랜섬웨어 방어 SSD에서의 감염 데이터 분리 및 고속 쓰레기 수집 연구

민 동 현, 안 진 우, 김 영 재
서강대학교 컴퓨터공학과
{mdh38112, jinu37, youkim}@sogang.ac.kr

Infected Data Segregation and Fast Garbage Collection on Ransomware Defense SSD

Donghyun Min, Jinwoo Ahn, Youngjae Kim
Department of Computer Science and Engineering, Sogang University, Seoul, South Korea

요 약

랜섬웨어는 사용자의 데이터를 암호화하고 금전을 요구하여 사용자의 중요한 데이터에 막대한 손상을 가하면서 금전적 피해를 야기시킨다. 이에 따라 최근에는 SSD단에서 랜섬웨어를 방어하는 연구가 활발히 진행되고 있다. 특히, Ameoba는 랜섬웨어로부터 디바이스 내부에서 랜섬웨어를 감지하고 자동 백업 및 복구를 하는 최신 기술의 랜섬웨어 방어 SSD이다. 하지만, Amoeba는 복구를 위한 백업 페이지들의 증가로 인해 Garbage Collection의 오버헤드가 증가되어 SSD성능을 낮추고 SSD 수명을 감소시키는 문제점을 가진다. 따라서 본 연구에서, 우리는 Amoeba-Div를 제안한다. Amoeba-div는 디바이스 내부에서 일반 유저 데이터와 랜섬웨어 감염 데이터 간의 분리 기법을 제안한다. 이를 통해 Garbage Collection의 오버헤드를 줄이고 랜섬웨어 감지 기반 방어 SSD의 성능을 증가시킨다. 실험에서 Amoeba-div는 Amoeba보다 성능 관점에서 13%만큼의 프로그램 최대 반응 시간을 낮추며, 최대 15%만큼의 지우기 연산 감소를 통해 SSD 수명을 연장시키는 효과를 증명한다.

1 서 론

랜섬웨어(ransomware)는 악성 소프트웨어의 한 종류로 사용자의 머신으로 침입하여 데이터를 암호화한 뒤, 사용자에게 금전을 요구하는 악성 프로그램이며 많은 피해량을 일으킨다 [1].

랜섬웨어의 공격으로 인한 피해를 줄이기 위해, NAND 플래시 메모리 기반의 Solid-State Drive (SSD) 에서의 데이터 백업 기술이 연구되었다 [2, 3, 4]. 이 기술들은 펌웨어 레벨의 백업이기 때문에 커널 특권을 갖는 랜섬웨어로부터 더욱 안전하다는 장점이 있다.

Amoeba [4]는 SSD 내부에서 랜섬웨어 감지, 자동 백업 및 복구가 가능한 랜섬웨어 방어 SSD이다. 하지만 Amoeba는 백업으로 인해 Garbage Collection (GC)의 페이지 복사량이 증가하여 GC 비용을 증가시키는 문제를 갖고 있다. 일반적으로 SSD는 무효화된 블럭을 회수하기 위해 GC의 지우기(erase) 연산을 수행한다. 만약 빅팀(victim) 블럭에 유효 페이지가 남아있다면 이를 다른 블럭으로 복사한다. Amoeba는 복구에 쓰일 백업(backup) 페이지도 함께 복사하도록 설계되었다. 왜냐하면 Amoeba는 백업 페이지를 유효한 데이터로 간주하기 때문이다. 따라서 전체 페이지 복사 횟수가 증가하고 결과적으로 GC 비용이 증가한다.

본 연구는 Amoeba의 GC 성능 효율을 높이기 위해 디바이스 레벨에서 일반 유저 데이터와 랜섬웨어 감염 데이터 간의 블록 분리 배치 기법을 구현한 Amoeba-Div를 제안한다. 랜섬웨어 감염 데이터들만 같은 블럭(SSD의 지우기 연산의 단위)에 따로 배치시킨다면 지우기 연산 수행 시, 필요한 페이지 복사량을 줄일 수 있어서 Amoeba의 GC 성능을 높일 수 있다. 또한, 블록 분리 기법을 통해 블럭을 지우기 위한 연산의 수행 빈도가 확률적으로 줄어들 수 있기 때문에 SSD 수명을 연장시키는 장점을 만들어낸다.

평가 섹션에서는 리눅스 환경을 대상으로 하는 실제 랜섬웨어인 Erebus을 사용하여 Amoeba와 Amoeba-Div을 비교하였다. 실험 결과에서 Amoeba-Div 메커니즘은 Amoeba에 비해 GC 과정에서 필요한 페이지 복사 횟수를 최대 15% 줄였고 3% 만큼의 성능 향상을 확인했다. 또한, 블럭 지우기 연산의 호출 빈도가 최대 15% 만큼 적어지는 것을 확인했다.

2 배경 지식

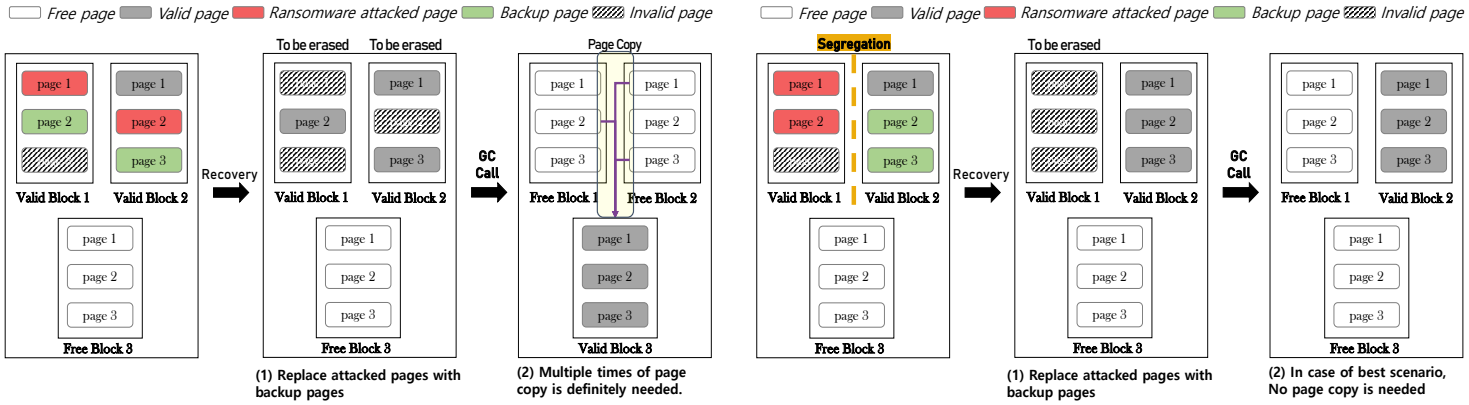
2.1 랜섬웨어 감지 기반의 백업/복구 Amoeba

Amoeba는 효과적인 랜섬웨어 감지 알고리즘 [5] 을 하드웨어 모듈에 구현해서 빠르고 정확한 랜섬웨어 감지 기술을 구현한 최신 기술의 SSD다. Amoeba는 랜섬웨어를 포함한 모든 덮어쓰기 요청을 항상 허용하는 대신에, 정확한 랜섬웨어 감지를 바탕으로 랜섬웨어의 침입을 판단 시, 백업을 만드는 메커니즘을 제안하였다.

이 메커니즘은 기존의 유효 페이지를 곧 바로 무효화 시키지 않고 백업으로 사용하기 때문에 백업 공간을 추가적으로 만들 필요가 없다. 더욱이, Amoeba는 랜섬웨어 침입을 감지 하는 순간에 생성한 백업 페이지가 유저의 최신 원본 데이터이기 때문에 다른 이전 버전의 백업을 만들지 않는다. 따라서 하나의 유효 페이지마다 최대 1개의 백업 페이지를 갖게 된다. 이는 기존의 SSD 백업 연구가 여러 버전의 백업을 가져서 생겨난 GC 오버헤드를 감소시켰다. 이러한 특징들로 Amoeba는 SSD 공간 효율성과 성능 향상을 만들어냈다. 또한, Amoeba는 데이터를 복구할 때, 랜섬웨어 감염 페이지를 무효화 시키고, 백업 페이지를 바로 유효 상태로 바꾸어 복구를 진행하므로 유저의 도움이 필요없는 빠른 복구 능력을 구현하였다.

2.2 Amoeba의 GC 동작 과정

SSD는 블록을 회수하고자 할 때, GC를 호출하여 블록 단위의 지우기 연산을 수행한다. 호출된 GC는 빅팀 블록에 존재하는 유효 페이지만 다른 유효 블럭으로 옮기는 과정이 필수적이다. 반면 Amoeba의 GC는 빅팀에 존재하는 백업 페이지도 유효페이지로 간주하여 다른 블록으로 복사한다. 왜냐하면 Amoeba는 복구 과정에서 필요한 백업 데이터를 유지시켜야 하기 때문이다. 만약 백업



(a) Amoeba (without block segregation)

(b) Amoeba-Div (with block segregation)

그림 1: Amoeba 기반 SSD에서 블록 분리 정책의 효과

페이지를 옮기지 않고 블럭을 지운다면 백업 데이터가 없어지기 때문에 복구가 불가능하다. 따라서 Amoeba는 GC 수행 시, 유효 페이지와 백업 페이지 모두 복사하기 때문에 백업 메커니즘이 없는 일반 SSD 보다 GC 비용을 더 크게 갖는다. 특히, 랜섬웨어가 자주 침입할수록 Amoeba는 이를 감지하여 백업을 많이 만들게 되므로 GC 비용이 더 커지게 되고 결과적으로 SSD 성능을 낮춘다.

Amoeba가 복구를 수행한 후에도 페이지 복사 오버헤드는 여전히 문제가 된다. Amoeba는 복구 수행 시, 랜섬웨어 감염 페이지를 무효화 시키고 백업 페이지를 유효 상태로 바꾼다. 따라서 복구가 수행된 블럭은 무효화된 페이지가 증가하고 이에 따라 GC의 빅팀 블럭이 될 가능성이 증가한다. 하지만 빅팀 블럭에 복사해야 할 유효 혹은 백업 페이지가 섞여 있다면 랜섬웨어 데이터가 존재하는 블럭을 곧 바로 지우지 못하고 페이지 복사를 수행해야만 한다.

3 설계 및 구현

이 장에서는 제안하는 Amoeba-Div의 설계와 구현을 제시한다. Amoeba-Div는 기존 Amoeba의 정확한 랜섬웨어 감지 기법 능력을 활용하여 랜섬웨어에 공격당한 페이지 데이터만을 다른 특정 블럭으로 모아두는 기법(Block Segregation)을 구현하였다. 이 기법을 통해 Amoeba가 만드는 GC의 오버헤드를 줄일 수 있다. 그림 1은 Amoeba가 백업 페이지를 활용하여 복구하는 것을 표현하였다. 이 예제는 세 개의 NAND 플래시 블럭이 각각 세 개의 페이지를 갖고 있다고 가정한 Amoeba 이다.

그림 1(a)는 블록 분리 기법이 없는 Amoeba이기 때문에 랜섬웨어 공격당한 페이지와 유효, 백업 페이지가 블럭 1과 블럭 2에 걸쳐 섞여 있다. 유저가 랜섬웨어의 침입을 인식하고 복구를 진행된다면 SSD에서 랜섬웨어 공격당한 페이지가 무효화가 되고 백업 페이지가 다시 유효 페이지가 된다. 이러한 복구 과정에 따라 무효화된 페이지와 유효 페이지가 섞이게 되어 GC 비용이 커지는 것을 발견할 수 있다. 위 예제의 경우, 총 3번의 페이지 복사 비용이 발견된다.

반면에, 그림 1(b)는 블록 분리 기법을 적용한 Amoeba-Div이다. 즉, 랜섬웨어에 감염된 페이지들을 블럭1에, 유효 페이지와 백업 페이지를 블럭2에 분리시켰다. 이 경우 복구를 진행시키면 블럭1에는 오직 무효화된 페이지만 존재하고 블럭2에는 오직 유효 페이지만 존재하게 된다. 따라서 이 경우의 지우기 연산은 페이지 복사가 필요없기 때문에 GC 비용을 줄일 수 있다.

표 1: SSD 시뮬레이터 환경 구성

Common SSD parameters.	
Blocks per package	16384
Minimum free block percentage	10
Cleaning Policy	Greedy
Pages per Block	128
Page Read Latency	0.025
Page Write Latency	0.204
Block Erase Latency	1.5
Page size	8KB

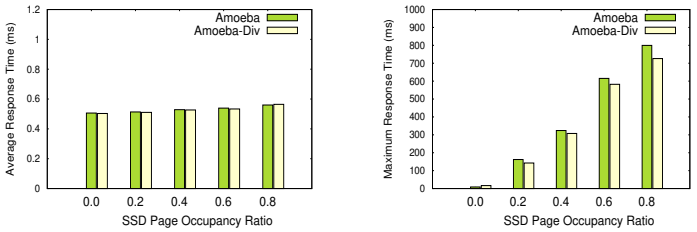
위 예제와 같이, 복구가 진행되면 랜섬웨어 감염 데이터는 무효화된다. 그림 1(b)와 같이 무효화가 될 데이터 (랜섬웨어 감염 데이터)들을 합친다면 GC 과정이 훨씬 더 효율적으로 수행될 수 있다.

Amoeba-Div을 구현하기 위해 SSD에서는 다음과 같은 두 가지의 블록 타입을 제공한다: 유효 페이지와, 백업 페이지를 저장하기 위한 블록(타입 1)과 랜섬웨어 감염 페이지를 위한 블록(타입 2). 우선 쓰기 요청이 들어오면 새롭게 쓰여질 데이터가 유저의 데이터인지, 랜섬웨어 데이터인지를 확인한다. 만약 유저의 데이터라고 판단하면 Amoeba와 동일한 쓰기 작업이 수행된다. 하지만 랜섬웨어 데이터라고 판단하면 해당 데이터를 보관할 타입 2 블럭을 따로 할당하여 쓰기 요청을 수행한다. GC에 의해 빅팀 블럭을 회수할 때, 빅팀 블럭의 타입과 동일한 타입의 유효 블럭으로 옮기는 기능을 GC 알고리즘에 추가하였다. 예를 들어, 빅팀 블럭이 타입 2 블럭이면, 이 블럭에 존재하는 유효 페이지들은 또 다른 타입 2의 유효 블럭으로 옮겨진다.

4 평가

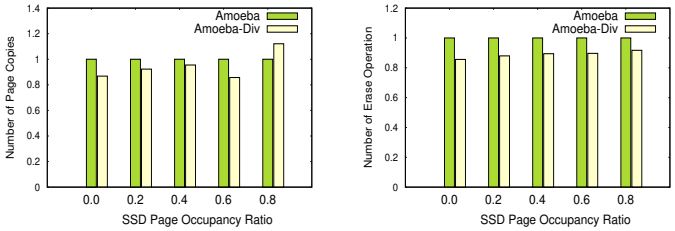
4.1 실험 설정

실험 환경: Amoeba-Div를 평가하기 위해, Microsoft Research의 Disksim SSD 시뮬레이터 [6] 를 활용하였다. 우리는 Baseline을 Amoeba로 정하여 블록 분리 기법을 적용한 Amoeba-Div를 비교 평가한다. SSD 시뮬레이터의 자세한 환경은 표1에 정리되었다.



(a) Average Response Time (b) Maximum Response Time

그림 2: SSD 페이지 점유율 변화에 따른 Amoeba와 Amoeba-Div 사이의 성능 비교



(a) Normalized Number of Page Copy (b) Normalized Erase counts

그림 3: SSD 페이지 점유율 변화에 따른 Amoeba와 Amoeba-Div 사이의 페이지 복사 횟수와 지우기 횟수

워크로드: Erebus 랜섬웨어 [7] 와 5초 간격으로 파일 데이터를 4MB씩 읽고 덮어쓰기를 하는 유저 프로그램을 동시적으로 돌려 블럭 내용을 갖는 I/O 집중적인 (I/O intensive) 트레이스를 뽑았다.

4.2 결 과

Amoeba와 Amoeba-Div간의 성능을 비교하기 위해, 우리는 프로그램 평균 응답 시간을 측정했다. 그림 2(a)은 Amoeba의 프로그램 평균 응답 시간에 정규화 시킨 Amoeba-Div의 평균 응답 시간이다. 그림 2(b)는 Amoeba과 Amoeba-Div의 최대 응답 시간을 비교한 결과 그래프이다. Amoeba-Div는 전체 성능에서는 최대 3% 만큼의 향상이 있었고, 최대 응답 시간에서 최대 13% 만큼의 향상을 확인하였다.

그림 3(a)는 프로그램 동작 중 발생한 페이지 복사 횟수를 의미한다. 블록 분리 기법을 통해 GC의 페이지 복사량이 최대 15% 만큼 감소했음을 확인할 수 있다. 반면 SSD 페이지 점유율이 80%인 경우엔 Amoeba-Div가 Amoeba 보다 성능이 오히려 감소한다. 하지만 증가된 페이지 복사량 때문에 생긴 이러한 성능 저하는 오직 0.8%로 매우 적다.

그림 3(b)는 프로그램 동작 중 발생한 지우기 연산의 빈도 수를 의미한다. 블록 분리 기법을 통해 모든 SSD 페이지 점유율 상태에서 확률적으로 줄어드는 지우기 연산의 호출 수를 볼 수 있다. Amoeba-Div는 Amoeba에 비해 최대 15%만큼의 지우기 연산을 줄일 수 있는 것을 확인했다. 따라서 블록 분리 기법은 결과적으로 SSD 수명 증가를 만들어낸다.

5 관련 연구

SSD에서 랜섬웨어를 감지하고 백업/복구 기능을 연구한 여러 접근 방법이 존재한다. FlashGuard [2]는 랜섬웨어의 *Read-After-Write* 패턴을 확인하여 감지하고 SSD 안에서 무효화되는 페이지를 백업으로 쓰는 기법을 처음으로 제안하였다. SSD-Insider [3]는 랜섬웨어의 *Overwrite* 패턴에 초점을 맞추었다. 랜섬웨어의 *Overwrite* 패턴과 일반 애플리케이션의 *Overwrite* 패턴의 차이들을 관찰하여 정확성이 향상된 랜섬웨어 감지를 수행한다. Multi-Streamed SSD [8]는 비슷한 수명을 갖는 데이터들을 같은 스트림(stream)으로 SSD로 보내서 분류된 데이터들 끼리만 같은 블록에 배치시키는 방식이다. 같은 블록에 묶인 데이터는 결과적으로 비슷한 시기에 무효화가 되기 때문에 GC 오버헤드를 줄일 수 있고 SSD의 수명을 늘릴 수 있다. Multi-Streamed SSD는 항상 호스트가 스트림을 지정해서 보내주어야 한다.

6 결 론

본 연구는 Amoeba의 GC 효율을 높이기 위해 데이터의 랜섬웨어 감염 여부를 파악하여 배치시킬 블럭을 분리 및 결정짓는 기법인 Amoeba-Div을 제안한다. 본 연구의 실험 결과는 백업 데이터를 유지하기 위해 필요한 GC의 페이지 복사 횟수를 줄이는 경향을 보여주었고 결과적으로 성능 향상을 이끌어 낼 수 있다는 것을 설명한다.

Acknowledgement

이 논문은 2017년도 정부(과학기술정보통신부)의 재원으로 한국연구재단-차세대정보•컴퓨팅기술개발사업의 지원 (2017M3C4A7080243)과 삼성전자 반도체의 연구 지원을 받아 수행된 연구임.

참고 문헌

[1] S. Morgan, "Cybercrime report," *Cybersecurity Ventures. Arvutivõrgus: https://cybersecurityventures. com/2015-wp/wp-content/uploads/2017/10/2017-Cybercrime-Report. pdf (24 märts 2018)*, 2017.

[2] J. Huang, J. Xu, X. Xing, P. Liu, and M. K. Qureshi, "FlashGuard: Leveraging intrinsic flash properties to defend against encryption ransomware," in *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, pp. 2231–2244, ACM, 2017.

[3] S. Baek, Y. Jung, A. Mohaisen, S. Lee, and D. Nyang, "SSD-Insider: Internal defense of solid-state drive against ransomware with perfect data recovery," in *2018 IEEE 38th International Conference on Distributed Computing Systems (ICDCS)*, pp. 875–884, IEEE, 2018.

[4] D. Min, D. Park, J. Ahn, R. Walker, J. Lee, S. Park, and Y. Kim, "Amoeba: An autonomous backup and recovery ssd for ransomware attack defense," in *IEEE Computer Architecture Letters*, IEEE, 2018.

[5] N. Scaife, H. Carter, P. Traynor, and K. R. Butler, "Cryptolock (and drop it): stopping ransomware attacks on user data," in *Distributed Computing Systems (ICDCS), 2016 IEEE 36th International Conference on*, pp. 303–312, IEEE, 2016.

[6] V. Prabhakaran and T. Wobber, "Ssd extension for disksim simulation environment," *Microsoft Reseach*, 2009.

[7] T. MICRO, "Erebus linux ransomware: Impact to servers and countermeasures," 2017. https://www.trendmicro.com/vinfo/us/security/news/cyber-attacks/erebus-linux-ransomware-impact-to-servers-and-countermeasures.

[8] J.-U. Kang, J. Hyun, H. Maeng, and S. Cho, "The multi-streamed solid-state drive.," in *HotStorage*, 2014.