특허증

CERTIFICATE OF PATENT

특 허 Patent Number	제 10-2656980 호
출원번호 Application Number	제 10-2022-0031506 호
출원일 Filing Date	2022년 03월 14일
등록일 Registration Date	2024년 04월 08일

발명의명칭 Title of the Invention

가상화 환경에서 인스턴스의 메모리를 연속적으로 기록하는 방법 및 장치

특허권자 Patentee

세종대학교산학협력단(240171-*****)**
서울특별시 광진구 능동로 209 (군자동, 세종대학교)

발명자 Inventor

등록사항란에 기재

위의 발명은 「특허법」에 따라 특허원부에 등록되었음을 증명합니다.

This is to certify that, in accordance with the Patent Act, a patent for the invention has been registered at the Korean Intellectual Property Office.

2024년 04월 08일

QR코드로 현재기준
등록사항을 확인하세요

특허청장
COMMISSIONER,
KOREAN INTELLECTUAL PROPERTY OFFICE

특허청
Korean Intellectual
Property Office

등록사항

특 허 **제 10-2656980 호**
Patent Number

발명자 Inventors

박기웅(791002-*****)**
서울특별시 광진구 능동로17길 21, 304호(화양동)

최상훈(890518-*****)**
서울특별시 광진구 긴고랑로2길 38-2, 301호(중곡동)

(19) 대한민국특허청(KR)
(12) 등록특허공보(B1)

(45) 공고일자 2024년04월15일
(11) 등록번호 10-2656980
(24) 등록일자 2024년04월08일

(51) 국제특허분류(Int. Cl.)
G06F 9/455 (2018.01) *G06F 21/53* (2013.01)
(52) CPC특허분류
G06F 9/45558 (2013.01)
G06F 21/53 (2013.01)
(21) 출원번호 10-2022-0031506
(22) 출원일자 2022년03월14일
심사청구일자 2022년03월14일
(65) 공개번호 10-2023-0134323
(43) 공개일자 2023년09월21일
(56) 선행기술조사문헌
최상훈 외 1명. 클라우드 보안성 강화를 위한 연산 효율적인 인스턴스 메모리 모니터링 기술. 2017년 8월
KR1020210105036 A
KR1020160030385 A

(73) 특허권자
세종대학교산학협력단
서울특별시 광진구 능동로 209 (군자동, 세종대학교)
(72) 발명자
박기웅
서울특별시 광진구 능동로17길 21, 304호(화양동)
최상훈
서울특별시 광진구 긴고랑로2길 38-2, 301호(중곡동)
(74) 대리인
양성보

전체 청구항 수 : 총 14 항 심사관 : 최정권

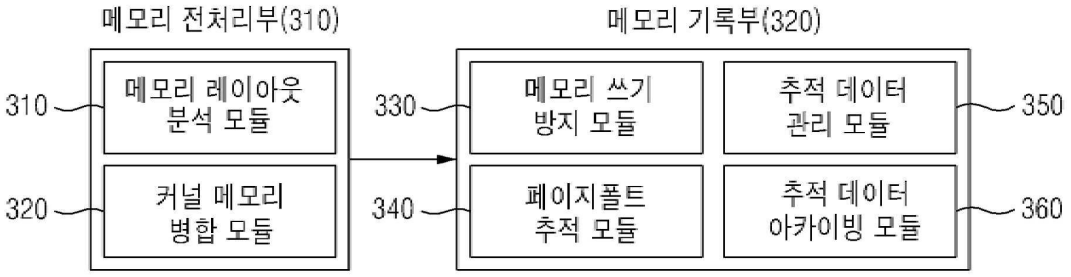
(54) 발명의 명칭 **가상화 환경에서 인스턴스의 메모리를 연속적으로 기록하는 방법 및 장치**

(57) 요 약

가상화 환경에서 인스턴스의 메모리를 연속적으로 기록하는 방법 및 장치가 제시된다. 본 발명에서 제안하는 가상화 환경에서 인스턴스의 메모리를 연속적으로 기록하는 장치는 복수의 서비스가 구동되는 클라우드 환경에서 구동 중인 서비스들의 메모리 레이아웃을 분석하고 그룹화하는 메모리 레이아웃 분석 모듈, 상기 분석된 메모리

(뒷면에 계속)

대 표 도 - 도3



레이아웃에 기초하여 메모리의 중복성 검사를 수행하고, 중복되는 메모리 영역을 하나의 메모리 영역으로 병합하는 커널 메모리 병합 모듈, 병합된 메모리 영역에서 변화되는 메모리를 추적하여 기록하고, 병합되지 않은 메모리 영역에 쓰기 연산을 방지할 수 있도록 메모리 영역을 보호하는 메모리 쓰기 방지 모듈, 상기 메모리 쓰기 방지모듈에 의해 보호된 메모리 영역에서 쓰기 연산이 발생하였을 때 유발되는 페이지폴트의 주소정보와 데이터 정보를 추적하는 페이지폴트 추적 모듈, 페이지폴트가 유발된 주소와 데이터 정보를 인 메모리 버퍼에 복사하고 추적 및 관리하는 추적 데이터 관리 모듈 및 상기 추적 데이터 관리 모듈에서 추적된 메모리를 인 메모리 버퍼에 복사하고, 사용자 개입 또는 버퍼가 가득 찼을 때 버퍼의 데이터를 비휘발성 스토리지 영역에 파일 형태로 생성하는 추적 데이터 아카이빙 모듈을 포함한다.

(52) CPC특허분류
G06F 2009/45583 (2019.08)

이 발명을 지원한 국가연구개발사업

과제고유번호	1711146331
과제번호	2020R1A2C4002737
부처명	과학기술정보통신부
과제관리(전문)기관명	한국연구재단
연구사업명	개인기초연구(과기정통부)(R&D)
연구과제명	IoT 침해사고 대응을 위한 지능형 분석 플랫폼 기술 연구
기 여 율	1/1
과제수행기관명	세종대학교
연구기간	2021.03.01 ~ 2022.02.28

명 세 서

청구범위

청구항 1

복수의 서비스가 구동되는 클라우드 환경에서 구동 중인 서비스들의 메모리 레이아웃을 분석하고 그룹화하는 메모리 레이아웃 분석 모듈;

상기 분석된 메모리 레이아웃에 기초하여 메모리의 중복성 검사를 수행하고, 중복되는 메모리 영역을 하나의 메모리 영역으로 병합하는 커널 메모리 병합 모듈;

병합된 메모리 영역에서 변화되는 메모리를 추적하여 기록하고, 병합되지 않은 메모리 영역에 쓰기 연산을 방지할 수 있도록 메모리 영역을 보호하는 메모리 쓰기 방지 모듈;

상기 메모리 쓰기 방지모듈에 의해 보호된 메모리 영역에서 쓰기 연산이 발생하였을 때 유발되는 페이지폴트의 주소정보와 데이터 정보를 추적하는 페이지폴트 추적 모듈;

페이지폴트가 유발된 주소와 데이터 정보를 인 메모리 버퍼에 복사하고 추적 및 관리하는 추적 데이터 관리 모듈; 및

상기 추적 데이터 관리 모듈에서 추적된 메모리를 인 메모리 버퍼에 복사하고, 사용자 개입 또는 버퍼가 가득 찼을 때 버퍼의 데이터를 비휘발성 스토리지 영역에 파일 형태로 생성하는 추적 데이터 아카이빙 모듈

을 포함하는 인스턴스 메모리 기록 장치.

청구항 2

제1항에 있어서,

상기 메모리 레이아웃 분석 모듈은,

클라우드 환경에서 구동 중인 VM(Virtual Machine) 또는 컨테이너들을 조회하고, 구동 중인 서비스들 중 동일한 이미지를 사용하는 서비스를 그룹화하며, 복수의 서비스가 구동되는 클라우드 환경에서 구동 중인 서비스들의 구동에 필요한 이미지 정보를 추출하여 그룹화하는

인스턴스 메모리 기록 장치.

청구항 3

제1항에 있어서,

상기 커널 메모리 병합 모듈은,

그룹화된 메모리 영역들의 레이아웃을 분석하고, 메모리 레이아웃 분석을 통해 유사 커널 메모리 영역을 추출하며, 추출된 커널 메모리가 그룹 내 다른 서비스들과 일치하는지 여부를 판단하고, 일치하는 경우 해당 메모리 영역을 병합하며, 추출된 커널 메모리가 그룹 내 다른 서비스들과 일치하지 않는 경우, 커널 메모리 영역에 대한 레이아웃 분석 및 커널 메모리 일치 여부 판단 과정을 반복 수행하는

인스턴스 메모리 기록 장치.

청구항 4

제1항에 있어서,

상기 메모리 쓰기 방지 모듈은,

상기 커널 메모리 병합이 완료되면 상기 커널 메모리의 변화 정보를 추적하고자 하는 타겟 메모리 영역을 추출하고, 타겟 메모리 영역이 추출되면 타겟의 모든 메모리 영역에 대해 쓰기 권한을 제거하는

인스턴스 메모리 기록 장치.

청구항 5

제1항에 있어서,

상기 페이지폴트 추적 모듈은,

메모리 수집 대기 상태가 되면, 서비스 내 메모리 쓰기 연산의 발생 여부를 판단하고, 서비스 내 메모리 쓰기 연산에 따라 발생하는 페이지폴트의 주소 정보 및 데이터 정보를 추적하는

인스턴스 메모리 기록 장치.

청구항 6

제1항에 있어서,

상기 추적 데이터 관리 모듈은,

페이지폴트에 대하여 추적된 주소 정보 및 데이터 정보를 호스트의 인 메모리 영역에 복사하고, 인 메모리 버퍼가 가득 차거나 또는 사용자에 의해 인 메모리 버퍼를 비워 달라는 요청 이벤트가 발생하는 경우, 인 메모리 버퍼에 쓰인 데이터를 비휘발성 스토리지 영역에 파일 형태로 생성하도록 관리하는

인스턴스 메모리 기록 장치.

청구항 7

제1항에 있어서,

상기 추적 데이터 아카이빙 모듈은,

메모리의 전체 영역이 아닌 변화되는 메모리만을 추적하여 기록하기 위하여 메모리 분석을 통해 유의미한 데이터를 획득하고 메모리 복원 절차를 수행하며, 체크포인트를 활용하여 메모리 랜덤 엑세스(Random-Access) 복원 성능을 보장하고, 각 프레임 마다 추적된 메모리 정보를 활용하여, 메타데이터를 생성함으로써 메모리를 분석하고, 사용자 요청에 따라 체크포인트 이전의 프레임을 무손실 압축하는

인스턴스 메모리 기록 장치.

청구항 8

메모리 레이아웃 분석 모듈을 통해 복수의 서비스가 구동되는 클라우드 환경에서 구동 중인 서비스들의 구동에 필요한 이미지를 분석하는 단계;

메모리 레이아웃 분석 모듈을 통해 구동 중인 서비스들 중 동일한 이미지를 사용하는 서비스를 그룹화하는 단계;

커널 메모리 병합 모듈을 통해 상기 그룹화된 서비스에 대하여 그룹별로 중복되는 커널 메모리 영역을 분석하여 병합하는 단계;

메모리 쓰기 방지 모듈을 통해 상기 중복되는 커널 메모리 영역이 병합된 서비스가 사용하는 메모리 영역의 쓰기 제한을 제거하는 단계;

상기 메모리 영역에서 쓰기 연산이 발생하였을 때, 추적 데이터 관리 모듈을 통해 쓰기 연산에 대한 주소정보와 데이터를 인 메모리 버퍼에 복사하는 단계; 및

추적 데이터 아카이빙 모듈을 통해 상기 인 메모리 버퍼의 내용을 비휘발성 스토리지로 복사하는 단계

를 포함하는 인스턴스 메모리 기록 방법.

청구항 9

제8항에 있어서,

상기 메모리 레이아웃 분석 모듈을 통해 복수의 서비스가 구동되는 클라우드 환경에서 구동 중인 서비스들의 구동에 필요한 이미지를 분석하는 단계는,

클라우드 환경에서 구동 중인 VM(Virtual Machine) 또는 컨테이너들을 조회하고, 구동 중인 서비스들 중 동일한 이미지를 사용하는 서비스를 그룹화하며, 복수의 서비스가 구동되는 클라우드 환경에서 구동 중인 서비스들의 구동에 필요한 이미지 정보를 추출하는

인스턴스 메모리 기록 방법.

청구항 10

제8항에 있어서,

상기 커널 메모리 병합 모듈을 통해 상기 그룹화된 서비스에 대하여 그룹별로 중복되는 커널 메모리 영역을 분석하여 병합하는 단계는,

그룹화된 메모리 영역들의 레이아웃을 분석하고, 메모리 레이아웃 분석을 통해 유사 커널 메모리 영역을 추출하며, 추출된 커널 메모리가 그룹 내 다른 서비스들과 일치하는지 여부를 판단하고, 일치하는 경우 해당 메모리 영역을 병합하며, 추출된 커널 메모리가 그룹 내 다른 서비스들과 일치하지 않는 경우, 커널 메모리 영역에 대한 레이아웃 분석 및 커널 메모리 일치 여부 판단 과정을 반복 수행하는

인스턴스 메모리 기록 방법.

청구항 11

제8항에 있어서,

상기 메모리 쓰기 방지 모듈을 통해 상기 중복되는 커널 메모리 영역이 병합된 서비스가 사용하는 메모리 영역의 쓰기 제한을 제거하는 단계는,

상기 커널 메모리 영역의 병합이 완료되면 상기 커널 메모리의 변화 정보를 추적하고자 하는 타겟 메모리 영역을 추출하고, 타겟 메모리 영역이 추출되면 타겟의 모든 메모리 영역에 대해 쓰기 권한을 제거하는

인스턴스 메모리 기록 방법.

청구항 12

제8항에 있어서,

상기 메모리 영역에서 쓰기 연산이 발생하였을 때, 추적 데이터 관리 모듈을 통해 쓰기 연산에 대한 주소정보와 데이터를 인 메모리 버퍼에 복사하는 단계는,

메모리 수집 대기 상태가 되면, 서비스 내 메모리 쓰기 연산의 발생 여부를 판단하고, 서비스 내 메모리 쓰기 연산에 따라 발생하는 페이지폴트의 주소 정보 및 데이터 정보를 추적하는

인스턴스 메모리 기록 방법.

청구항 13

제8항에 있어서,

상기 메모리 영역에서 쓰기 연산이 발생하였을 때, 추적 데이터 관리 모듈을 통해 쓰기 연산에 대한 주소정보와 데이터를 인 메모리 버퍼에 복사하는 단계는,

페이지폴트에 대하여 추적된 주소 정보 및 데이터 정보를 호스트의 인 메모리 영역에 복사하고, 인 메모리 버퍼가 가득 차거나 또는 사용자에 의해 인 메모리 버퍼를 비워 달라는 요청 이벤트가 발생하는 경우, 인 메모리 버퍼에 쓰인 데이터를 비휘발성 스토리지 영역에 파일 형태로 생성하도록 관리하는

인스턴스 메모리 기록 방법.

청구항 14

제8항에 있어서,

상기 추적 데이터 아카이빙 모듈을 통해 상기 인 메모리 버퍼의 내용을 비휘발성 스토리지로 복사하는 단계는,

메모리의 전체 영역이 아닌 변화되는 메모리만을 추적하여 기록하기 위하여 메모리 분석을 통해 유의미한 데이

터를 획득하고 메모리 복원 절차를 수행하며, 체크포인트를 활용하여 메모리 랜덤 엑세스(Random-Access) 복원 성능을 보장하고, 각 프레임 마다 추적된 메모리 정보를 활용하여, 메타데이터를 생성함으로써 메모리를 분석하고, 사용자 요청에 따라 체크포인트 이전의 프레임을 무손실 압축하는

인스턴스 메모리 기록 방법.

발명의 설명

기 술 분 야

[0001] 본 발명은 가상화 환경에서 인스턴스의 메모리를 연속적으로 기록하는 방법 및 장치에 관한 것이다.

배 경 기 술

[0002] 클라우드(Cloud) 컴퓨팅 이란, 시간과 장소에 구애 받지 않고 요구하는 성능에 상응하는 요금을 지불하면, 인터넷을 통해 컨테이너 및 가상 머신 형태의 컴퓨팅 자원을 제공 받을 수 있는 기술을 의미한다. 최근 들어, 가상화 기술이 발전함에 따라 클라우드 컴퓨팅 기술의 성능도 함께 발전하고 있으며, 클라우드 컴퓨팅에 대한 수요 및 시장 규모는 빠르게 증가하고 있다. 클라우드 컴퓨팅 기술은 기존에 직접 서비스 제공을 위한 하드웨어 장비를 구매하는 방식인 레거시(Legacy) 시스템과 비교하여 하드웨어 유지 보수 비용을 아낄 수 있다는 장점이 있으며, 자원 사용량에 따라 유동적으로 가상 자원을 요청하여 사용할 수 있어, 컴퓨팅 자원 사용 측면의 비용 절감 효과가 있다.

[0003] 클라우드 컴퓨팅 환경에서 자원의 활용성을 극대화하기 위해 방안으로 자원 관련 기술과 서비스 응용 기술이 많이 등장하였지만, 구동되는 서비스에 대한 보안성을 강화하거나 사고가 발생하였을 대응할 수 있는 기반기술이 마련되어 있지 않다. 따라서 클라우드 컴퓨팅 환경에서 서비스 환경에서 사이버 공격(예를 들어, 악성코드 감염, DDoS, 기밀 유출) 등이 발생하였을 때, 사고의 원인을 파악하는 데 많은 어려움을 겪는다. 사고의 원인을 파악하기 위해서는 포렌식이 필요하지만, 클라우드 컴퓨팅 특성상 물리적인 서버에 접근하여 포렌식 데이터를 추출하는 것이 매우 까다롭다. 게다가 진화된 최신 사이버 공격의 경우 메모리에서만 동작하기 때문에, 악의적인 사용자는 재부팅 또는 메모리 덮어쓰기를 통해 쉽게 악의적인 흔적을 제거할 수 있다.

[0004] 즉, 클라우드 컴퓨팅 환경에서 발생할 수 있는 사이버 공격에 대응하기 위해서는 물리적인 접근 없이 클라우드 컴퓨팅 환경에서 구동되는 서비스의 모든 행위정보를 효율적으로 비휘발성 스토리지에 기록할 수 있는 기술이 필요하다.

선행기술문헌

비특허문헌

[0005] (비특허문헌 0001) [1] N. Gruschka, M. Jensen, Attack surfaces: A taxonomy for attacks on cloudservices, in: 2010 IEEE 3rd international conference on cloud computing,IEEE, 2010, pp. 276-279.

(비특허문헌 0002) [2] A. Singh, D. M. Shrivastava, Overview of attacks on cloud computing,International Journal of Engineering and Innovative Technology (IJEIT) 1 (4).

(비특허문헌 0003) [3] F. Liu, Q. Ge, Y. Yarom, F. Mckeen, C. Rozas, G. Heiser, R. B. Lee, Catalyst: Defeating last-level cache side channel attacks in cloud computing,in: 2016 IEEE international symposium on high performance computer architecture (HPCA), IEEE, 2016, pp. 406-418.

발명의 내용

해결하려는 과제

[0006] 본 발명이 이루고자 하는 기술적 과제는 클라우드 플랫폼 환경에서 구동되는 가상머신의 메모리를 최소한의 연산으로 빠르게 수집할 수 있는 가상화 환경에서 인스턴스의 메모리를 연속적으로 기록하는 방법 및 장치를 제공하는데 있다. 본 발명을 통해 하이퍼바이저(Kernel-based Virtual Machine; KVM) 환경에서 구동되는 가상머신이나 도커 엔진 기반의 컨테이너 메모리를 초저지연 수집할 수 있고, 수집 대상이 증가함에도 지연 및 오버헤드의

증가 폭이 현저히 낮아 병렬적인 메모리 수집이 가능하다.

과제의 해결 수단

[0007] 일 측면에 있어서, 본 발명에서 제안하는 가상화 환경에서 인스턴스의 메모리를 연속적으로 기록하는 장치는 복수의 서비스가 구동되는 클라우드 환경에서 구동 중인 서비스들의 메모리 레이아웃을 분석하고 그룹화하는 메모리 레이아웃 분석 모듈, 상기 분석된 메모리 레이아웃에 기초하여 메모리의 중복성 검사를 수행하고, 중복되는 메모리 영역을 하나의 메모리 영역으로 병합하는 커널 메모리 병합 모듈, 병합된 메모리 영역에서 변화되는 메모리를 추적하여 기록하고, 병합되지 않은 메모리 영역에 쓰기 연산을 방지할 수 있도록 메모리 영역을 보호하는 메모리 쓰기 방지 모듈, 상기 메모리 쓰기 방지모듈에 의해 보호된 메모리 영역에서 쓰기 연산이 발생하였을 때 유발되는 페이지폴트의 주소정보와 데이터 정보를 추적하는 페이지폴트 추적 모듈, 페이지폴트가 유발된 주소와 데이터 정보를 인 메모리 버퍼에 복사하고 추적 및 관리하는 추적 데이터 관리 모듈 및 상기 추적 데이터 관리 모듈에서 추적된 메모리를 인 메모리 버퍼에 복사하고, 사용자 개입 또는 버퍼가 가득 찼을 때 버퍼의 데이터를 비휘발성 스토리지 영역에 파일 형태로 생성하는 추적 데이터 아카이빙 모듈을 포함한다.

[0008] 상기 메모리 레이아웃 분석 모듈은 클라우드 환경에서 구동 중인 VM(Virtual Machine) 또는 컨테이너들을 조회하고, 구동 중인 서비스들 중 동일한 이미지를 사용하는 서비스를 그룹화하며, 복수의 서비스가 구동되는 클라우드 환경에서 구동 중인 서비스들의 구동에 필요한 이미지 정보를 추출하여 그룹화한다.

[0009] 상기 커널 메모리 병합 모듈은 그룹화된 메모리 영역들의 레이아웃을 분석하고, 메모리 레이아웃 분석을 통해 유사 커널 메모리 영역을 추출하며, 추출된 커널 메모리가 그룹 내 다른 서비스들과 일치하는지 여부를 판단하고, 일치하는 경우 해당 메모리 영역을 병합하며, 추출된 커널 메모리가 그룹 내 다른 서비스들과 일치하지 않는 경우, 커널 메모리 영역에 대한 레이아웃 분석 및 커널 메모리 일치 여부 판단 과정을 반복 수행한다.

[0010] 상기 메모리 쓰기 방지 모듈은 상기 커널 메모리 병합이 완료되면 상기 커널 메모리의 변화 정보를 추적하고자 하는 타겟 메모리 영역을 추출하고, 타겟 메모리 영역이 추출되면 타겟의 모든 메모리 영역에 대해 쓰기 권한을 제거한다.

[0011] 상기 페이지폴트 추적 모듈은 메모리 수집 대기 상태가 되면, 서비스 내 메모리 쓰기 연산의 발생 여부를 판단하고, 서비스 내 메모리 쓰기 연산에 따라 발생하는 페이지폴트의 주소 정보 및 데이터 정보를 추적한다.

[0012] 상기 추적 데이터 관리 모듈은 페이지폴트에 대하여 추적된 주소 정보 및 데이터 정보를 호스트의 인 메모리 영역에 복사하고, 인 메모리 버퍼가 가득 차거나 또는 사용자에 의해 인 메모리 버퍼를 비워 달라는 요청 이벤트가 발생하는 경우, 인 메모리 버퍼에 쓰인 데이터를 비휘발성 스토리지 영역에 파일 형태로 생성하도록 관리한다.

[0013] 상기 추적 데이터 아카이빙 모듈은 메모리의 전체 영역이 아닌 변화되는 메모리만을 추적하여 기록하기 위하여 메모리 분석을 통해 유의미한 데이터를 획득하고 메모리 복원 절차를 수행하며, 체크포인트를 활용하여 메모리 랜덤 엑세스(Random-Access) 복원 성능을 보장하고, 각 프레임 마다 추적된 메모리 정보를 활용하여, 메타데이터를 생성함으로써 메모리를 분석하고, 사용자 요청에 따라 체크포인트 이전의 프레임을 무손실 압축한다.

[0014] 또 다른 일 측면에 있어서, 본 발명에서 제안하는 가상화 환경에서 인스턴스의 메모리를 연속적으로 기록하는 방법은 복수의 서비스가 구동되는 클라우드 환경에서 구동 중인 서비스들의 구동에 필요한 이미지를 분석하는 단계, 구동 중인 서비스들 중 동일한 이미지를 사용하는 서비스를 그룹화하는 단계, 상기 그룹화된 서비스에 대하여 그룹별로 중복되는 커널 메모리 영역을 분석하여 병합하는 단계, 상기 중복되는 커널 메모리 영역이 병합된 서비스가 사용하는 메모리 영역의 쓰기 제한을 제거하는 단계, 상기 서비스 영역에서 쓰기 연산이 발생하였을 때, 쓰기 연산에 대한 주소정보와 데이터를 인 메모리 버퍼에 복사하는 단계 및 상기 인 메모리 버퍼의 내용을 비휘발성 스토리지로 복사하는 단계를 포함한다.

발명의 효과

[0015] 본 발명의 실시예들에 따르면 가상화 환경에서 인스턴스의 메모리를 연속적으로 기록하는 방법 및 장치를 통해 클라우드 컴퓨팅 환경에서 구동 중인 모든 서비스의 중복되는 메모리 데이터를 효율적으로 제거하고, 서비스별 메모리 변화 정보만을 추적하고 기록하여 저장 및 관리를 수행하기 때문에 단일 서비스뿐만 아니라 다수의 서비스가 구동되는 환경에서도 효율적인 메모리 수집이 가능하다. 또한, 수집된 메모리 데이터는 타임라인에 따라 저장되기 때문에 사용자는 특정 시점에 구동 중이었던 모든 서비스를 연계 분석할 수 있을 뿐만 아니라, 시간에

변화에 따른 서비스의 행위정보를 상세하게 추적 및 분석할 수 있는 효과가 있다.

도면의 간단한 설명

[0016] 도 1은 본 발명의 일 실시예에 따른 저지연 메모리 기록 시스템이 적용되는 복수의 서비스가 구동되는 클라우드 환경을 나타내는 도면이다.

도 2는 본 발명의 일 실시예에 따른 가상화 환경에서의 저지연 메모리 기록 하는 Cloud-BlackBox의 개념을 설명하기 위한 도면이다.

도 3은 본 발명의 일 실시예에 따른 가상화 환경에서의 저지연 메모리 기록 장치의 구성을 나타내는 도면이다.

도 4는 본 발명의 일 실시예에 따른 가상화 환경에서의 저지연 메모리 기록 방법을 설명하기 위한 흐름도이다.

도 5는 본 발명의 일 실시예에 따른 클라우드 블랙박스(Cloud-BlackBox)의 전체 동작 과정을 설명하기 위한 도면이다.

도 6은 본 발명의 일 실시예에 따른 가상화 환경에서의 저지연 메모리 기록 장치가 구동 중인 인스턴스의 메모리를 병합하는 과정을 설명하기 위한 도면이다.

도 7은 본 발명의 일 실시예에 따른 가상화 환경에서의 저지연 메모리 기록 장치에서 인스턴스의 메모리 변화를 추적하여 기록하는 병합하는 과정을 설명하기 위한 도면이다.

도 8은 본 발명의 일 실시예에 따른 VM 메모리를 추적 및 기록하는 과정을 설명하기 위한 도면이다.

도 9는 본 발명의 일 실시예에 따른 메모리 복원 과정을 설명하기 위한 도면이다.

도 10은 본 발명의 일 실시예에 따른 효율적인 대규모 메모리 분석을 위한 SAMI 메타데이터를 설명하기 위한 도면이다.

발명을 실시하기 위한 구체적인 내용

[0017] 본 발명은 클라우드 플랫폼 환경에서 구동 중인 인스턴스의 메모리를 효율적으로 기록할 수 있는 방법 및 장치에 관한 것으로서, 더욱 상세하게는 가상화 환경에서 구동되는 인스턴스(다시 말해, 프로세스)의 메모리를 수집하는 과정에서 발생하는 메모리 데이터의 중복성을 최소화할 수 있는 두 가지의 방법을 통해 메모리 수집 연산의 속도를 가속하고, 불필요하게 낭비될 수 있는 스토리지 공간을 확보할 수 있는 시스템에 관한 것이다.

[0018] 기존의 클라우드 플랫폼에서 구동 중인 서비스(Virtual Machine; VM)의 행위정보를 기록하기 위해서는 메모리에 대한 접근이 필요하다. 메모리는 휘발성 영역으로 운영체제에서 발생할 수 있는 행위에 대한 데이터가 일시적으로 저장되는 공간이다. 따라서 사이버 공격이 발생한 시점의 메모리를 획득할 수 있다면 메모리 데이터 분석을 통해 공격의 원인을 파악할 수 있을 뿐만 아니라, 법적 증거물로도 활용될 수 있다. 하지만 메모리를 수집하는 연산은 많은 연산 오버헤드를 발생시키기 때문에 서비스가 구동되는 환경에서 메모리를 실시간으로 수집하는 것은 매우 어렵다.

[0019] 따라서 본 발명에서는 클라우드 플랫폼 환경에서 구동되는 가상머신의 메모리를 최소한의 연산으로 빠르게 수집할 수 있는 실시간 메모리 기록 방법 및 장치를 제공하고자 한다.

[0020] 본 발명을 통해 하이퍼바이저(Kernel-based Virtual Machine; KVM) 환경에서 구동되는 가상머신이나, 도커 엔진 기반의 컨테이너 메모리를 초저지연 수집하는 것이 가능하며, 수집 대상이 증가함에도 지연 및 오버헤드의 증가폭이 현저히 낮아 병렬적인 메모리 수집이 가능하다. 따라서 추후 본 발명을 활용할 수 있는 분야에는 악성코드 분석, 안티바이러스, 포렌식, AI 학습을 위한 대규모 데이터 확보 분야 등이 있다. 이하, 본 발명의 실시 예를 첨부된 도면을 참조하여 상세하게 설명한다.

[0022] 본 발명의 실시예에 따른 용어 클라우드(또는 클라우드 환경, 클라우드 컴퓨팅)에 있어서, 클라우드란 구름(Cloud)와 같이 무형의 형태로 존재하는 하드웨어, 소프트웨어 등의 컴퓨팅 자원을 자신이 필요한 만큼 빌려 쓰고 이에 대한 사용요금을 지급하는 방식의 컴퓨팅 서비스로, 서로 다른 물리적인 위치에 존재하는 컴퓨팅 자원을 가상화 기술로 통합해 제공하는 환경을 말한다.

[0023] 본 발명의 실시예에 따른 용어 메모리 병합에 있어서, 커널 동일 페이지 병합은 하이퍼 바이저 시스템이 여러 프로세스간에 동일한 내용을 가진 메모리 페이지를 공유할 수 있도록 하는 커널 기능을 말한다. 하지만 메모리

를 병합에서 대상을 전체 영역으로 지정하면 병합하는데 많은 시간이 소요되는 단점이 있다.

[0024] 본 발명의 실시예에 따른 용어 포렌식에 있어서, 컴퓨터 포렌식(computer forensics, 컴퓨터 법의학) 또는 디지털 포렌식은 전자적 증거물 등을 사법기관에 제출하기 위해 데이터를 수집, 분석, 보고서를 작성하는 일련의 작업을 말한다. 과거에 얻을 수 없었던 증거나 단서들을 제공해주고, 컴퓨터 포렌식은 사이버 해킹 공격, 사이버 범죄시 범죄자들은 컴퓨터, 이메일, IT 기기, 스마트폰 등의 운영체제, 애플리케이션, 메모리 등에 다양한 전자적 증거를 남기게 되면서, 사이버 범죄자 추적 및 조사에 핵심적인 요소가 되고 있다. 특히 최근에 메모리 포렌식은 침해사고 분석과 악성코드 분석 등 여러 분야에서 활용되고 있다.

[0025] 본 발명의 실시예에 따른 용어 메모리 포렌식은 디지털 포렌식에서의 사고 대응(Incident Response) 방안 중에 하나로 휘발성의 특징을 가지고 있는 메모리에 대해서 정보를 수집하여 의미 있는 증거 및 데이터를 도출하는 기술을 의미한다.

[0026] 본 발명의 실시예에 따른 용어 VMI(Virtual Machine Introspection)는 가상머신의 내부 메모리에 읽기 또는 쓰기를 위한 기술을 의미한다. 추가적으로 CPU 레지스터에 접근하는 것도 가능하며 가상머신의 정지 또는 재구동, 바이너리 데이터 출력 등의 가상화 내부에 대해 매우 깊은 수준의 탐색 및 제어가 가능하다. 본 발명에서 구동중인 프로세스의 PID(Process ID) 정보 등을 가져오는데 사용할 수 있다.

[0027] 본 발명의 실시예에 따른 용어 페이지 폴트는 프로그램이 자신의 주소 공간에는 존재하지만, 시스템의 RAM에는 현재 없는 데이터나 코드에 접근 시도하였을 경우 발생하는 현상을 말한다.

[0029] 도 1은 본 발명의 일 실시예에 따른 저지연 메모리 기록 시스템이 적용되는 복수의 서비스가 구동되는 클라우드 환경을 나타내는 도면이다.

[0030] 본 발명의 실시 예에 따른 저지연 메모리 기록 기술은 원격에 존재하는 클라우드 플랫폼(110)과 클라우드 플랫폼 서비스를 관리하는 관리자 또는 서비스를 이용하는 사용자(120)가 존재하는 환경에서 적용된다.

[0031] 클라우드 컴퓨팅 인프라는 성장단계를 넘어서 성숙단계에 접어들고 있다. 클라우드 인프라 사이버 위협에 대응하기 위해 클라우드 서비스의 안정성과 보안성 강화를 위한 다양한 접근들이 시도되고 있다. 특히 클라우드 인프라에서 발생하는 보안사고 및 서비스 에러에 대한 사고 원인을 파악하고 이를 해결하기 위한 수단으로 클라우드 서비스의 시스템 메모리를 활용하려는 시도들이 큰 관심을 받고 있다.

[0032] 시스템 메모리는 운영체제 내의 대부분의 행위 정보가 저장되는 휘발성 저장소이다. 따라서, 메모리 분석을 통해 프로세스, 커널 드라이버, 활성화 네트워크 정보, 레지스트리 정보, 사용자 활동 등 운영체제 시스템의 거시적인 뷰와 미시적인 뷰를 투명하게 모니터링할 수 있다. 특히 클라우드 인프라와 같이 하나의 호스트 운영체제에서 다수의 서비스들이 구동되는 환경(예를 들어, IaaS, PaaS, SaaS)에서는 각 서비스 내부에 에이전트(Agent) 설치 없이, 호스트 운영체제 레벨에서 모든 서비스의 라이브 메모리 영역을 오염 없이 스냅샷할 수 있다는 장점이 있다. 이와 같은 장점으로 인해, 가상화 환경에서의 메모리 분석기술은 포렌식 분야, 악성코드 분석 분야, 안티바이러스 분야, 서비스 에러 복구 분야 등에서 활용되고 있다. 특히 비휘발성 데이터(예를 들어, System-logging, Storage Forensics, 등) 분석기반의 보안성 강화에 초점이 맞춰있었던 클라우드 분야에서도 최근 메모리에 관한 활발한 연구들이 진행되고 있다. 이와 같이 많은 기술들이 클라우드 시스템 메모리에 관심을 보이는 이유는 최신 공격기법들이 고도화되어 탐지가 까다로워졌기 때문이다. 고도화된 공격기법의 예시로는 스토리지에 흔적을 남기지 않는 File-less Attack(예를 들어, Angler Exploit, Poweliks, Kovter, Bootkit)이 있으며, 하나의 대상을 장시간에 걸쳐 집중적으로 공격하는 APT(Advanced Persistent Threat)등이 있다. 이와 같이 고도화된 공격들은 비휘발성 스토리지에 저장되는 데이터 분석 및 모니터링을 통해 탐지하는 데 한계가 있다. 또한, 메모리 분석은 침해사고 분석, 악성 행위 탐지에 활용될 수 있을 뿐만 아니라 서비스 장애 원인을 파악하는데 활용될 수 있다. 그 예로 Windows, Linux 운영체제에서는 시스템 크래쉬가 발생하면 활성화된 메모리 데이터를 비휘발성 영역에 남기는 것이다.

[0033] 시스템 메모리 분석기술은 보안 및 서비스 복구에 활용할 수 있다는 다양한 장점으로 인해 클라우드 컴퓨팅 인프라에 적용하여 활용하려는 연구들이 많이 있었지만, 실질적으로 COTS(Commercial Off-The-Shelf)에 적재되어 활용되지 못하고 있다. 활용되지 못하는 이유는 서비스 사고 원인에 대해 심층 분석하거나 서비스 보안 위협을 탐지하기 위해서는 구동되고 있는 다수의 VM의 메모리를 스냅샷 해야 하기 때문이다. 이를 위한 반복적인 메모리 스냅샷 연산은 서비스 사용자 관점에서는 VM 성능저하, VM Downtime 발생등의 성능 관련 오버헤드가 유발되며, 관리자 관점에서는 메모리 스냅샷 소요시간, 방대한 메모리 데이터 생성, 분석 대상의 폭발적인 증가 등의 문제가 유발된다. 즉, 대규모의 서비스가 구동되는 클라우드 컴퓨팅 인프라에서 메모리 수집 및 분석기술을 서

비스 안정성 및 보안성 강화를 위한 목적으로써 활용하기 위해서 반드시 해결해야 하는 문제는 크게 3가지로 정리할 수 있다:

[0034] 첫째, 메모리 수집 오버헤드로 인한 VM Downtime 발생 및 VM 시스템 성능 저하 문제이다. 클라우드 컴퓨팅 플랫폼에서 VM 메모리 스냅샷 행위는 연산적으로 많은 오버헤드를 발생시킨다. 복수의 VM이 구동되는 클라우드 플랫폼에서 병렬적으로 복수의 VM 메모리를 수집한다면 시스템 운영이 불가능한 수준의 연산 오버헤드가 유발될 것이다.

[0035] 둘째, 메모리 수집으로 인해 소비되는 스토리지 공간이다. 메모리는 비휘발성 저장장치와 비교하면 상대적으로 작은 크기를 갖는다. 따라서, 단 한 번의 메모리 수집은 비휘발성 스토리지 공간에서 비교적 작은 공간을 차지한다. 하지만 메모리를 장시간 누적하여 저장한다는 관점으로 접근한다면 엄청난 양의 스토리지 공간이 필요하다. 복수의 VM이 구동되는 클라우드 플랫폼에서 메모리를 누적하여 수집한다는 것은 엄청난 양의 데이터가 생성된다는 것을 의미한다.

[0036] 셋째, 분석대상(메모리)의 폭발적인 증가이다. 클라우드 플랫폼에서 다수의 VM의 메모리를 누적하여 스냅샷 한다고 하더라도, 즉시 수집된 메모리를 보안사고의 원인 및 해결을 하기 위한 요소로써 활용될 수 있는 것은 아니다. 유의미한 정보는 메모리 분석을 통해 찾아내야 하기 때문이다. 이미 빅데이터 분야에서도 대규모의 비정형 데이터를 분석하는 것은 매우 어려운 문제라는 것은 수없이 언급되었다. 즉 메모리를 단순히 누적하여 기록하는 것은 오히려 메모리 분석을 어렵게 만들기 때문에, 대규모 메모리 기록과 분석에 특화된 아카이빙이 필요하다.

[0038] 도 2는 본 발명의 일 실시예에 따른 가상화 환경에서의 저지연 메모리 기록 하는 Cloud-BlackBox의 개념을 설명하기 위한 도면이다.

[0039] 본 발명에서는 대규모의 클라우드 컴퓨팅(다시 말해, 가상화) 환경에서 VM들의 메모리를 연산 효율적으로 기록하고 재생하여 분석할 수 있는 Cloud-BlackBox를 제안한다. Cloud-BlackBox의 개념은 도 2와 같다.

[0040] 본 발명의 실시예에 따른 Cloud-BlackBox는 클라우드 플랫폼에서 구동중인 다수의 VM의 메모리 정보를 효율적으로 추적할 수 있다. 또한, 구동중인 VM들의 이미지 정보를 활용하여, 커널 메모리 영역(Global Memory-MDr)과 사용자 영역(Private Memory-MDp)으로 분리한다. 글로벌 메모리(Global Memory) 영역은 동일한 운영체제를 구동하는 VM들끼리 공유될 수 있도록 병합한다. 그리고, 추적된 메모리 정보와 글로벌 메모리(Global Memory) 병합 정보를 활용하여, 타겟 메모리를 효율적으로 분석할 수 있게 인코딩하여 스토리지에 저장한다. 분리된 메모리 영역을 분석하기 위해서는 MDr 과 MDp 의 XOR 연산을 통한 메모리 복원이 필요하다. 메모리 복원 시간을 보장하기 위해서 체크 포인트(Checkpoint)(MDc)를 활용한다.

[0042] 도 3은 본 발명의 일 실시예에 따른 가상화 환경에서의 저지연 메모리 기록 장치의 구성을 나타내는 도면이다.

[0043] 도 3을 참조하면, 본 발명의 실시 예에 따른 가상화 환경에서의 저지연 메모리 기록 장치는 구동 중인 서비스들의 메모리 전처리를 수행하는 메모리 전처리부(310) 및 변화되는 메모리를 추적하여 기록하기 위한 메모리 기록부(320)를 포함한다.

[0044] 본 발명의 실시예에 따른 메모리 전처리부(310)는 메모리 레이아웃 분석 모듈(311) 및 커널 메모리 병합 모듈(312)을 포함한다.

[0045] 본 발명의 실시예에 따른 메모리 레이아웃 분석 모듈(311)은 복수의 서비스가 구동되는 환경에서 서비스들의 메모리 레이아웃을 분석하고 그룹화하는데 사용된다. 다시 말해, 구동 중인 서비스들의 메모리 전처리를 위해 서비스의 메모리를 분석한다.

[0046] 종래기술에서 널리 사용되고 있는 메모리 분석 프레임워크(Volatilty, Rekall) 들은 단일 메모리에 대한 분석에 초점이 맞춰져 설계되었다. 하지만, 클라우드 환경에서 다수의 VM을 메모리를 기록한다는 것은 분석해야 할 메모리가 대규모로 생성한다는 것을 의미한다. 따라서 현재의 메모리 프레임워크를 통해 기록된 대규모의 메모리를 분석한다는 것은 비효율적이다. 게다가 메모리 분석의 의도, 목적, 특성들을 고려하지 않고 반복적으로 메모리를 수집할 경우 의미 있는 분석결과를 도출하는 것은 매우 어렵다.

[0047] 본 발명에서는 메모리 분석의 의도, 목적, 메모리의 특성을 고려하여 대규모의 메모리를 효율적으로 분석할 수 있는 메모리 코덱이라는 메모리 아카이빙 메커니즘을 제안한다. 메모리 코덱은 메모리 추적을 통해 생성된 데이터를 시간적 특성에 따른 메모리 변화정보(프로세스, 라이브러리, Network, 등), 공간적 특성에 따른 내부(Inter)-VM 연계분석 등에 활용한다. 또한, 체크포인트를 활용하여 기록된 부분 메모리 데이터를 분석이 가능한

온전한 메모리 상태로 빠르게 복원해준다. 마지막으로, 메모리 코덱을 통해 기록된 메모리는 기존의 메모리 포렌식 프레임워크를 활용하여 분석할 수 있는 확장성을 제공하기 때문에 다양한 플러그인(예를 들어, apihooks, malfind, yarascan 등)을 활용할 수 있다는 장점이 있다.

[0048] 본 발명의 실시예에 따른 커널 메모리 병합 모듈(312)은 분석된 레이아웃을 바탕으로 메모리의 중복성 검사를 수행하여 중복되는 메모리 영역을 하나의 메모리 영역으로 병합한다.

[0049] 본 발명의 실시예에 따르면, 메모리 기록 시 소모되는 스토리지 공간을 최소화하기 위해 내부(Inter)-VM 커널 메모리 병합 후 아카이빙 하는 메커니즘을 제안하였다. 본 발명에서는 클라우드라는 특수한 환경의 특징을 적극적으로 활용하여 일부 데이터만 저장하는 방법으로 수집 데이터를 최소화하였다. 다수의 VM이 구동되는 클라우드 플랫폼의 특성상 비슷한 OS의 이미지가 구동된다. AWS(Amazon Web Services)에서 구동 중인 VM의 OS는 90%가 Linux 계열이고, Azure에서 구동 중인 OS는 40%가 Linux 계열이다.

[0050] 본 발명의 실시예에 따른 KVM(Kernel-based Virtual Machine) 환경에서 2GB를 할당 받는 Ubuntu 18.04(64bit) VM을 다수 구동시키고, Inter-VM 간의 중복되는 메모리 영역을 검증해본 결과 30%의 메모리 영역이 중복된다는 것을 확인하였다. 본 발명에서는 클라우드 플랫폼의 특성을 고려하여 동일한 OS 이미지를 사용하는 VM을 그룹화하고, 중복되는 메모리 영역(다시 말해, 커널)을 글로벌 페이지(Global Page)로 관리함으로써, 대상 메모리 범위를 축소하는 글로벌 메모리 병합 메커니즘을 제안한다. 또한 글로벌 메모리 병합 메커니즘은 OS 별 메모리 레이아웃 유사도를 통해 다수의 VM의 메모리를 매우 빠르게 스캐닝할 수 있는 특징을 갖고 있다. 메모리 트레킹 메커니즘과 글로벌 메모리 병합 메커니즘을 혼합하여 사용함으로써 메모리 기록 시 발생하는 스토리지 공간을 12.85배 이상 절감할 수 있다.

[0051] 본 발명의 실시예에 따른 메모리 기록부(320)는 메모리 쓰기 방지 모듈(321), 페이지폴트 추적 모듈(322), 추적 데이터 관리 모듈(323), 추적 데이터 아카이빙 모듈(324)을 포함한다.

[0052] 본 발명의 실시예에 따른 메모리 쓰기 방지 모듈(321)은 변화되는 메모리를 추적하여 기록한다. 병합되지 않은 메모리 영역에 쓰기 연산을 방지할 수 있도록 메모리 영역을 보호한다.

[0053] 본 발명의 실시예에 따른 페이지폴트 추적 모듈(322)은 쓰기 방지모듈에 의해 보호된 메모리 영역에서 쓰기 연산이 발생하였을 때 페이지폴트가 유발되는데 이때 페이지폴트를 유발한 주소정보와 데이터 정보를 추적한다.

[0054] 본 발명의 실시예에 따른 추적 데이터 관리 모듈(323)은 페이지폴트가 유발된 주소와 데이터 정보를 인 메모리 버퍼에 복사하고 관리한다.

[0055] 본 발명의 실시예에 따른 추적 데이터 아카이빙 모듈(324)은 추적된 메모리를 인 메모리 버퍼에 복사하고, 사용자 개입 또는 버퍼가 가득 찼을 때 버퍼의 데이터를 비휘발성 스토리지 영역에 파일 형태로 생성한다.

[0056] 메모리를 기록할 때 발생할 수 있는 첫 번째 문제점은 메모리 수집 오버헤드로 인한 시스템 성능 저하 문제이다. 본 발명의 실시예에 따른 메모리 스냅샷 과정에서 성능 저하의 보틀넥 포인트(Bottleneck Point)(blocking)를 찾기 위해 메모리 수집의 모든 과정을 심층적으로 분석하였다. 그 결과 메모리 데이터를 스토리지 영역으로 복사하는 과정인 File I/O 연산이 메모리 수집 전체연산 과정에서 93% 이상을 차지한다는 것을 알 수 있었다.

[0057] 따라서, 본 발명에서는 메모리 수집 시 발생하는 File I/O 연산 최적화를 위한 방안으로 메모리 트레킹 메커니즘을 제안하였다. 메모리 트레킹은 VM이 사용하는 모든 메모리 영역에서 변화되는 페이지 정보를 추적하는 메커니즘이다. 본 발명의 실시예에서 4GB를 할당 받은 단일 Native VM에 대해 1,000ms 간격으로 메모리 수집해본 결과 약 96.7%의 데이터가 중복된다는 사실 알 수 있었다. VM 내에서 서비스가 구동된다고 하여도 메모리의 일부 영역만 변경된다는 사실을 실험을 통해 검증하였기 때문에 인지 스케일 비트맵(Cognitive-Scale Bitmap)을 통해 메모리의 변화되는 메모리만 추적하여 캐싱하고(다시 말해, 인 메모리 영역에 복사), 캐시 데이터를 디스크에 복제시키는 방법으로 디스크 I/O를 최소화하였다. 결론적으로 블로킹(Blocking) 연산으로 발생하는 메모리 수집 시간은 14.85배 감소시켰으며, 메모리 수집 시 발생하는 VM 일시정지문제는 서비스 구동 시 최초 1회에만 심리스(Seam-less) 하게 발생한다.

[0059] 도 4는 본 발명의 일 실시예에 따른 가상화 환경에서의 저지연 메모리 기록 방법을 설명하기 위한 흐름도이다.

[0060] 제안하는 가상화 환경에서의 저지연 메모리 기록 방법은 복수의 서비스가 구동되는 클라우드 환경에서 구동 중인 서비스들의 구동에 필요한 이미지를 분석하는 단계(410), 구동 중인 서비스들 중 동일한 이미지를 사용하는 서비스를 그룹화하는 단계(420), 상기 그룹화된 서비스에 대하여 그룹별로 중복되는 커널 메모리 영역을 분석하

여 병합하는 단계(430), 상기 중복되는 커널 메모리 영역이 병합된 서비스가 사용하는 메모리 영역의 쓰기 제한을 제거하는 단계(440), 상기 서비스 영역에서 쓰기 연산이 발생하였을 때, 쓰기 연산에 대한 주소정보와 데이터를 인 메모리 버퍼에 복사하는 단계(450) 및 상기 인 메모리 버퍼의 내용을 비휘발성 스토리지로 복사하는 단계(460)를 포함한다.

[0061] 단계(410)에서, 복수의 서비스가 구동되는 클라우드 환경에서 구동 중인 서비스들의 구동에 필요한 이미지를 분석한다. 먼저 현재 클라우드 플랫폼(예를 들어, 리눅스 운영체제)에서 VMI와 같은 가상화 관리 모듈을 통해 구동 중인 VM 또는 컨테이너들을 조회한다.

[0062] 단계(420)에서, 구동 중인 서비스들 중 동일한 이미지를 사용하는 서비스를 그룹화한다. 복수의 서비스가 구동되는 클라우드 환경에서 구동 중인 서비스들의 구동에 필요한 이미지인 Base-Image 정보를 추출한다.

[0063] 이때 Base-Image는 VMI를 활용하여 추출할 수 있으며, 이미지 정보가 제공되지 않을 때는 서비스의 커널 메모리 영역을 분석하면 Base-Image 정보 확인이 가능하다. 이후 동일한 Base-Image 정보 또는 커널 버전을 사용하는 서비스들을 그룹화한다.

[0064] 단계(430)에서, 상기 그룹화된 서비스에 대하여 그룹별로 중복되는 커널 메모리 영역을 분석하여 병합한다. 그룹화된 메모리 영역들의 레이아웃을 분석하고(431), 메모리 레이아웃 분석을 통해 유사 커널 메모리 영역을 추출한다(342). 이후 커널 메모리가 그룹 내 다른 서비스들과 일치하는지 여부를 판단하고(433), 일치한다면 해당 메모리 영역을 병합한다(434). 만약에 일치하지 않는다면 커널 메모리 영역에 대한 레이아웃 분석(431) 및 커널 메모리 일치 여부 판단(433)을 반복적으로 수행한다.

[0065] 단계(440)에서, 상기 중복되는 커널 메모리 영역이 병합된 서비스가 사용하는 메모리 영역의 쓰기 제한을 제거한다.

[0066] 메모리 병합이 완료되면 메모리의 변화 정보를 추적하고자 하는 타겟 메모리 영역을 추출한다(441). 타겟 메모리 영역이 추출되면 타겟의 모든 메모리 영역에 대해 쓰기 권한을 제거한다(442).

[0067] 단계(450)에서, 상기 서비스 영역에서 쓰기 연산이 발생하였을 때, 쓰기 연산에 대한 주소정보와 데이터를 인 메모리 버퍼에 복사한다.

[0068] 메모리 수집 대기 상태(451)에 진입하고, 서비스 내 메모리 쓰기 연산의 발생 여부를 판단(452)하여, 서비스 내 메모리 쓰기 연산이 발생한 경우 페이지폴트가 유발된다. 메모리 쓰기 연산이 발생하면 쓰기 연산에 대한 주소 및 데이터 정보를 추출한다(453).

[0069] 단계(460)에서, 상기 인 메모리 버퍼의 내용을 비휘발성 스토리지로 복사한다.

[0070] 추적된 주소 및 데이터를 호스트의 인 메모리 영역에 복사한다(461). 이후 인 메모리 버퍼가 가득 차거나, 사용자에 의해 인 메모리 버퍼를 비워 달라는 요청 이벤트가 발생하면(462), 인 메모리 버퍼에 쓰인 데이터를 비휘발성 스토리지 영역에 파일 형태로 생성한다(463).

[0072] 도 5는 본 발명의 일 실시예에 따른 클라우드 블랙박스(Cloud-BlackBox)의 전체 동작 과정을 설명하기 위한 도면이다.

[0073] 도 5를 참조하여, 클라우드 컴퓨팅(다시 말해, 가상화) 환경에서 연산 효율적으로 메모리를 기록하고 재생하여 분석할 수 있는 클라우드 블랙박스(Cloud-BlackBox)에 대해 설명한다. Cloud-BlackBox는 연산 효율적으로 다수의 VM의 메모리를 빠르게 기록할 수 있으며, 수집 시 발생하는 스토리지 비용이 최소화된다는 특징이 있다. 또한 대규모로 수집된 메모리 데이터 셋들에 대해 빠르게 심층분석 할 수 있는 특징이 있다.

[0074] 본 발명의 실시예에 따른 Cloud-BlackBox의 구조는 도 5와 같이 크게 4가지의 주요 구성요소(Global Memory Manager(510), Page Tracking Manager(520), Memory I/O Scheduler(530), M-Frame Manager(540))를 포함한다.

[0075] 본 발명의 실시예에 따른 Global Memory Manager(510)(도 3의 메모리 전처리부(310)에 해당)는 물리적 머신에서 구동 중인 VM들의 메모리 레이아웃을 교차 검증하여, 동일한 혹은 유사한 커널을 사용하는 VM을 그룹화하고, 중복되는 Kernel-Memory 영역을 병합하는 메커니즘이다. Global Memory Manager(510)는 물리머신에서 VM이 생성되거나 제거될 때 커널 메모리를 재 그룹 및 병합처리를 위해 호출되는 모듈이다.

[0076] 본 발명의 실시예에 따른 Page Tracking Manager(520)(도 3의 메모리 쓰기 방지 모듈(321) 및 페이지 폴트 추적 모듈(322)에 해당)는 VM의 변화하는 메모리를 추적하여 관리하기 위한 페이지 추적 메커니즘이다. Page

Tracking Manager(520)는 Global Memory Manager(510)를 통해 병합된 Global Page와 VM 고유의 메모리 영역인 보안 페이지(Private Page)의 메모리 변화정보를 추적하는 모듈이다. 메모리 변화정보를 체크하기 위해 Cognitive-Scale Bitmap(예를 들어, Macro Bitmap, Mircro Bitmap)을 활용한다. 본 발명에서는 2개의 비트맵(Bitmap)을 활용하였으며, 사용자 요구사항에 따라 2개 이상의 비트맵(Bitmap)을 활용할 수 있다. 해당 모듈은 사용자 혹은 다른 모듈로부터 VM의 메모리 기록 요청이 발생하면 수행된다.

[0077] 본 발명의 실시예에 따른 Memory I/O Scheduler(530)(도 3의 추적 데이터 관리 모듈(323)에 해당)는 휘발성 데이터(다시 말해, VM 메모리)를 비휘발성 영역(다시 말해, 영구적 스토리지)에 효율적으로 저장하기 위한 메커니즘이다. Page Tracking Manager(520)를 통해 추적된 페이지의 실제 메모리 데이터를 비동기식 접근 메모리 버퍼에 복제하고, 복제된 메모리 데이터는 백그라운드 스레드(Background thread)에 의해 스토리지에 플러싱(Flushing) 된다.

[0078] 본 발명의 실시예에 따른 M-Frame Manager(530)(도 3의 추적 데이터 아카이빙 모듈(324)에 해당)는 수집된 메모리를 효율적으로 관리 및 분석하기 위한 메커니즘이다. 본 발명에서는 수집된 하나의 메모리 파일을 M-Frame이라고 명명한다. 본 발명의 실시예에 따른 Cloud-BlackBox는 메모리의 전체 영역이 아닌 변화되는 메모리만을 추적하여 기록하기 때문에 메모리 분석 도구를 통해 유의미한 데이터를 획득하기 위해 반드시 메모리 복원 절차가 필요하다. M-Frame은 체크포인트를 활용해 메모리 랜덤 엑세스(Random-Access) 복원 성능을 보장해주며, 각 프레임 마다 추적 된 메모리 정보를 활용하여, 메타데이터(SMAI)를 생성해줌으로써 대규모의 메모리를 효율적으로 분석할 수 있게 해준다. 또한, 스토리지 비용 절감을 요구하는 사용자 요청에 따라 체크포인트 이전의 프레임을 무손실 압축해주는 기능을 제공한다.

[0080] 도 6은 본 발명의 일 실시예에 따른 가상화 환경에서의 저지연 메모리 기록 장치가 구동 중인 인스턴스의 메모리를 병합하는 과정을 설명하기 위한 도면이다.

[0081] 도 6을 참조하면, 발명의 실시 예에 따른 가상화 환경에서의 저지연 메모리 기록 시스템이 중복되는 메모리 영역을 제거하기 위해 메모리 전처리하는 과정은 다음과 같다.

[0082] 본 발명의 실시예에 따르면, 클라우드 컴퓨팅 환경에서 구동 중인 서비스들의 메모리 할당 정보를 기반으로 메모리 레이아웃 분석 모듈은 서비스가 할당받아 사용하고 있는 메모리 영역에서 커널 메모리 영역을 찾아내고 중복성을 검사한다. 서비스가 할당받은 메모리 영역 중 커널 메모리 영역이 중복되는 영역이 판별되면 커널 메모리 병합 모듈은 중복되는 커널 메모리에 대해 서비스들이 포인팅으로 접근할 수 있도록 공유 메모리 영역으로 만든다. 이를 통해 각 서비스의 전체 메모리 영역 중 일부는 공유 메모리로 관리 되기 때문에 메모리 기록 대상의 메모리 범위가 축소된다.

[0083] VM 메모리 스냅샷 시 가장 많은 오버헤드가 발생하는 요소는 메모리를 반복적으로 복제하는 연산이다. 반복되는 메모리의 연산 복잡도는 할당된 메모리 크기에 비례한다. 일반적으로 VM의 GB 단위 이상의 메모리를 할당시키기 때문에 메모리 스냅샷 시 수많은 메모리 탐색 및 반복적인 복제 연산이 필요하다. 다수의 VM이 구동되는 대규모 클라우드 플랫폼이라면 메모리 스냅샷 연산은 더더욱 큰 오버헤드를 유발할 것이다.

[0084] 본 발명의 실시예에 따른 Cloud-BlackBox는 메모리 스냅샷 시 유발되는 연산 오버헤드를 최적화하고, 스냅샷되어야 하는 메모리 범위를 최소화하기 위해 Inter-VM 메모리 병합에 집중하였다. 본 발명의 실험결과에 따르면 동일한 커널을 사용하는 Inter-VM은 약 30% 이상의 메모리가 중복된다는 것을 확인하였다. 이와 같은 특징은 다수의 VM이 구동되는 클라우드 플랫폼이라는 환경에서 가장 잘 활용될 수 있다. 클라우드 플랫폼은 대부분 Base-Image를 사용자에게 제공해주기 때문에 클라우드 플랫폼에서 구동될 수 있는 커널은 한정된다.

[0085] AWS에서 발표한 자료에 따르면 EC2를 사용하는 사용자들의 OS는 90%가 Linux 계열이라는 것을 알 수 있다. 즉 동일한 커널을 사용하는 VM들의 메모리를 그룹화하고, 메모리를 공유하도록 병합을 시키면 탐색 및 스냅샷 되어야 하는 메모리 범위를 축소시킬 수 있다. 하지만 병합되는 공유메모리 영역에 사용자의 보안(Private) 데이터가 올라갔을 때 다른 VM의 메모리에 접근할 수 있다는 보안 문제를 유발할 수 있다. 따라서 병합되어야 하는 메모리는 반드시 유저레벨에서 접근할 수 없는 커널 메모리(Kernel Memory) 영역이면서, 보안 데이터(Private Data)가 올라갈 수 없는 공간이어야 한다. 이와 같이 안전한 메모리 영역은 클라우드 환경에서 구동시킬 수 있는 커널 별로 메모리 분석과 반복적인 워크로드 구동을 통해 특정시킬 수 있다. 게다가 커널 메모리 영역이 변경되면 즉각적으로 탐지할 수 있기 때문에 메모리 병합이 커널 위변조 공격을 탐지할 수 있다는 장점이 있다. 정리하면 Global Memory Manager는 메모리 병합을 통해 중복되는 메모리 탐색 및 복제 연산을 최소화하는 메커니즘이다.

[0086] 본 발명의 실시예에 따른 Global Memory Manager는 먼저 VM 커널 검증을 수행한다(610). VM의 특정 커널 메모리를 확인하면 VM이 사용하는 커널 버전을 확인할 수 있다. 또한, VMM을 통해 VM의 Base Image를 확인할 수 있다. 다음으로, VM 그룹화이다. 동일한 커널을 사용하는 VM은 동일한 커널 메모리 규격을 갖고 있다. 따라서 동일한 커널을 사용하는 VM이 그룹화하였을 메모리 병합 시 안정성이 있으며, 탐색시간이 최소화된다. 이후, 글로벌 메모리 탐색을 수행한다(620). 먼저, 전처리 과정을 통해 Kernel에 따라 병합되어도 안전한 영역 메모리 구조가 파악되었다고 가정한다. 파악된 메모리 구조가 병합되어도 되는지 VM별로 교차검증을 통해 글로벌 메모리를 선정한다. 글로벌 메모리 교차검증에는 Rabin-Karp 검색 알고리즘을 활용하였다. Rabin-Karp 검색 알고리즘은 바이너리를 해싱 후 비교를 통해 불일치 여부를 판단할 수 있다. 글로벌 메모리에서 선정되지 못한 영역은 전부 보안 메모리(Private Memory) 영역으로 지정한다. 마지막으로 메모리 병합이다(630). 글로벌 메모리 병합은 linux kernel 2.6.32에서 제공해주는 KSM(Kernel Samepage Merging)의 기능의 일부 기능을 활용한다. KSM은 공유되어야 하는 메모리를 찾기 위한 탐색 과정이 매우 오랜 시간 소요된다는 단점이 있다. 하지만 Cloud-BlackBox의 경우 그룹별로 전처리를 통해 병합되어야 하는 메모리 영역이 확정되어 있으므로 탐색 과정이 불필요하다. 따라서, Cloud-BlackBox는 KSM에서 메모리를 병합하는 특정 기능만을 활용한다. Global Memory Manager를 통해 글로벌 메모리가 병합되면, 병합된 메모리 영역은 쓰기 제한(Write Protect)을 설정해두어 메모리 위변조 시 탐지할 수 있도록 한다. 이후 주기적인 Memory Recorder와 Memory I/O Scheduler 호출을 통해 Global M Frame을 생성한다.

[0088] 도 7은 본 발명의 일 실시예에 따른 가상화 환경에서의 저지연 메모리 기록 장치에서 인스턴스의 메모리 변화를 추적하여 병합 및 기록하는 과정을 설명하기 위한 도면이다.

[0089] 도 7(a)는 종래기술에 따른 메모리 스냅샷 과정을 나타내는 도면이고, 도 7(b)는 본 발명의 일 실시예에 따른 클라우드 블랙박스(Cloud BlackBox)를 이용한 스냅샷 과정을 나타내는 도면이다.

[0090] 본 발명의 실시예에 따른 메모리 쓰기 방지 모듈은 클라우드 컴퓨팅 환경에서 구동 중인 서비스(다시 말해, 인스턴스)가 할당 받아 사용하고 있는 메모리 영역에 대해 쓰기 권한을 제거한다. 이후 서비스에서 메모리 쓰기 연산이 발생하면 페이지폴트 추적 모듈을 통해 메모리 쓰기가 발생한 주소와 데이터 정보를 추적한다. 본 발명의 실시예에 따른 추적 데이터 관리 모듈은 페이지폴트가 유발된 주소와 데이터 정보를 인 메모리 버퍼에 복사하고 관리한다. 본 발명의 실시예에 따른 추적 데이터 아카이빙 모듈은 인 메모리 버퍼가 가득 차거나, 사용자가 인 메모리 버퍼를 비워달라고 요청하였을 때 비휘발성 스토리지 영역에 추적된 메모리 정보를 파일 형태로 생성한다.

[0091] VM의 메모리를 기록한다는 것은 메모리 스냅샷 연산이 반복된다는 것을 의미한다. 앞서 Cloud-BlackBox는 Global Manager를 통해 메모리 스냅샷 연산 시 탐색 되어야 하는 메모리를 병합하여 첫 번째 최적화를 이루었다. 하지만 여전히 VM 메모리 스냅샷은 많은 연산 오버헤드를 유발한다. VM 메모리 스냅샷 연산의 가장 큰 한계점은 메모리가 스냅샷이 수행되는 동안 VM이 일시정지된다는 문제와 메모리 스냅샷 수행시간이 오래 걸린다는 것이다.

[0092] 도 7을 참조하면 메모리 스냅샷을 위해서는 VM 일시 정지가 필요하고, 메모리 스냅샷 수행이 끝난 이후에 VM을 재가동 시킨다는 것을 확인할 수 있다. 즉 VM이 일시정지되는 시간은 메모리 스냅샷 수행시간은 비례한다. 본 발명의 실험결과에 따르면 4GB의 메모리를 할당받은 VM의 메모리를 QEMU를 통해 수집해 본 결과 약 9.3초의 시간이 소요되었다. 즉, 단 한 번의 메모리 스냅샷 수행으로 인해 사용자는 9.3초 동안 클라우드 서비스를 보장받을 수 없다는 것을 의미한다. 반복적인 메모리 스냅샷 연산을 수행한다는 것은 더더욱 비효율적이다. 더군다나, 메모리 스냅샷 연산이 호스트 OS의 자원을 VM과 나누어 사용하기 때문에, 자원 경합이 유발된다. 이로 인해 동일한 호스트 머신에서 구동 중인 다른 VM의 성능에 저하 현상을 유발시킨다. 즉. 메모리 스냅샷 연산은 SLA(Service-Level Agreement)은 보장할 수 없으므로 사용 클라우드에서 활용할 수 없다고 여겨지고 있다.

[0093] 본 발명의 실시예에 따른 Cloud-BlackBox는 이와 같은 한계점을 해결하기 위해 기존의 메모리 스냅샷 과정에서 지연이 발생하는 보틀넥 포인트(Bottleneck Point)(blocking)를 심층적으로 분석하였다. 메모리 수집 연산은 크게 3가지의 요소로부터 지연이 발생한다. 첫째 VM의 페이지(Page)를 탐색하는 과정이다. VM의 메모리를 스토리지에 파일 형태로 생성하기 위해서는 할당된 페이지에 맵핑 된 물리 메모리 영역을 탐색해야 한다. Cloud-BlackBox는 앞서 Global Manager를 통해 탐색 되어야 하는 메모리 크기 축소시켰다. 두 번째는 읽기 메모리(Read Memory)(Page) 연산이다. 메모리에 저장된 데이터를 스토리지에 쓰기 위해서는 반드시 메모리 옵셋(Offset)에 대한 읽기(Read) 연산이 필요하다. 하지만 메모리-투-메모리(Memory to Memory)로 데이터가 복제되기 때문에 성능 지연이 크게 발생하진 않는다. 세 번째는 메모리를 스토리지 복사하는 과정이다. 본 발명의 실

험결과에 따르면 파일(File) I/O 연산이 메모리 수집 전체연산 과정에서 93% 이상을 차지한다는 것을 알 수 있었다. Cloud-BlackBox는 File I/O Bottleneck Point를 Cognitive-Scale Bitmap을 활용한 페이지 추적(Page Tracking)과 인 메모리 기반의 비동기 버퍼를 통해 해결하였다.

[0094] 본 발명의 실시예에 따른 Cloud-BlackBox는 메모리 기록 과정은 도 7(b)와 같다. Cloud-BlackBox는 기존의 메모리 스냅샷과 다르게 VM 구동 시 최초 1회에 VM 일시정지가 발생한다. 이를 초기 덤프(init dump)라고 말한다. init dump 과정은 기존 스냅샷과 같이 메모리 탐색 과정과 읽기 메모리(Read Memory) 과정이 포함되어 있다. 하지만 스토리지에 복제되어야 하는 대상 데이터(다시 말해, 메모리)를 인 메모리 기반의 비동기 버퍼에 삽입해준다. 해당 과정은 스토리지와 메모리의 대역폭(Bandwith)을 해결하기 위한 수단이다. 이후 Macro Bitmap을 생성해주고, Bitmap을 0으로 초기화 해준다. Macro Bitmap은 VM의 페이지(4kb)를 큰 그룹으로 관리하여 특정 메모리가 변조되었을 때 표기해주는 역할을 수행한다. Macro Bitmap이 생성되면 VM을 재가동 시킨다. 백그라운드 스레드에서는 비동기 버퍼에 삽입된 메모리를 스토리지 영역에 파일로 생성해준다. 본 발명은 최초에 스냅샷 된 메모리 스냅샷 파일을 MDr이라고 한다. 이후 두 번째 메모리 스냅샷부터는 VM 일시정지 현상이 발생하지 않으며, 메모리 스냅샷 연산이 간소화된다. Cognitive-Scale Bitmap를 통해 스냅샷 연산이 간소화되는 과정은 다음과 같다. Macro Bitmap의 변조 지역성 모니터링을 통해, 빈번한 변화가 발생하는 영역은 Micro Bitmap으로 생성하고, 변화가 적은 영역은 큰 범위로 관리할 수 있는 Macro Bitmap으로 묶어준다. 사용자 혹은 다른 모듈로부터 메모리 스냅샷 연산 요청이 들어오면 Macro/Micro Bitmap 영역의 Write 권한을 제거한다. VM에서 메모리 쓰기(Memory Write) 요청이 들어오면 페이지 폴트(Page-Fault)가 발생하고, Macro/Micro Bitmap에 Dirty-bit를 마킹해준다. 사용자가 지정한 지속 시간(Duration Time) 이후 쓰기 제한(Write Protect) 권한을 복구해주고 Macro/Micro Bitmap에 표기된 페이지(4kb) 내용을 비동기 버퍼에 삽입해준다. 백그라운드에서는 비동기 버퍼에 삽입된 메모리를 offset 정보가 담긴 메타데이터와 함께 스토리지 영역에 파일로 생성해준다. Macro/Micro Bitmap을 통해 생성된 파일을 MDp라고 한다. 이후, 반복적인 과정을 통해 Mircro Bitmap은 디테일해지고, Macro Bitmap의 영역은 넓어지기 때문에 관리되어야 하는 비트맵의 개수는 점진적으로 줄어들게 되므로 성능이 최적화된다.

[0096] 도 8은 본 발명의 일 실시예에 따른 VM 메모리를 추적 및 기록하는 과정을 설명하기 위한 도면이다.

[0097] MDp가 생성되는 과정은 도 8을 참조하여 더욱 상세히 설명한다. 초기 덤프(init dump) 과정에서는 VM의 메모리 전체영역을 Memory I/O Scheduler를 통해 생성한다. 이후 메모리 스냅샷(delta dump)는 Macro Bitmap과 Mircro Bitmap을 활용하여 MDp를 생성한다. Macro Bitmap은 메모리의 변화정보를 큰 범위에서 관찰하여 단순 표기하는 공간으로 쓰기 제한(Write Protect)으로 인한 가상 성능 저하 현상이 발생하지 않는다. Mircro Bitmap의 경우 하나의 비트맵을 페이지 단위로 쪼개어 표기하는 공간이다. Mircro Bitmap 경우 면밀한 메모리 추적을 위해 쓰기 제한이 필요함으로써 약간의 성능 저하 현상이 유발될 수 있다. 하지만 메모리 스냅샷 요청 시점의 정확한 메모리 상태를 저장할 수 있다는 큰 장점이 있다. MDp를 생성은 Macro Bitmap에 Mircro Bitmap을 오버레이(Overlay) 함으로써 가능해진다. 큰 범위에서 페이지를 추적하는 Macro Bitmap 과 페이지 단위까지 미세하게 메모리를 트레킹 Mircro Bitmap이 오버레이되었을 때 스냅샷 요청 시점의 VM의 상태가 완전한 Bitmap으로 표기된다. 오버레이를 통해 최종 생성된 Bitmap은 Memory I/O Scheduler에 의해 비동기 버퍼를 거쳐 스토리지에 MDp 파일로 생성된다. Cloud-Blac6Box의 장점은 최초의 MDr 생성을 제외하고는 VM 서스펜드(Suspend) 현상이 거의 발생하지 않는다는 것과 스토리지 비용이 최소화된다는 것이다. 메모리 스냅샷 시 발생하는 반복적으로 발생하는 쓰기 제한 이벤트는 VM 성능 저하를 유발하지만, 본 발명이 제안한 Cloud-BlackBox는 Cognitive-Scale Bitmap을 통해 쓰기 제한 이벤트 호출을 최소화시켰다.

[0099] 도 9는 본 발명의 일 실시예에 따른 메모리 복원 과정을 설명하기 위한 도면이다.

[0100] 본 발명의 실시예에 따른 Cloud-BlackBox를 통해 기록되는 메모리는 하나의 MDr 과 다수의 MDp 를 포함한다. 이와 같은 구조는 특정 시점의 VM 메모리를 분석하기 위해서는 반드시 메모리 복원 절차가 필요하다. 메모리 복원 절차는 도 9와 같다. 메모리 복원 절차는 최초에 생성된 MDr과 누적된 MDp의 XOR 연산을 통해 복구할 수 있다. 하지만 MDp가 누적될수록 메모리를 복원하는 데 시간이 많이 소요될 것이다. 즉 랜덤 엑세스 복원(Random Access Restore) 성능을 보장하지 못한다. Cloud-BlackBox는 랜덤 엑세스 메모리 복원 성능을 보장하기 위해 MDc라는 체크포인트 파일을 생성하도록 고안하였다. MDc는 누적 MDp의 누적 사이즈가 사용자가 지정한 크기를 초과하게 되면 생성되는 파일이다. MDc는 이전 MDr 또는 MDc 이후에 생성된 MDp 들을 XOR 하여 생성된다. MDc로 인해 메모리 복원 시 랜덤 엑세스 성능을 보장할 수 있다. MDc 생성은 M-Frame Manager에 의해서 제어된다. M-Frame Manager는 Cgorups를 통해 독립된 자원을 할당받아 백그라운드에서 구동되기 때문에 VM 성능 및 메모리

스냅샷 성능 저하를 유발시키지 않는다.

[0102] 도 10은 본 발명의 일 실시예에 따른 효율적인 대규모 메모리 분석을 위한 SAMI 메타데이터를 설명하기 위한 도면이다.

[0103] 본 발명의 실시예에 따른 Cloud-BlackBox는 대규모의 메모리를 효율적으로 분석하기 위해 도 10과 같이 SAMI(Synchronized Accessible Memory Interchange) 라는 메타데이터를 생성한다. SAMI는 Volatility, Rekall과 같은 메모리 분석 프레임워크에서 활용될 수 있으며, 시간에 따른 VM 메모리 변화정보를 타임라인으로 생성하는데 활용할 수 있다. 대규모의 메모리셋에 대해 병렬적으로 분석하기 위해서는 최초 1회 옵셋 추출이 필요하다. 플러그인에 대한 옵셋 추출은 MDr 데이터 분석을 통해 이뤄진다. 메모리 분석 플러그인은 volatility과 rekall에서 지원하는 pslist, dlllist, sockets 등을 활용할 수 있으며, 자체적인 플러그인을 제작하여 사용할 수 있다. 메모리 분석 플러그인 테이블(Memory Analysis Plugin Table)에 입력된 플러그인 정보를 기준으로 VM별 메모리 옵셋을 추출한다. 추출된 옵셋 정보는 MDp 생성 시 생성되는 SAMI(Synchronized Accessible Memory Interchange) 데이터를 확인을 통해 맵핑 시킬 수 있다. 예를 들어 pslist 변화정보에 대해 빠르게 분석하기 위해서는 SAMI에 기록된 옵셋 정보를 활용하여 pslist 옵셋에 접근 또는 수정이 이루어진 MDp 들을 분석하면 된다.

[0105] 이상에서 설명된 장치는 하드웨어 구성요소, 소프트웨어 구성요소, 및/또는 하드웨어 구성요소 및 소프트웨어 구성요소의 조합으로 구현될 수 있다. 예를 들어, 실시예들에서 설명된 장치 및 구성요소는, 예를 들어, 프로세서, 콘트롤러, 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(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 기술분야에서 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 콘트롤러를 포함할 수 있다. 또한, 병렬 프로세서(parallel processor)와 같은, 다른 처리 구성(processing configuration)도 가능하다.

[0106] 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, 구성요소(component), 물리적 장치, 가상 장치(virtual equipment), 컴퓨터 저장 매체 또는 장치에 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.

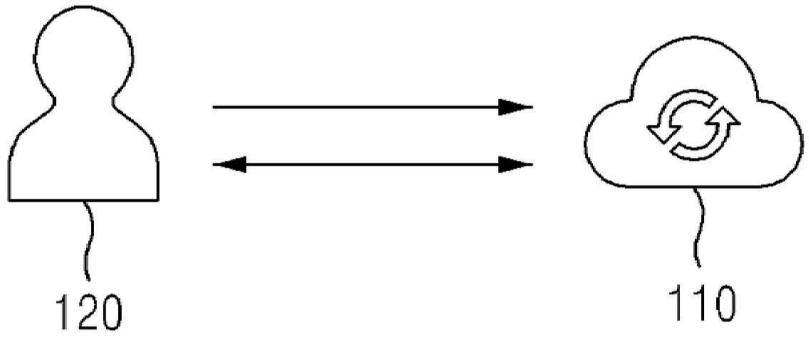
[0107] 실시예에 따른 방법은 다양한 컴퓨터 수단을 통하여 수행될 수 있는 프로그램 명령 형태로 구현되어 컴퓨터 판독 가능 매체에 기록될 수 있다. 상기 컴퓨터 판독 가능 매체는 프로그램 명령, 데이터 파일, 데이터 구조 등을 단독으로 또는 조합하여 포함할 수 있다. 상기 매체에 기록되는 프로그램 명령은 실시예를 위하여 특별히 설계되고 구성된 것들이거나 컴퓨터 소프트웨어 당업자에게 공지되어 사용 가능한 것일 수도 있다. 컴퓨터 판독 가능 기록 매체의 예에는 하드 디스크, 플로피 디스크 및 자기 테이프와 같은 자기 매체(magnetic media), CD-ROM, DVD와 같은 광기록 매체(optical media), 플롭티컬 디스크(floptical disk)와 같은 자기-광 매체(magneto-optical media), 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.

[0108] 이상과 같이 실시예들이 비록 한정된 실시예와 도면에 의해 설명되었으나, 해당 기술분야에서 통상의 지식을 가진 자라면 상기의 기재로부터 다양한 수정 및 변형이 가능하다. 예를 들어, 설명된 기술들이 설명된 방법과 다른 순서로 수행되거나, 및/또는 설명된 시스템, 구조, 장치, 회로 등의 구성요소들이 설명된 방법과 다른 형태로 결합 또는 조합되거나, 다른 구성요소 또는 균등물에 의하여 대치되거나 치환되더라도 적절한 결과가 달성될 수 있다.

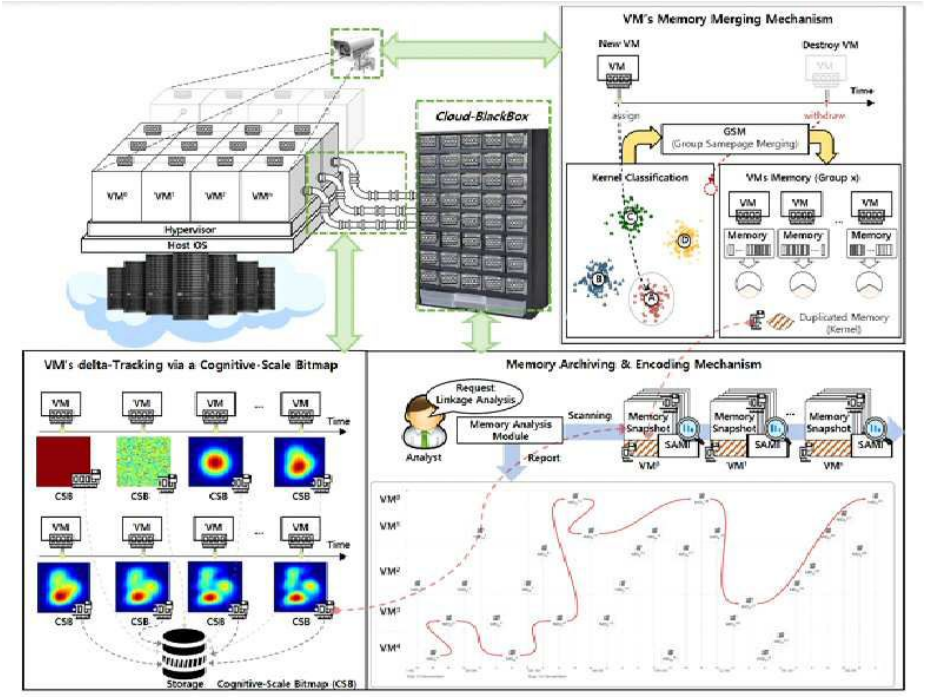
[0109] 그러므로, 다른 구현들, 다른 실시예들 및 특허청구범위와 균등한 것들도 후술하는 특허청구범위의 범위에 속한다.

도면

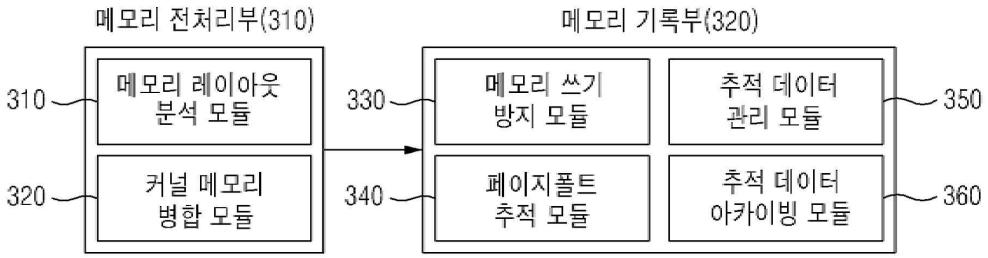
도면1



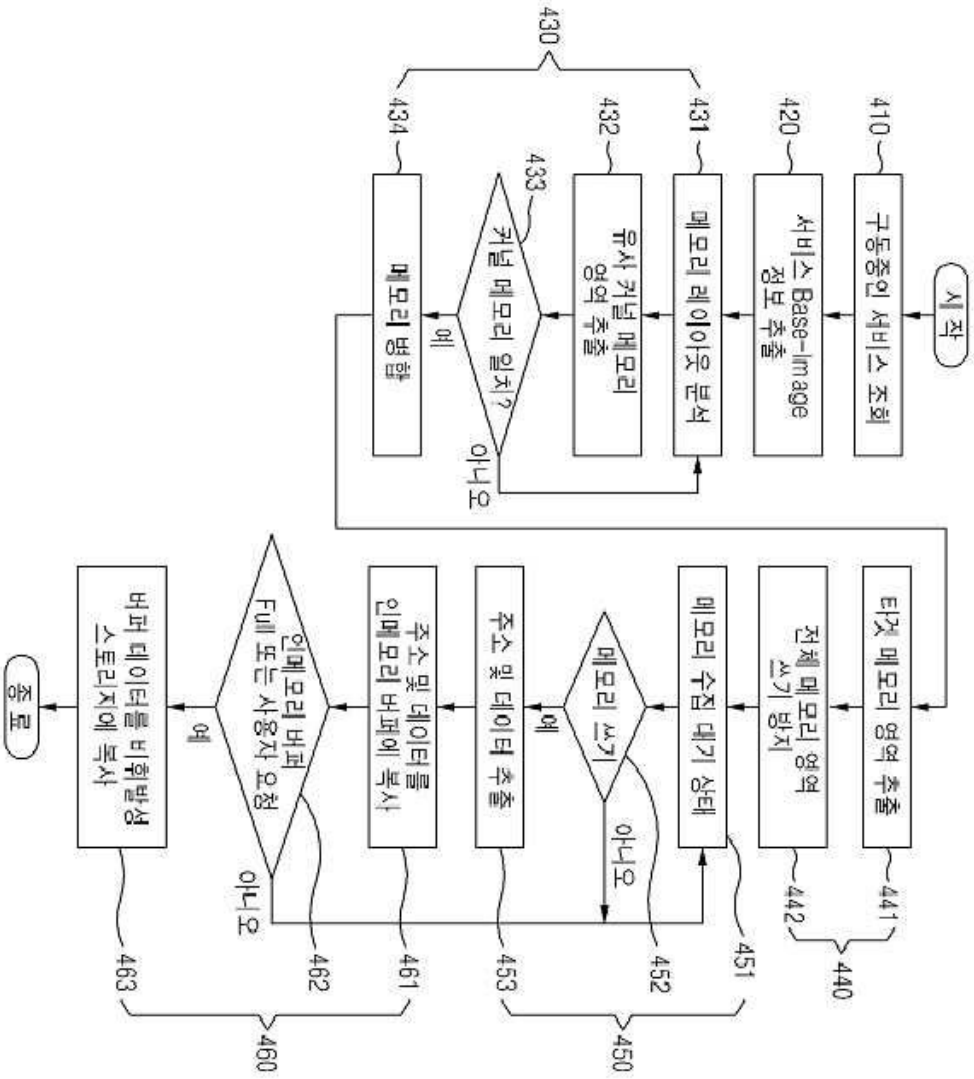
도면2



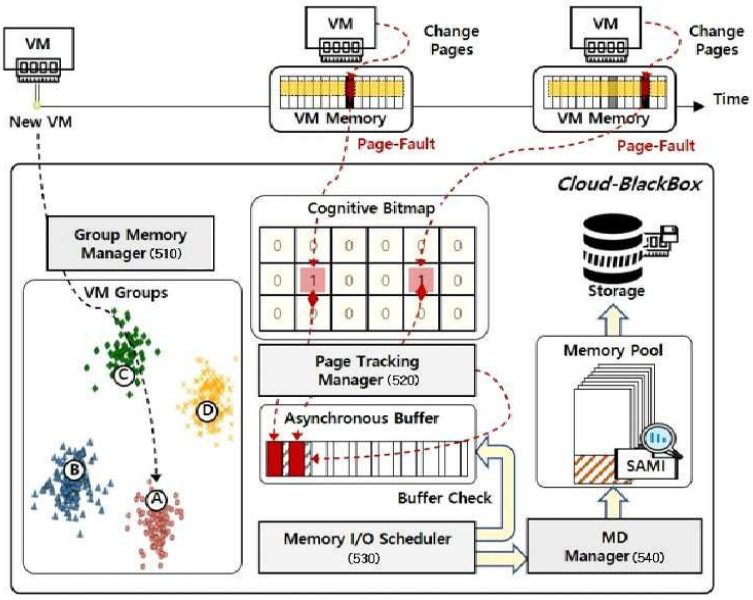
도면3


메모리 전처리부(310)
메모리 기록부(320)
310
메모리 레이아웃 분석 모듈
320
커널 메모리 병합 모듈
330
메모리 쓰기 방지 모듈
340
페이지폴트 추적 모듈
350
추적 데이터 관리 모듈
360
추적 데이터 아카이빙 모듈

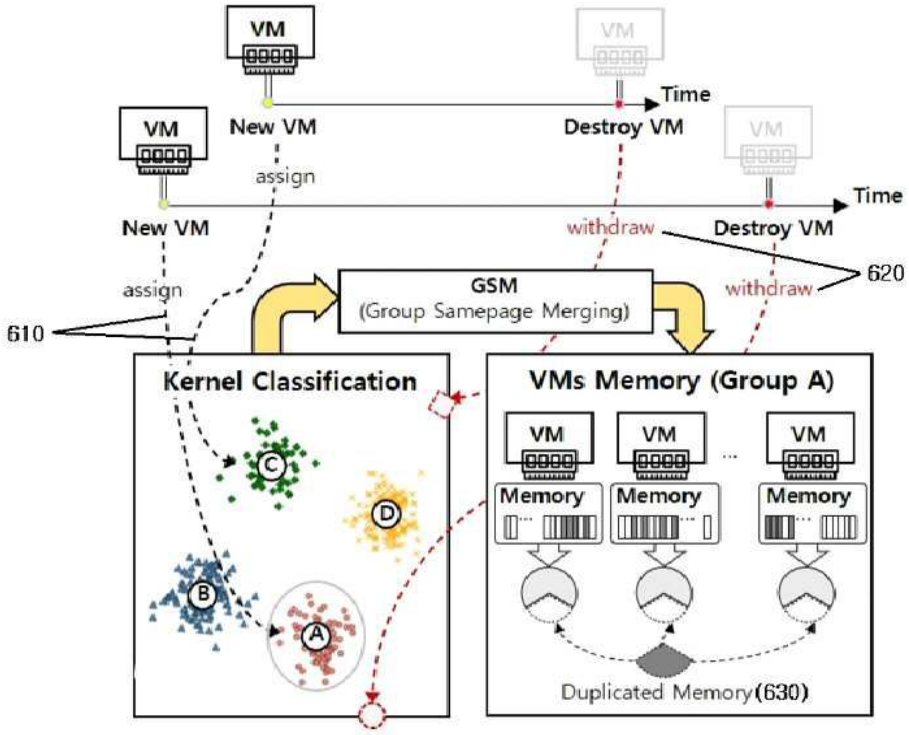
도면4


시작
410
구동중인 서비스 조회
420
서비스 Base-Image 정보 추출
430
431
메모리 레이아웃 분석
432
유사 커널 메모리 영역 추출
433
커널 메모리 일치?
예
아니오
434
메모리 병합
441
타겟 메모리 영역 추출
442
전체 메모리 영역 쓰기 방지
440
451
메모리 수집 대기 상태
452
메모리 쓰기
아니오
453
주소 및 데이터 추출
예
461
주소 및 데이터를 인메모리 버퍼에 복사
462
인메모리 버퍼 Full 또는 사용자 요청
아니오
예
463
버퍼 데이터를 비휘발성 스토리지에 복사
450
460
종료

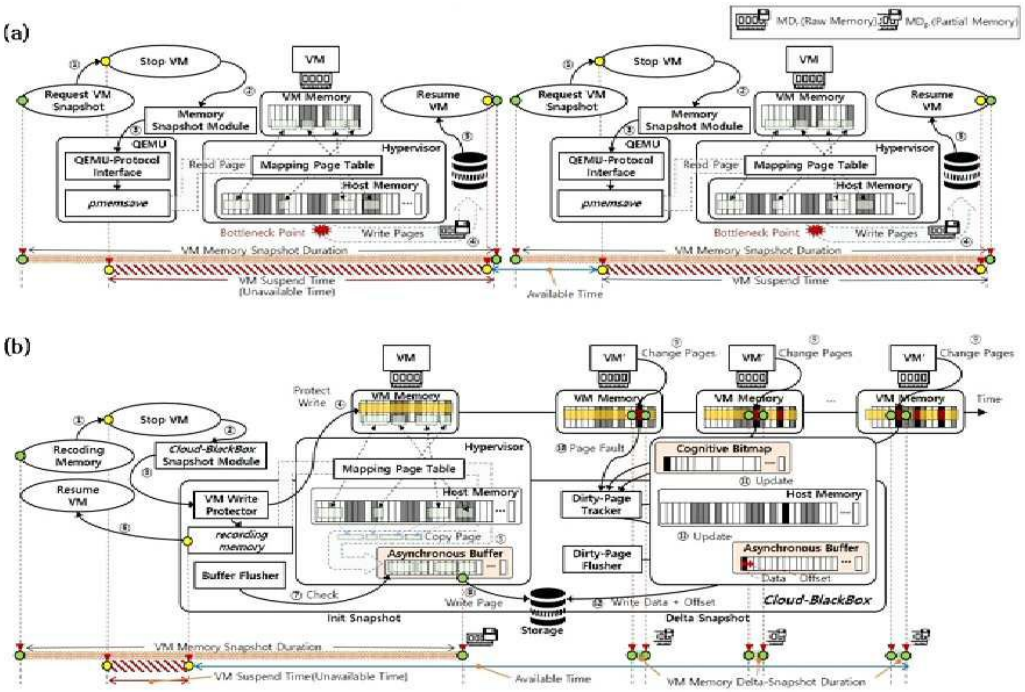
도면5


VM
New VM
VM
Change Pages
VM Memory
Page-Fault
VM
Change Pages
VM Memory
Page-Fault
Time
Cloud-BlackBox
Group Memory Manager (510)
VM Groups
Cognitive Bitmap
Page Tracking Manager (520)
Asynchronous Buffer
Buffer Check
Memory I/O Scheduler (530)
Storage
Memory Pool
SAMI
MD Manager (540)

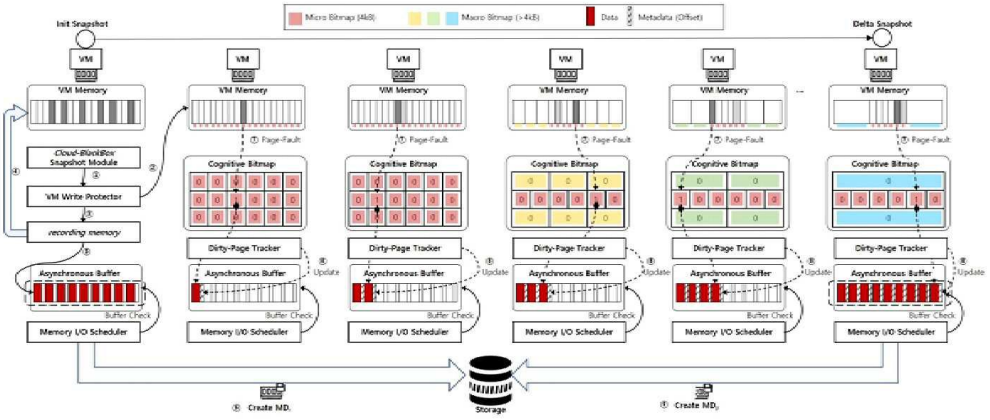
도면6


VM
New VM
VM
Time
Destroy VM
VM
New VM
assign
Time
withdraw
Destroy VM
620
assign
610
GSM
(Group Samepage Merging)
withdraw
Kernel Classification
VMs Memory (Group A)
VM
VM
VM
Memory
Memory
Memory
Duplicated Memory(630)

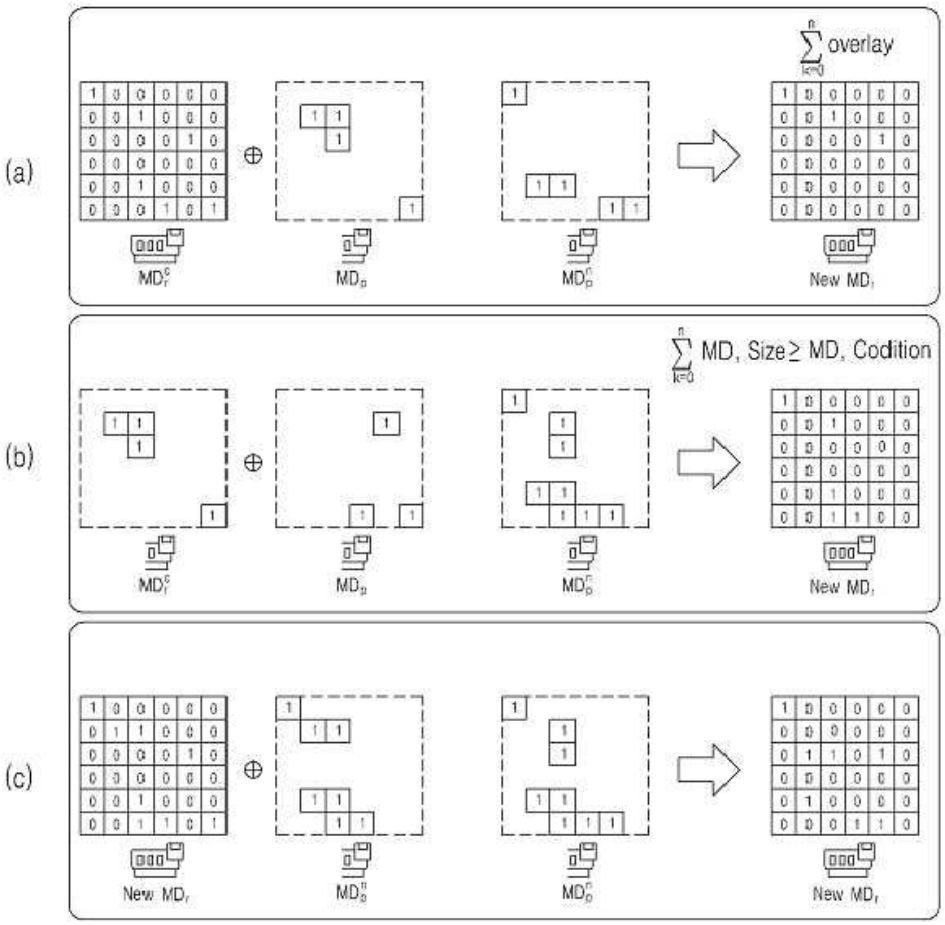
도면7


(a)
Stop VM
Request VM Snapshot
Memory Snapshot Module
VM
VM Memory
Resume VM
QEMU
QEMU-Protocol Interface
pmemsave
Hypervisor
Mapping Page Table
Host Memory
Bottleneck Point
Write Pages
VM Memory Snapshot Duration
VM Suspend Time (Unavailable Time)
Available Time
VM Suspend Time
(b)
Recoding Memory
Cloud-BlackBox Snapshot Module
VM Write Protector
recording memory
Buffer Flusher
Protect Write
Asynchronous Buffer
Copy Page
Page Fault
Cognitive Bitmap
Update
Dirty-Page Tracker
Dirty-Page Flusher
Change Pages
Time
Write Page
Write Data + Offset
Storage
Init Snapshot
Delta Snapshot
Cloud-BlackBox
VM Suspend Time(Unavailable Time)
VM Memory Delta-Snapshot Duration

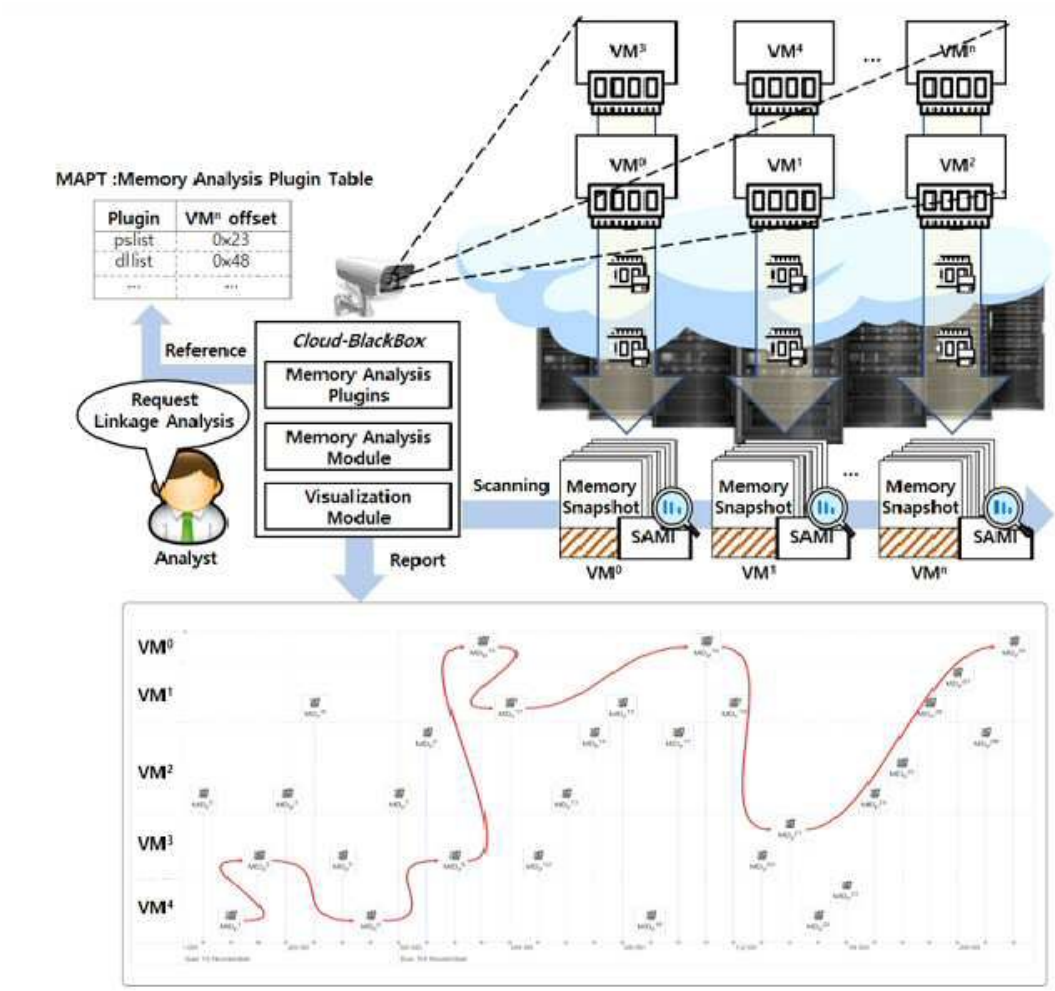
도면8


Init Snapshot
Delta Snapshot
VM
VM Memory
Cloud-BlackBox Snapshot Module
VM Write Protector
recording memory
Page-Fault
Cognitive Bitmap
Dirty-Page Tracker
Asynchronous Buffer
Buffer Check
Memory I/O Scheduler
Update
Storage

도면9



도면10


MAPT :Memory Analysis Plugin Table
Plugin
VMn offset
pslist
0x23
dllist
0x48
Reference
Cloud-BlackBox
Memory Analysis Plugins
Memory Analysis Module
Visualization Module
Request Linkage Analysis
Analyst
Report
Scanning
Memory Snapshot
SAMI
VM0
VM1
VM2
VM3
VM4
VMn