



(19) 대한민국특허청(KR)
(12) 공개특허공보(A)

(11) 공개번호 10-2024-0168076
(43) 공개일자 2024년11월29일

- (51) 국제특허분류(Int. Cl.)
G06F 21/56 (2013.01) G06F 21/52 (2013.01)
G06N 3/08 (2023.01)
- (52) CPC특허분류
G06F 21/56 (2013.01)
G06F 21/52 (2013.01)
- (21) 출원번호 10-2023-0065643
(22) 출원일자 2023년05월22일
심사청구일자 2023년05월22일
- (71) 출원인
고려대학교 산학협력단
서울특별시 성북구 안암로 145, 고려대학교 (안암동5가)
- (72) 발명자
김선권
대전광역시 유성구 가정로 43, 107동 903호
이동훈
서울특별시 종로구 사직로8길 34, 1404호(내수동, 경희궁의아침3단지아파트)
(뒷면에 계속)
- (74) 대리인
김동용

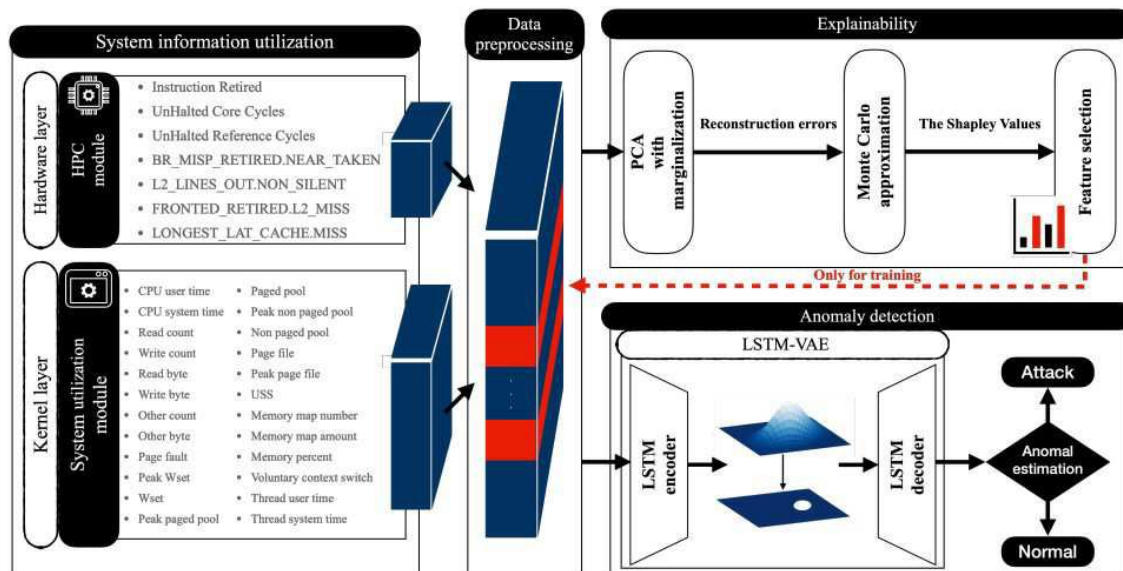
전체 청구항 수 : 총 8 항

(54) 발명의 명칭 시스템 정보를 활용한 ROP 공격 탐지 장치 및 방법

(57) 요약

시스템 정보를 활용한 ROP(Return-Oriented Programming) 공격 탐지 방법 및 장치가 개시된다. 상기 ROP 공격 탐지 방법은 적어도 프로세서(processor)를 포함하는 컴퓨팅 장치에 의해 수행되는 ROP 공격 탐지 방법으로써, 학습 데이터를 수집하는 단계, 상기 학습 데이터를 전처리하는 단계, 및 전처리된 학습 데이터를 이용하여 신경망 모델을 학습함으로써 ROP 공격 탐지 모델을 생성하는 단계를 포함하고, 상기 학습 데이터는 HPC(Hardware Performance Counter) 데이터와 시스템 활용도(system utilization) 데이터를 포함하는 시스템 정보(system information)인 것을 특징으로 한다.

대표도 - 도1



(52) CPC특허분류

G06N 3/08 (2023.01)

(72) 발명자

진홍주

서울특별시 성북구 고려대로27길 34-8, B동 505호

주경호

서울특별시 성북구 지봉로 24길 52-8

이지원

서울특별시 성북구 고려대로1길 83, 404호

이 발명을 지원한 국가연구개발사업

과제고유번호

1711160981

과제번호

2021R1A2C2014428

부처명

과학기술정보통신부

과제관리(전문)기관명

한국연구재단

연구사업명

개인기초연구(과기정통부)

연구과제명

PHY-level에서 획득 가능한 부채널 정보를 이용한 기기 보안서비스 기술 연구

기 여 율

1/1

과제수행기관명

고려대학교

연구기간

2022.03.01 ~ 2024.02.29

명세서

청구범위

청구항 1

적어도 프로세서(processor)를 포함하는 컴퓨팅 장치에 의해 수행되는 ROP 공격(Return-Oriented Programming attack) 탐지 방법에 있어서,

학습 데이터를 수집하는 단계;

상기 학습 데이터를 전처리하는 단계; 및

전처리된 학습 데이터를 이용하여 신경망 모델을 학습함으로써 ROP 공격 탐지 모델을 생성하는 단계를 포함하고,

상기 학습 데이터는 HPC(Hardware Performance Counter) 데이터와 시스템 활용도(system utilization) 데이터를 포함하는 시스템 정보(system information)인,

ROP 공격 탐지 방법.

청구항 2

제1항에 있어서,

상기 전처리하는 단계는,

상기 HPC 데이터와 상기 시스템 활용도 데이터의 타임 포인트(time point)와 타임 인터벌(time interval)을 일치시키는 단계를 포함하는,

ROP 공격 탐지 방법.

청구항 3

제2항에 있어서,

상기 전처리하는 단계 이후에,

상기 전처리된 학습 데이터 각각의 샐플리 밸류(Shapley value)를 산출하는 단계; 및

가장 높은 샐플리 밸류를 갖는 미리 정해진 개수의 시스템 정보를 선택하는 단계를 더 포함하는,

ROP 공격 탐지 방법.

청구항 4

제3항에 있어서,

상기 ROP 공격 탐지 모델을 생성하는 단계는 선택된 학습 데이터만을 이용하여 수행되는,

ROP 공격 탐지 방법.

청구항 5

제4항에 있어서,

상기 ROP 공격 탐지 모델을 생성하는 단계는 ROP 공격이 수행되지 않은 환경에서 수집된 정상 데이터만을 이용하여 학습을 수행하는,

ROP 공격 탐지 방법.

청구항 6

제5항에 있어서,
 상기 전처리하는 단계는,
 알고리즘 1을 이용하여 상기 학습 데이터를 변환하는 단계를 더 포함하는,
 ROP 공격 탐지 방법.
 [알고리즘 1]

Algorithm 1: Data preprocessing algorithm for system information.

Input: $X_{hpc} \in \mathbb{R}^{samples \times features}$
 $X_{util} \in \mathbb{R}^{samples \times features}$
Output: $X_{sysInfo} \in \mathbb{R}^{resamples \times features}$
 $N1 \leftarrow samples;$
 $N2 \leftarrow features;$
while $N1 \neq 0$ **do**
 $IPC1 \leftarrow \frac{X_{hpc}(N1, \text{Instruction Retired})}{X_{hpc}(N1, \text{UnHalted Core Cycles})};$
 $IPC2 \leftarrow \frac{X_{hpc}(N1, \text{Instruction Retired})}{X_{hpc}(N1, \text{UnHalted Reference Cycles})};$
 $X_{hpcA} \leftarrow \text{merge } IPC1, IPC2, \text{ and } X_{hpc}(N1, \text{others});$
 $N1 \leftarrow N1 - 1;$
end
while $N2 \neq 0$ **do**
 $X_{utilZ} \leftarrow \text{zscore}(X_{util}(:, N2));$
 $N2 \leftarrow N2 - 1;$
end
 $X_{hpcR} \in \mathbb{R}^{resamples \times features} \leftarrow \text{resample } X_{hpcA} \text{ to } 1ms;$
 $X_{utilR} \in \mathbb{R}^{resamples \times features} \leftarrow \text{resample } X_{utilZ} \text{ to } 1ms;$
 $X_{hpcZ} \leftarrow \text{interpolate } X_{hpcR} \text{ with zero value};$
 $X_{utilN} \leftarrow \text{interpolate } X_{utilR} \text{ with nearest value};$
 $X_{sysInfo} \leftarrow \text{merge } X_{hpcZ}, X_{utilN};$

청구항 7

제6항에 있어서,
 타겟 프로그램의 실행 중에 ROP 공격 발생 여부를 탐지하는 단계를 더 포함하는,
 ROP 공격 탐지 방법.

청구항 8

제7항에 있어서,
 상기 탐지하는 단계는,
 상기 타겟 프로그램의 실행 중에 상기 미리 정해진 개수의 시스템 정보를 수집하는 단계를 포함하는,
 ROP 공격 탐지 방법.

발명의 설명

기술 분야

본 발명은 소프트웨어 보안에 관한 것으로, 특히 침입 탐지 시스템으로써 정상 프로그램과 ROP 공격(Return-Oriented Programming attack) 발생 사이의 시스템 정보(system information) 차이에 기반한 딥러닝 기술을 활용한 공격 탐지 기법에 관한 것이다.

[0001]

배경 기술

- [0002] 메모리 취약점은 현대 컴퓨터 시스템 상에서 다양한 종류의 소프트웨어 개발 및 동작에서 발생한다. 특히, 스택 메모리 영역에서 발생하는 메모리 취약점은 현대 컴퓨터 시스템을 대상으로 하는 공격자에게 시스템으로부터 민감한 정보를 탈취하고 멀웨어 설치에 유용한 도구로써 사용된다.
- [0003] 현대 컴퓨터 시스템을 대상으로 하는 공격자는 메모리 취약점을 기반으로 하여 스택의 반환 주소(return address)를 공격자가 의도대로 변조하고 프로그램의 제어 흐름을 탈취하는 ROP 공격(Return-Oriented Programming attack)을 수행한다. 다시 말하면, 공격자는 버퍼 오버플로와 같은 메모리 취약점을 활용해 스택의 반환 주소를 변조하여 공격자가 원하는 임의의 코드를 동작시킨다. 이와 같이, 스택의 반환 주소를 변조하는 ROP 공격은 실제 현대 컴퓨터 시스템 해킹에 빈번하게 사용된다.
- [0004] ROP 공격에 대한 최신 보호 기법들은 스택의 반환 주소 변조를 성능 오버헤드 관점에서 실용적으로 방어하거나 탐지하기 위해 소프트웨어 측면과 하드웨어 측면을 활용한 하이브리드 방식으로 구현된다.
- [0005] ROP 공격에 대한 하드웨어 측면의 하이브리드 방어 기법 가운데는 부가적인 하드웨어 지원을 통해 스택 반환 주소 인증 메커니즘을 제시한다. 하지만, 부가적인 하드웨어 지원을 통한 ROP 공격 방어 기법은 기존 시스템에 추가적으로 필요한 하드웨어를 설치하기 위한 비용 문제가 발생한다.
- [0006] ROP 공격에 대한 소프트웨어 측면의 하이브리드 방어 기법 가운데는 기존 시스템에 존재하는 컴파일러에 스택 반환 주소의 변조를 확인하는 명령어를 추가 및 보완한다. 하지만, 기존 시스템에 존재하는 컴파일러에 대한 수정이 필요하기 때문에 수정된 컴파일러가 제공하는 ISA(Instruction Set Architecture)와 기존 시스템과의 호환성 문제가 발생한다.

선행기술문헌

특허문헌

- [0007] (특허문헌 0001) 대한민국 공개특허 제2021-0155985호 (2021.12.24. 공개)
- (특허문헌 0002) 대한민국 공개특허 제2018-0041953호 (2018.04.25. 공개)
- (특허문헌 0003) 대한민국 등록특허 제2156891호 (2020.09.16. 공고)

발명의 내용

해결하려는 과제

- [0008] 본 발명이 이루고자 하는 기술적인 과제는 시스템 정보를 활용한 ROP(Return-Oriented Programming) 공격을 탐지하는 장치 및 방법을 제공하는 것이다.

과제의 해결 수단

- [0009] 본 발명의 일 실시예에 따른 ROP 공격 탐지 방법은 적어도 프로세서(processor)를 포함하는 컴퓨팅 장치에 의해 수행되는 ROP 공격(Return-Oriented Programming attack) 탐지 방법으로써, 학습 데이터를 수집하는 단계, 상기 학습 데이터를 전처리하는 단계, 및 전처리된 학습 데이터를 이용하여 신경망 모델을 학습함으로써 ROP 공격 탐지 모델을 생성하는 단계를 포함하고, 상기 학습 데이터는 HPC(Hardware Performance Counter) 데이터와 시스템 활용도(system utilization) 데이터를 포함하는 시스템 정보(system information)인 것을 특징으로 한다.

발명의 효과

- [0010] 본 발명의 실시예에 따른, 시스템 정보를 활용한 ROP 공격 탐지 장치 및 방법에 의할 경우, 추가적인 하드웨어나 컴파일러의 수정 없이 효과적으로 ROP 공격을 탐지할 수 있는 효과가 있다.

도면의 간단한 설명

- [0011] 본 발명의 상세한 설명에서 인용되는 도면을 보다 충분히 이해하기 위하여 각 도면의 상세한 설명이 제공된다.

도 1은 본 발명의 전체적인 개요도이다.

도 2는 ROP 공격에 대한 30개의 시스템 정보의 기여도를 나타낸다.

도 3은 시스템 정보와 딥러닝 모델의 공격 판단에 대한 시각화 결과를 도시한다.

도 4는 본 발명의 일 실시예에 따른 ROP 공격 탐지 모델 생성 방법을 설명하기 위한 흐름도이다.

발명을 실시하기 위한 구체적인 내용

- [0012] 본 명세서에 개시되어 있는 본 발명의 개념에 따른 실시예들에 대해서 특정한 구조적 또는 기능적 설명들은 단지 본 발명의 개념에 따른 실시예들을 설명하기 위한 목적으로 예시된 것으로서, 본 발명의 개념에 따른 실시예들은 다양한 형태로 실시될 수 있으며 본 명세서에 설명된 실시예들에 한정되지 않는다.
- [0013] 본 발명의 개념에 따른 실시예들은 다양한 변경들을 가할 수 있고 여러 가지 형태들을 가질 수 있으므로 실시예들을 도면에 예시하고 본 명세서에서 상세하게 설명하고자 한다. 그러나, 이는 본 발명의 개념에 따른 실시예들을 특정한 개시 형태들에 대해 한정하려는 것이 아니며, 본 발명의 사상 및 기술 범위에 포함되는 모든 변경, 균등물, 또는 대체물을 포함한다.
- [0014] 제1 또는 제2 등의 용어는 다양한 구성 요소들을 설명하는데 사용될 수 있지만, 상기 구성 요소들은 상기 용어들에 의해 한정되어서는 안 된다. 상기 용어들은 하나의 구성 요소를 다른 구성 요소로부터 구별하는 목적으로만, 예컨대 본 발명의 개념에 따른 권리 범위로부터 벗어나지 않은 채, 제1 구성 요소는 제2 구성 요소로 명명될 수 있고 유사하게 제2 구성 요소는 제1 구성 요소로도 명명될 수 있다.
- [0015] 어떤 구성 요소가 다른 구성 요소에 "연결되어" 있다거나 "접속되어" 있다고 언급된 때에는, 그 다른 구성 요소에 직접적으로 연결되어 있거나 또는 접속되어 있을 수도 있지만, 중간에 다른 구성 요소가 존재할 수도 있다고 이해되어야 할 것이다. 반면에, 어떤 구성 요소가 다른 구성 요소에 "직접 연결되어" 있다거나 "직접 접속되어" 있다고 언급된 때에는 중간에 다른 구성 요소가 존재하지 않는 것으로 이해되어야 할 것이다. 구성 요소들 간의 관계를 설명하는 다른 표현들, 즉 "~사이에"와 "바로 ~사이에" 또는 "~에 이웃하는"과 "~에 직접 이웃하는" 등도 마찬가지로 해석되어야 한다.
- [0016] 본 명세서에서 사용한 용어는 단지 특정한 실시예를 설명하기 위해 사용된 것으로서, 본 발명을 한정하려는 의도가 아니다. 단수의 표현은 문맥상 명백하게 다르게 뜻하지 않는 한, 복수의 표현을 포함한다. 본 명세서에서, "포함하다" 또는 "가지다" 등의 용어는 본 명세서에 기재된 특징, 숫자, 단계, 동작, 구성 요소, 부분품 또는 이들을 조합한 것이 존재함을 지정하려는 것이지, 하나 또는 그 이상의 다른 특징들이나 숫자, 단계, 동작, 구성 요소, 부분품 또는 이들을 조합한 것들의 존재 또는 부가 가능성을 미리 배제하지 않는 것으로 이해되어야 한다.
- [0017] 다르게 정의되지 않는 한, 기술적이거나 과학적인 용어를 포함해서 여기서 사용되는 모든 용어들은 본 발명이 속하는 기술 분야에서 통상의 지식을 가진 자에 의해 일반적으로 이해되는 것과 동일한 의미를 가진다. 일반적으로 사용되는 사전에 정의되어 있는 것과 같은 용어들은 관련 기술의 문맥상 가지는 의미와 일치하는 의미를 갖는 것으로 해석되어야 하며, 본 명세서에서 명백하게 정의하지 않는 한, 이상적이거나 과도하게 형식적인 의미로 해석되지 않는다.
- [0018] 이하, 본 명세서에 첨부된 도면들을 참조하여 본 발명의 실시예들을 상세히 설명한다. 그러나, 특허출원의 범위가 이러한 실시예들에 의해 제한되거나 한정되는 것은 아니다. 각 도면에 제시된 동일한 참조 부호는 동일한 부재를 나타낸다.
- [0019] 본 발명에서는 시스템 정보(system information)를 이용하여 ROP 공격(Return-Oriented Programming attack)을 탐지하는 장치 및 방법을 제안한다. 이를 위해, 시스템의 운영체제 커널을 통해 하드웨어 성능 카운터(Hardware Performance Counter, HPC) 데이터와 시스템 활용도(system utilization) 데이터가 수집될 수 있다. 즉, 본 발명에서는 HPC 데이터의 비결정론적 특성으로 인한 성능 제약을 극복하기 위해 HPC 데이터와 더불어 프로세스(process)의 시스템 자원 사용량을 나타내는 시스템 활용도 데이터를 함께 이용한다.
- [0020] 이와 관련하여, 표 1은 수집되는 HPC 데이터를, 표 2는 수집되는 시스템 활용도 데이터를 나타낸다.

[표 1]

Index	HPC data	Explanation
1	Instruction Retired	Number of instructions at retirement
2	UnHalted Core Cycles	Core clock cycles
3	UnHalted Reference Cycles	Reference clock cycles
4	BR_MISP_RETIREDD.NEAR_TAKEN	Number of near branch instructions retired that were mispredicted and taken
5	L2_LINES_OUT.NON_SILENT	Number of lines evicted by L2 cache
6	FRONTEND_RETIREDD.L2_MISS	Number of L2 cache true misses
7	LONGEST_LAT_CACHE.MISS	Number of the last-level cache misses

수집되는 HPC 데이터는 Instruction Retired, UnHalted Core Cycles, UnHalted Reference Cycles, BR_MISP_RETIREDD.NEAR_TAKEN, L2_LINES_OUT.NON_SILENT, FRONTEND_RETIREDD.L2_MISS, 및 LONGEST_LAT_CACHE.MISS 중 적어도 하나를 포함하고, 이에 대한 구체적인 설명은 표 1에 기재되어 있다.

HPC는 CPU의 PMU(Performance Monitoring Unit)에 포함되는 특수 목적의 레지스터 집합(special-purpose register set)을 의미한다. 이러한 레지스터 집합은 하드웨어 성능 이벤트 정보를 저장한다. 일반적으로, HPC는 인텔 CPU에서 코어당 3 개의 고정된 카운터와 4 개의 프로그램 가능한 카운터를 제공한다. 그리고, 최근의 프로세서들은 많은 하드웨어 성능 이벤트들을 제공한다. 다양한 프로세서 성능들을 평가하기 위해, PMU는, 모델-특화된(model-specific) HPCs를 통해, 하드웨어 성능 이벤트들을 모니터링한다.

[표 2]

Index	System utilization	Explanation
1	CPU user time	Process time spent in user mode
2	CPU system time	Process time spent in kernel mode
3	Read count	Number of read operations performed by a process
4	Write count	Number of write operations performed by a process
5	Read byte	Number of bytes read
6	Write byte	Number of bytes written
7	Other count	Number of I/O operations performed (other than read and write operations)
8	Other byte	Number of bytes transferred (other than read and write operations)
9	Page fault	Number of page faults
10	Peak Wset	Peak non-swapped physical memory used by a process
11	Wset	Non-swapped physical memory used by a process
12	Peak paged pool	Peak paged pool usage by a process
13	Paged pool	The current paged pool usage by a process
14	Peak non-paged pool	Peak non-paged pool usage by a process
15	Non-paged pool	Current non-paged pool usage by a process
16	Page file	Total amount of virtual memory used by a process
17	Peak page file	Peak total amount of virtual memory used by a process
18	USS	Memory size (unique to a process)
19	Memory map number	Number of memory maps per process
20	Memory map amount	Total amount of memory maps per process
21	Memory percent	Process memory utilization as a percentage
22	Voluntary context switch	Number of voluntary context switches by a process
23	Thread user time	Time spent by threads in user mode
24	Thread system time	Time spent by threads in kernel mode

수집되는 시스템 활용도 데이터는 CPU user time, CPU system time, Read count, Write count, Read byte, Write byte, Other count, Other byte, Page fault, Peak Wset, Wset, Peak paged pool, Paged pool, Peak non-paged pool, Non-paged pool, Page file, Peak page file, USS, Memory map number, Memory map amount, Memory percent, Voluntary context switch, Thread user time, 및 Thread system time 중 적어도 하나를 포함하고, 이에 대한 상세한 설명은 표 2에 기재되어 있다.

HPC 데이터는 HPC에 의한 인터럽트 방식으로 동작하는 샘플링 모드로 동작한다. HPC 데이터에 대한 샘플링 주기는 미리 정해진 샘플링율을 갖도록 설정될 수 있다. 예컨대, CPU의 명령어 처리 속도에 따른 실제 프로그램 명

령어 처리수가 약 1ms가 되도록 설정할 수 있다. 시스템 활용도 데이터는 미리 정해진 주기, 예컨대 1ms 주기로 운영체제의 프로세스 자원 관리 파일을 읽을 수 있다. 즉, HPC 데이터와 시스템 활용도 데이터는 미리 정해진 샘플링 주기(예컨대, 1ms)를 가질 수 있다. 실시예에 따라, HPC 데이터는 제1 샘플링율을 가질 수 있고, 시스템 활용도 데이터는 제2 샘플링율을 가질 수 있다. 이때, 제1 샘플링율과 제2 샘플링율은 동일하거나 상이할 수 있다. 또한, 소정의 전처리 동작을 수행하여 동일한 샘플링율을 갖는 HPC 데이터와 시스템 활용도 데이터를 생성할 수도 있다.

[0029] 본 발명에서는, 하드웨어 레이어에서의 HPC 데이터와 커널 레이어에서의 시스템 활용도 데이터를 시스템 정보로 정의한다. HPC 데이터는 HPCs를 통해 측정되는 하드웨어 성능 이벤트들에 대한 정보를 의미한다. 다음으로, 시스템 활용도 데이터는 프로세스들이 이용하는 시스템 리소스와 관련된다. 다시 말하면, 시스템 활용도 정보는 프로세스가 이용 중인 시스템 리소스를 나타낸다. 예컨대, 시스템 활용도 데이터는 CPU 이용, 메모리, 디스크, 네트워크 등등에 대한 정보를 포함한다.

[0030] 도 1은 본 발명의 전체적인 개요도이다.

[0031] 도 1을 참조하면, 본 발명은 4 개의 메인 요소들, 시스템 정보 이용(system information utilization), 데이터 전처리(data preprocessing), 설명가능성(explainability), 및 이상 감지(anomaly detection)로 구성된다. 시스템 정보 이용에서, 두 개의 모듈이 HPC 데이터와 시스템 활용도 데이터를 샘플링한다.

[0032] **시스템 정보 이용.** 우선, 제안하는 시스템은 ROP 공격과 관련된 모든 필요한 시스템 정보를 수집한다.

[0033] **데이터 전처리.** 데이터 전처리를 위해, 수집된 HPC 데이터와 시스템 활용도 데이터는 IPC(instructions per cycle) 또는 Z-스코어(Z-Score)와 같은 의미있는 데이터(meaningful data)로 변환될 수도 있다. 또한, 매 1ms 마다, 제로 보간법(zero interpolation) 및/또는 최근접 보간법(nearest interpolation)으로, 데이터를 업샘플링(upsampling)할 수 있다. 업샘플링 모듈은 HPC 데이터와 시스템 활용도 데이터 사이의 타임 포인트(time point)와 타임 인터벌(time interval)을 매칭하고, 하나의 데이터 셋으로 병합할 수 있다.

[0034] **설명가능성.** 본 발명은, 인공지능 모델(예컨대, LSTM-VAE)의 예측에 대한 설명가능성을 제공하기 위해 샵플리 밸류(Shapley values)를 이용한다. 샵플리 밸류를 계산하기 위해 정의되는 밸류 함수(value function)는 확률론적 PCA(probabilistic Principal Component Analysis)의 재구성 오류로 근사된다. 효율적인 샵플리 밸류 계산을 위해, 밸류 함수는 주변화(marginalization)를 이용하여 간소화 된다. 밸류 함수에 대하여, 본 발명에서는, 몬테 카를로 근사(Monte Carlo approximation)를 이용하여 샵플리 밸류를 계산한다. 샵플리 밸류는 예측 결과에 대한 각 시스템 정보의 기여도를 나타낸다. 인공지능 모델(예컨대, LSTM-VAE)에 이용되는 시스템 정보의 개수를 최소화하기 위하여 기여도에 기초하여 미리 정해진 개수의 특징을 선택할 수 있다. 즉, 가장 높은 샵플리 밸류를 나타내는 미리 정해진 개수의 특징만을 선택하여 학습 동작을 수행할 수 있다. 이와 같이, 본 발명에서 제안하는 시스템(또는 모델)은 공격 탐지 성능을 유지하면서 최적의 오버헤드를 보인다.

[0035] **이상 탐지.** 이상 탐지에서, 본 발명에서는, 준지도학습 방식에 의한 LSTM-VAE를 제안한다. 준지도학습 방식의 사용은 ROP 공격에 대한 충분한 양의 데이터를 수집하기 어렵기 때문이다. LSTM-VAE는 다변량 시계열 시스템 정보의 시간적 종속성을 효과적으로 분석할 수 있다. 이상 스코어(anomaly score)에 기초하여, 제안하는 모델은 ROP 공격을 탐지할 수 있다.

[0036] 이하에서는, 시스템 정보 이용(System information utilization)에 대해 보다 상세하게 설명한다.

[0037] ROP 공격의 두드러진 특징은, 오직 소수의 명령만을 갖는 가젯(gadgets)이 체인을 이용하는 페이로드(payload)에 있다. 시스템 상에서, 페이로드의 실행은 ROP 공격에 대한 몇몇의 단서들을 남긴다. 이러한 단서들 중 하나는, 프로그램 실행 동안에 콜 명령(call instructions)은 리턴 명령(return instructions)과 쌍을 이루지 않는다는 것이다.

[0038] **시스템 정보 이용(System information utilization)**

[0039] 이러한 단서에 따라, 본 발명에서는 ROP 공격과 관련성이 있는 HPC 데이터(표 1 참조) 뿐만 아니라 시스템 활용도 데이터(표 2 참조)를 모니터링한다. 제안하는 시스템은 두 개의 모듈을 포함한다. 제1 모듈은, 파이썬 라이브러리(library)를 통해, 시스템 활용도 데이터를 미리 정해진 주기로(예컨대, 매 1ms 마다) 샘플링한다. 또한, 본 발명에서는, HPC 데이터를 측정하는 동안, 성능 모니터링 인터럽트(performance monitoring interrupt)를 이용한다. 제2 모듈은, 윈도우 커널 드라이브(Windows kernel drive)를 통해, 2.9×10^6 싸이클의 CS-PMI로, HPC 데이터를 샘플링한다. CS-PMI는 타겟 프로세스의 HPC 데이터가 다른 프로세스에 의해 오염되는 것을 예방할 수

있다.

[0040] 데이터 전처리(Data preprocessing)

[0041] 프로그램 실행 동안 시스템 정보를 수집하고, 수집된 시스템 정보를 전처리한다. 알고리즘 1은 데이터 전처리를 위한 알고리즘이다. IPC를 정제하기 위하여, HPC 데이터로부터의 두 가지 종류의 사이클을 이용하고, IPC1 및 IPC2를 계산한다. 다음으로, 프로세스의 시스템 활용도의 변화를 모니터링하기 위하여, 시스템 활용도 데이터에 대해 Z-스코어를 적용할 수 있다. 또한, 타임 인터벌과 타임 포인트를 매칭시키기 위하여, 1ms의 타임 인터벌에서 측정된 HPC 데이터와 시스템 활용도 데이터를 업샘플링할 수 있다. 업샘플링으로 인한 결손치(missing values)는 소정의 보간 기법을 통해 보간될 수 있다. HPC 데이터를 보간하기 위하여 체로 보간법이 이용되고, 시스템 활용도 데이터를 보간하기 위하여 최근접 보간법이 이용될 수 있다. 다만, 본 발명의 권리범위가 사용되는 보간법에 제한되는 것은 아니며, 실시예에 따라 상이한 보간법이 적용될 수도 있다.

[0042] [알고리즘 1]

Algorithm 1: Data preprocessing algorithm for system information.

Input: $X_{hpc} \in \mathbb{R}^{samples \times features}$
 $X_{util} \in \mathbb{R}^{samples \times features}$
Output: $X_{sysInfo} \in \mathbb{R}^{resamples \times features}$
 $N1 \leftarrow samples;$
 $N2 \leftarrow features;$
while $N1 \neq 0$ **do**
 $IPC1 \leftarrow \frac{X_{hpc}(N1, \text{Instruction Retired})}{X_{hpc}(N1, \text{UnHalted Core Cycles})};$
 $IPC2 \leftarrow \frac{X_{hpc}(N1, \text{Instruction Retired})}{X_{hpc}(N1, \text{UnHalted Reference Cycles})};$
 $X_{hpcA} \leftarrow \text{merge } IPC1, IPC2, \text{ and } X_{hpc}(N1, \text{others});$
 $N1 \leftarrow N1 - 1;$
end
while $N2 \neq 0$ **do**
 $X_{utilZ} \leftarrow \text{zscore}(X_{util}(:, N2));$
 $N2 \leftarrow N2 - 1;$
end
 $X_{hpcR} \in \mathbb{R}^{resamples \times features} \leftarrow \text{resample } X_{hpcA} \text{ to } 1\text{ms};$
 $X_{utilR} \in \mathbb{R}^{resamples \times features} \leftarrow \text{resample } X_{utilZ} \text{ to } 1\text{ms};$
 $X_{hpcZ} \leftarrow \text{interpolate } X_{hpcR} \text{ with zero value};$
 $X_{utilN} \leftarrow \text{interpolate } X_{utilR} \text{ with nearest value};$
 $X_{sysInfo} \leftarrow \text{merge } X_{hpcZ}, X_{utilN};$

[0043] 설명가능성(Explainability)

[0044] 제안하는 시스템의 예측 결과를 해석하기 위하여, 이상 탐지를 설명하기 위해 샐플리 밸류를 활용한다. 샐플리 밸류 계산에는 제안하는 시스템의 예측 능력을 나타내는 밸류 함수를, 확률론적 PCA(probabilistic PCA)의 재구성 오류를 이용함으로써, 정의한다. 또한, 샐플리 밸류를 효율적으로 계산하기 위해서, 밸류 함수를 단순화하는 주변화 기법(Naoya Takeishi. Shapley values of reconstruction errors of pca for explaining anomaly detection. In 2019 international conference on data mining workshops (icdmw), pages 793-798. IEEE, 2019.)을 이용할 수 있다. 그러면, 특징들의 서브셋을 임의로 선택하는 몬테 카를로 근사를 갖는 밸류 함수를 이용하여 샐플리 밸류를 계산할 수 있다. 30개의 시스템 정보에 대한 샐플리 밸류에 의해 기여도를 측정한다. 제안하는 시스템의 예측 성능을 정의하기 위해서, 정상 동작으로부터의 110개의 데이터 셋을 이용한다. PCA의 차원은 8로 설정될 수 있다. ROP 공격 실행으로부터의 110 개의 데이터 셋이 샐플리 밸류를 계산하기 위해 이용된다. 몬테 카를로 반복은 1,000으로 설정된다. 데이터 셋의 샘플당 각 특징에 대한 샐플리 밸류가 평가된다. ROP 공격에 대한 30개의 시스템 정보의 기여도는 도 2에 도시되어 있다.

[0045] 요컨대, 시계열 딥러닝 모델인 LSTM-VAE의 성능 오버헤드를 줄이기 위해 특징 선택(feature selection)을 수행할 수 있다. 특징 선택에서는 각 시스템 정보 데이터 가운데 공격 탐지에 높은 기여도를 가지는 데이터를 선택할 수 있다. 확률론적 PCA를 통해 각 시스템 정보 데이터 가운데 공격 탐지에 얼마만큼의 기여도를 갖는지 샐플리

밸류를 계산한다. 특히, 새플리 밸류에 대한 계산은 몬테 카를로 근사를 통해 수행하게 되고, 이를 해당 딥러닝 모델의 입력으로 학습에 활용할 수 있다.

[0047] **이상 탐지(Anomaly detection)**

[0048] 이상 탐지는, 정상과 비정상 데이터 사이에 클래스 불균형이 존재할 때 또는 정상 데이터에 비해 비정상 데이터가 충분하지 않을 때, 유용하다. 실제 ROP 공격 데이터를 수집하기 어려운 환경에서, 탐지 성능을 향상시키기 위해, 본 발명에서는 모델 학습 과정에서 공격에 대한 사전 정보를 요구하지 않는 준지도 방식의 이상 탐지에 기초한다.

[0049] 본 발명에서는 LSTM-VAE를 이용한다. LSTM-VAE는 3D 시퀀스(샘플들×타임 윈도우 사이즈×특징들)를 입력 데이터로 수신한다. 정상 실행과 ROP 공격 실행 사이의 입력 데이터를 구분하기 위해서, LSTM-VAE 재구성 에러를 이상 스코어로 이용한다. 재구성 에러는 MSE(Mean Squared Error)로 계산될 수 있다. 높은 이상 스코어는, 입력 데이터가 LSTM-VAE에 의해 잘 재구성되지 않은 것을 나타낸다. 최적 임계값을 획득하기 위해서, PRC(Precision-Recall Curve)를 이용하여, 정밀도와 재현율 스코어(precision and recall scores) 사이의 차이가 최소화되는 가장 높은 스코어를 이상 스코어의 임계치로 설정할 수 있다. 모델의 학습 과정에서는 정상 데이터만을 이용하여 학습될 수 있다.

[0050] 정상 실험 데이터는 윈도우 운영체제 타겟 머신에서 동작 중인 특정 프로그램에 대하여 모니터링된 데이터를 사용할 수 있고, 공격 실험 데이터는 칼리 리눅스 운영체제 공격 머신에 설치된 메타스플로잇 프레임워크를 통해 윈도우 운영체제 타겟 머신에서 동작 중인 특정 프로그램에 ROP 공격을 수행하고 이를 모니터링함으로써 획득할 수 있다.

[0051] 전처리된 시스템 정보를 입력으로 받는 시계열 딥러닝 모델인 LSTM-AVE는 케라스 딥러닝 라이브러리의 stateless LSTM 모델을 활용하여 구현될 수 있으며, 사용된 옵티마이저는 adam, 학습율(learning rate)은 0.0001, 활성화 함수는 tanh가 적용될 수 있다.

[0052] 도 3은 시스템 정보와 딥러닝 모델의 공격 판단에 대한 시각화 결과를 도시한다.

[0053] 도 3을 참조하면, 상단 6개의 그래프는 모니터링된 시스템 정보를, 최하단은 딥러닝 모델의 공격 판단 근거인 이상 점수를 나타낸다. 핑크색 세로 선들에 의한 영역은 ROP 공격이 발생한 시점을 나타낸다. 구체적으로, 도 3의 (a)는 프로그램의 정상 실행을 나타내고, (b)는 cve-2015-0359를 기반으로 한 ROP 공격 발생을 나타내고, (c)는 cve-2012-1876을 기반으로 한 ROP 공격 발생을 나타낸다. 정상 동작을 나타내는 (a)의 그래프에서는 이상 점수가 1을 넘지 않음에 비해, ROP 공격이 수행된 (b)와 (c)의 이상 점수는 1 보다 매우 큰 값을 가짐을 볼 수 있다. 이에 따라, ROP 공격을 효과적으로 탐지할 수 있다.

[0054] 도 4는 본 발명의 일 실시예에 따른 ROP 공격 탐지 모델 생성 방법을 설명하기 위한 흐름도이다. 이하에서는, 앞선 설명과 중복되는 내용에 관한 구체적인 기재를 생략하기로 한다.

[0055] ROP 공격 탐지 모델 생성 방법은 ROP 공격 탐지 방법 등으로 명명될 수도 있으며, 적어도 프로세서(processor) 및/또는 메모리(memory)를 포함하는 컴퓨팅 장치에 의해 수행될 수 있다. 컴퓨팅 장치는 PC(Personal Computer), 서버(server), 노트북 PC, 랩탑 컴퓨터, 태블릿 PC 등을 포함할 수 있다. 또한, 방법을 이루는 각 단계는 물리적으로 분리된 복수의 컴퓨팅 장치에 의해 수행되는 것도 가능하다.

[0056] 우선, 시스템 정보가 수집된다(S110). 시스템 정보는 HPC 데이터와 시스템 활용도 데이터를 포함한다. 시스템 정보는 임의의 프로그램을 실행하는 도중에, ROP 공격이 수행되는 환경 및/또는 정상 동작 환경에서, 수집될 수 있다. 실시예에 따라, 시스템 정보는 USB 메모리 장치와 같은 저장 장치로부터 입출력 인터페이스를 통해 수신되거나, 미리 수신되어 컴퓨팅 장치에 저장되어 있을 수도 있다.

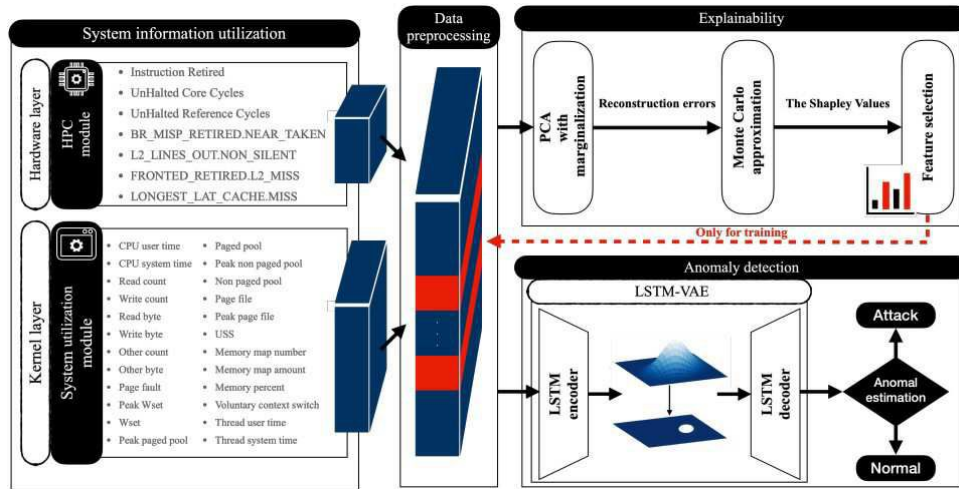
[0057] 시스템 정보에 대한 전처리 동작이 수행된다(S120). 전처리 동작은 HPC 데이터와 시스템 활용도 데이터 사이의 타임 포인트 및/또는 타임 인터벌을 일치시키는 동작을 포함할 수 있다. 또한, 전처리 동작은 알고리즘 1에 의한 데이터 변환 동작을 포함할 수도 있다.

[0058] 전처리된 시스템 정보에 대하여, 학습에 이용된 특징 선택이 수행된다(S130). 선택되는 특징은 가장 높은 새플리 밸류를 갖는 미리 정해진 개수의 시스템 정보를 의미한다. 실시예에 따라, S130 단계는 생략되는 것도 가능하다. 이 경우, 미리 정해진 시스템 정보가 수집되거나, 및/또는 미리 정해진 시스템 정보에 대한 전처리 동작만이 수행될 수 있다.

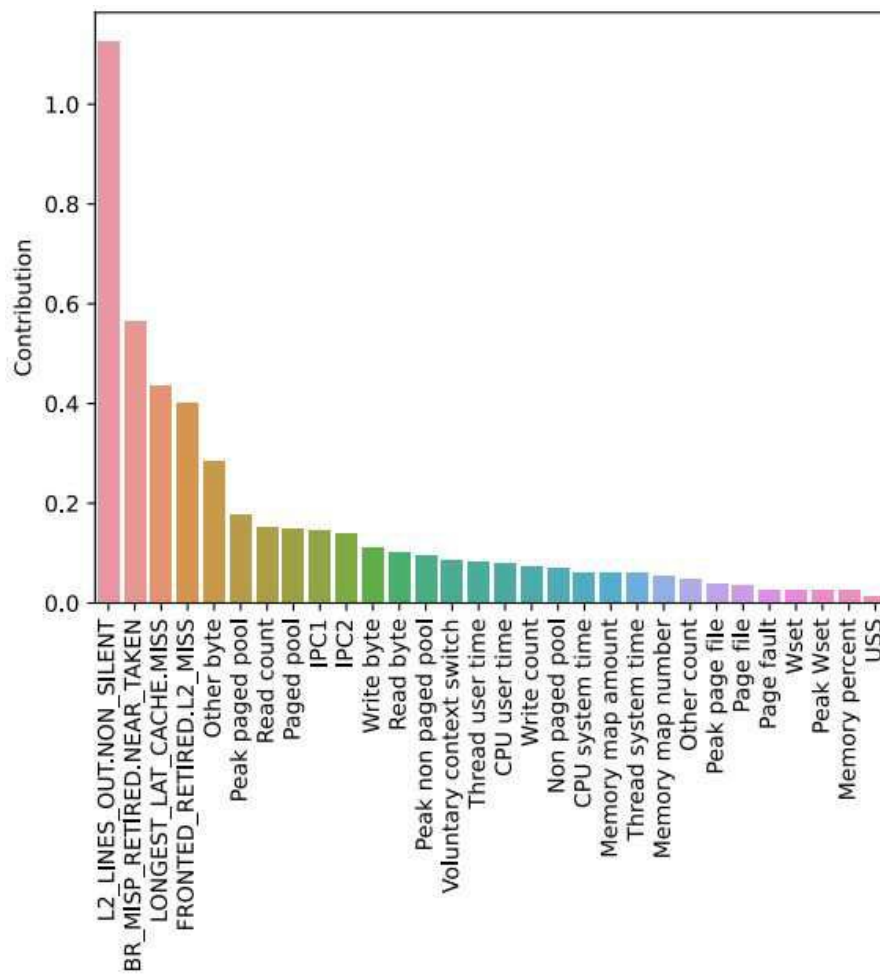
- [0059] 전처리된 시스템 정보 중에서 선택된 시스템 정보를 학습 데이터로 이용하여 신경망 모델(예컨대, LSTM-AVE)을 학습시킴으로써, ROP 공격 탐지 모델을 생성할 수 있다(S140). 학습 과정에서는, 정상 데이터(ROP 공격이 수행되지 않은 환경에서 수집된 시스템 정보)만을 이용하여 학습될 수 있다.
- [0060] 다음으로, ROP 공격을 탐지하는 단계가 더 수행될 수 있다(S150). 보다 구체적으로, 소정의 프로그램(타겟 프로그램이라 칭함)을 실행하는 도중에 시스템 정보가 수집될 수 있다. 수집되는 시스템 정보는 선택된 특징에 대응하는 시스템 정보만을 의미할 수 있다. 수집된 시스템 정보에 대한 전처리 동작이 수행되고, 전처리된 시스템 정보를 ROP 공격 탐지 모델에 입력함으로써, ROP 공격을 탐지할 수 있다.
- [0061] 이상에서 설명된 장치는 하드웨어 구성 요소, 소프트웨어 구성 요소, 및/또는 하드웨어 구성 요소 및 소프트웨어 구성 요소의 집합으로 구현될 수 있다. 예를 들어, 실시예들에서 설명된 장치 및 구성 요소는, 예를 들어, 프로세서, 컨트롤러, ALU(Arithmetic Logic Unit), 디지털 신호 프로세서(Digital Signal Processor), 마이크로컴퓨터, FPA(Field Programmable array), PLU(Programmable Logic Unit), 마이크로프로세서, 또는 명령(instruction)을 실행하고 응답할 수 있는 다른 어떠한 장치와 같이, 하나 이상의 범용 컴퓨터 또는 특수 목적 컴퓨터를 이용하여 구현될 수 있다. 처리 장치는 운영 체제(Operation System, OS) 및 상기 운영 체제 상에서 수행되는 하나 이상의 소프트웨어 애플리케이션을 수행할 수 있다. 또한, 처리 장치는 소프트웨어의 실행에 응답하여, 데이터를 접근, 저장, 조작, 처리 및 생성할 수도 있다. 이해의 편의를 위하여, 처리 장치는 하나가 사용되는 것으로 설명된 경우도 있지만, 해당 기술 분야에서 통상의 지식을 가진 자는, 처리 장치가 복수 개의 처리 요소(Processing Element) 및/또는 복수 유형의 처리 요소를 포함할 수 있음을 알 수 있다. 예를 들어, 처리 장치는 복수 개의 프로세서 또는 하나의 프로세서 및 하나의 컨트롤러를 포함할 수 있다. 또한, 병렬 프로세서(Parallel Processor)와 같은, 다른 처리 구성(Processing Configuration)도 가능하다.
- [0062] 소프트웨어는 컴퓨터 프로그램(Computer Program), 코드(Code), 명령(Instruction), 또는 이들 중 하나 이상의 조합을 포함할 수 있으며, 원하는 대로 동작하도록 처리 장치를 구성하거나 독립적으로 또는 결합적으로(Collectively) 처리 장치를 명령할 수 있다. 소프트웨어 및/또는 데이터는, 처리 장치에 의하여 해석되거나 처리 장치에 명령 또는 데이터를 제공하기 위하여, 어떤 유형의 기계, 구성 요소(Component), 물리적 장치, 가상 장치(Virtual Equipment), 컴퓨터 저장 매체 또는 장치, 또는 전송되는 신호 파(Signal Wave)에 영구적으로, 또는 일시적으로 구체화(Embody)될 수 있다. 소프트웨어는 네트워크로 연결된 컴퓨터 시스템 상에 분산되어서, 분산된 방법으로 저장되거나 실행될 수도 있다. 소프트웨어 및 데이터는 하나 이상의 컴퓨터 판독 가능 기록 매체에 저장될 수 있다.
- [0063] 실시예에 따른 방법은 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 실시예를 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), 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 실시예의 동작을 수행하기 위해 하나 이상의 소프트웨어 모듈로서 작동하도록 구성될 수 있으며, 그 역도 마찬가지이다.
- [0064] 본 발명은 도면에 도시된 실시예를 참고로 설명되었으나 이는 예시적인 것에 불과하며, 본 기술 분야의 통상의 지식을 가진 자라면 이로부터 다양한 변형 및 균등한 타 실시예가 가능하다는 점을 이해할 것이다. 예를 들어, 설명된 기술들이 설명된 방법과 다른 순서로 수행되거나, 및/또는 설명된 시스템, 구조, 장치, 회로 등의 구성 요소들이 설명된 방법과 다른 형태로 결합 또는 조합되거나, 다른 구성 요소 또는 균등물에 의하여 대치되거나 치환되더라도 적절한 결과가 달성될 수 있다. 따라서, 본 발명의 진정한 기술적 보호 범위는 첨부된 등록청구범위의 기술적 사상에 의해 정해져야 할 것이다.

도면

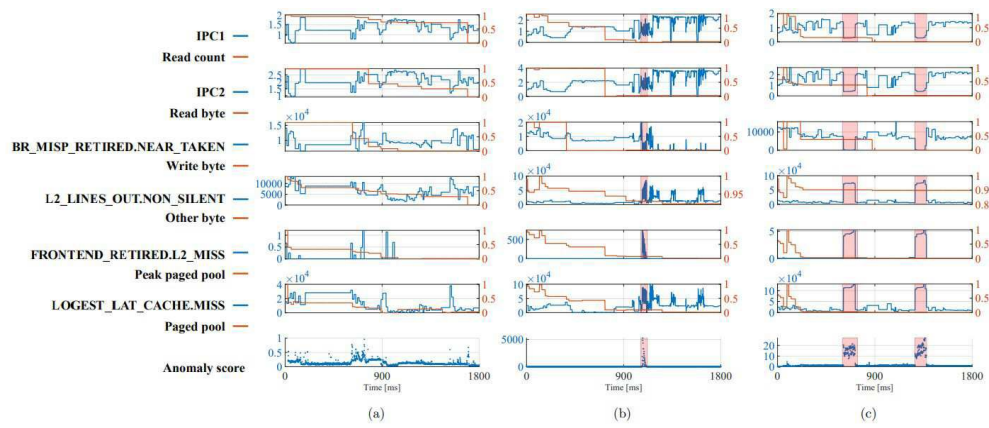
도면1



도면2



도면3



도면4

