그 출력은 센서(3820)에 의해서 모니터되고 아날로그-디지털 변환기(3830)에 의해 변환된 다음 암호(3850)에 공급된다. 암호(3850)는 비트 샘플들의 블록 암호화 또는 이들에 대한 스캔블링(효과적으로는 키가 순열인 치환 암호)을 포함해서 몇몇 방식으로 사이드-채널 공격자에게 적절한 정보를 숨길 수 있다.

[0275] 외부 모니터의 경우, 트레이스들에 대한 접근을 제공하는 물리적인 연결은 패스워드를 요구하는 디지털 스위치에 의해서 가능해진다. 이 경우에, 외부 모니터에 대한 접촉점은 플랫폼상의 전력 관리 칩에 의해 제공된다. 전력 관리 칩은 전압 조정기만큼 간단할 수 있지만, 현대의 스마트폰에 이용되는 대부분의 상업적인 프로세서의 경우에는, 전력 관리 칩은 훨씬 더 복잡하다. 적절한 모니터가 연결될 때, PPF-가능 전력 매니저는 일단 연결되면 외부 모니터로부터 패스워드를 관독하고 나서, 외부 모니터가 순간 전류 드레인 또는 전력 소비를 캡처할 수 있도록 해주는 외부 센서에 전원 전류를 통과시킨다. 도 39는 이러한 프로세스를 그래픽으로 표현하였다.

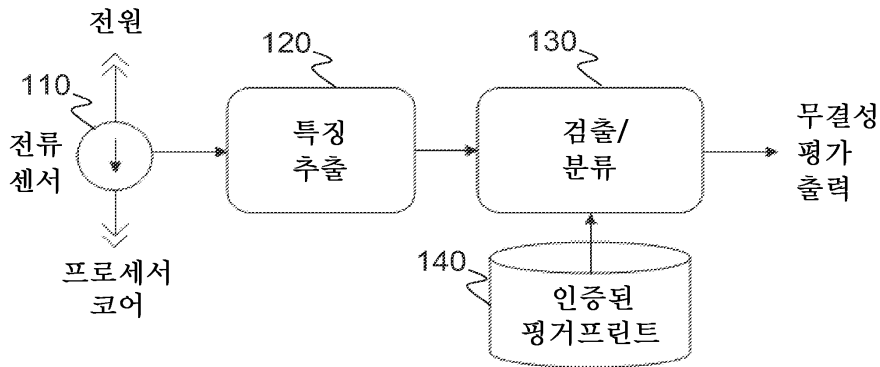
[0276] 여기에서 설명한 솔루션들은 공격자가 우리의 타겟 시스템에 대한 사이드 채널 공격을 실행하는 것을 방지하기 위해 의도된 것은 아니라는 점에 주목하는 것이 중요하다. 대신에 이들은 사이드-채널 공격을 위해 PPF 모니터링 설비를 남용하는 것을 방지하기 위한 것으로 의도된 것이다. 이들 조치가 적소에 있다면, 잠재적 공격자는 PPF 모니터링이 있는 보드에 대해 PPF 모니터링이 없는 보드에 대한 것과 동일한 하드웨어 수정을 해야만 할 것

이다.

[0277] 본 발명이 양호한 실시예를 들어서 설명되었을지라도, 첨부된 청구항들의 정신 및 범위 내에서 이 기술 분야에 숙련된 자는 본 발명을 수정하여 실행할 수 있음을 인식할 것이다.

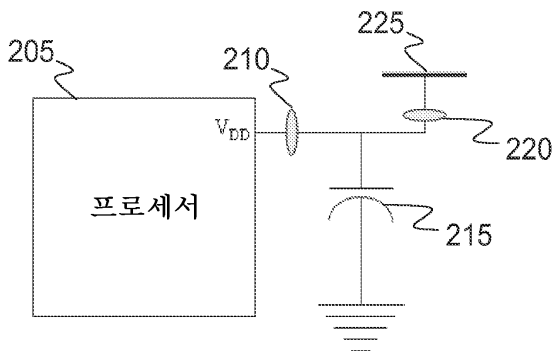
도면

도면1

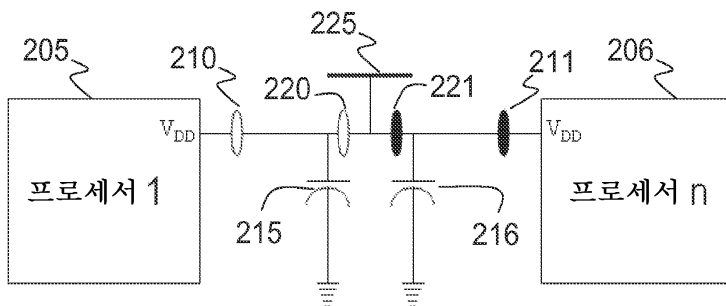


종래 기술

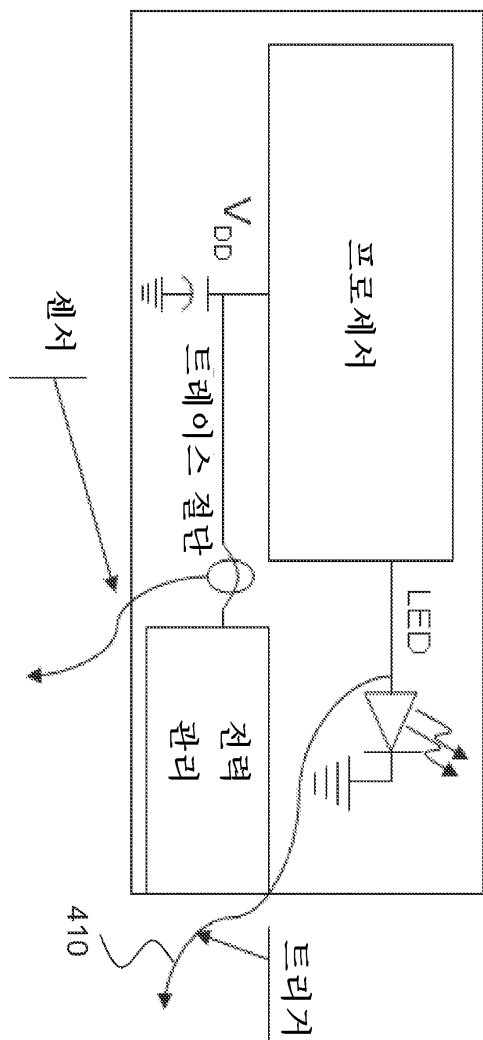
도면2



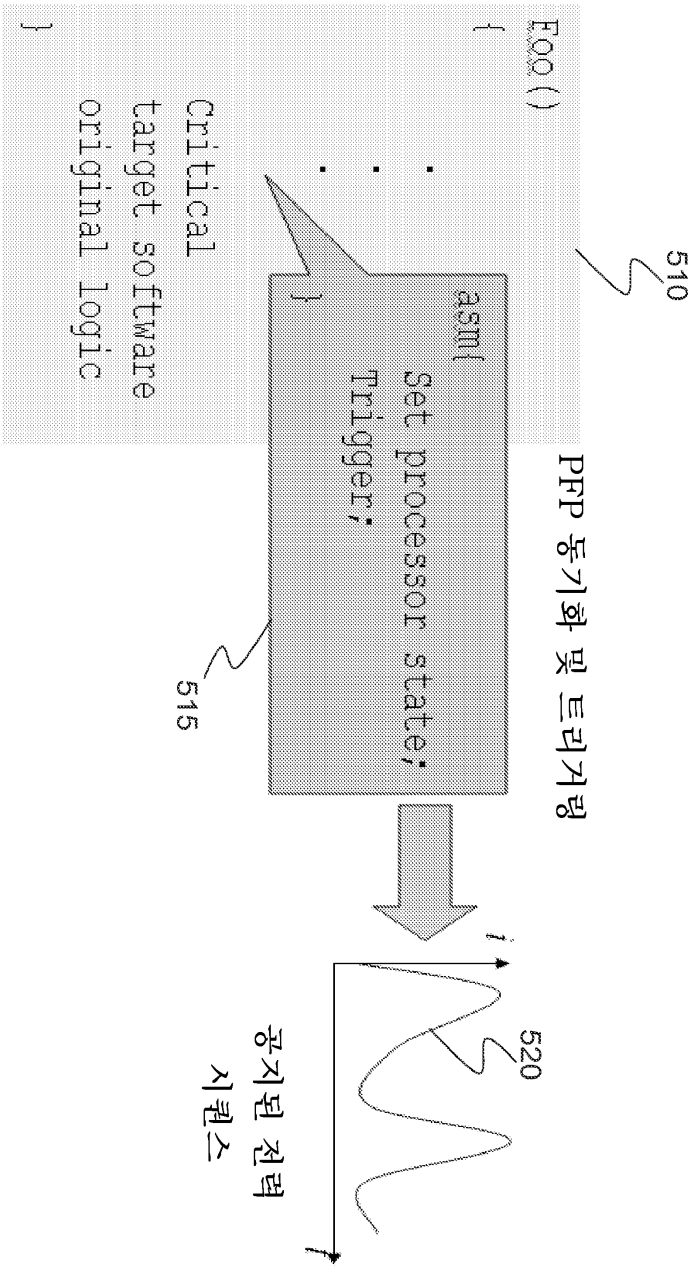
도면3



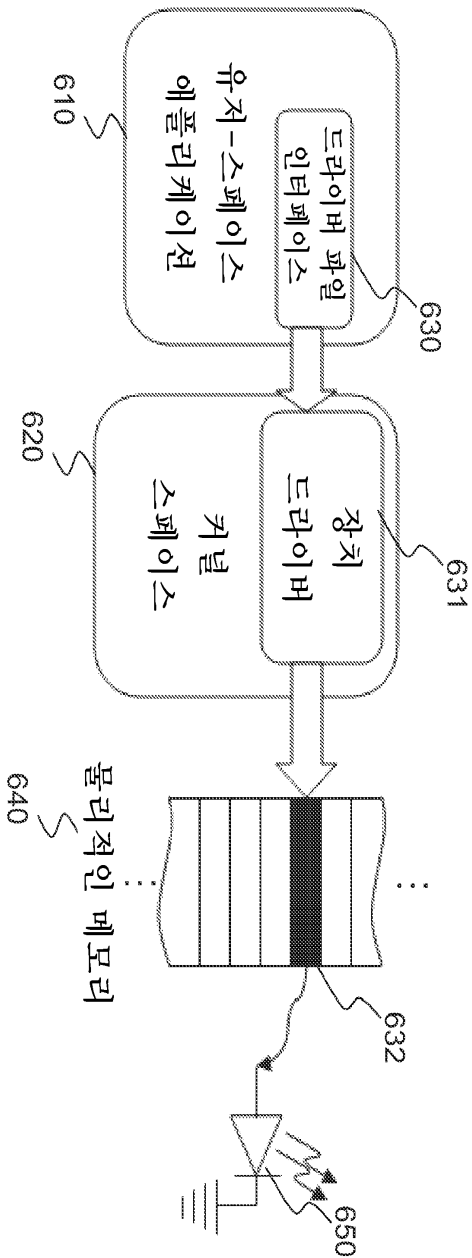
도면4



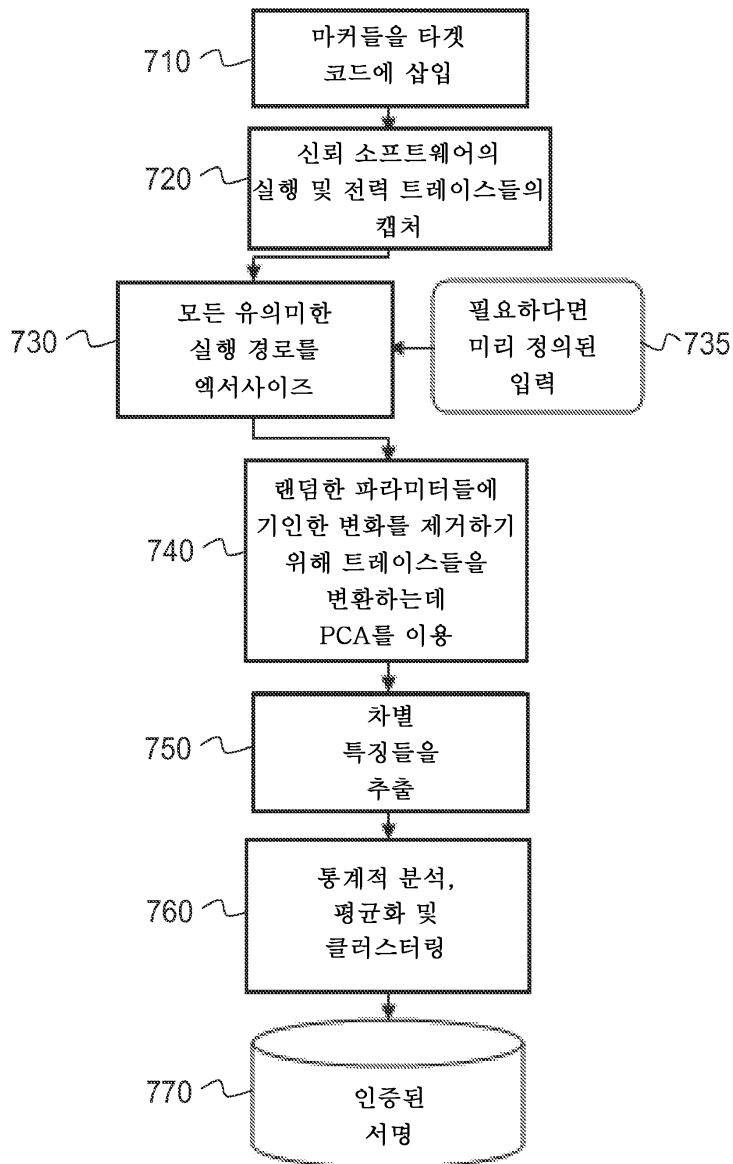
도면5



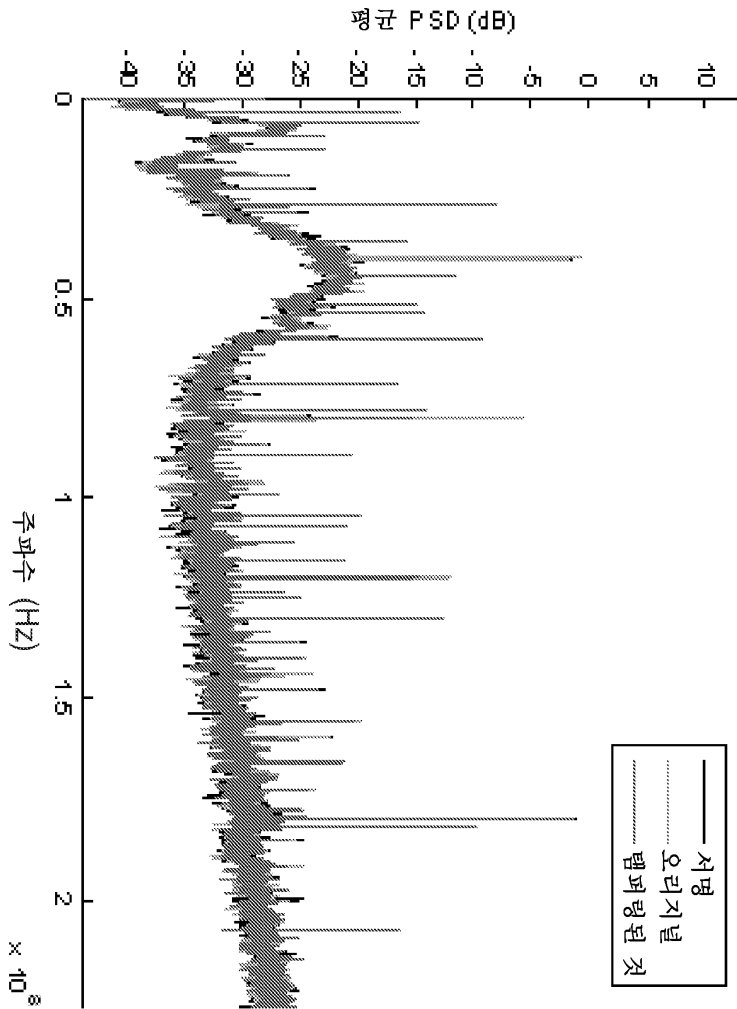
도면6



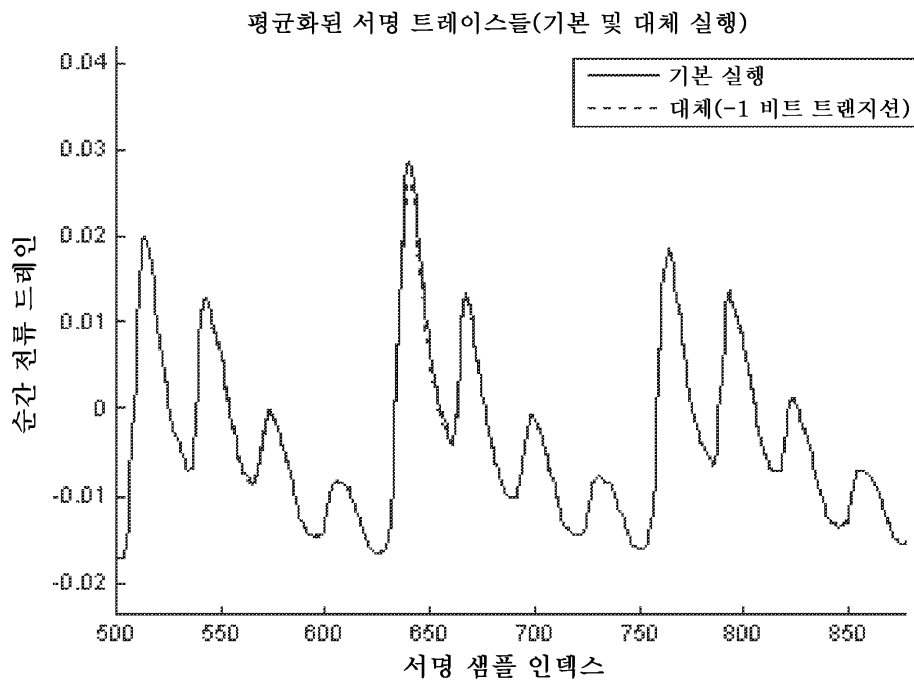
도면7



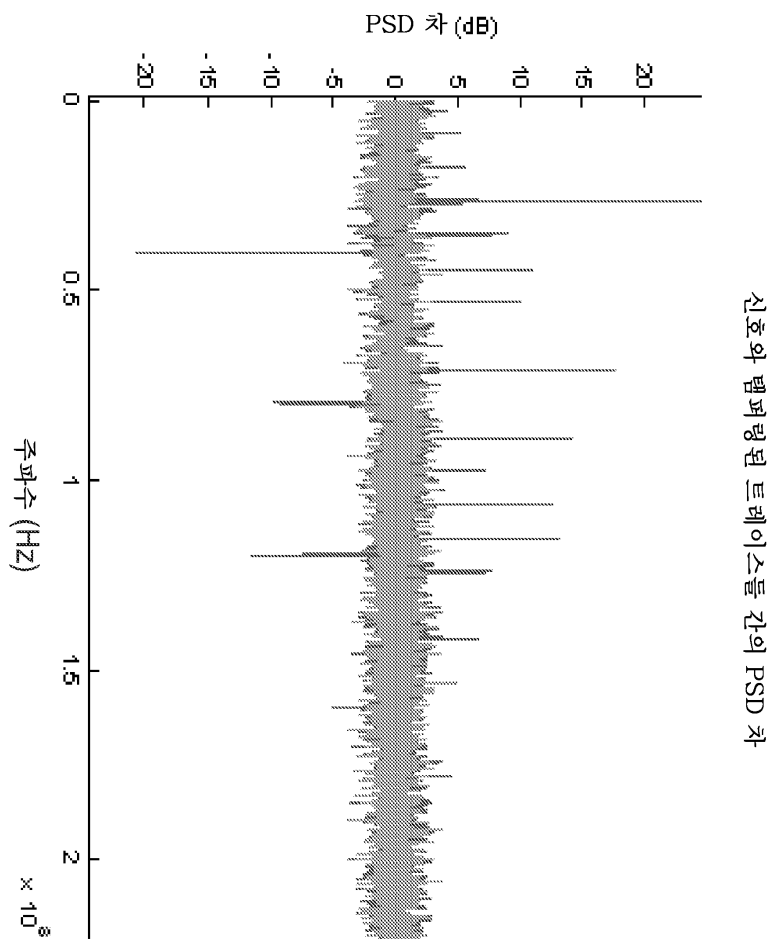
도면8



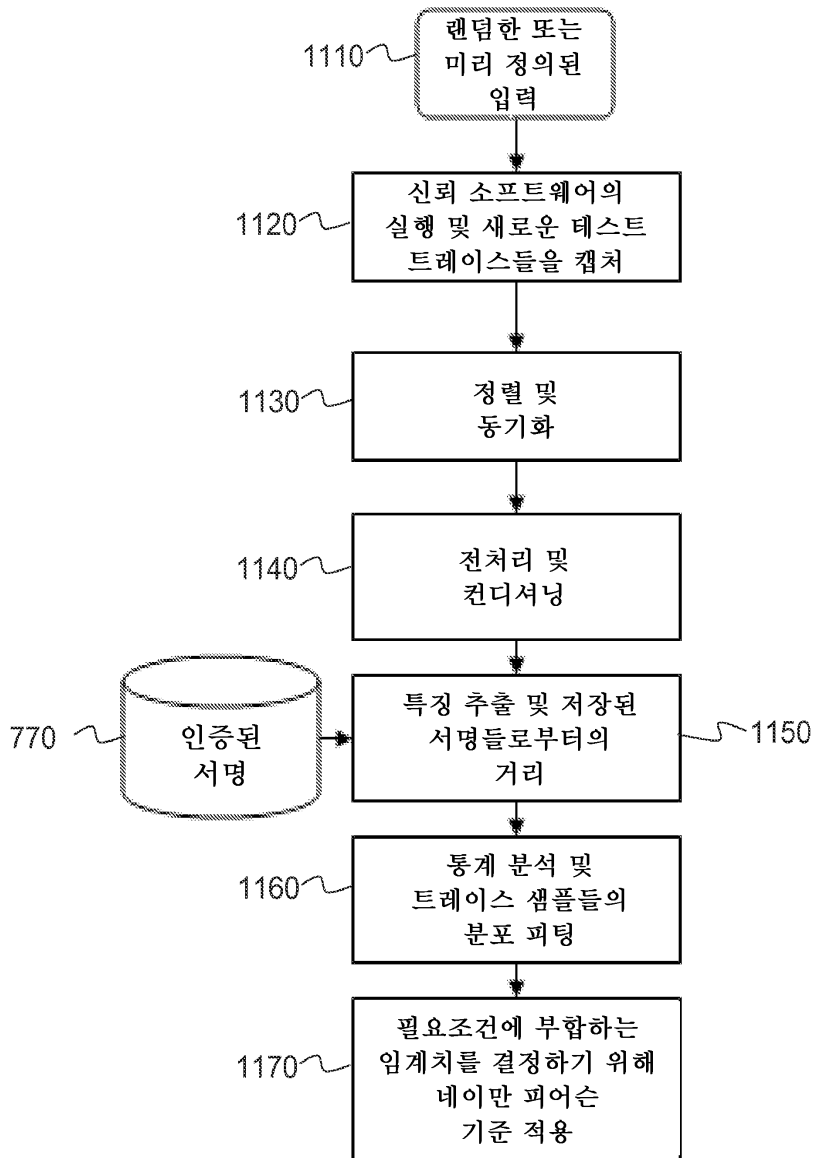
도면9



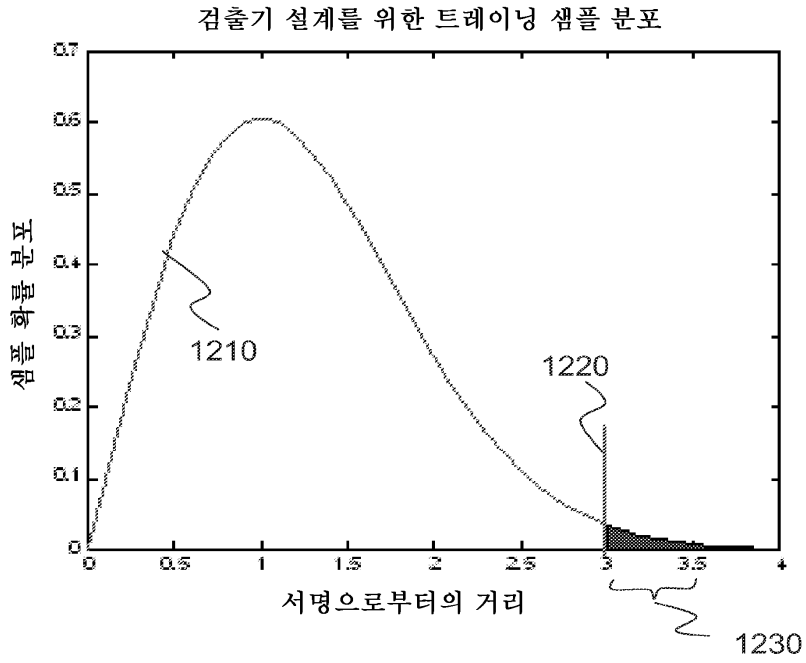
도면10



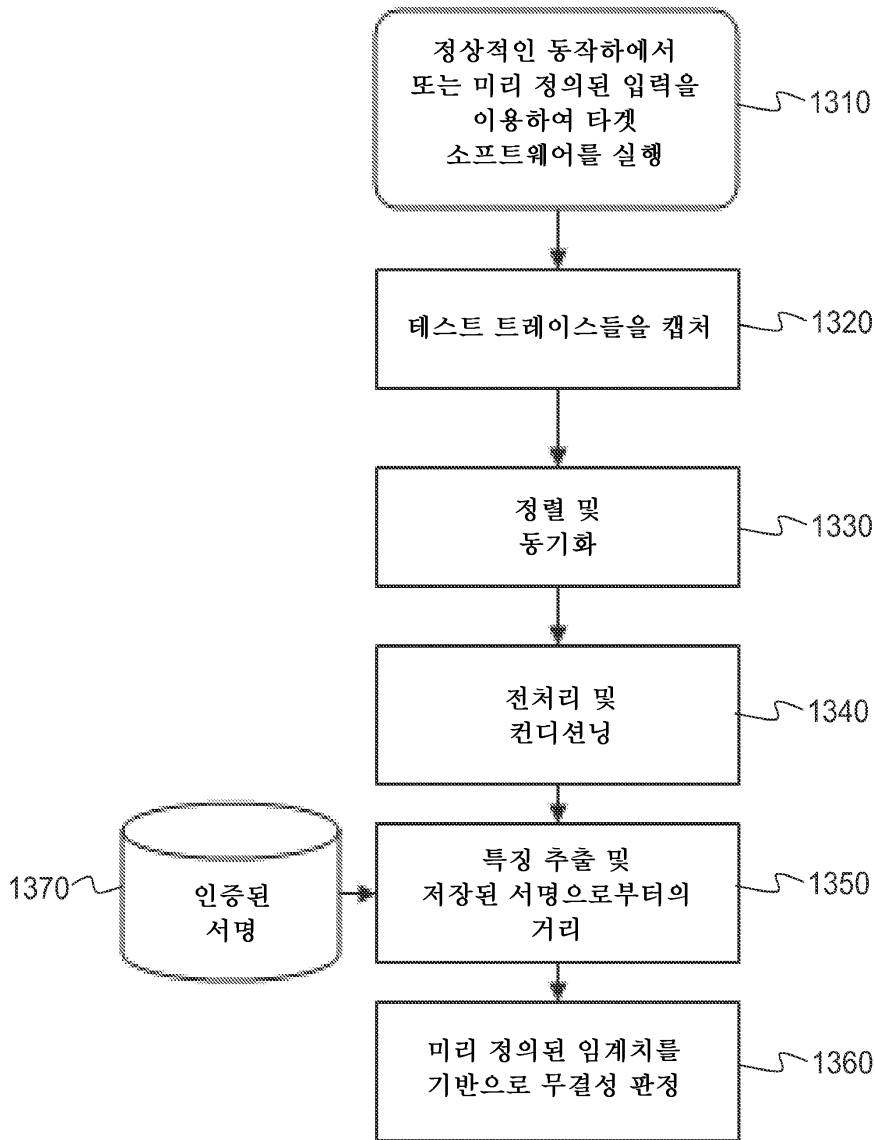
도면11



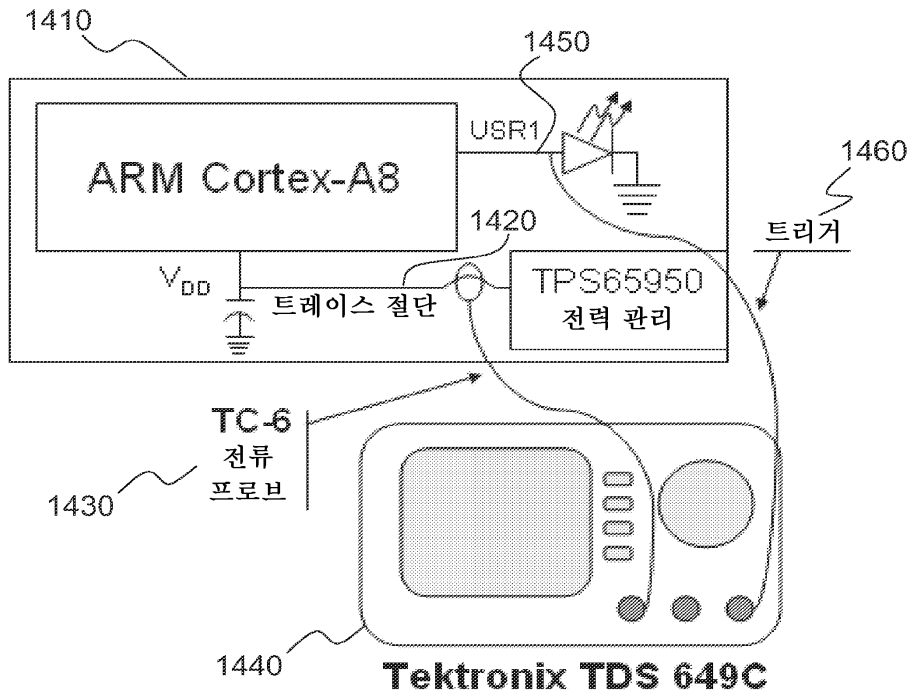
도면12



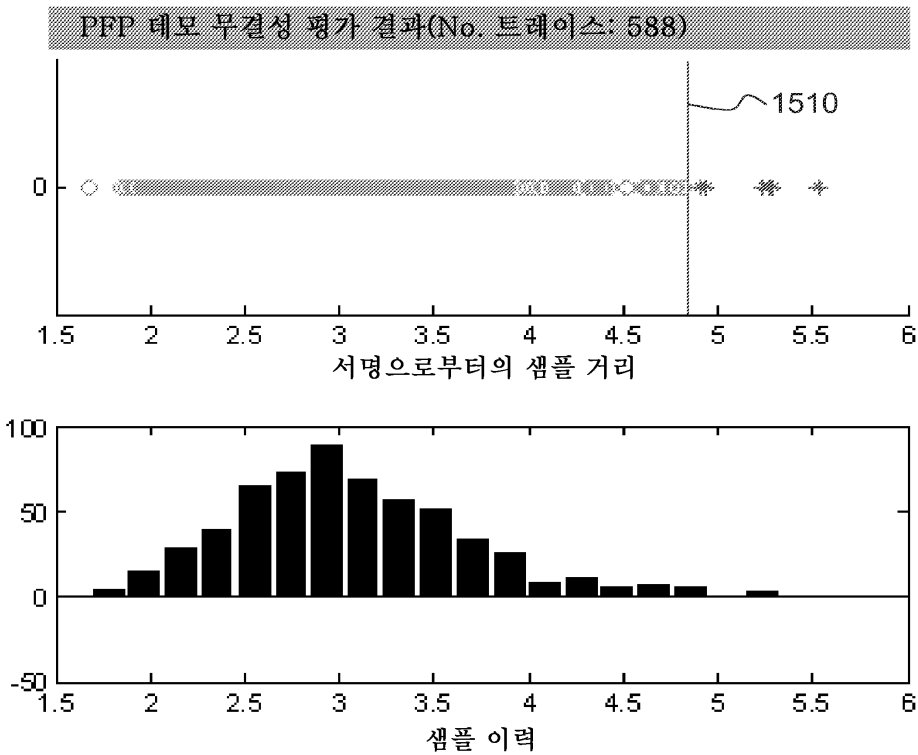
도면13



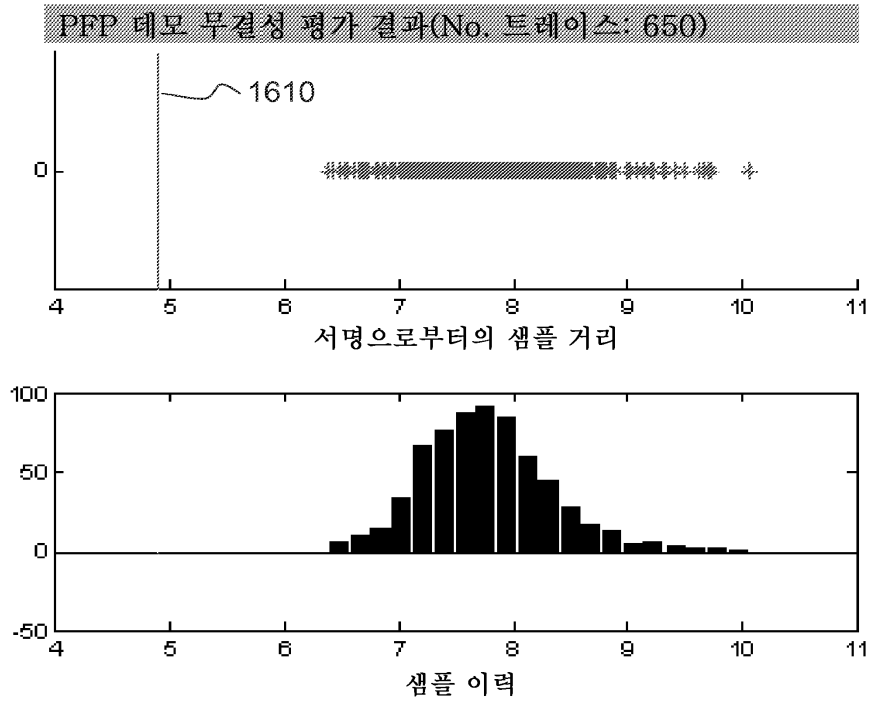
도면14



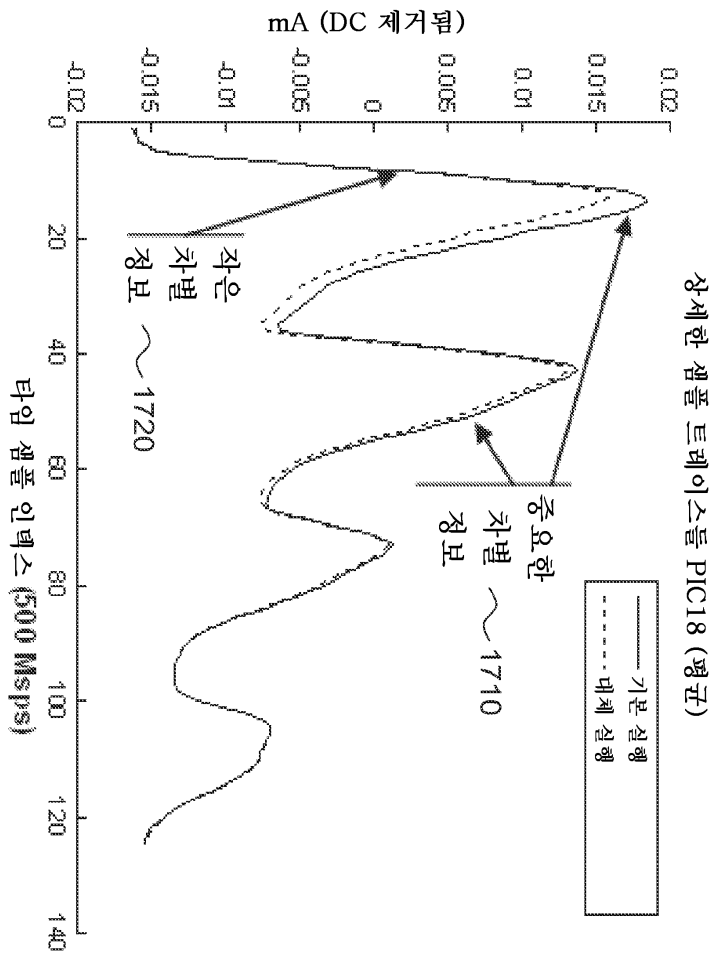
도면15



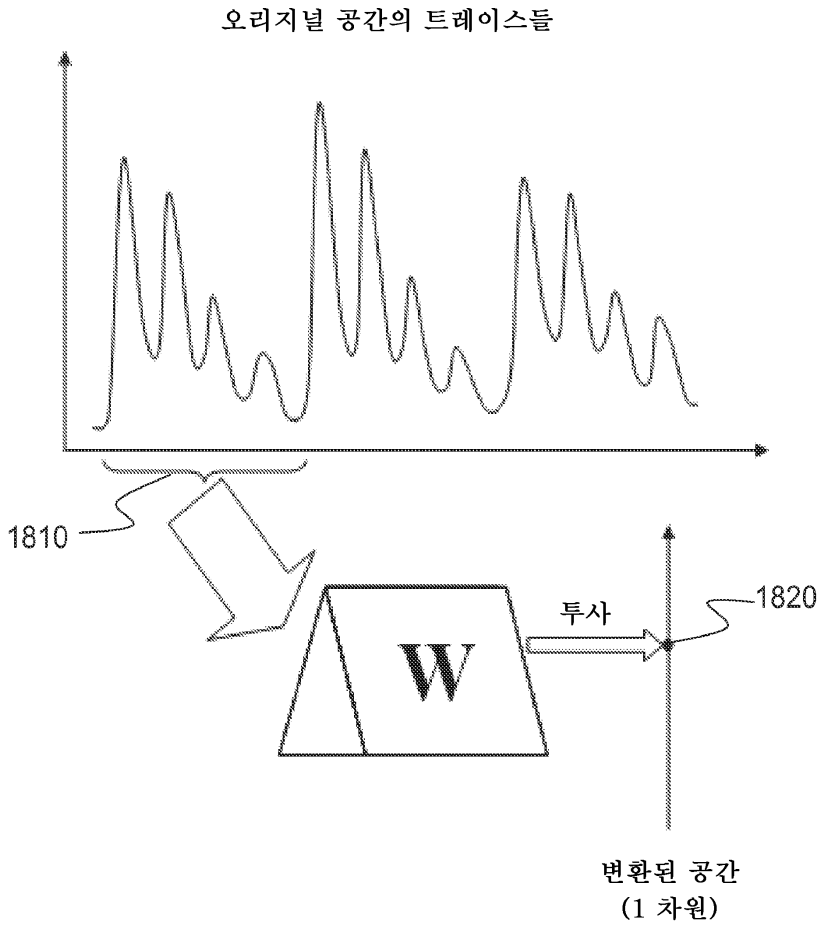
도면16



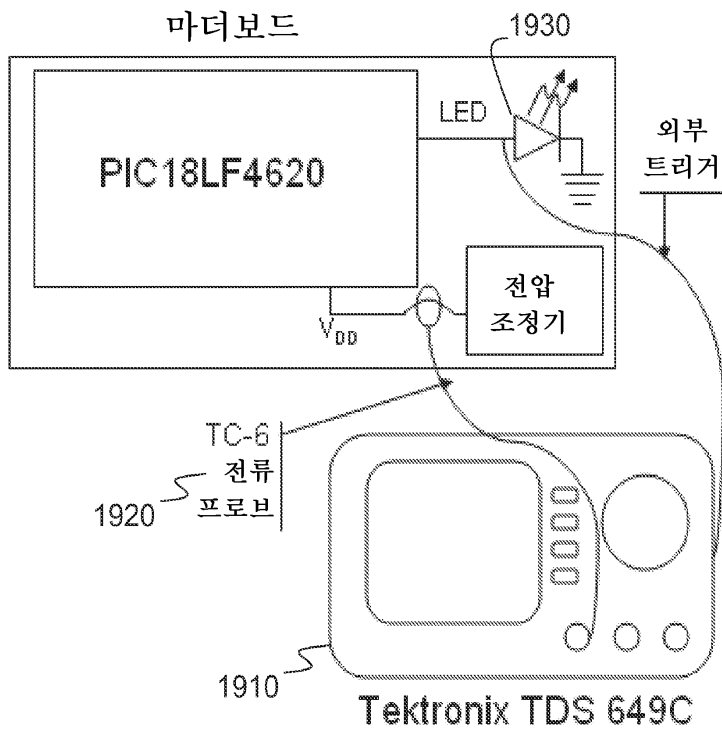
도면17



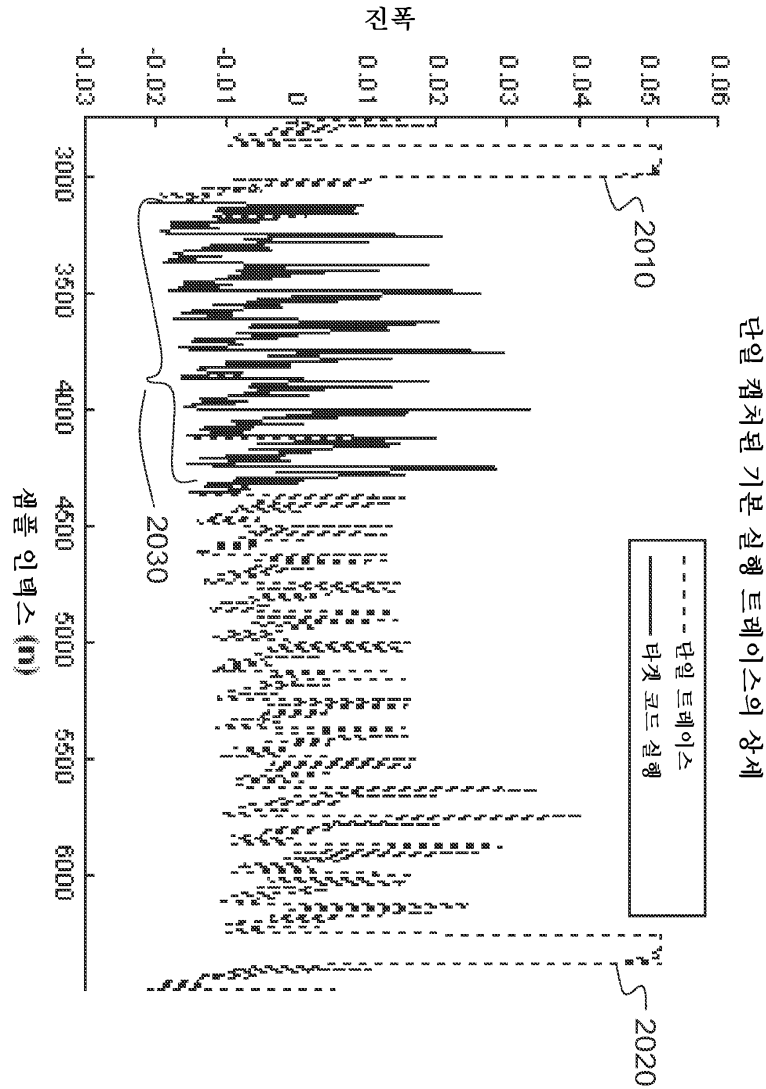
도면18



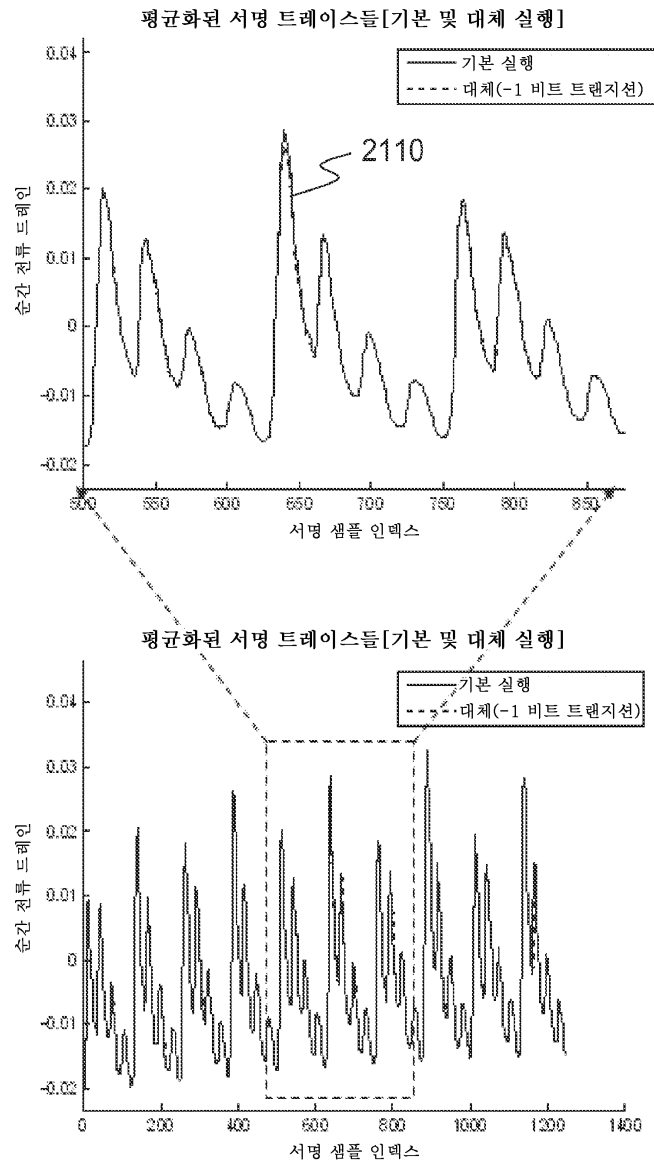
도면19



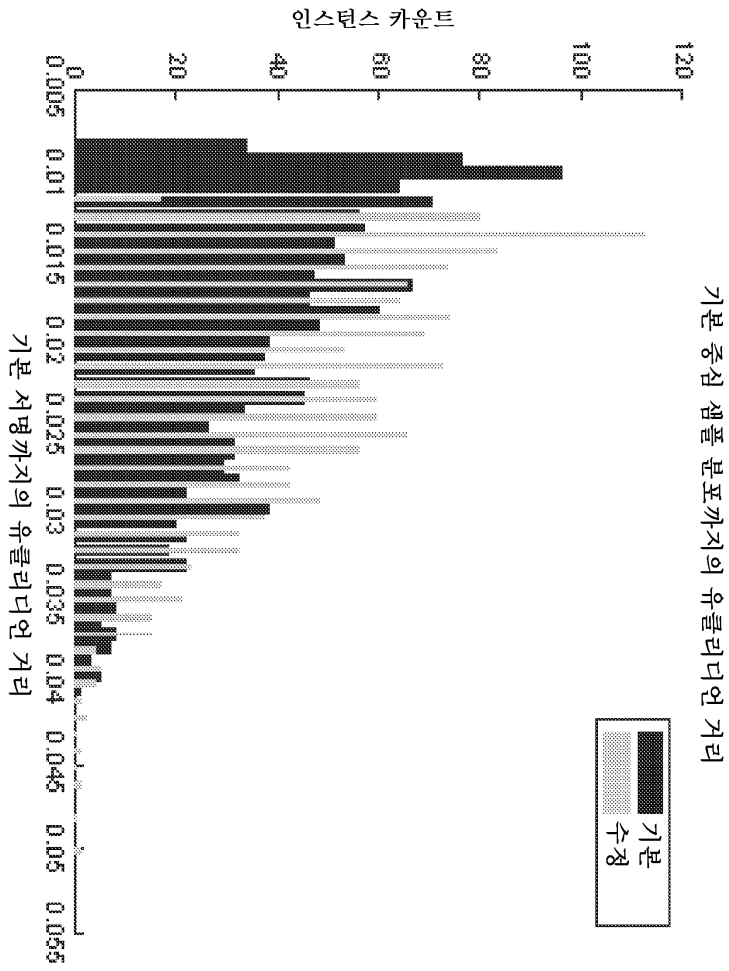
도면20



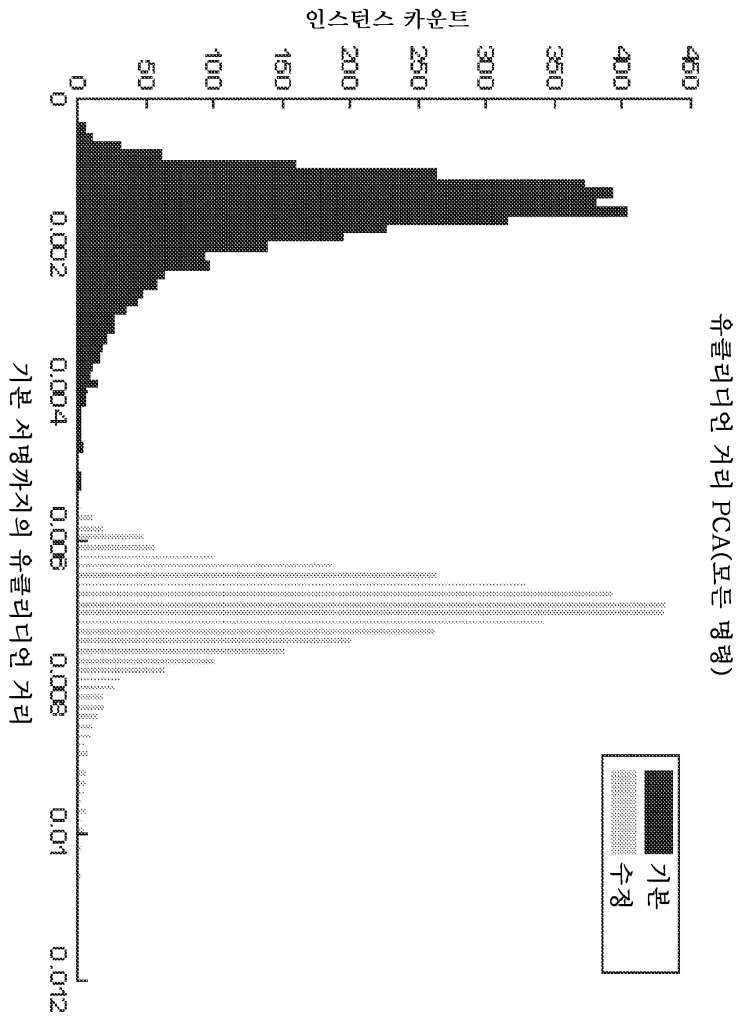
도면21



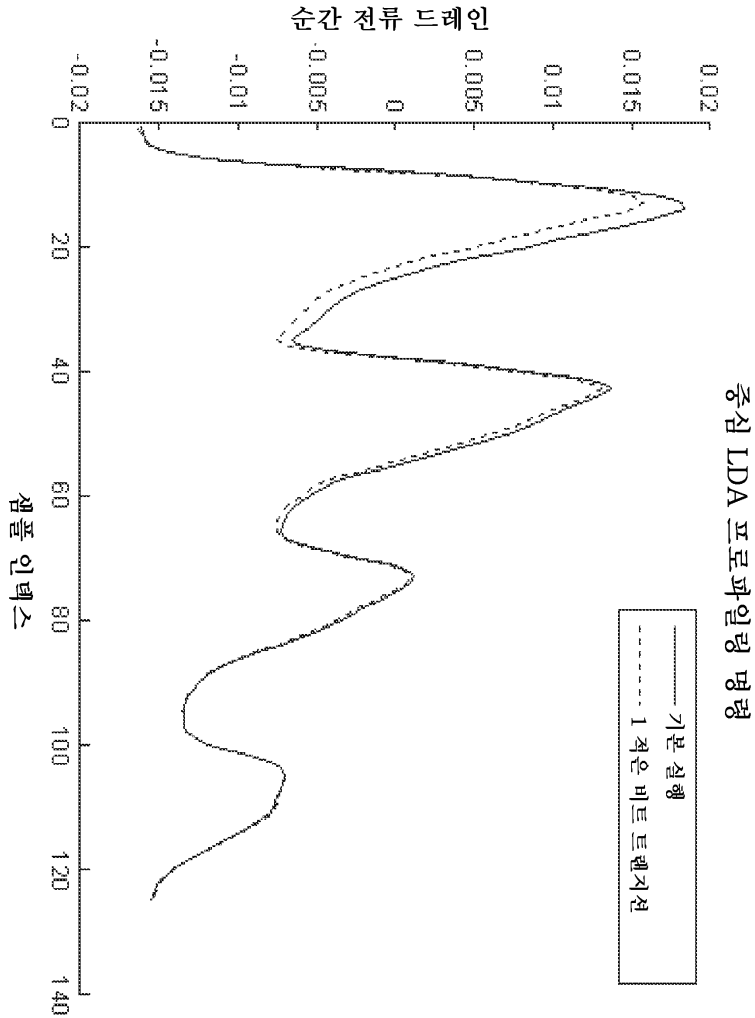
도면22



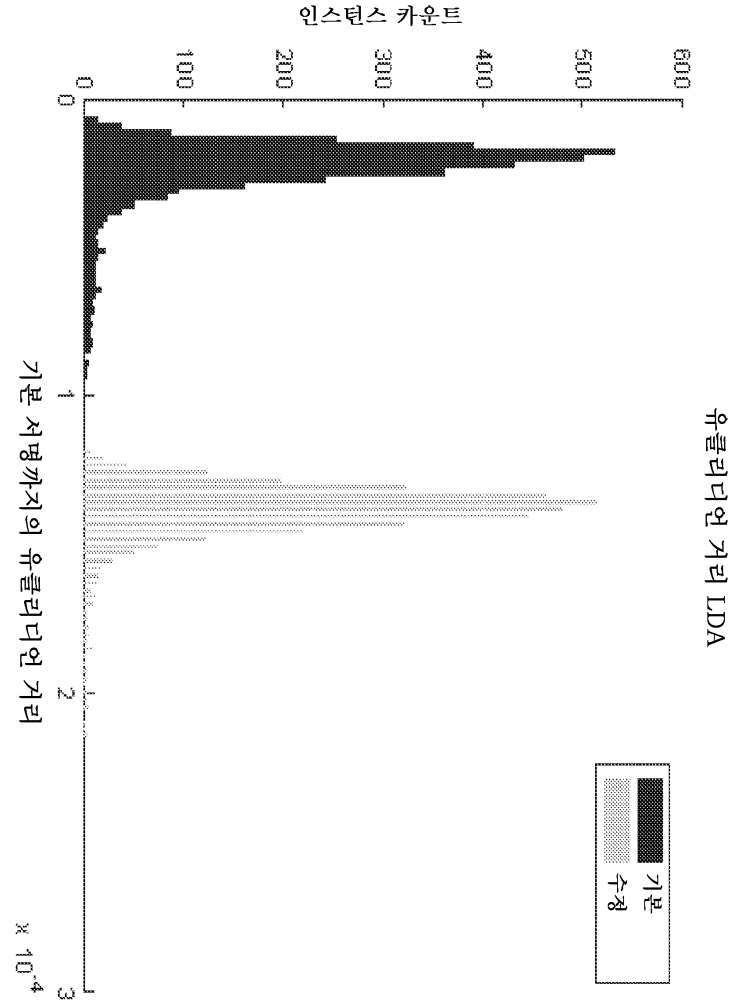
도면23



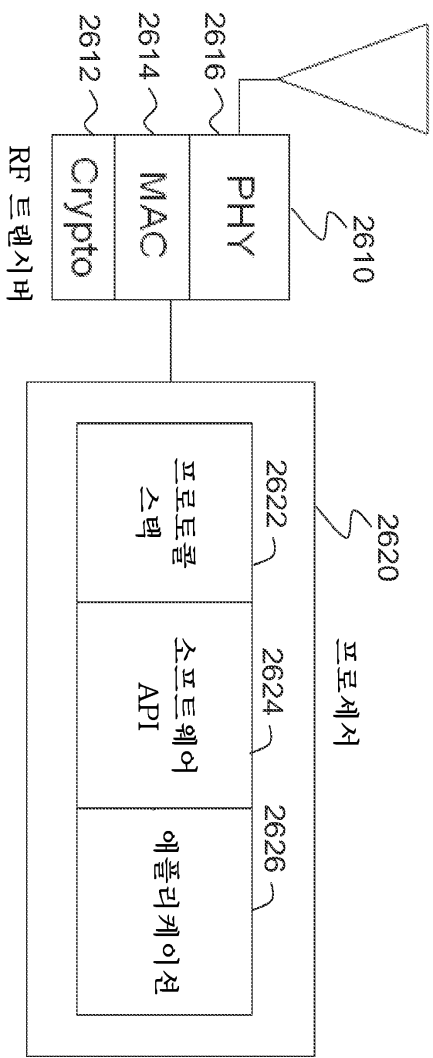
도면24



도면25

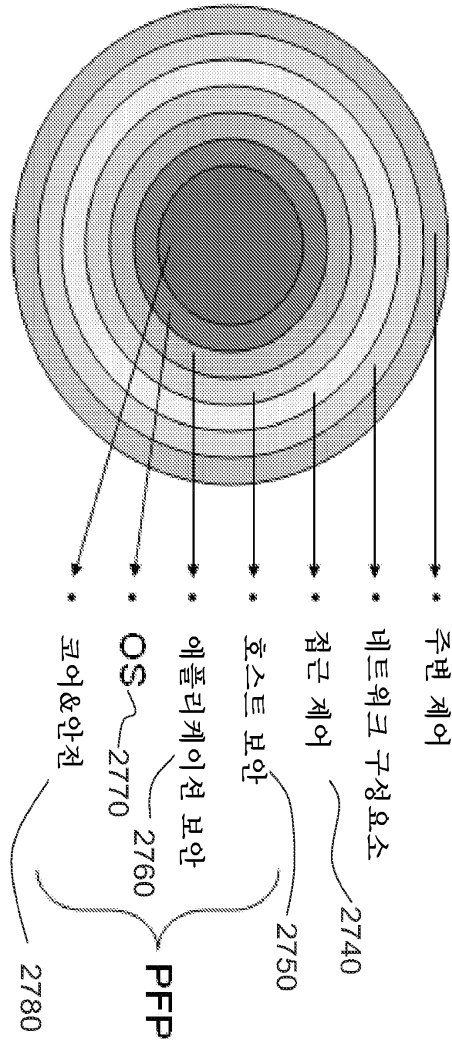


도면26

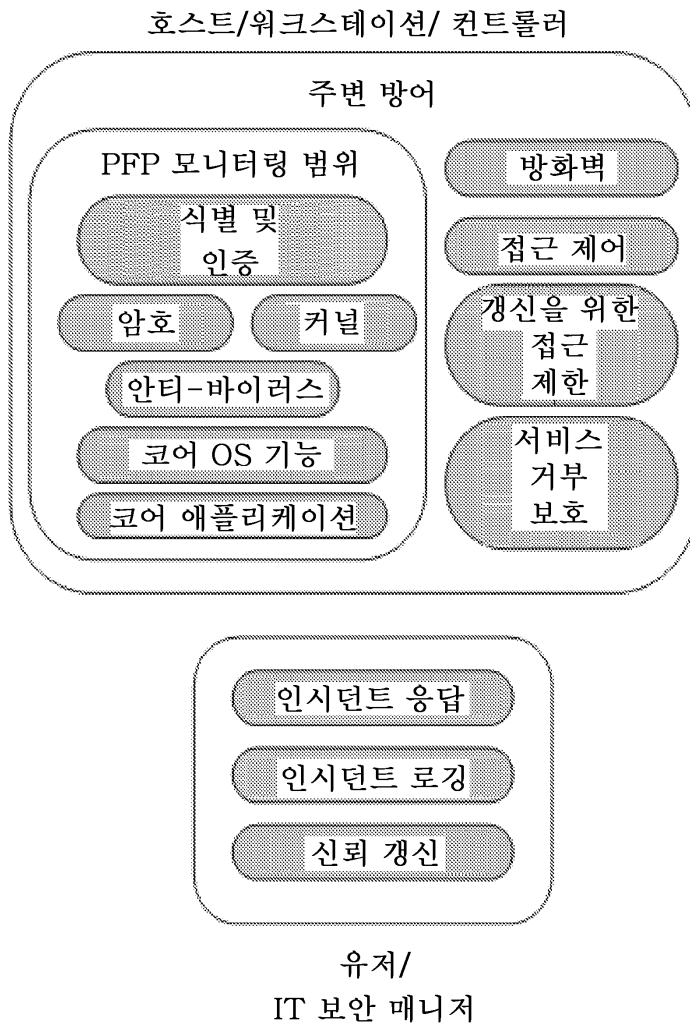


도면27

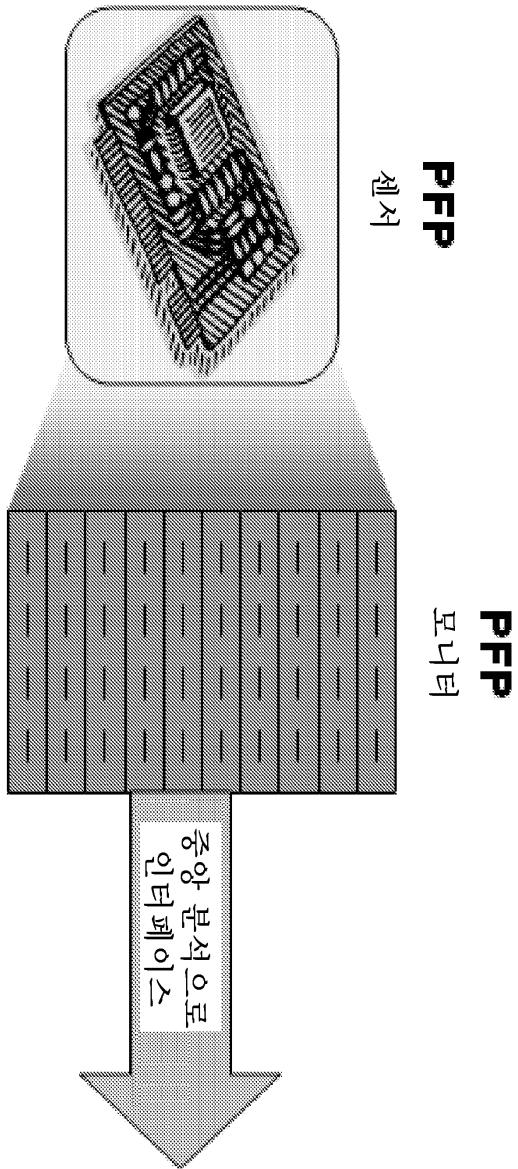
2700



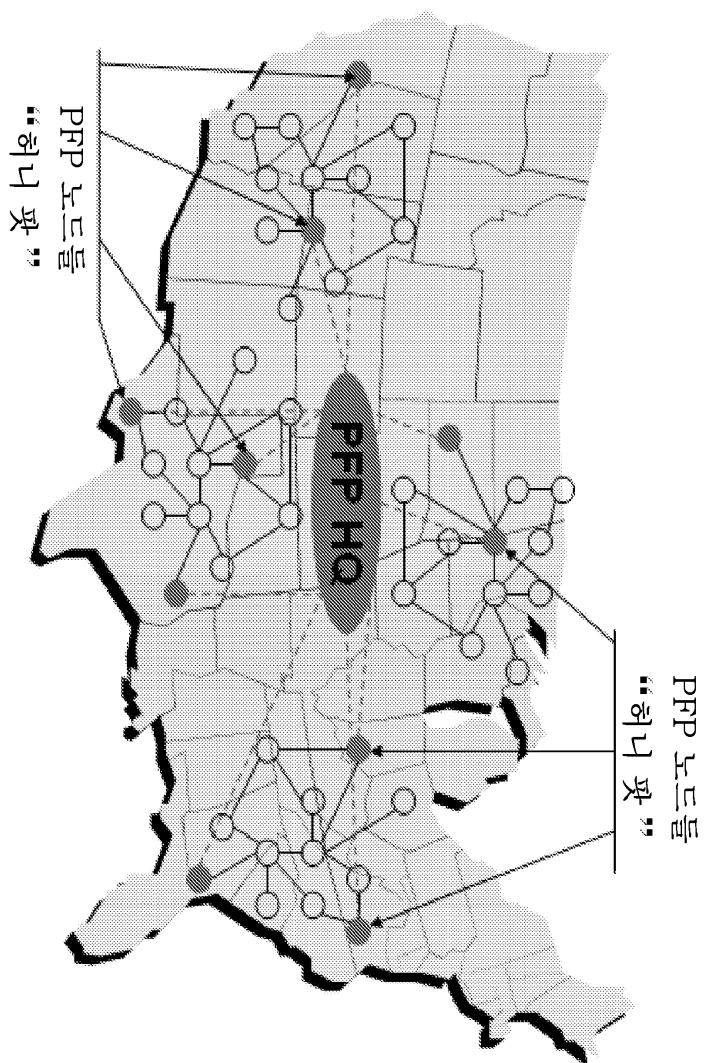
도면28



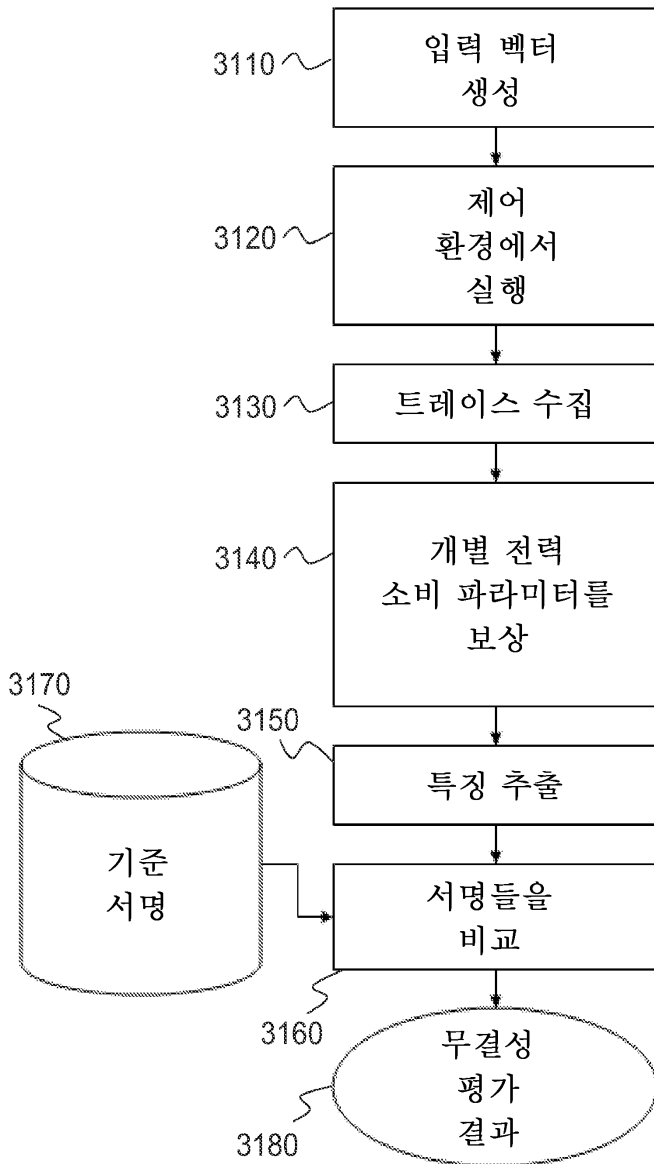
도면29



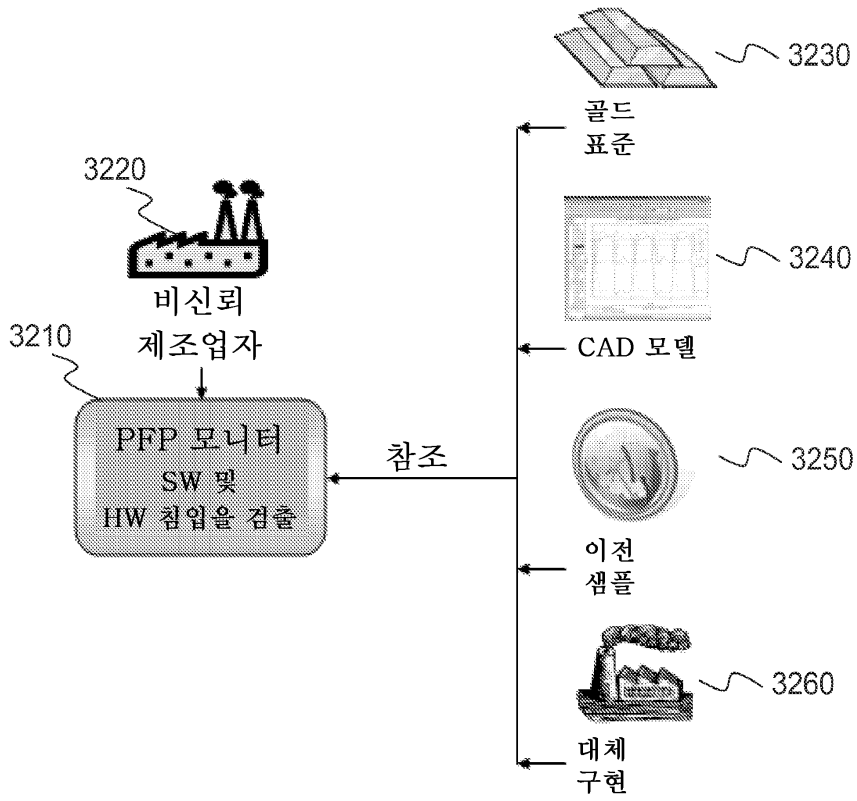
도면30



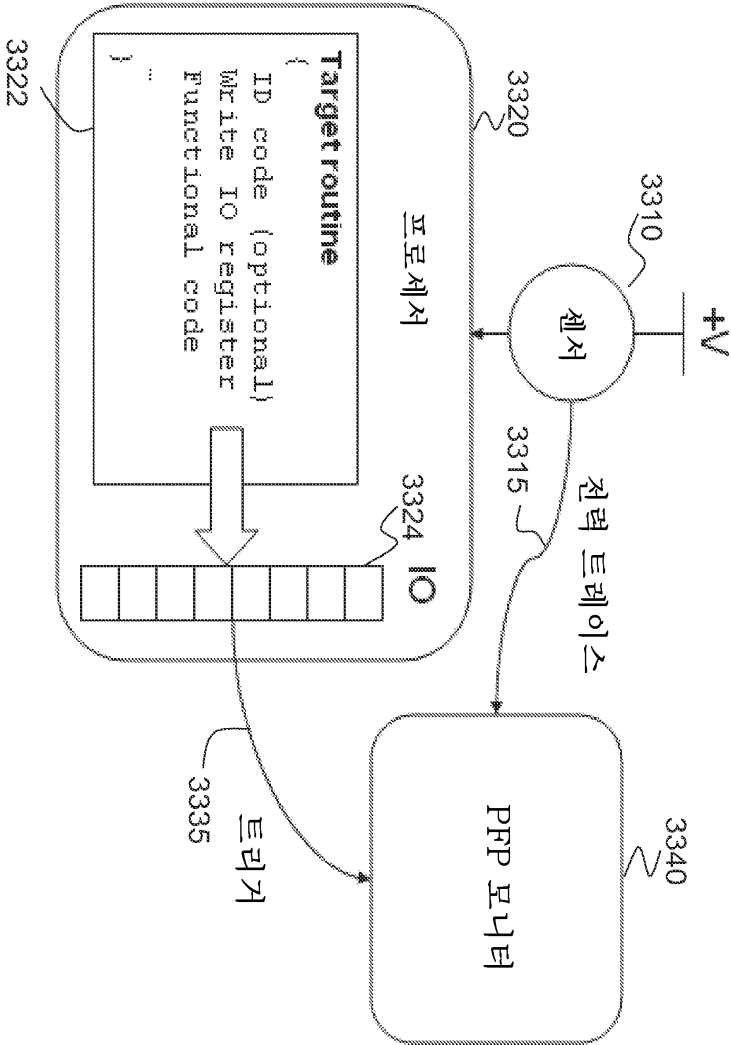
도면31



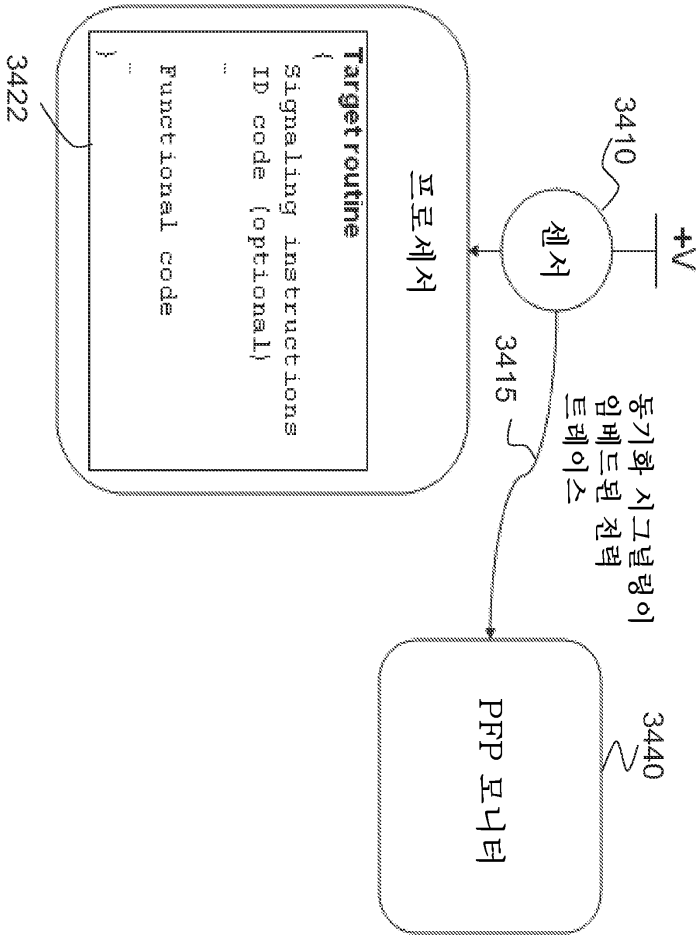
도면32



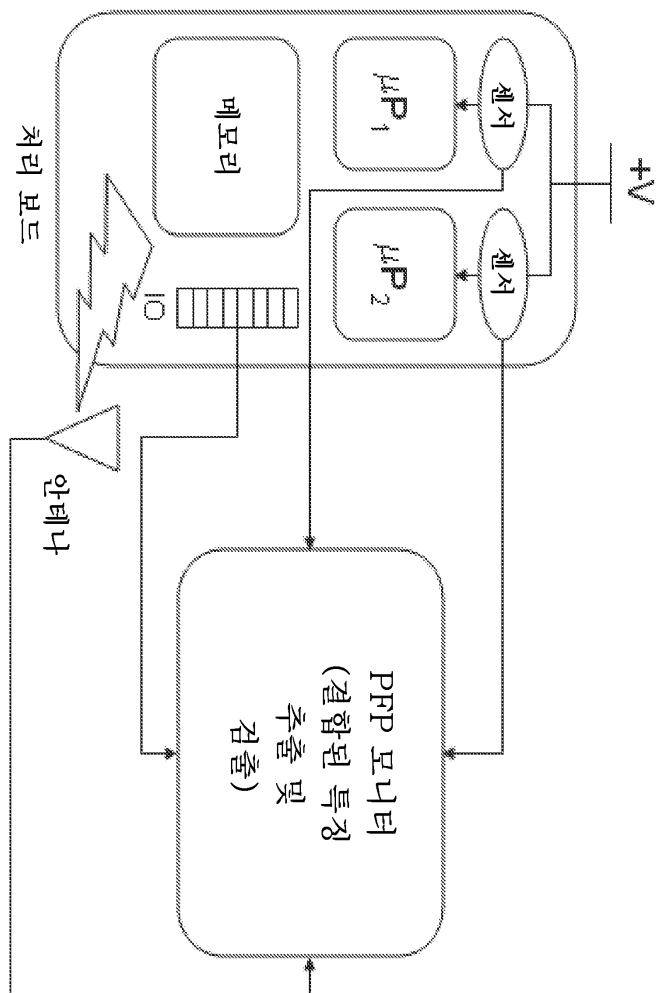
도면33



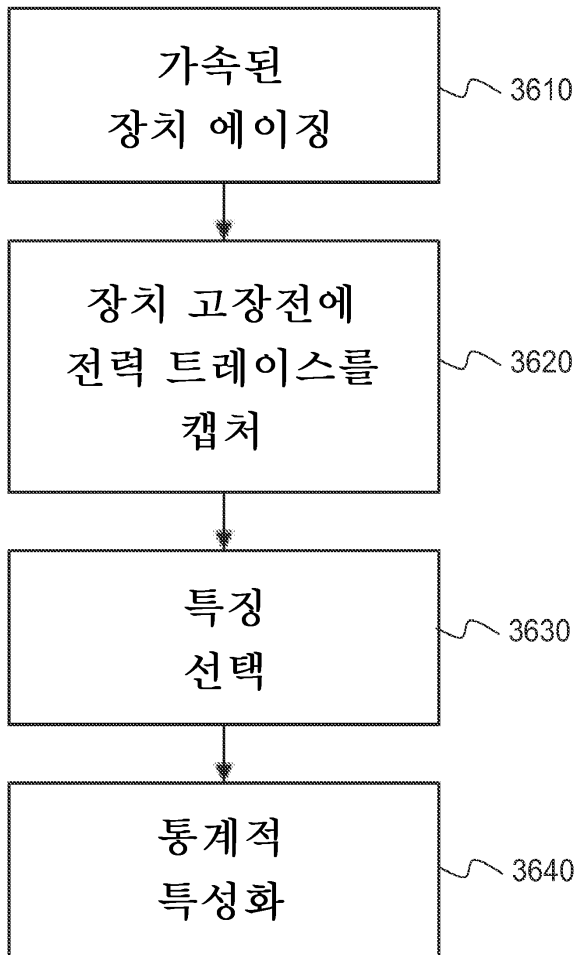
도면34



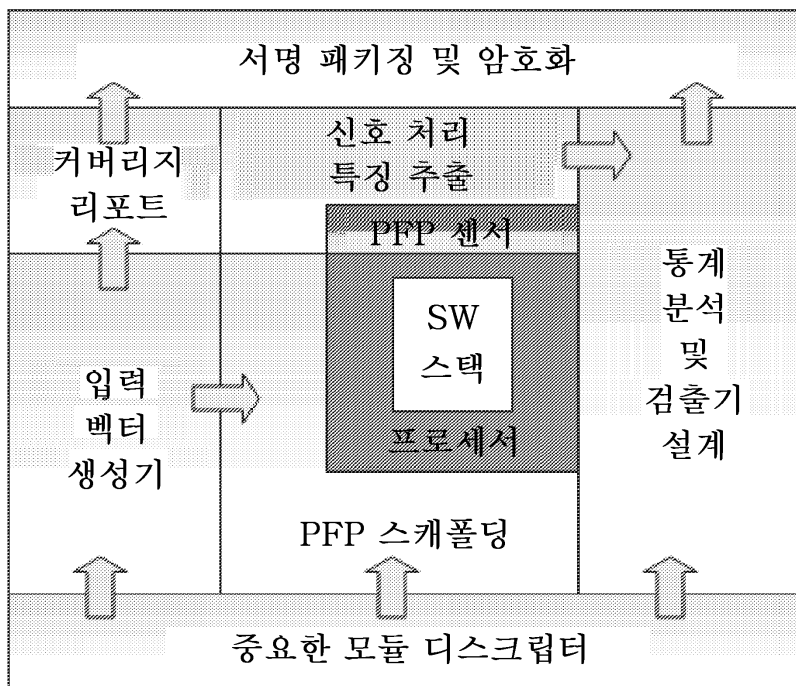
도면35



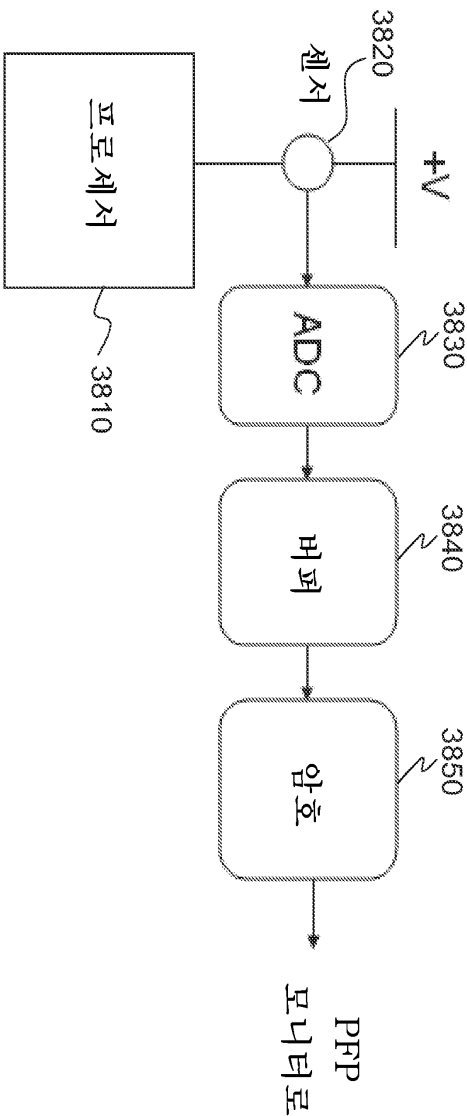
도면36



도면37



도면38



도면39

