JKIICE Journal of the Korea Institute of Information and Communication Engineering

한국정보통신학회논문지 Vol. 25, No. 10: 1397~1402, Oct. 2021

Overlay2 파일 시스템의 소스 보호 방법에 관한 연구

한성화*

Overlay2 file system's Source Protection Methodology

Sung-Hwa Han*

*Professor, Department of Information Security, Tongmyung University, Busan, 48520 Korea

요 약

Overlay2 파일 시스템은 다수의 directory를 하나로 통합하여 mount 하는 union 파일 시스템 중의 하나이다. 이 overlay2 일 시스템 마운트에 생성되는 write-able layer는 source directory에 독립적으로 동작하는 특징이 있어 application 배포를 위한 컨테이너 플랫폼 등에 많이 사용되고 있다. 그러나 overlay2 파일 시스템은 mount에 사용되는 source directory에 있는 파일을 임의 변조할 경우, 그 변조 내용이 write-able layer에 적용되는 보안 취약점이 있다. 본 연구에서는 보안 취약점을 제거하기 위한 overlay2 파일 시스템의 source directory 보호 기술을 제안하였다. 제안하는 아키텍처에 따라 실증 구현 후 source directory 보호 기능을 검증한 결과, 본 연구에서 제안하는 보호 기술은 실효적이라고 판단되었다. 다만 본 연구에서 제안하는 방법은 수동적인 보호 방식이므로, 이를 운영체제 레벨에서 자동 보호하기 위한 후속 연구가 필요하다.

ABSTRACT

The overlay2 file system is one of the union file systems that mounts multiple directories into one. The source directory used for this overlay2 file system mount has a characteristic that it operates independently of the write-able layer after mounting, so it is often used for container platforms for application delivery. However, the overlay2 file system has a security vulnerability that the write-able layer is also modified when file in the source directory is modified. In this study, I proposed the overlay2 file system protection technology to remove the security vulnerabilities of the overlay2 file system. As a result of empirically implementing the proposed overlay2 file system protection technology and verifying the function, the protection technology proposed in this study was verified to be effective. However, since the method proposed in this study is a passive protection method, a follow-up study is needed to automatically protect it at the operating system level.

키워드 : 오버레이 파일 시스템, 접근통제, 디렉토리 보호, 마운트, 유니온 파일 시스템

Keywords : Overlay file system, Access control, Directory protection, Mount, Union file system

Received 9 August 2021, **Revised** 3 September 2021, **Accepted** 13 September 2021

*** Corresponding Author** Sung-hwa Han(E-mail:shhan@tu.ac.kr, Tel:+82-51-629-1285)
Professor, Department of Information Security, Tongmyung University, Busan, 48520 Korea

Open Access http://doi.org/10.6109/jkiice.2021.25.10.1397 print ISSN: 2234-4772 online ISSN: 2288-4165

Ⅰ. 서 론

Overlay2 file system은 union file system중의 하나로, linux/uninx 운영체제에서 지원하는 uninn mount를 file system으로 확장한 기술이다. 하나 이상의 source directory를 file system으로 mount 한 상태에서 다른 directory를 같은 file system으로 mount 하여, 통합된 단일 file system으로 활용하는 기술이다. 기존에는 linux/unix 운영체제에서만 활용되어왔으나 현재는 windows 운영체제에서도 그 연구를 진행하고 있다.

Overlay2 file system은, mount로 인해 생성된 write-able layer가 mount에 사용되는 source directory에 독립적으로 동작하는 것이 가장 큰 장점이다. 이러한 장점으로 인해 중요 파일의 무결성을 유지하면서 독립적인 write, execute 기능을 활용하는 Docker 등의 플랫폼에서 많이 사용되고 있다[1].

Overlay2 file system은 많은 장점을 제공하지만, 정보보호 관점에서는 취약점이 있다. Overlay2 file system으로 mount 할 때 사용되는 source directory에 존재하는 파일을 임의 변조 한 경우, 해당 변조 내용이 write-able Layer에 그대로 반영되는 단점이 있다.

본 연구에서는, overlay2 file system의 단점을 제거할 수 있는 보안 기술을 제안한다. 제안하는 보안 기술의 실효성을 검증하기 위하여, 실제 구현하고 제공하는 보안 기능을 검증한다.

Ⅱ. 관련 연구

2.1. Overlay2 file system

Overlay2 file system은 union mount를 실체화한 대표적인 file system이다[2]. Union mount는 동일 경로로 여러 file system을 동시에 mount 하는 것으로, 처음 mount 한 상태의 file system을 그대로 유지하고 다음에 mount 한 file system을 over-wrapping 한다. 중첩되지 않은 file은 유지하되, 중복된 file은 마지막에 mount 한 file로 over-wrapping 한다[3].

그림 1은 overlay2 file system의 mount 구조를 나타낸다. 사용자는 mount 할 source directory를 선택하여 mount 하면, mount logic에 의해 사용자에게 하나의 view를 제공한다. Mount에 사용된 source directory는 read-only 권한만 적용되며, 다른 overlay2 file system의 source directory로 사용될 수 있다[4].

Overlay2 file system을 mount 하였을 때 생성되는 write-able layer는 temporary 한 virtual file system이다. 사용자에게 제공되는 read·write layer는 mount 직후에 생성되고, unmount 후에는 모두 삭제된다. 사용자가 union file system에 file copy나 modify 한 내역은 모두 read·write layer에 저장되고, 별도의 directory로 commit 하지 않는 이상 저장되지 않는다. Commit 할 때 사용되는 target directory도 source directory에 over-wrapping 될 수 없다[5].

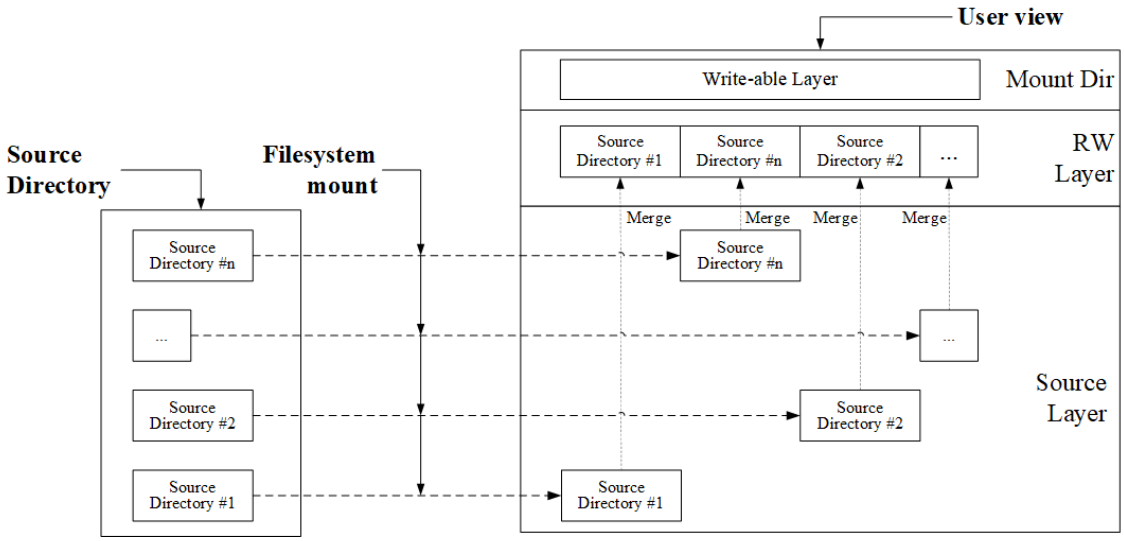

Fig. 1 Overlay2 file system mount structure

2.2. Overlay2 file system의 보안 취약점

모든 정보서비스는 보안 원칙에 따라 안전한 환경에서 제공되어야 한다. Overlay2 file system을 사용할 때에도 마찬가지로 하는 정보서비스도 안전해야 한다. 그러나 overlay2 file system은 자체적으로는 안전하지 않다.

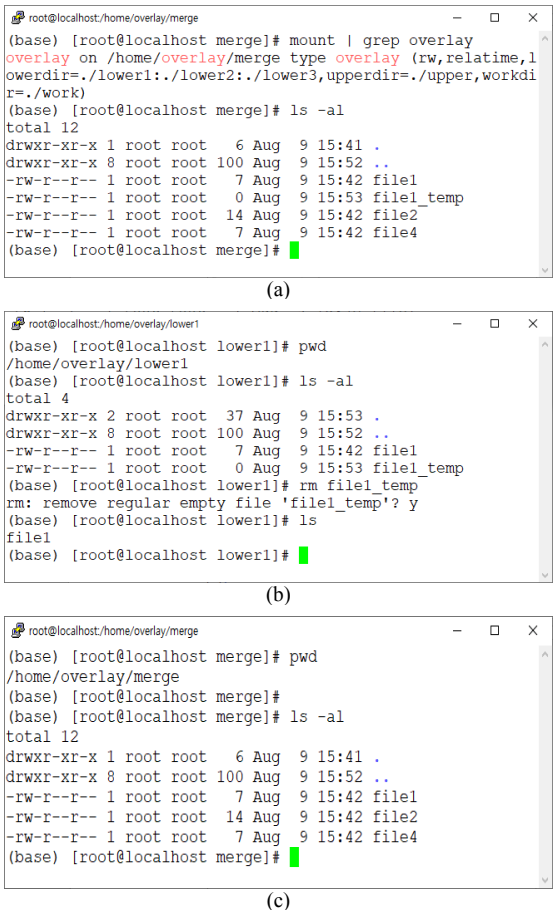

Fig. 2 Overlay2 file system's vulnerability

그림 2는 overlay2 file system의 보안 취약점을 악용하는 시나리오를 보여준다. (a)에서는 3개의 source directory를 overlay2 file system으로 mount하였다, 이 상태에서 (b)와 같이 다른 계정으로 source directory의 파일을 임의 변조·삭제하면, 해당 파일은 (c)와 같이 write-able layer에 반영된다.

이와 같이 overlay2 file 시스템은 mount 이후에 source directory에 대한 별도의 보안 조치는 하지 않고 있다. 이러한 보안 취약점을 공격자가 악용한다면, overlay2 file system을 활용하는 정보서비스의 중요 내용에 접근하여 정보를 유출하거나 인가되지 않은 변경으로 정보서비스에 오동작을 발생시킬 수 있다. 특히, overlay2 file system은 container platform에서 다양한 application 배포에 활용되므로, 이러한 취약점을 악용할 경우 같은 container image를 사용하는 모든 application에 영향을 미칠 수 있다[6].

Legacy system에서는 이러한 비인가 접근을 차단하기 위하여 SELinux나 AppArmor등의 보안 솔루션을 사용하여 중요 파일을 보호하고 있다[7, 8]. 그러나 이러한 보안 솔루션은 overlay2 file system 사용을 고려하지 않기 때문에, 이 보안 취약점을 개선할 수 없다.

이와 같은 보안 환경을 종합하면, overlay2 file system을 사용하는 정보서비스는, 안전한 환경에서 운영된다고 할 수 없다.

Ⅲ. Overlay2 file system 보호 기술

3.1. Overlay2 file system에 대한 보안 요구사항

Overlay2 file system을 mount 하였을 때 사용자가 접근할 수 있는 대상은 write-able layer이므로, 보안 원칙에 따라 이를 생성하기 이전에 source directory에 대한 접근은 사용자에게 허용되어야 한다[9, 10]. 다만 overlay2 file system으로 mount 한 후 source directory 내 파일의 임의 변경은 mount 되었을 때의 write-able layer에 반영되므로, source directory에 대한 보호는 mount 된 이후에 보호되어야 한다[11, 12].

3.2. Overlay2 file system 보호 아키텍처

본 연구에서는 union file system의 목표를 달성하면서, overlay2 file system을 보호하기 위한 그림 3의 source directory에 대한 접근통제 기능을 제공하는 보안 기술을 제안한다. 본 연구에서 제안하는 overlay2 file system의 source directory 보호 기능을 제공하기 위한 architecture는 크게 4개의 module로 구성되어 있다.

Policy Identifier는 사용자의 overlay2 file system의 mount를 모니터링하고, mount에 사용된 source directory에 대한 정보를 kernel에 전달한다.

Kernel Policy Manager는 Policy Identifier가 전달한 source directory 정보와 mount directory 정보를 수신하

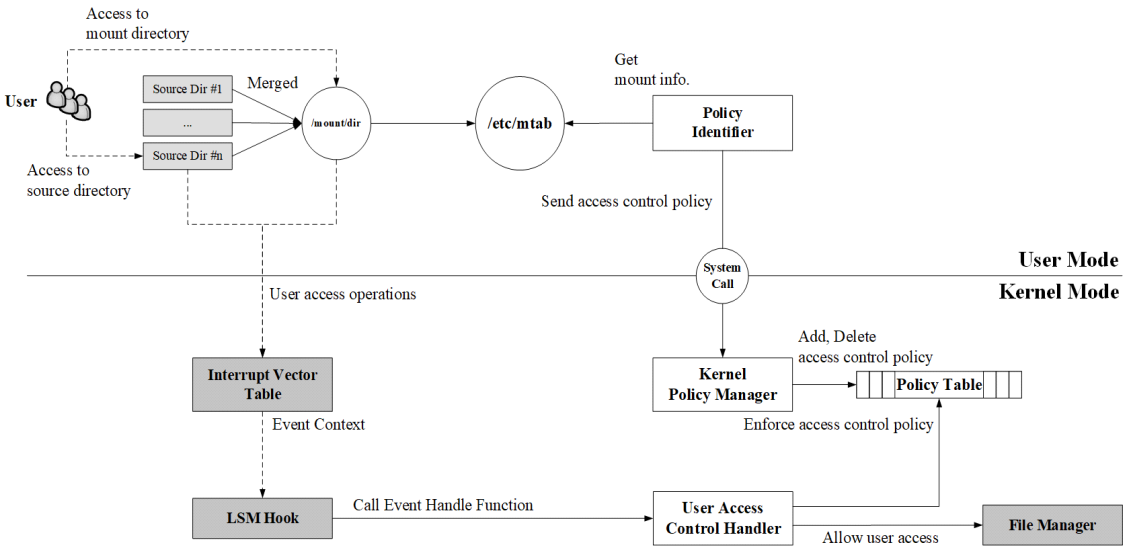

Fig. 3 Overlay2 file system's vulnerability

여, 각각에 대한 사용자의 file I/O를 filtering 하는 접근 통제 정책을 생성하여 Policy Table에 등록한다.

Policy Table은 source directory 접근을 차단하기 위한 hashtable 형태의 access control policy structure이다.

User Access Control Handler는 사용자의 File I/O가 발생하였을 때, 이에 대한 접근 경로를 추출하고 Policy Table과 비교하여 접근 허용 여부를 결정한다. 사용자의 접근 경로가 source directory로 접근하는 경우에는 차단한다.

3.2. source directory 보호 기능 동작 순서

Overlay2 file system의 source directory를 보호하기 위해서는, 사용자의 overlay2 file system mount를 모니터링하여 보호 대상 directory path를 식별하는 과정과 식별된 directory에 대한 접근통제 정책을 생성하여 이를 kernel에 전달하는 과정, 등록된 접근통제 정책을 적용하는 과정, 마지막으로 overlay2 file system을 unmount 하였을 때 등록된 접근통제 정책을 삭제하는 과정으로 나누어진다.

Overlay2 file system이 생성되었을 때에는, mount 된 file system에 대한 정보는 시스템에 등록된다. Policy Identifier는 이 정보를 모니터링하여, mount된 overlay2 file system에 대한 세부 정보를 확인한다. 이후 확인된 정보를 Kernel Policy Manager에 전달한다.

Kernel Policy Manager는 이를 수신하여, source directory에 대해서는 모든 사용자의 접근을 차단하는 정책을 생성하여 Policy Table에 등록한다.

사용자의 file I/O는 모두 운영체제의 Interrupt Vector Table에 등록된다. 이 Interrupt Vector Table에는 사용자의 file I/O에 대한 세부 정보가 등록되는데, 사용자가 접근하는 대상 경로도 포함되어 있다. LSM(Linux Security Module)은 사용자가 정의한 file I/O Handler가 존재할 때 이를 등록하는 interface로, file I/O가 발생하게 되면 이벤트에 대한 정보는 LSM을 경유하여 User Access Control Hanlder에 전달된다.

여기서 사용자의 파일 접근이 overlay2 file system의 source directory에 대한 접근인 경우, User Access Control Hanlder 에서는 이 파일 접근을 차단한다.

사용자에 의해 overlay2 file system을 unmount한다면, 더 이상 source directory에 대한 비인가 접근 차단 정책은 적용되지 말아야 한다. 그러므로 Policy Identifier는 overlay2 file system의 unmount를 확인하면, 이에 대한 정보를 전달하여 Policy Table에 등록된 접근통제 정책을 삭제한다.

Ⅳ. Overlay2 file system 보호 기술 검증

4.1. 기능 검증 환경 및 검증 항목

본 연구에서 제안하는 overlay2 file system의 source directory 보호 기능 제공을 위한 아키텍처의 실효성을 확인하기 위하여, 본 연구에서 제안하는 아키텍처에 따라 실증 구현 후 기능 검증을 수행한다.

기능 검증 환경은 본 연구에서 제안하는 아키텍처를 CentOS 8.3에서 구현하였다. 커널 모듈로 Kernel Policy Manager와 User Access Control Handler를 통합 구현하였으며, Application daemon으로 Policy Identifier를 구현하였다. Kernel Policy Manager와 Policy Identifier의 통신은 System Call 호출 방식을 사용하였다.

실시간 모니터링 기능 검증을 위한 테스트 항목은 표 1과 같다.

Table. 1 Security function verification items

Test ID	Test Object
Func_Test_01	• Check overlay2 file system mount monitoring function • Check getting source directory's absolute path
Func_Test_02	• Check security policy to protect source directory transfer and set to kernel when overlay2 file system mounted
Func_Test_03	• Check security policy enforcement for user file access in source directory

4.2. 기능 검증 결과

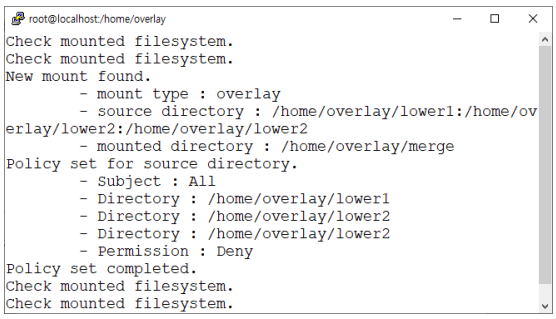

Fig. 4 Policy Identifier's access control setting verification

Func_Test_01을 확인한 결과, 그림 4와 같이 Policy Identifier는 overlay2 file system이 시스템에서 mount되었을 때 이에 대한 source directory에 대한 정보를 정상 수집하는 것을 확인하였다.

Func_Test_02를 확인한 결과, 그림 5와 같이 Kernel Policy Manager는 Policy Identifier가 전달한 source directory 정보를 수신하여, 이를 access control policy로 등록하는 것을 확인하였다.

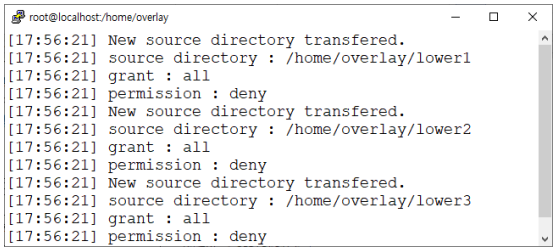

Fig. 5 Kernel Policy Manager's access control registering function verification

Func_Test_03을 확인한 결과 그림 6과 같이, overlay2 file system이 mount 된 상태에서 사용자가 source directory에 접근하고자 할 때, 이를 User Access Control Handler가 차단함을 확인하였다.

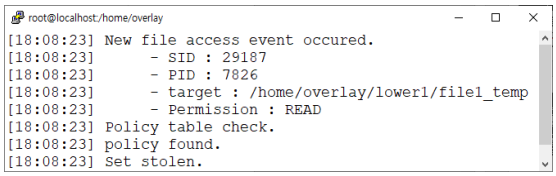

Fig. 6 User Access Control Handler's access control enforcing function verification

Ⅴ. 결 론

Overlay2 file system는, mount 하였을 때 실제 사용되는 write-able layer는 source directory에 전혀 영향을 미치지 않는다는 장점으로 인해 그 활용도는 계속 증가할 것이다. 그러나 overlay2 file system의 Image layer의 임의 변경은 write-able layer에 즉시 반영되는 취약점이 확인되었다.

본 연구에서는 이 취약점을 개선하기 위하여, overlaye2 file system이 mount 되었을 때, source directory에 대한 비인가 접근을 차단하기 위한 아키텍처를 제안하였으며, 이를 실증 구현 후 단위 기능 검증을 통해 제안한 아키텍처의 실효성을 확인하였다.

다만, 본 연구는 union file system 중의 하나인

overlay2 file system에 국한된 연구이므로, 여타 union file system인 AUFS나 overlay file system으로 확장하기 위한 추가 연구가 필요하다.

REFERENCES

[1] C. Zheng, L. Rupprecht, V. Tarasov, D. Thain, M. Mohamed, D. Skourtis, A. S. Warke, and D. Hildebrand, "Wharf: Sharing docker images in a distributed file system," in *Proceedings of the ACM Symposium on Cloud Computing* pp. 174-185, Oct. 2018.

[2] C. Wu and Q. A. Chen, "Research on Union file system for Linux and Its Performance Analysis," *Computer Knowledge and Technology*, 2013.

[3] D. Seybold, C. B. Hauser, G. Eisenhart, S. Volpert, and J. Domaschka, "The impact of the storage tier: A baseline performance analysis of containerized dbms," in *European Conference on Parallel Processing Springer, Cham*, pp. 93-105, Aug. 2018.

[4] L. Ma, S. Yi, and Q. Li, "Efficient service handoff across edge servers via docker container migration," in *Proceedings of the Second ACM/IEEE Symposium on Edge Computing*, pp. 1-13, Oct. 2017.

[5] C. Zheng and D. Thain, "Integrating containers into workflows: A case study using makeflow, work queue, and docker," in *Proceedings of the 8th International Workshop on Virtualization Technologies in Distributed Computing*, pp. 31-38, Jun. 2015.

[6] G. Kappes and S. V. Anastasiadis, "A user-level toolkit for storage I/O isolation on multitenant hosts," in *Proceedings of the 11th ACM Symposium on Cloud Computing*, pp. 74-89, Oct. 2020.

[7] W. Findlay, A. Somayaji, and D. Barrera, "bpfbox: Simple Precise Process Confinement with eBPF," in *Proceedings of the 2020 ACM SIGSAC Conference on Cloud Computing Security Workshop*, pp. 91-103, Nov. 2020.

[8] I. Borate and R. K. Chavan, "Sandboxing in linux: From smartphone to cloud," *International Journal of Computer Applications*, vol. 148, no. 8, 2016.

[9] M. Nieles, K. Dempsey, and V. Y. Pillitteri, "An introduction to information security," NIST special publication, vol. 800, no. 12, 2017.

[10] J. A. Bullock, G. D. Haddow, and D. P. Coppola, "Introduction to homeland security: Principles of all-hazards risk management," *Butterworth-Heinemann*, 2011.

[11] S. De Capitani di Vimercati, S. Paraboschi, and P. Samarati, "Access control: principles and solutions," *Software: Practice and Experience*, vol. 33, no. 5, pp. 397-421, 2003.

[12] P. Bellavista and A. Montanari, "Context awareness for adaptive access control management in IoT environments," *Security and Privacy in Cyber-Physical Systems: Foundations, Principles and Applications*, vol. 2, no. 5, pp. 157-178, 2017.

한성화(Sung-Hwa Han)

동명대학교 정보보호학과 교수
숭실대학교 공학박사
SW영향평가 전문위원
※관심분야 : IT융합보안, 시스템 보안, 네트워크 보안, 인공지능, 악성코드 탐지