Incident Analysis and Forensics in Docker Environments

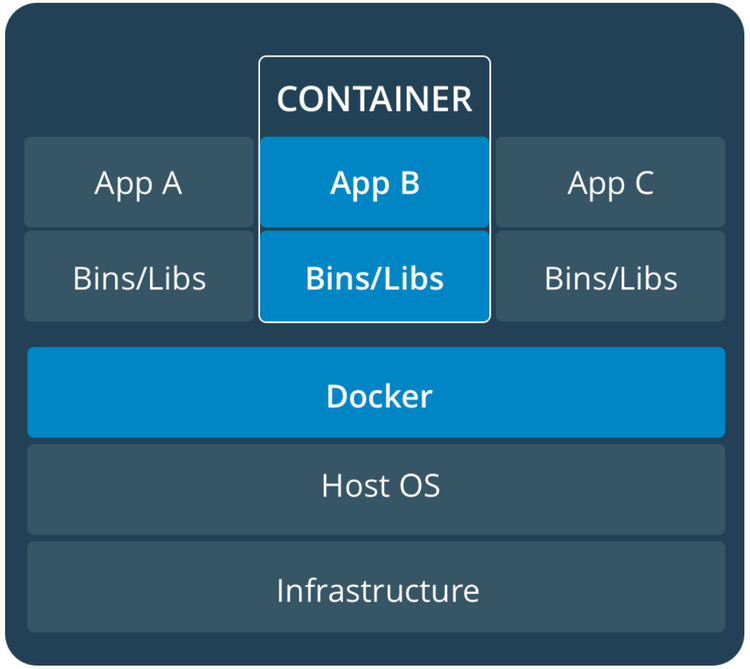
demantos

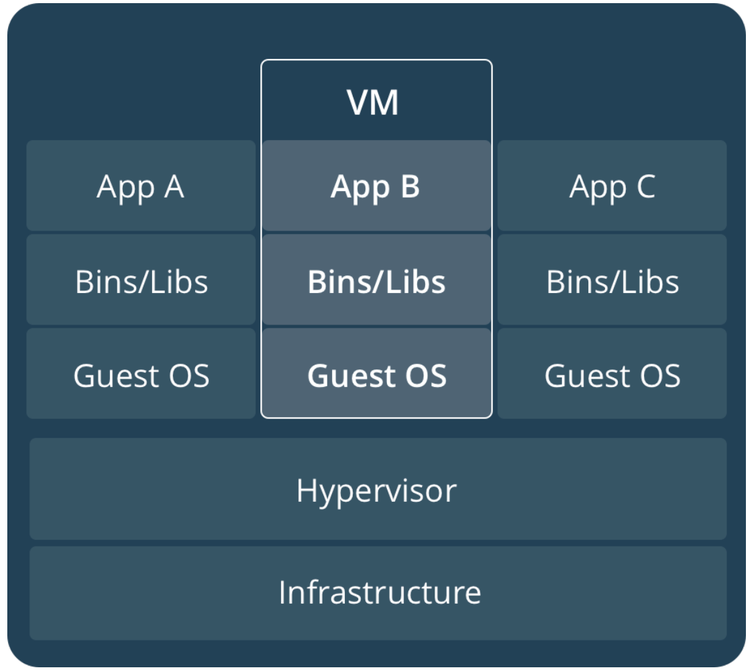
security.engineer@TOSS

목차

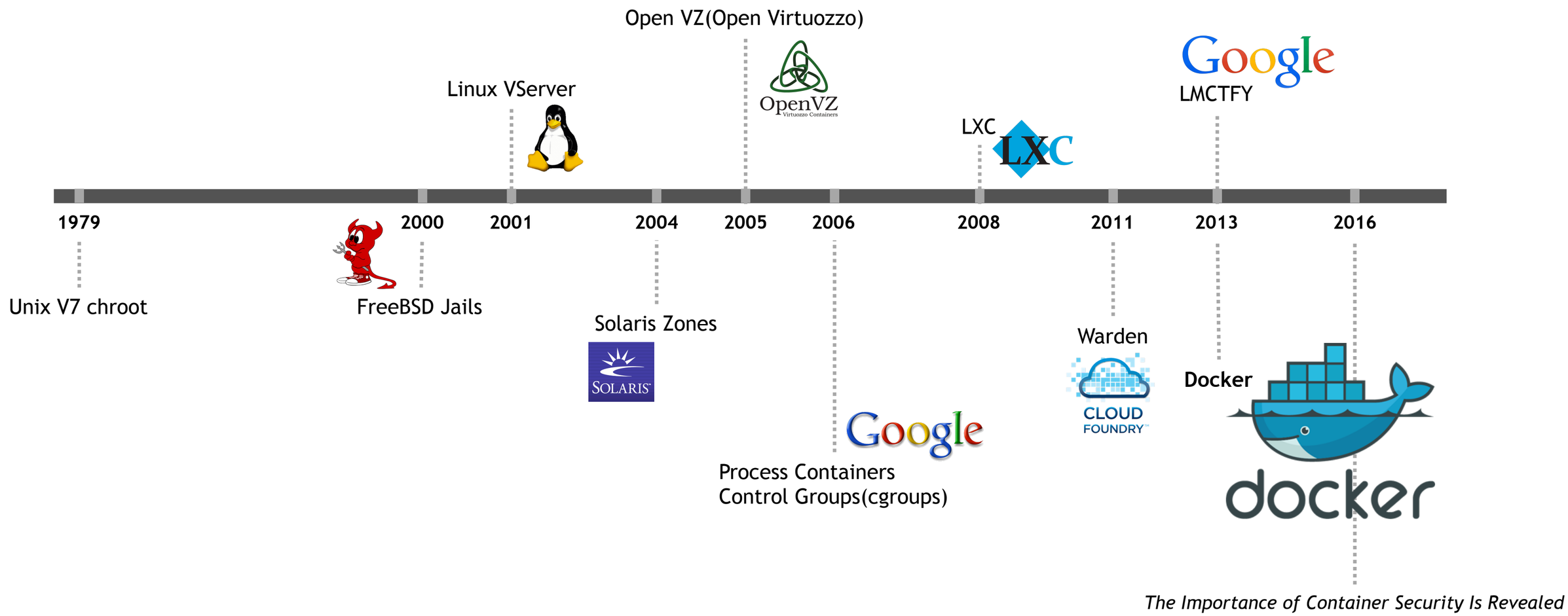
- Container
- Docker
- Forensic analysis on Docker hosts
- Considerations (a.k.a Painfull things)
- Conclusion

- **Container**
  - 소프트웨어를 한 컴퓨팅 환경에서 다른 컴퓨팅 환경으로 이동하면서도 안정적으로 실행하는 방법이 없을까?
  - 애플리케이션과 그 실행에 필요한 라이브러리, 바이너리, 구성 파일 등을 패키지로 묶어 배포
  - 운영체제 위에서 구현된 가상화, 즉 '운영체제 레벨 가상화'라고 부르기도 함
  - '크기와 속도' 면에서 '서버 가상화' 보다 훨씬 효율적임





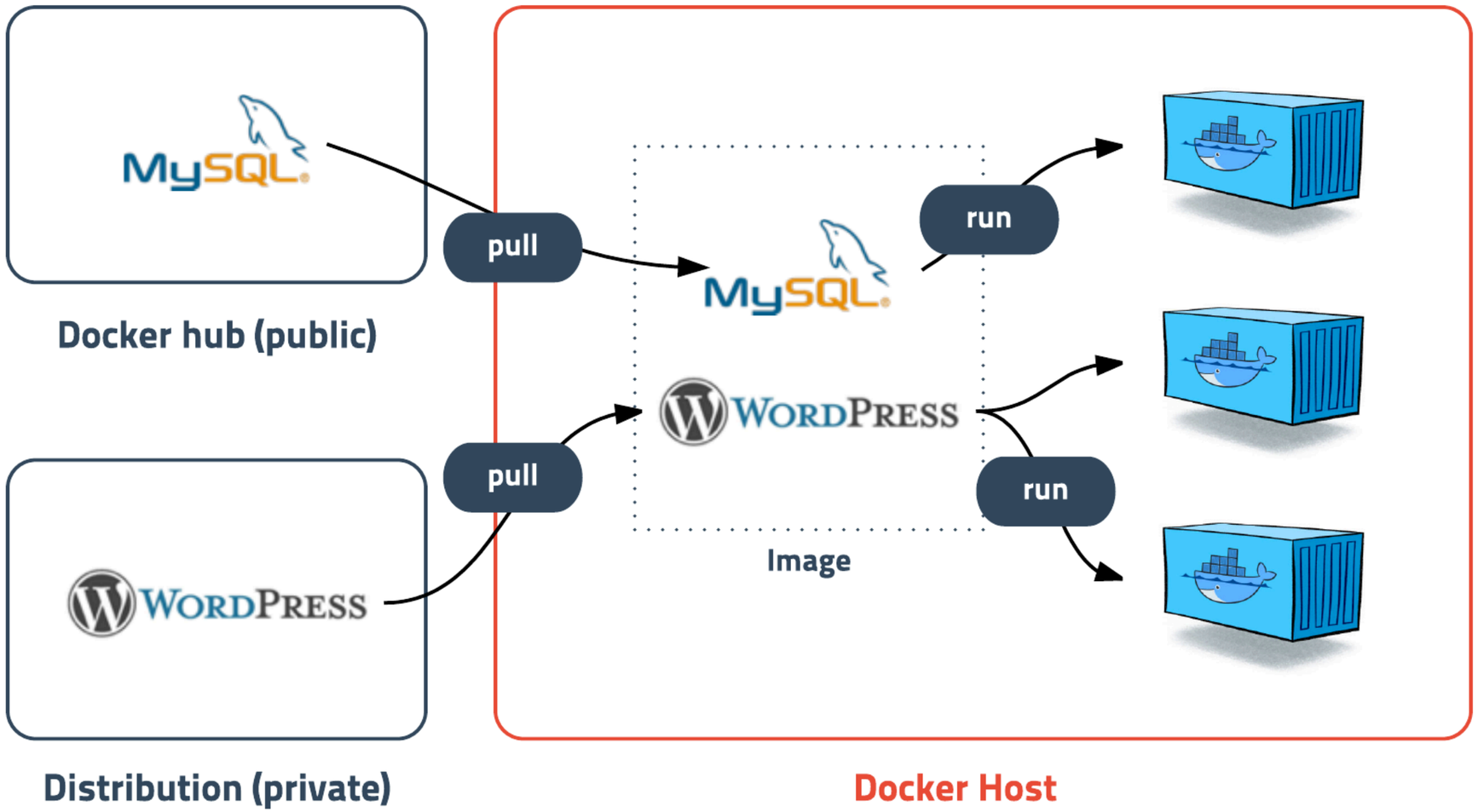
- **History of Container**



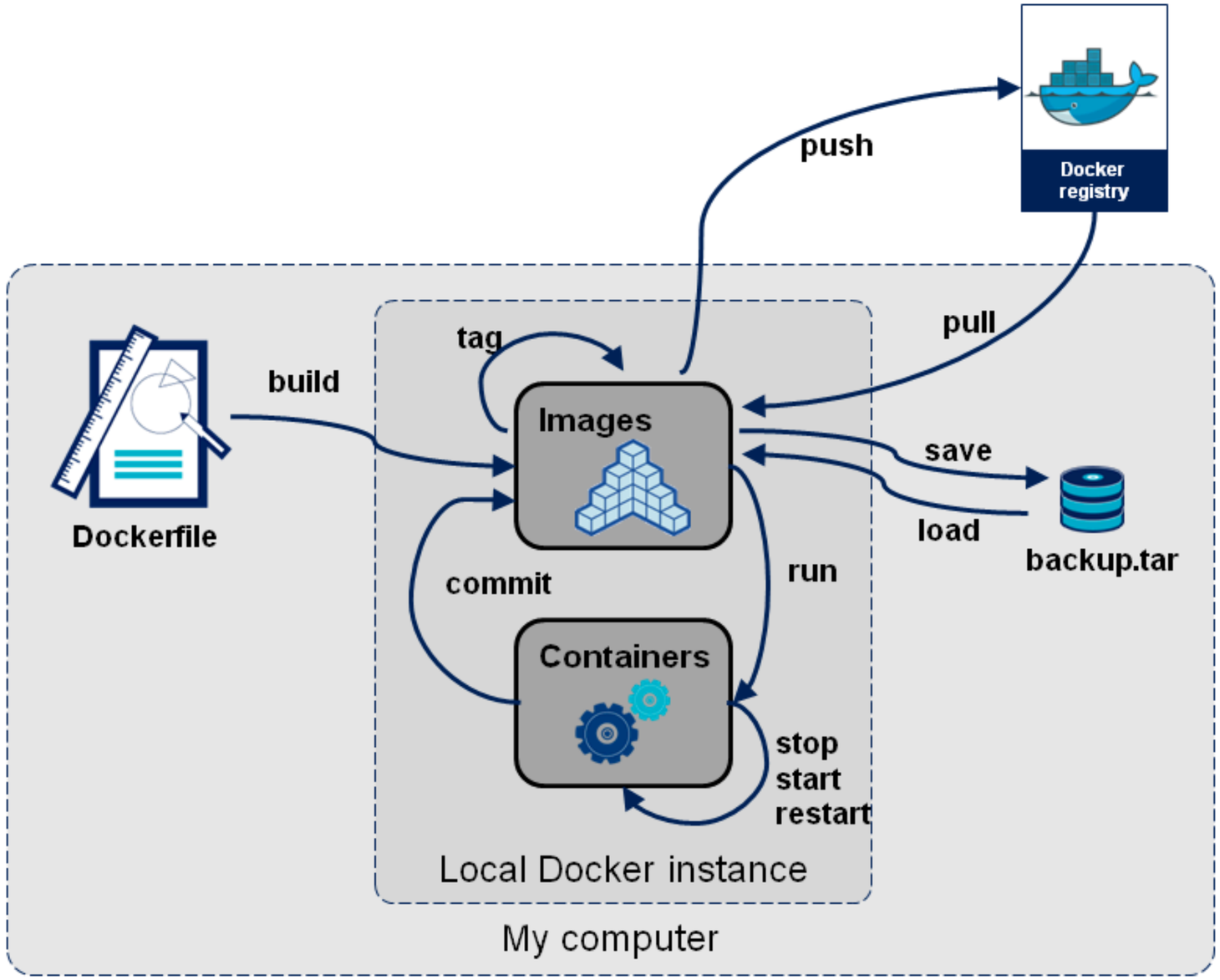
- **컨테이너 기반의 오픈소스 가상화 플랫폼**
- **2013년 3월 Pycon Conference에서 Solomon Hykes가 The future of Linux Containers 세션에서 공개**
- **다양한 프로그램, 실행환경을 컨테이너로 추상화하고 동일한 인터페이스를 제공하여 프로그램의 배포 및 관리를 단순화**
  - 백엔드 프로그램, 데이터베이스 서버, 메시지 큐 등 어떤 프로그램도 컨테이너로 추상화 가능
  - 조립PC, AWS, Azure, Google cloud등 어디에서든 실행 가능
  - 구글은 모든 서비스들이 컨테이너로 동작하고 매주 20억개의 컨테이너 구동 (2014년 자료)
    - ✓ https://speakerdeck.com/jbeda/containers-at-scale
- **LXC를 기반으로 시작**
  - 0.9버전에서는 자체적인 libcontainer 기술을 사용, 추후 runC기술에 병합

- **Image**
  - 컨테이너 실행에 필요한 파일과 설정값 등을 포함하고 있는 것
  - 상태값을 가지지 않고 변하지 않음 (Immutable)
  - 베이스 이미지 (Base Image)
    - ✓ 리눅스 배포판의 커널 영역을 제외한 **사용자 공간/영역만을 이미지에 포함한 파일**을 의미
    - ✓ Redis나 Nginx, PHP 등을 설치해서 베이스 이미지로 배포
  - 도커 이미지 (Docker Image)
    - ✓ 베이스 이미지에서 몇 가지를 라이브러리, 프로그램, 소스 파일등을 추가/설치/저장한 이미지
    - ✓ 베이스 이미지에서 추가되거나 변형되는 부분만 도커 이미지에 저장
    - ✓ **베이스 이미지 + 변형된 도커 이미지 + 변형된 도커 이미지 + ...**

- **Image**


MySQL
pull
Docker hub (public)
WordPress
pull
Distribution (private)
MySQL
WordPress
Image
run
run
Docker Host

- **Container**
  - 이미지가 실행된 상태
  - 이미지 하나로 수백개의 컨터이너 생성 가능
  - 실행 파일과 프로세스의 관계와 비슷함



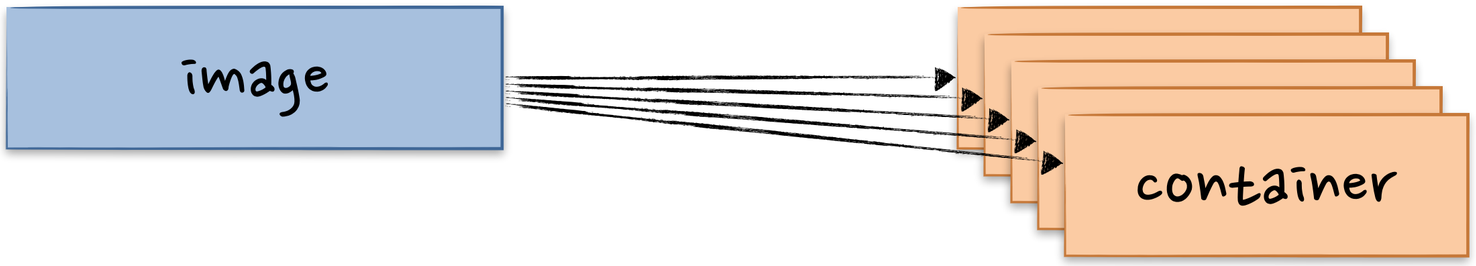
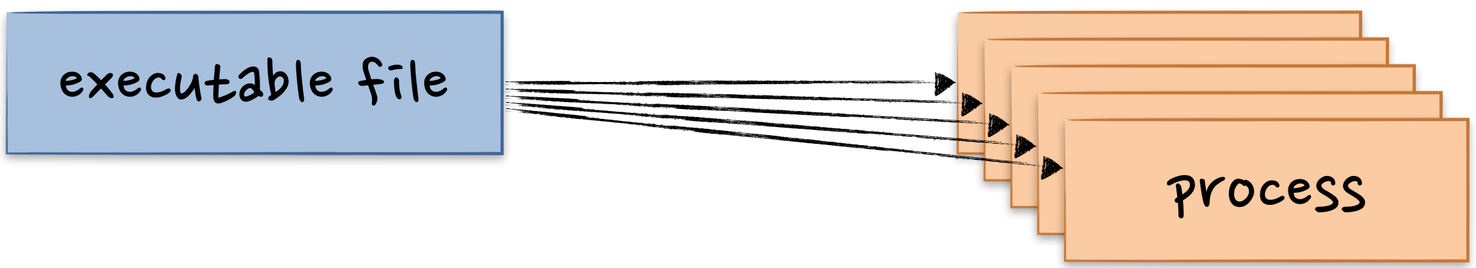
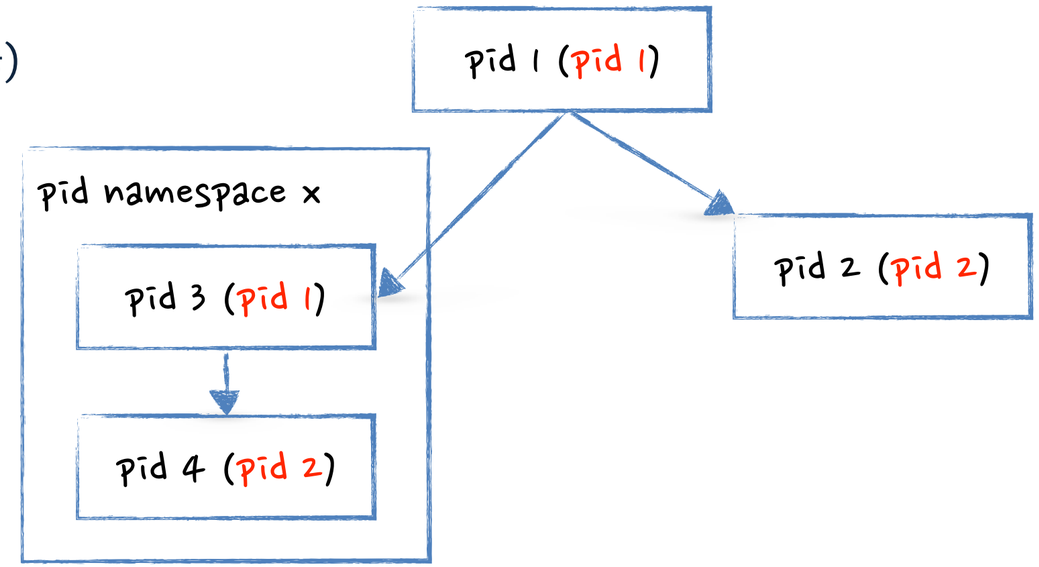


image
container

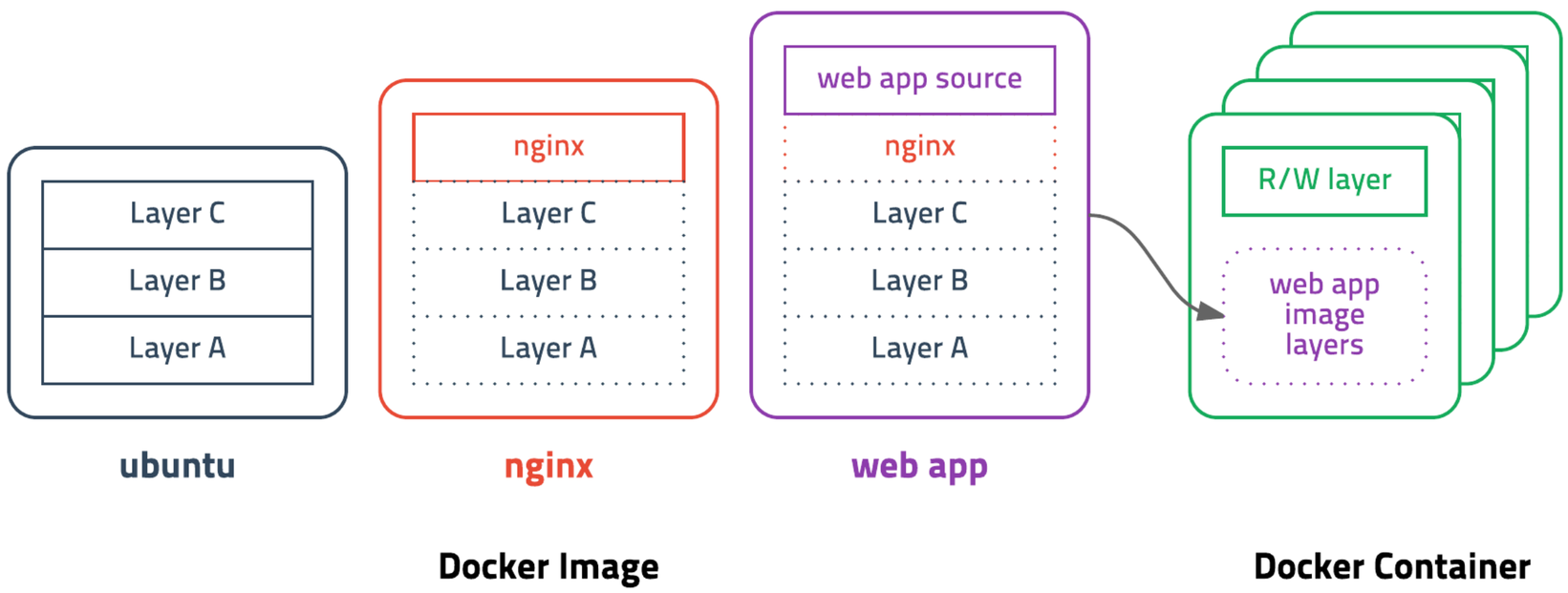

executable file
process

- **cgroups (Control Groups)**
  - 프로세스 또는 프로세스 그룹의 리소스 사용을 제어하고 제한하는 커널 기능
  - 사용자 공간을 설정하고 해당 프로세스를 관리하는 초기화 시스템인 systemd를 사용하여 이러한 격리된 프로세스를 더 효과적으로 제어할 수 있게 함
  - 제어 가능 리소스
    - ✓ Memory
    - ✓ CPU
    - ✓ I/O
    - ✓ Network
    - ✓ device node (/dev/)

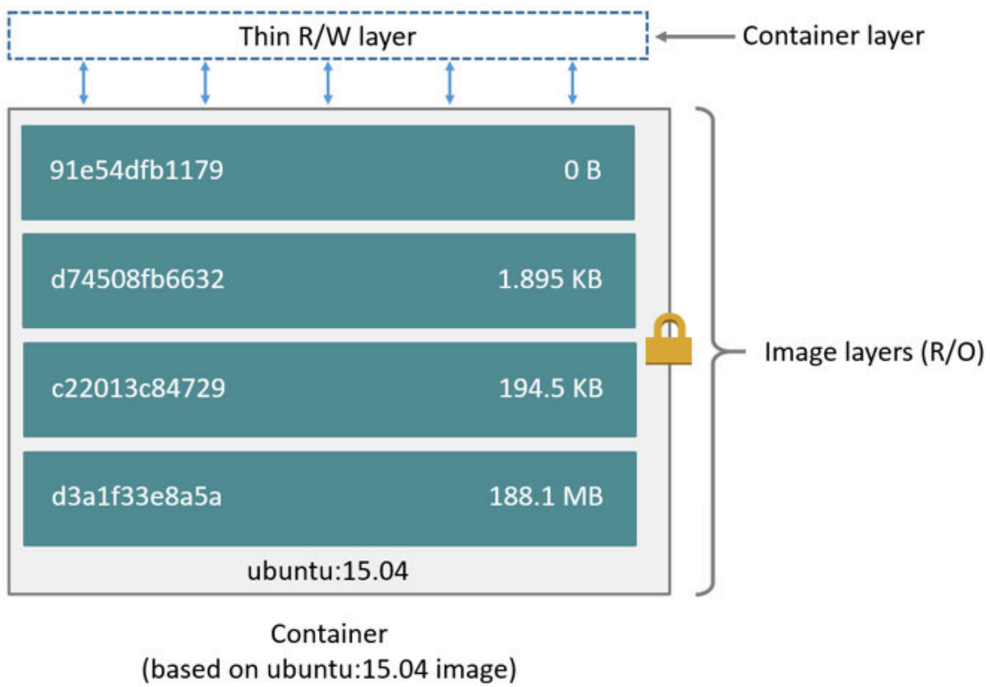
- **Namespaces**
  - 사용자와 그룹 ID를 네임스페이스별로 매핑
  - 컨테이너의 측면에서 이는 사용자와 그룹이 컨테이너 밖에서 특정 작업을 수행할 권한이 없어도 컨테이너 내에서 그러한 권한을 가질 수 있음을 의미
    - ✓ Jail의 개념과 비슷하지만 프로세스를 추가로 격리하고 보안을 강화한다는 점에서 Jail의 수정된 환경의 개념과 다름
  - 6개의 네임스페이스 존재
    - ✓ mnt (파일시스템 마운트) : 호스트 파일시스템에 구애받지 않고 독립적으로 파일시스템을 마운트하거나 언마운트 가능
    - ✓ pid (프로세스) : 독립적인 프로세스 공간을 할당
    - ✓ net (네트워크) : namespace간에 network 충돌 방지 (중복 포트 바인딩 등)
    - ✓ ipc (SystemV IPC) : 프로세스간의 독립적인 통신통로 할당
    - ✓ uts (hostname) : 독립적인 hostname 할당
    - ✓ user (UID) : 독립적인 사용자 할당



- **Layered Filesystem**
  - 유니온 파일 시스템을 이용하여 여러 개의 레이어를 하나의 파일시스템으로 사용
  - 여러 개의 읽기 전용 레이어로 구성되고 파일이 수정되고 추가되면 새로운 레이어(**R/W layer**) 생성
  - 컨테이너 생성시에도 레이어 방식을 사용하는데 기존의 이미지 레이어 위에 읽기/쓰기 레이어 추가
  - 이미지 레이어를 그대로 사용하면서 컨테이너가 실행중에 생성하는 파일이나 변경된 내용은 읽기/쓰기 레이어에 저장되므로 여러 개의 컨테이너를 생성해도 최소한의 용량만 사용



- **Union File System**
  - 다른 파일 시스템 및 디렉토리(called branches)의 콜렉션을 단일 논리 파일 시스템으로 통합하는 파일 시스템
  - 중첩된 디렉토리 중에서 같은 경로를 가진 디렉토리들의 내용은 하나로 합쳐진 디렉토리로 인식
  - Copy-on-Write
    - ✓ 공통되는 데이터들을 그대로 사용
    - ✓ 변경되는 정보만 컨테이너 레이어에 저장 -> 디스크 공간을 효율적으로 사용할 수 있음



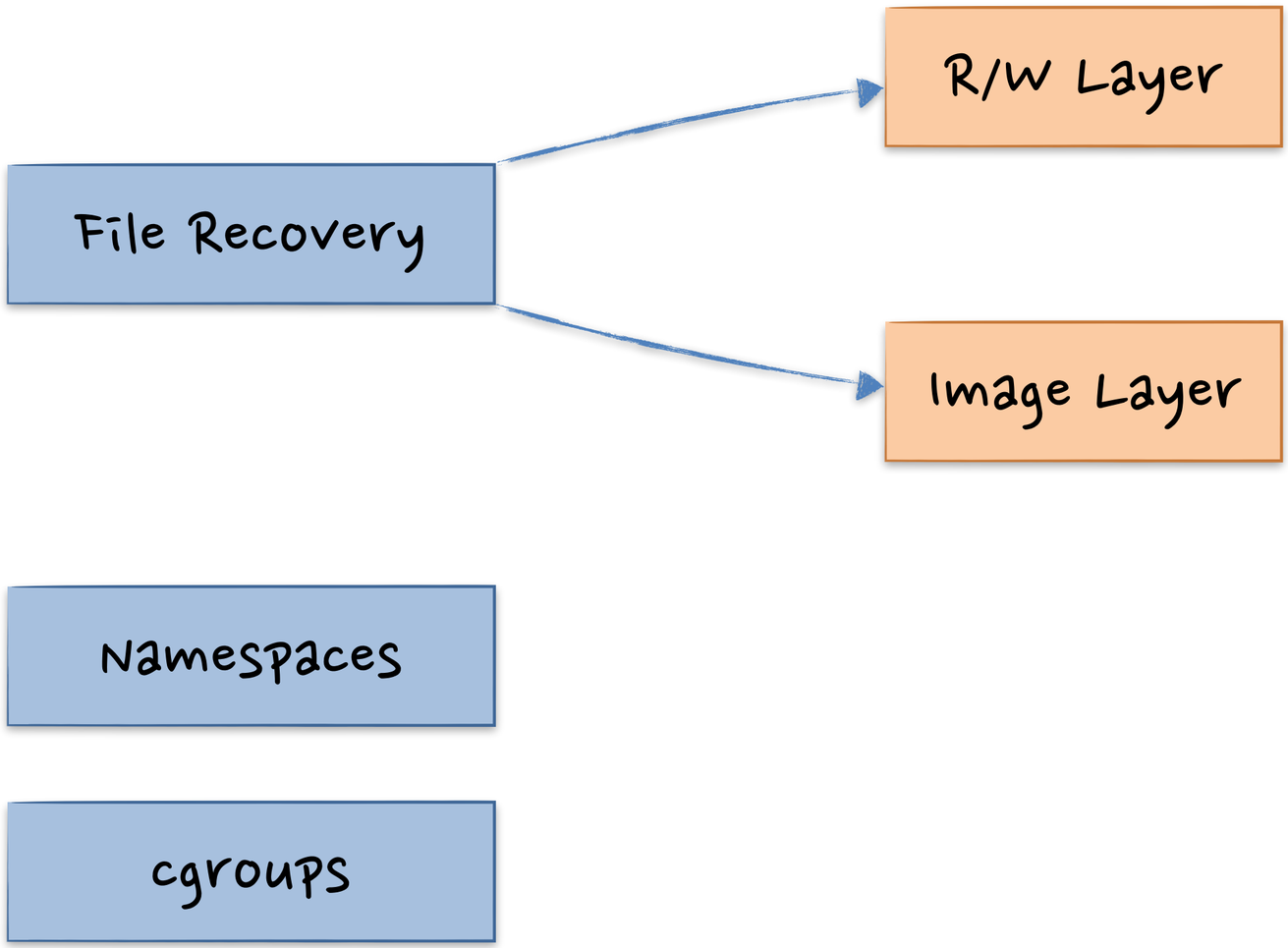
keep this things into your brain...

- when you upload file?
- when you modify uploaded file through another uploaded file?
- when you modify system file through uploaded file?

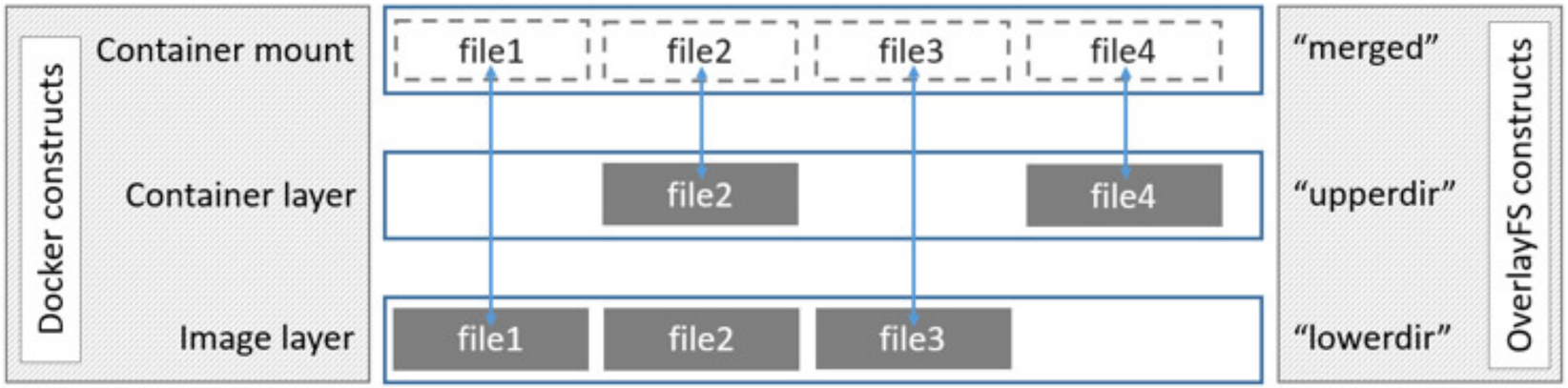
traditional forensic analysis method?

or

new forensic analysis method?


R/W Layer
File Recovery
Image Layer
Namespaces
cgroups

- **File Recovery**
  - 고려사항
    - ✓ 전통적인 운영체제처럼 파일 시스템이 블럭 디바이스에 매핑되어 있지 않고, **layered 파일 시스템 구조**를 가지고 있음
    - ✓ 호스트 시스템의 디스크 덤프에서 파일을 찾는 것은 도커 이미지에서도 찾을 수 있음
    - ✓ 도커 이미지는 R/W Layer를 이용해서 여러 개의 컨테이너에서 사용될 수 있음
    - ✓ 파일과 컨테이너를 연관 짓기 위해 런타임 정보나 특정 설정 정보가 필요



- ☑ Which image provided the given file?
- ☑ Which containers used the given file?
- ☑ Was the file deleted at container level?

- **File Recovery**
  - 런타임 환경에서 docker ps 명령으로 실행된 컨테이너 확인 가능

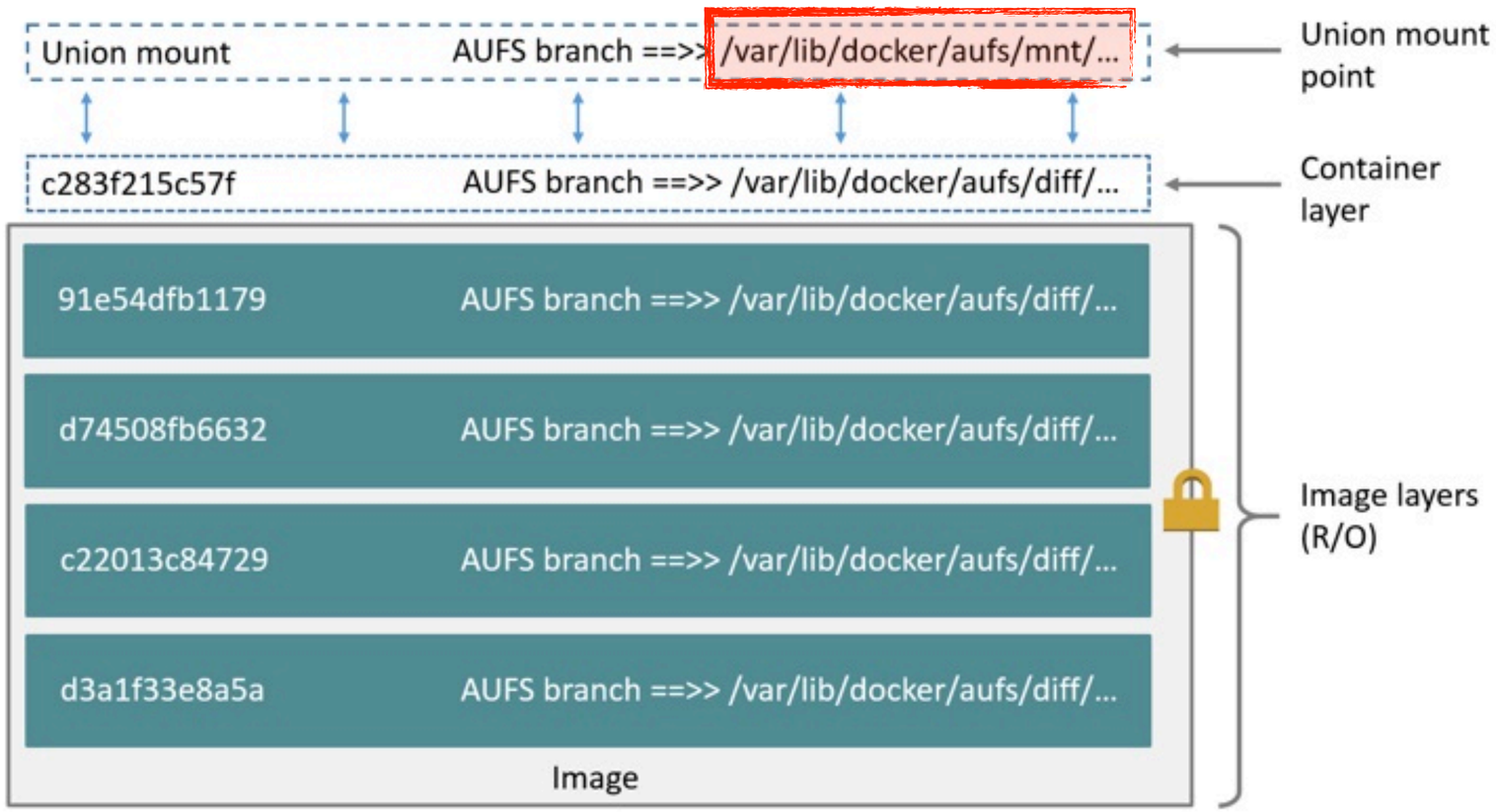
```
root@gotham:~#
root@gotham:~# docker ps -a
CONTAINER ID        IMAGE                    COMMAND                  CREATED             STATUS                     PORTS                                                                    NAMES
24e3b2dffeb6        mysql                    "docker-entrypoint..."   4 hours ago         Exited (1) 4 hours ago                                                                              mymy
9c740e3f83af        fios-nginx-php-fpm:0.1   "docker-php-entryp..."   31 hours ago        Exited (0) 31 hours ago                                                                             mystifying_fermi
38412d1a76a8        willdurand/elk           "/usr/bin/supervis..."   31 hours ago        Exited (0) 31 hours ago                                                                             elk_stack
dda3ed9e5082        willdurand/elk           "/usr/bin/supervis..."   31 hours ago        Exited (0) 31 hours ago                                                                             sleepy_colden
6982c8116d71        willdurand/elk           "/usr/bin/supervis..."   31 hours ago        Exited (0) 31 hours ago                                                                             suspicious_jepsen
aaf706d65311        nginx                    "nginx -g 'daemon ..."   31 hours ago        Exited (0) 3 hours ago                                                                              mynginx
01e4296ad690        fios-nginx-php-fpm:0.1   "docker-php-entryp..."   2 days ago          Up 3 hours                 0.0.0.0:80->80/tcp, 0.0.0.0:443->443/tcp, 0.0.0.0:9000->9000/tcp   5th_fios
0f55c35a92f9        fios-nginx-php-fpm:0.1   "docker-php-entryp..."   2 days ago          Exited (0) 2 days ago                                                                               tender_dubinsky
root@gotham:~#
root@gotham:~# docker inspect 01e4296ad690
[
    {
        "Id": "01e4296ad690aa3d88b84bf35f80731739c1825d9033e3055262ec4df602263a",
        "Created": "2018-06-12T09:52:28.742364385Z",
        "Path": "docker-php-entrypoint",
        "Args": [
            "/start.sh"
        ],
        "State": {
            "Status": "running",
            "Running": true,
            "Paused": false,
            "Restarting": false,
            "OOMKilled": false,
            "Dead": false,
            "Pid": 116588,
            "ExitCode": 0,
            "Error": "",
            "StartedAt": "2018-06-14T14:19:33.094149038Z",
            "FinishedAt": "2018-06-14T14:19:22.502680523Z"
        },
        "Image": "sha256:4dbb6370e33a8d8f4136f769668d238851c8144b3e79a94c2d6da30ae4dc14c2",
        "ResolvConfPath": "/var/lib/docker/containers/01e4296ad690aa3d88b84bf35f80731739c1825d9033e3055262ec4df602263a/resolv.conf",
        "HostnamePath": "/var/lib/docker/containers/01e4296ad690aa3d88b84bf35f80731739c1825d9033e3055262ec4df602263a/hostname",
        "HostsPath": "/var/lib/docker/containers/01e4296ad690aa3d88b84bf35f80731739c1825d9033e3055262ec4df602263a/hosts",
        "LogPath": "/var/lib/docker/containers/01e4296ad690aa3d88b84bf35f80731739c1825d9033e3055262ec4df602263a/01e4296ad690aa3d88b84bf35f80731739c1825d9033e3055262ec4df602263a-json.log",
        "Name": "/5th_fios",
        "RestartCount": 0,
        "Driver": "aufs",
        "MountLabel": "",
        "ProcessLabel": "",
        "AppArmorProfile": "",
        "ExecIDs": [
            "7b646b23e3f53108074b81056df24c2548b4fcfc2452fef1a4916503a92e1514"
        ],
        "HostConfig": {
            "Binds": null,
```

- **File Recovery**
  - 런타임 분석이 불가능한 경우 -> 설정 파일 이용
    - ✓ 기본 도커 설정 경로 : /var/lib/docker
    - ✓ 컨테이너 설정 : /var/lib/docker/container/{ContainerID}
    - ✓ 컨테이너 실행 여부 확인
      - config.v2.json -> Running : true
      - /var/lib/docker/container/{ContainerID}/shm -> perm 1777
  - 파일이 컨테이너 내에서 삭제된 경우 다음과 같은 두 가지 가능성이 존재함
    - ✓ 삭제된 파일이 R/W 레이어에 있었던 경우
      - 일반적인 운영체제 매카니즘에 의해서 파일 시스템 상에서 파일이 삭제됨
    - ✓ 삭제된 파일이 이미지 레이어에 있었던 경우
      - 삭제된 파일의 레퍼런스는 R/W 레이어에 있으나, 파일은 이미지 레이어에 남아 있음

- **Recovery of Files of the R/W Layer**
  - R/W layer의 파일 삭제는 호스트 파일 시스템에서 삭제된 것이다.
  - 메타데이터는 파일 시스템에 남아 있으며 호스트 파일 시스템에 따라 복구할 수 있는 방법이 다르다.
    - ✓ ext3, ext4, zfs, …
  - 즉, 전통적인 디스크 포렌식 분석 기법으로 삭제된 파일을 복구할 수 있다.
    - ✓ File Carving
    - ✓ File System Analysis

- **Recovery of Files of the R/W Layer**
  - File System Analysis
    - ✓ MFT, inode table과 같은 파일 시스템 구조를 이용해서 분석
    - ✓ 이러한 정보를 바탕으로 삭제된 파일 복구 가능
      - sleuthkit
  - Docker Storage Backend
    - ✓ AUFS
    - ✓ Overlay2

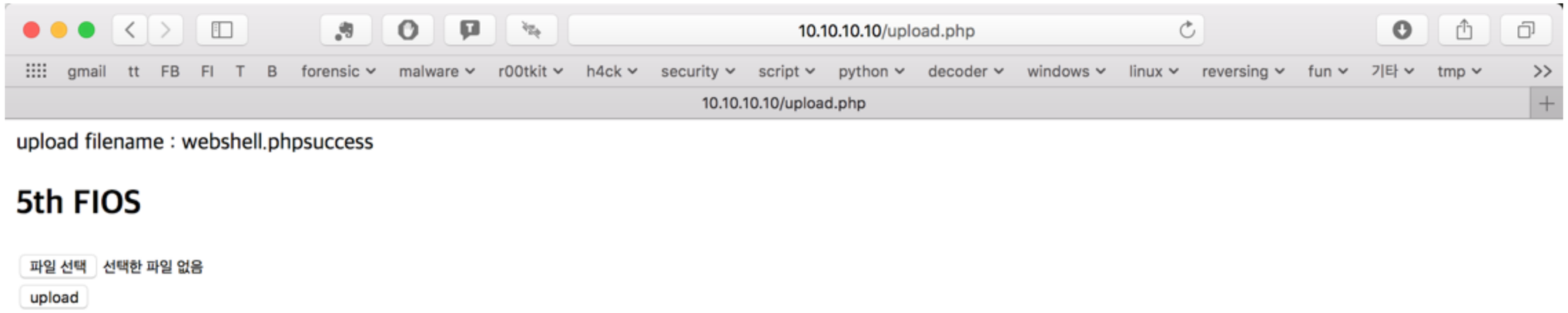
- **Recovery of Files of the R/W Layer**
  - AUFS
    - ✓ 여러 개의 디렉토리를 일련의 스택으로 묶어서 하나의 단일 뷰로 보여 줌



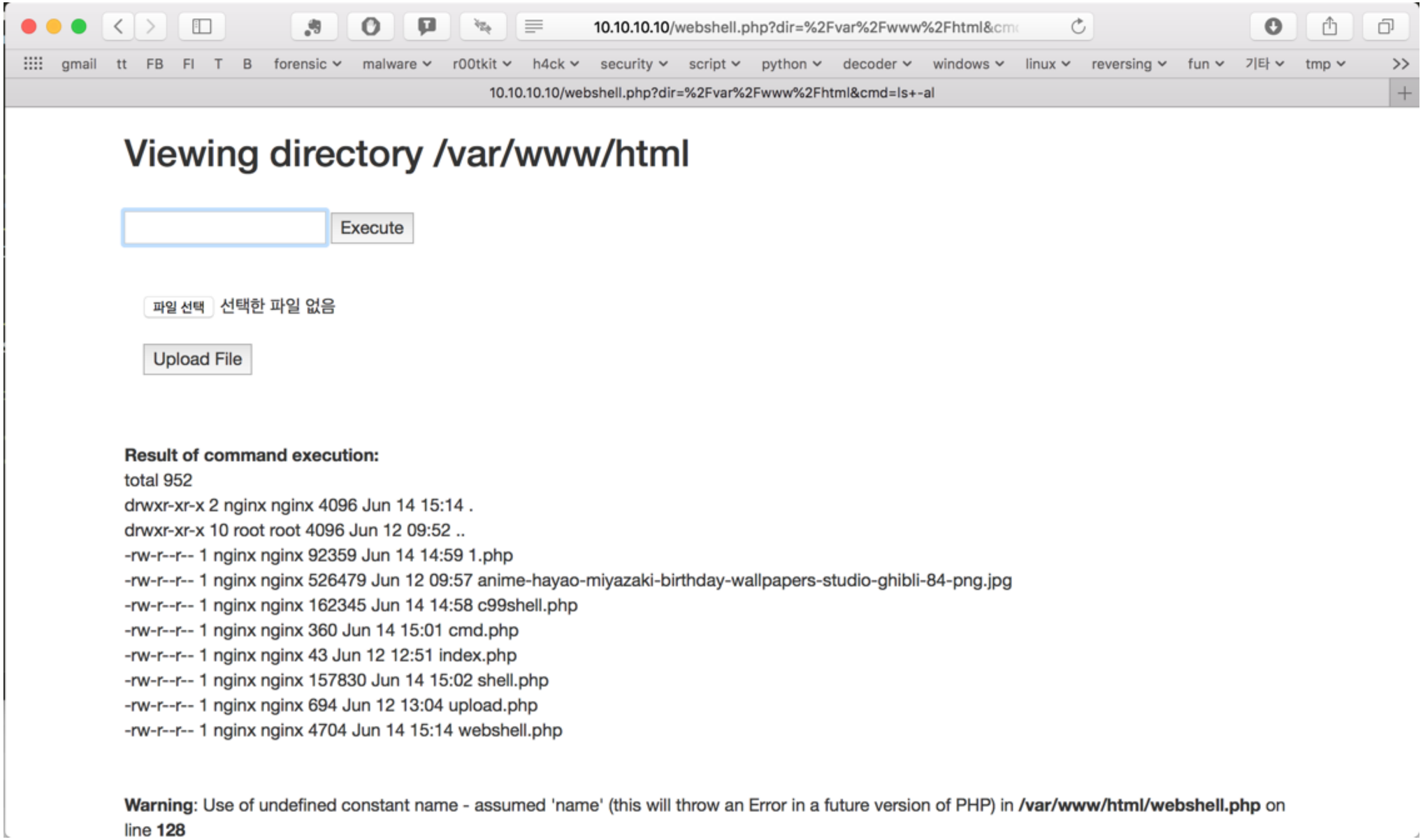
delete R/W layer file and try to recover test…

```
root@gotham:/var/lib/docker# docker ps
CONTAINER ID        IMAGE                    COMMAND                  CREATED             STATUS              PORTS                                      NAMES
01e4296ad690        fios-nginx-php-fpm:0.1   "docker-php-entryp..."   2 days ago          Up 14 seconds       0.0.0.0:80->80/tcp, 0.0.0.0:443->443/tcp   5th_fios

root@gotham:/var/lib/docker# cat /var/lib/docker/image/aufs/layerdb/mounts/01e4296ad690aa3d88b84bf35f80731739c1825d9033e3055262ec4df602263a/mount-id
7b88a94070766f37fdab3a39592579f4898245fe4bf445c4443714f6c9b1d7bf

root@gotham:/var/lib/docker# ls -al /var/lib/docker/aufs/mnt/7b88a94070766f37fdab3a39592579f4898245fe4bf445c4443714f6c9b1d7bf
total 88
drwxr-xr-x 90 root root  4096 Jun 12 18:52 .
drwx------ 93 root root 12288 Jun 14 22:57 ..
drwxr-xr-x  2 root root  4096 Jun 12 17:20 bin
drwxr-xr-x  4 root root  4096 Jun 12 18:52 dev
-rwxr-xr-x  1 root root     0 Jun 12 18:52 .dockerenv
drwxr-xr-x 38 root root  4096 Jun 13 17:08 etc
drwxr-xr-x  3 root root  4096 Jan 10 11:16 home
drwxr-xr-x 12 root root  4096 Jun 12 18:13 lib
drwxr-xr-x  5 root root  4096 Jan 10 04:36 media
drwxr-xr-x  2 root root  4096 Jan 10 04:36 mnt
dr-xr-xr-x  2 root root  4096 Jan 10 04:36 proc
drwx------  5 root root  4096 Jun 12 18:57 root
drwxr-xr-x  3 root root  4096 Jun 12 18:29 run
drwxr-xr-x  2 root root  4096 Jan 10 04:36 sbin
drwxr-xr-x  2 root root  4096 Jan 10 04:36 srv
-rwxr-xr-x  1 root root  7394 Jun 12 16:37 start.sh
drwxr-xr-x  2 root root  4096 Jan 10 04:36 sys
drwxrwxrwt  3 root root  4096 Jun 14 23:19 tmp
drwxr-xr-x 56 root root  4096 Jun 12 18:52 usr
drwxr-xr-x 33 root root  4096 Jun 12 18:52 var

root@gotham:/var/lib/docker# ls -al /var/lib/docker/aufs/diff/7b88a94070766f37fdab3a39592579f4898245fe4bf445c4443714f6c9b1d7bf/var/www/html/
total 532
drwxr-xr-x 2 systemd-timesync systemd-journal   4096 Jun 12 21:54 .
drwxr-xr-x 3 root             root              4096 Jun 12 18:52 ..
-rw-r--r-- 1 systemd-timesync systemd-journal 526479 Jun 12 18:57 anime-hayao-miyazaki-birthday-wallpapers-studio-ghibli-84-png.jpg
-rw-r--r-- 1 systemd-timesync systemd-journal     43 Jun 12 21:51 index.php
-rw-r--r-- 1 systemd-timesync systemd-journal    694 Jun 12 22:04 upload.php
```

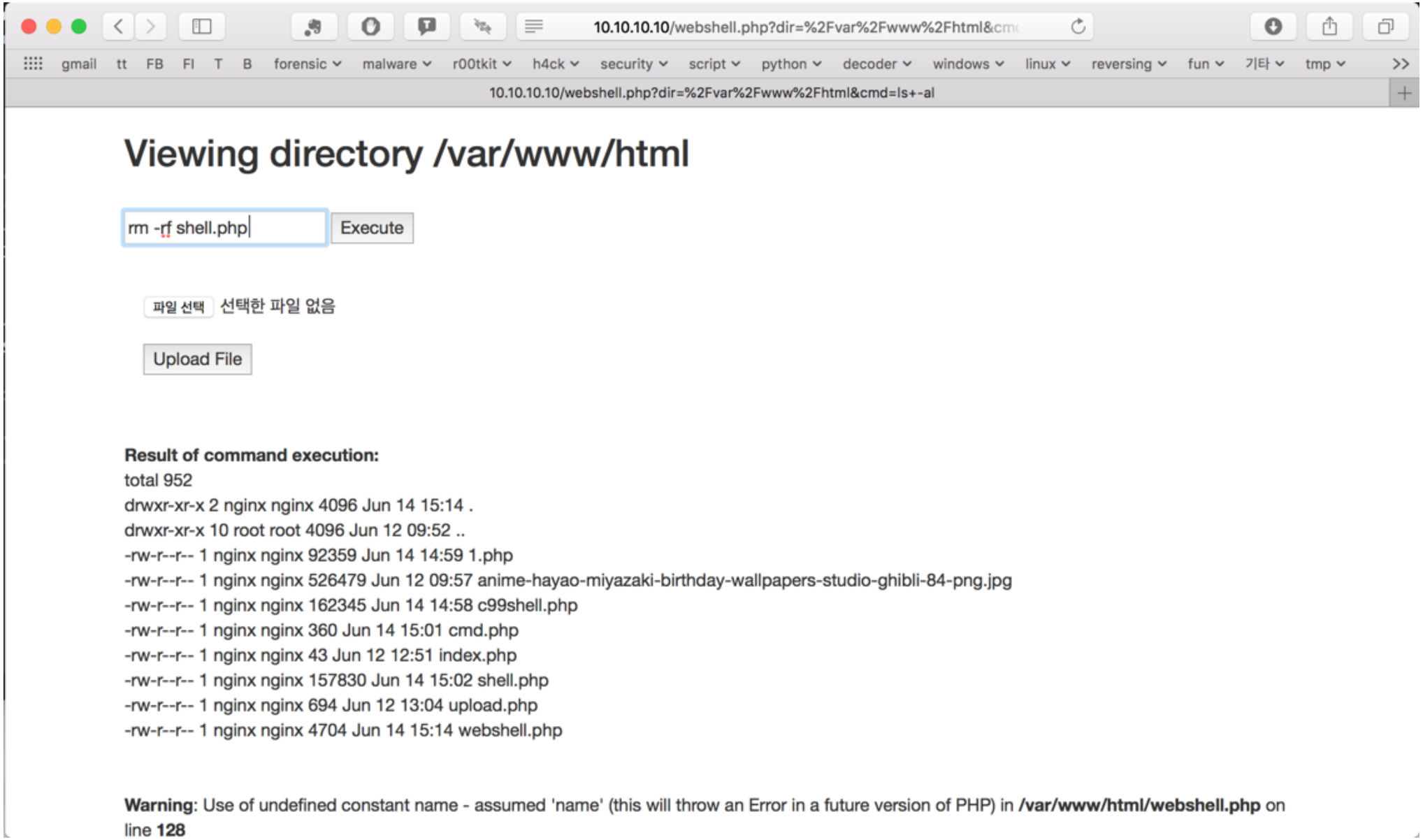
delete R/W layer file and try to recover test…



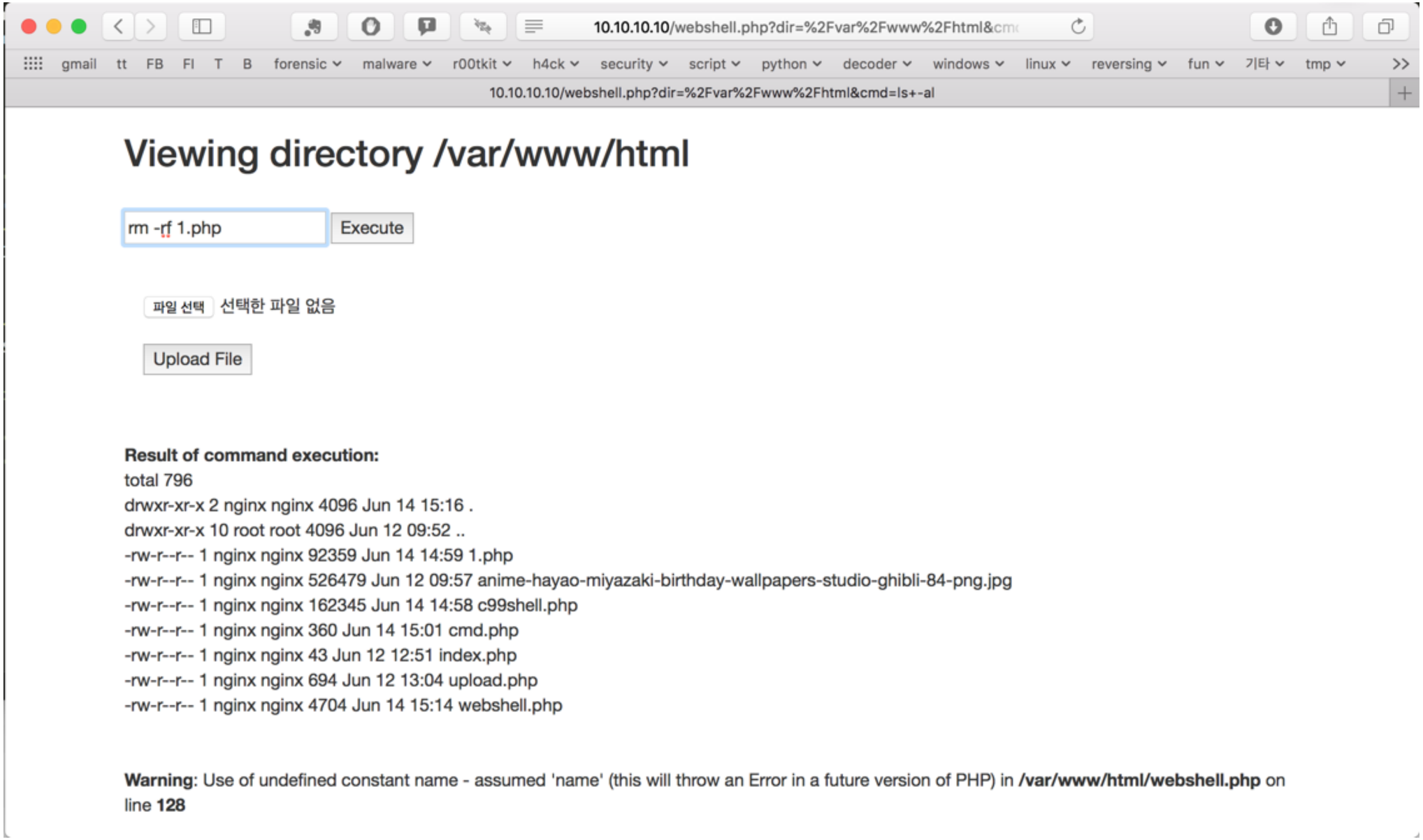
delete R/W layer file and try to recover test…



delete R/W layer file and try to recover test…



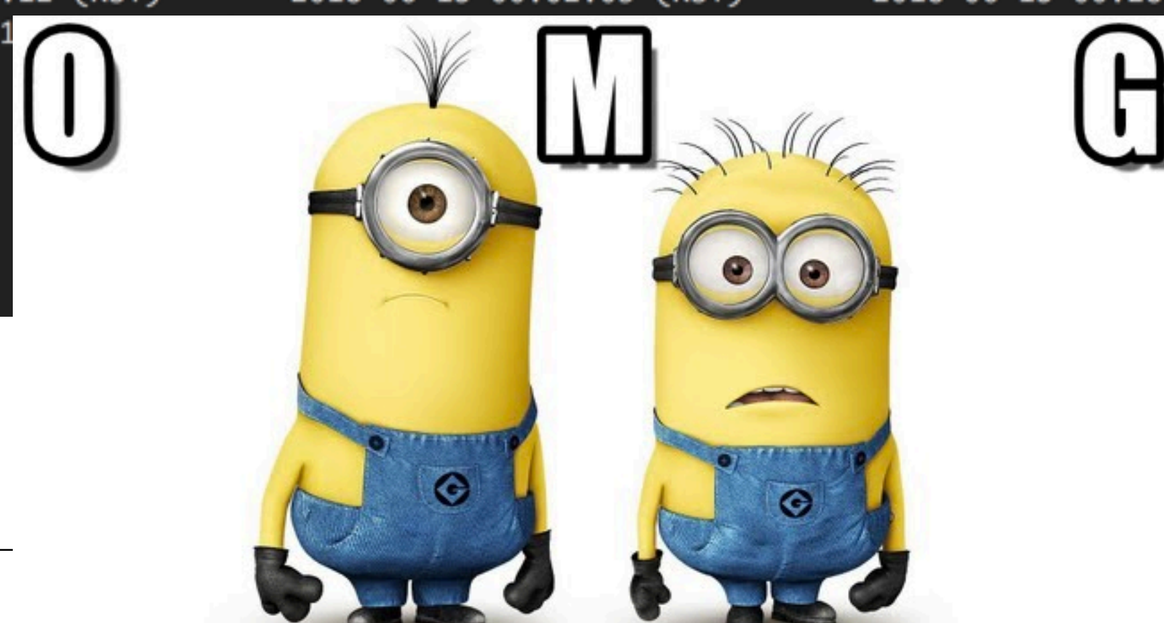
delete R/W layer file and try to recover test…



delete R/W layer file and try to recover test… fail

```
root@gotham:~# fls -lp /dev/sda1 267643
r/r 267662:      upload.php       2018-06-12 22:04:46 (KST) 2018-06-14 23:58:37 (KST) 2018-06-14 23:19:33 (KST) 2018-06-12 18:57:19 (KST) 694      101      100
r/r 267644:      index.php        2018-06-12 21:51:17 (KST) 2018-06-14 22:11:57 (KST) 2018-06-14 23:19:33 (KST) 2018-06-12 18:52:29 (KST) 43       101      100
r/r 267665:      anime-hayao-miyazaki-birthday-wallpapers-studio-ghibli-84-png.jpg     2018-06-12 18:57:50 (KST) 2018-06-14 22:11:57 (KST) 2018-06-14 23:19:33 (KST)
2018-06-12 18:57:50 (KST) 526479   101      100
r/r 271366:      c99shell.php     2018-06-14 23:58:37 (KST) 2018-06-14 23:59:15 (KST) 2018-06-14 23:58:37 (KST) 2018-06-14 23:58:37 (KST) 162345   101      100
r/r * 271367:    1.php    2018-06-15 00:23:58 (KST) 2018-06-14 23:59:49 (KST) 2018-06-15 00:23:58 (KST) 2018-06-14 23:59:43 (KST) 0        101      100
r/r * 271370:    shell.php        2018-06-15 00:16:00 (KST) 2018-06-15 00:02:53 (KST) 2018-06-15 00:16:00 (KST) 2018-06-15 00:02:46 (KST) 0        101      100
r/r * 271365:    cmd.php  2018-06-15 00:28:12 (KST) 2018-06-15 00:02:03 (KST) 2018-06-15 00:28:12 (KST) 2018-06-15 00:01:44 (KST) 0        101      100
r/r 271368:      webshell.php     2018-06-15 00:14:12 (KST) 2018-06-15 00:14:33 (KST) 2018-06-15 00:14:12 (KST) 2018-06-15 00:14:12 (KST) 4704     101      100
```

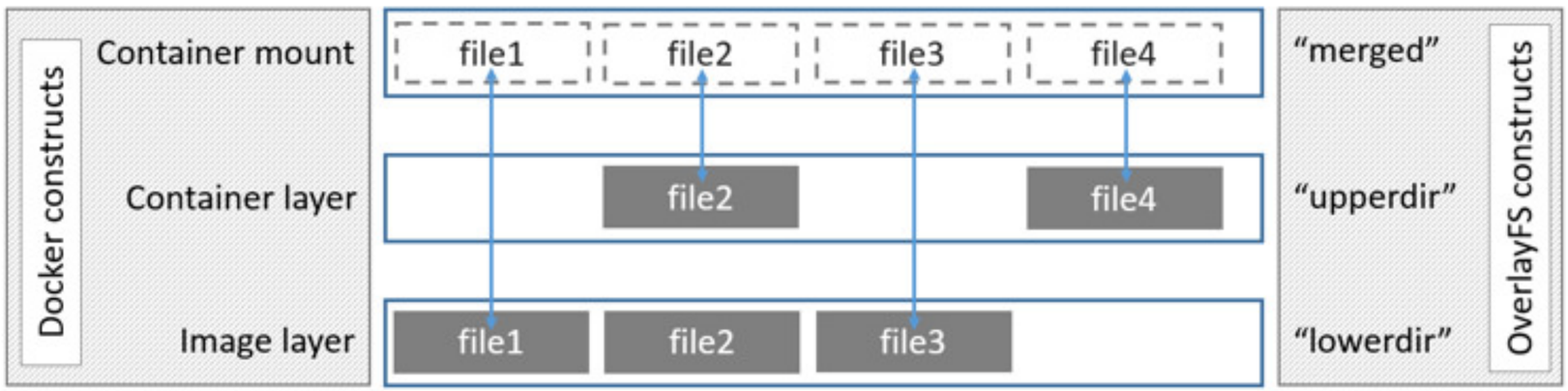
```
root@gotham:~# fls -lp /dev/sda1 267643
r/r 267662:      upload.php       2018-06-12 22:04:46 (KST)       2018-06-14 23:58:37 (KST)       2018-06-14 23:19:33 (KST)
r/r 267644:      index.php        2018-06-12 21:51:17 (KST)       2018-06-14 22:11:57 (KST)       2018-06-14 23:19:33 (KST)
r/r 267665:      anime-hayao-miyazaki-birthday-wallpapers-studio-ghibli-84-png.jpg       2018-06-12 18:57:50 (KST)       201
r/r 271366:      c99shell.php     2018-06-14 23:58:37 (KST)       2018-06-14 23:59:15 (KST)       2018-06-14 23:58:37 (KST)
r/r * 271367:    1.php    2018-06-15 00:23:58 (KST)       2018-06-14 23:59:49 (KST)       2018-06-15 00:23:58 (KST)       201
r/r * 271370:    shell.php        2018-06-15 00:16:00 (KST)       2018-06-15 00:02:53 (KST)       2018-06-15 00:16:00 (KST)
r/r * 271365:    cmd.php 2018-06-15 00:28:12 (KST)       2018-06-15 00:02:03 (KST)       2018-06-15 00:28:12 (KST)       201
r/r 271368:      webshell.php     2018-06-1                                                       5 00:14:12 (KST)
root@gotham:~#
root@gotham:~#
root@gotham:~# icat /dev/sda1 271367
root@gotham:~# icat /dev/sda1 271370
root@gotham:~# icat /dev/sda1 271365
root@gotham:~#
```



delete R/W layer file and try to recover test… why?

```
root@gotham:~# mount
sysfs on /sys type sysfs (rw,nosuid,nodev,noexec,relatime)
proc on /proc type proc (rw,nosuid,nodev,noexec,relatime)
udev on /dev type devtmpfs (rw,nosuid,relatime,size=994644k,nr_inodes=248661,mode=755)
devpts on /dev/pts type devpts (rw,nosuid,noexec,relatime,gid=5,mode=620,ptmxmode=000)
tmpfs on /run type tmpfs (rw,nosuid,noexec,relatime,size=202988k,mode=755)
/dev/sda1 on / type ext4 (rw,relatime,errors=remount-ro,data=ordered)
securityfs on /sys/kernel/security type securityfs (rw,nosuid,nodev,noexec,relatime)
tmpfs on /dev/shm type tmpfs (rw,nosuid,nodev)
tmpfs on /run/lock type tmpfs (rw,nosuid,nodev,noexec,relatime,size=5120k)
tmpfs on /sys/fs/cgroup type tmpfs (ro,nosuid,nodev,noexec,mode=755)
cgroup on /sys/fs/cgroup/systemd type cgroup (rw,nosuid,nodev,noexec,relatime,xattr,release_agent=/lib/systemd/systemd-cgroups-agent,name=systemd)
pstore on /sys/fs/pstore type pstore (rw,nosuid,nodev,noexec,relatime)
cgroup on /sys/fs/cgroup/memory type cgroup (rw,nosuid,nodev,noexec,relatime,memory)
cgroup on /sys/fs/cgroup/net_cls,net_prio type cgroup (rw,nosuid,nodev,noexec,relatime,net_cls,net_prio)
cgroup on /sys/fs/cgroup/cpu,cpuacct type cgroup (rw,nosuid,nodev,noexec,relatime,cpu,cpuacct)
cgroup on /sys/fs/cgroup/cpuset type cgroup (rw,nosuid,nodev,noexec,relatime,cpuset)
cgroup on /sys/fs/cgroup/devices type cgroup (rw,nosuid,nodev,noexec,relatime,devices)
cgroup on /sys/fs/cgroup/blkio type cgroup (rw,nosuid,nodev,noexec,relatime,blkio)
cgroup on /sys/fs/cgroup/pids type cgroup (rw,nosuid,nodev,noexec,relatime,pids)
cgroup on /sys/fs/cgroup/perf_event type cgroup (rw,nosuid,nodev,noexec,relatime,perf_event)
cgroup on /sys/fs/cgroup/hugetlb type cgroup (rw,nosuid,nodev,noexec,relatime,hugetlb)
cgroup on /sys/fs/cgroup/freezer type cgroup (rw,nosuid,nodev,noexec,relatime,freezer)
systemd-1 on /proc/sys/fs/binfmt_misc type autofs (rw,relatime,fd=35,pgrp=1,timeout=0,minproto=5,maxproto=5,direct)
hugetlbfs on /dev/hugepages type hugetlbfs (rw,relatime)
mqueue on /dev/mqueue type mqueue (rw,relatime)
debugfs on /sys/kernel/debug type debugfs (rw,relatime)
fusectl on /sys/fs/fuse/connections type fusectl (rw,relatime)
lxcfs on /var/lib/lxcfs type fuse.lxcfs (rw,nosuid,nodev,relatime,user_id=0,group_id=0,allow_other)
tmpfs on /run/user/1000 type tmpfs (rw,nosuid,nodev,relatime,size=202988k,mode=700,uid=1000,gid=1000)
/dev/sda1 on /var/lib/docker/aufs type ext4 (rw,relatime,errors=remount-ro,data=ordered)
none on /var/lib/docker/aufs/mnt/7b88a94070766f37fdab3a39592579f4898245fe4bf445c4443714f6c9b1d7bf type aufs (rw,relatime,si=810de8de00cc7c4b,dio,dirperm1)
shm on /var/lib/docker/containers/01e4296ad690aa3d88b84bf35f80731739c1825d9033e3055262ec4df602263a/shm type tmpfs (rw,nosuid,nodev,noexec,relatime,size=65536k)
nsfs on /run/docker/netns/d7f81bee3cda type nsfs (rw)
```

- **Recovery of Files of the R/W Layer**
  - Overlay2
    - ✓ lowerdir은 읽기 전용의 이미지 레이어
    - ✓ 컨테이너 레이어는 이미지 레이어로 부터 만들어지며, 변경되는 정보들이 여기에 저장
    - ✓ 최종적으로 컨테이너 마운트를 통해서 이들 정보가 merged
    - ✓ AUFS와는 달리 단지 두 개의 레이어만 가지고 있기 때문에 AUFS와 같은 멀티레이어 이미지를 구성할 수 없음
    - ✓ 대신 각 이미지 레이어마다 /var/lib/docker/overlay2 디렉토리 밑에 자신만의 디렉토리를 가지는 것으로 이 문제를 해결



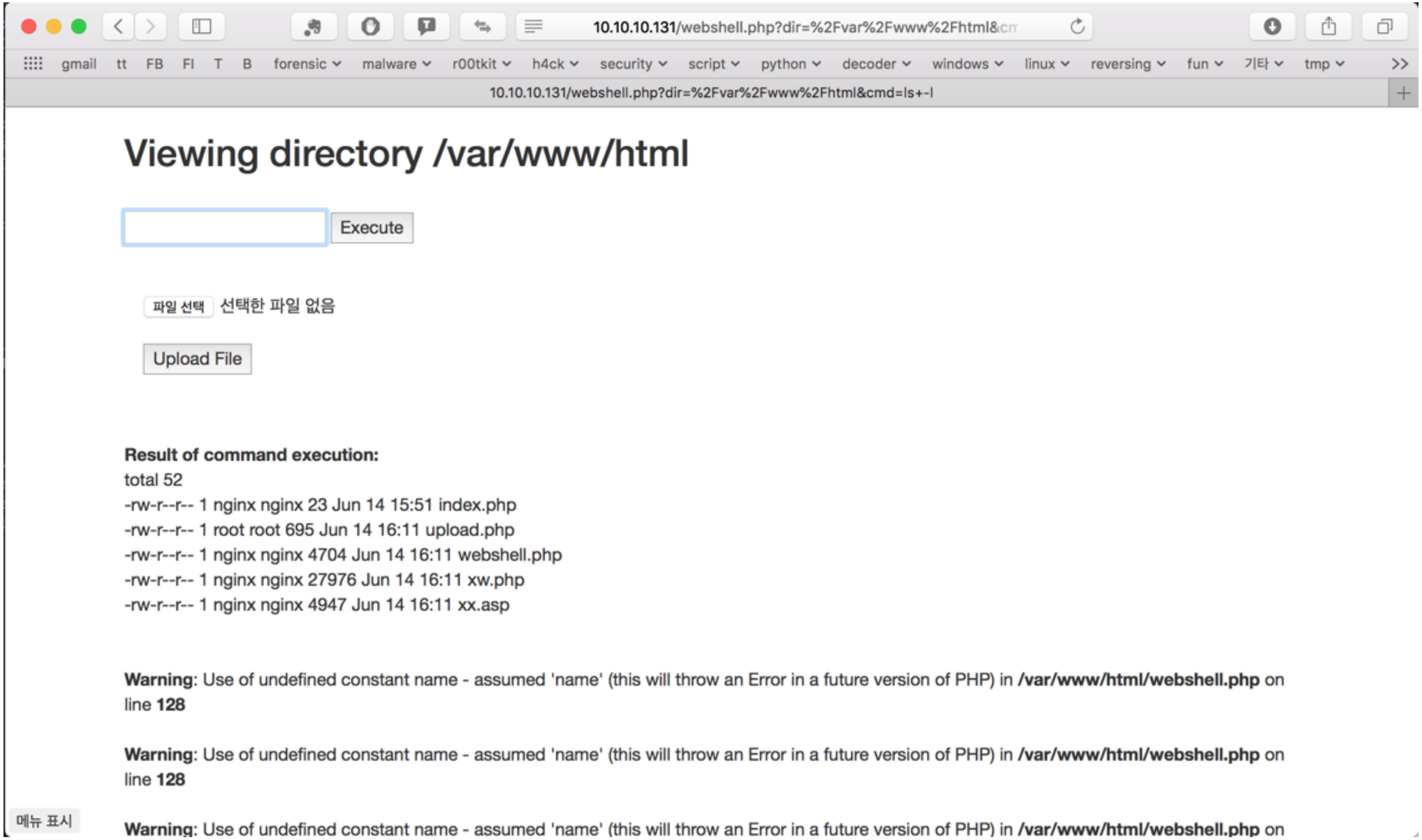
delete R/W layer file and try to recover test in overlay2 storage driver environment…

```
[root@arkham ~]# docker ps
CONTAINER ID        IMAGE                    COMMAND                  CREATED             STATUS              PORTS                                      NAMES
6a4a91c3f5c2        fios-nginx-php-fpm:0.1   "docker-php-entryp..."   6 minutes ago       Up 6 minutes        0.0.0.0:80->80/tcp, 0.0.0.0:443->443/tcp   5th_fios
f3e2674e7007        redis                    "docker-entrypoint..."   30 hours ago        Up 30 hours         6379/tcp                                   youthful

[root@arkham ~]# cat /var/lib/docker/image/overlay2/layerdb/mounts/6a4a91c3f5c2e3648ff0d0e35505da44865008449853974be54fda17a5d90d7f/mount-id
86e5e059b940727d7a0f88bdbdd281887bb46e0780d21be12e04a8d3981ceae1

[root@arkham ~]# ls -al /var/lib/docker/overlay2/86e5e059b940727d7a0f88bdbdd281887bb46e0780d21be12e04a8d3981ceae1
합계 20
drwx------.  5 root root   69  6월 15 01:09 .
drwx------. 61 root root 8192  6월 15 01:09 ..
drwxr-xr-x.  7 root root   62  6월 15 01:09 diff
-rw-r--r--.  1 root root   26  6월 15 01:09 link
-rw-r--r--.  1 root root  898  6월 15 01:09 lower
drwxr-xr-x.  1 root root   62  6월 15 01:09 merged
drwx------.  3 root root   18  6월 15 01:09 work
```

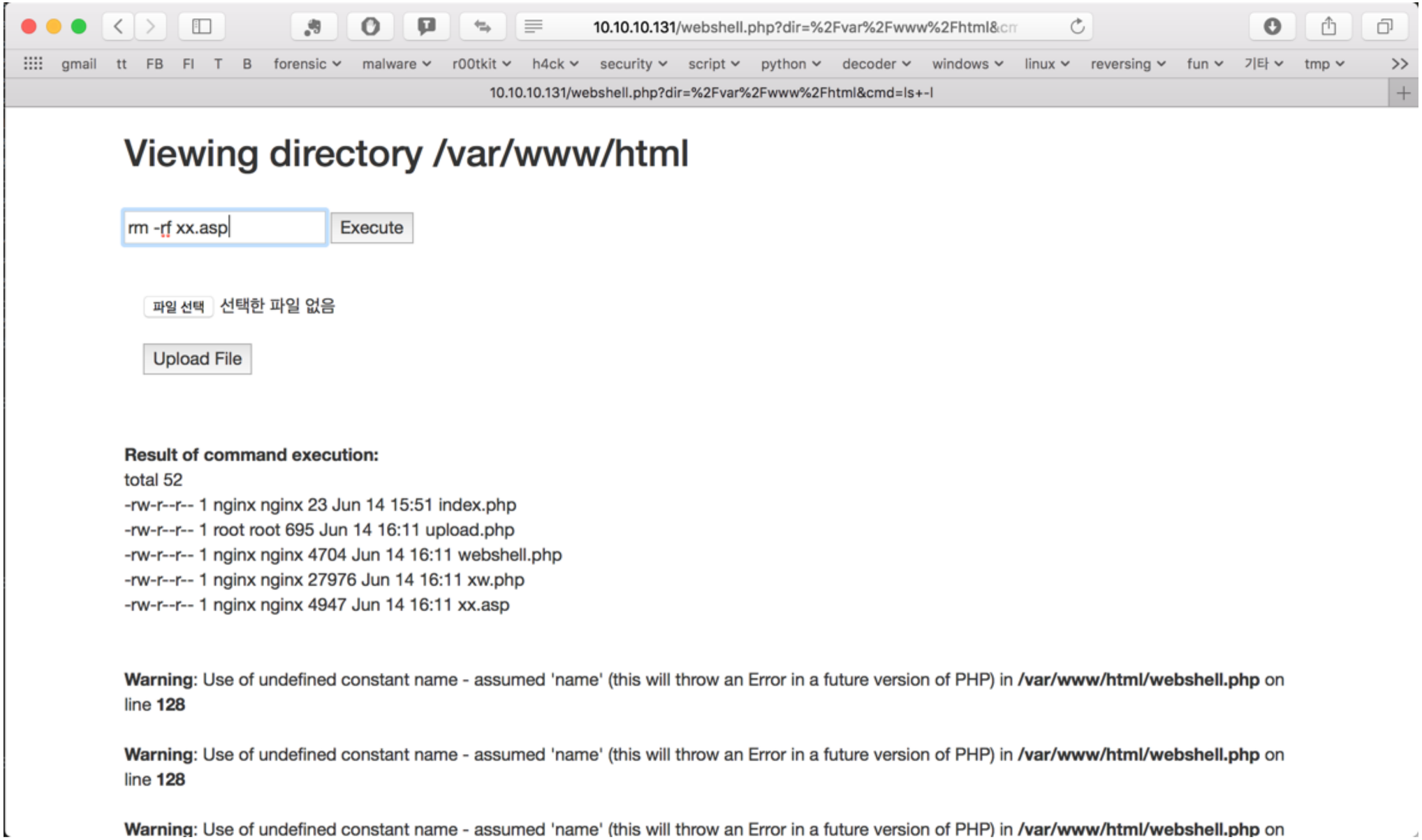
created and not yet deleted in the R/W layer

view of the entire file system across all layers

delete R/W layer file and try to recover test in overlay2 storage driver environment…



delete R/W layer file and try to recover test in overlay2 storage driver environment…

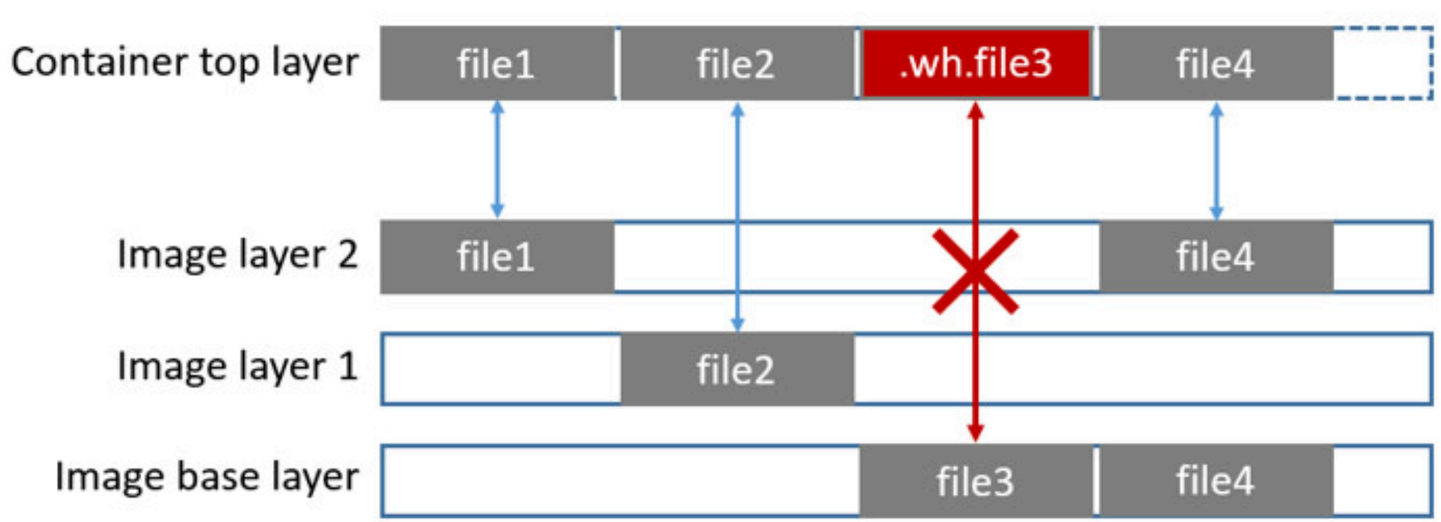


delete R/W layer file and try to recover test in overlay2 storage driver environment…

```
[root@arkham ~]# ls -al /var/lib/docker/overlay2/86e5e059b940727d7a0f88bdbdd281887bb46e0780d21be12e04a8d3981ceae1/diff/var/www/html
합계 44
drwxr-xr-x. 2  100  101    75  6월 15 01:17 .
drwxr-xr-x. 3 root root    18  6월 15 01:06 ..
-rw-r--r--. 1  100  101    23  6월 15 00:51 index.php
-rw-r--r--. 1 root root   695  6월 15 01:11 upload.php
-rw-r--r--. 1  100  101  4704  6월 15 01:11 webshell.php
-rw-r--r--. 1  100  101 27976  6월 15 01:11 xw.php

[root@arkham ~]# ls -al /var/lib/docker/overlay2/86e5e059b940727d7a0f88bdbdd281887bb46e0780d21be12e04a8d3981ceae1/merged/var/www/html
합계 44
drwxr-xr-x. 1  100  101    75  6월 15 01:17 .
drwxr-xr-x. 1 root root    18  6월 15 01:06 ..
-rw-r--r--. 1  100  101    23  6월 15 00:51 index.php
-rw-r--r--. 1 root root   695  6월 15 01:11 upload.php
-rw-r--r--. 1  100  101  4704  6월 15 01:11 webshell.php
-rw-r--r--. 1  100  101 27976  6월 15 01:11 xw.php
```

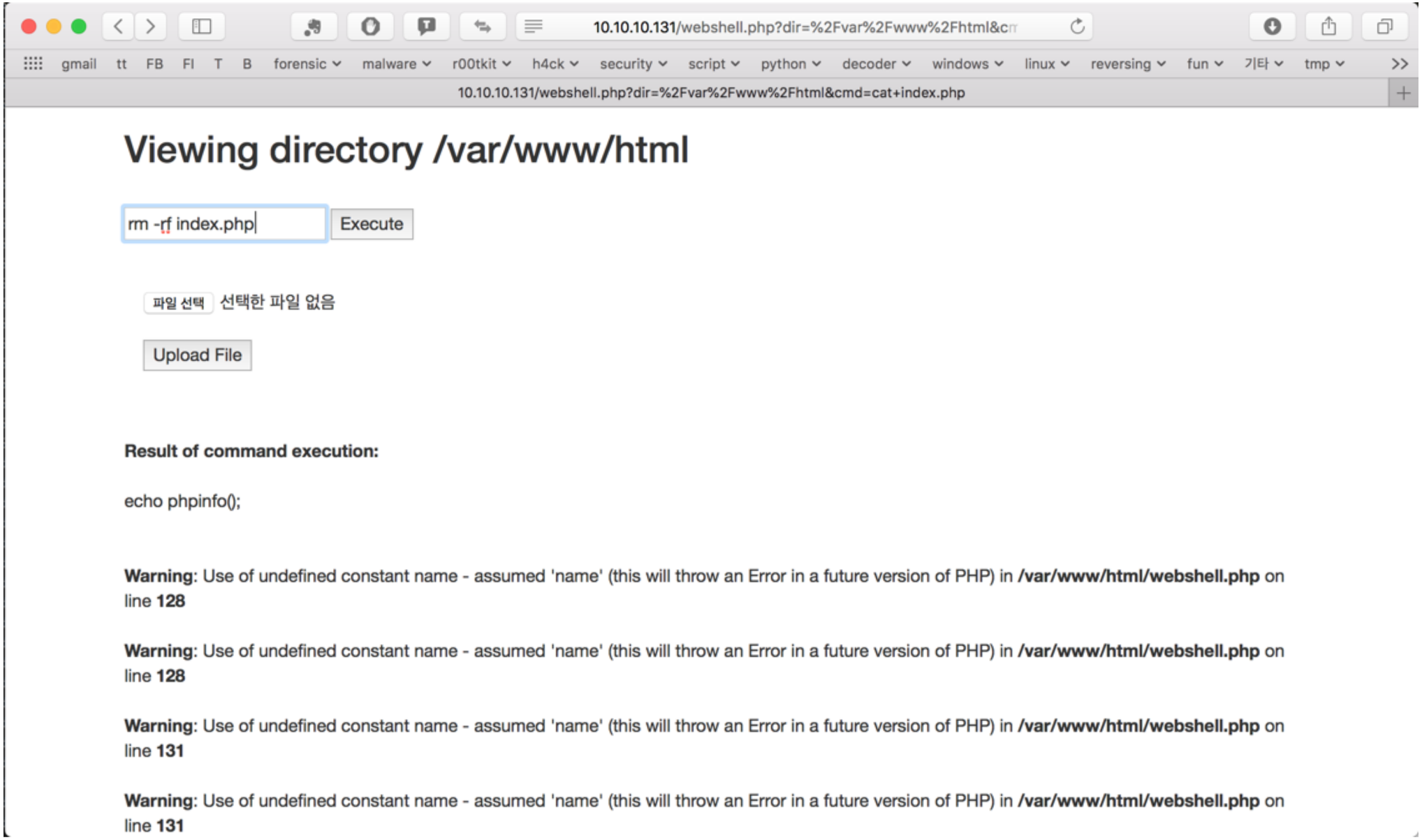
```
[root@arkham ~]# fls /dev/sda1
Cannot determine file system type
[root@arkham ~]# fls -f list
Supported file system types:
        ntfs (NTFS)
        fat (FAT (Auto Detection))
        ext (ExtX (Auto Detection))
        iso9660 (ISO9660 CD)
        hfs (HFS+)
        ufs (UFS (Auto Detection))
        raw (Raw Data)
        swap (Swap Space)
        fat12 (FAT12)
        fat16 (FAT16)
        fat32 (FAT32)
        exfat (exFAT)
        ext2 (Ext2)
        ext3 (Ext3)
        ext4 (Ext4)
        ufs1 (UFS1)
        ufs2 (UFS2)
        yaffs2 (YAFFS2)
```

- **Recovery of Files of the Image Layer**
  - 삭제된 파일에 대한 레퍼런스는 R/W 레이어에 있으나, 파일은 이미지 레이어에 남아 있음
  - overlay2 스토리지 드라이버를 사용하는 도커라면 삭제된 파일 이름을 가진 inode를 R/W 레이어에 할당
  - 삭제된 파일은 캐릭터 디바이스(character device)로 파일 시스템 플래그를 변경
    - ✓ 즉, overlay2 환경에서 삭제된 파일을 찾으려면 캐릭터 디바이스 파일을 찾아라!!

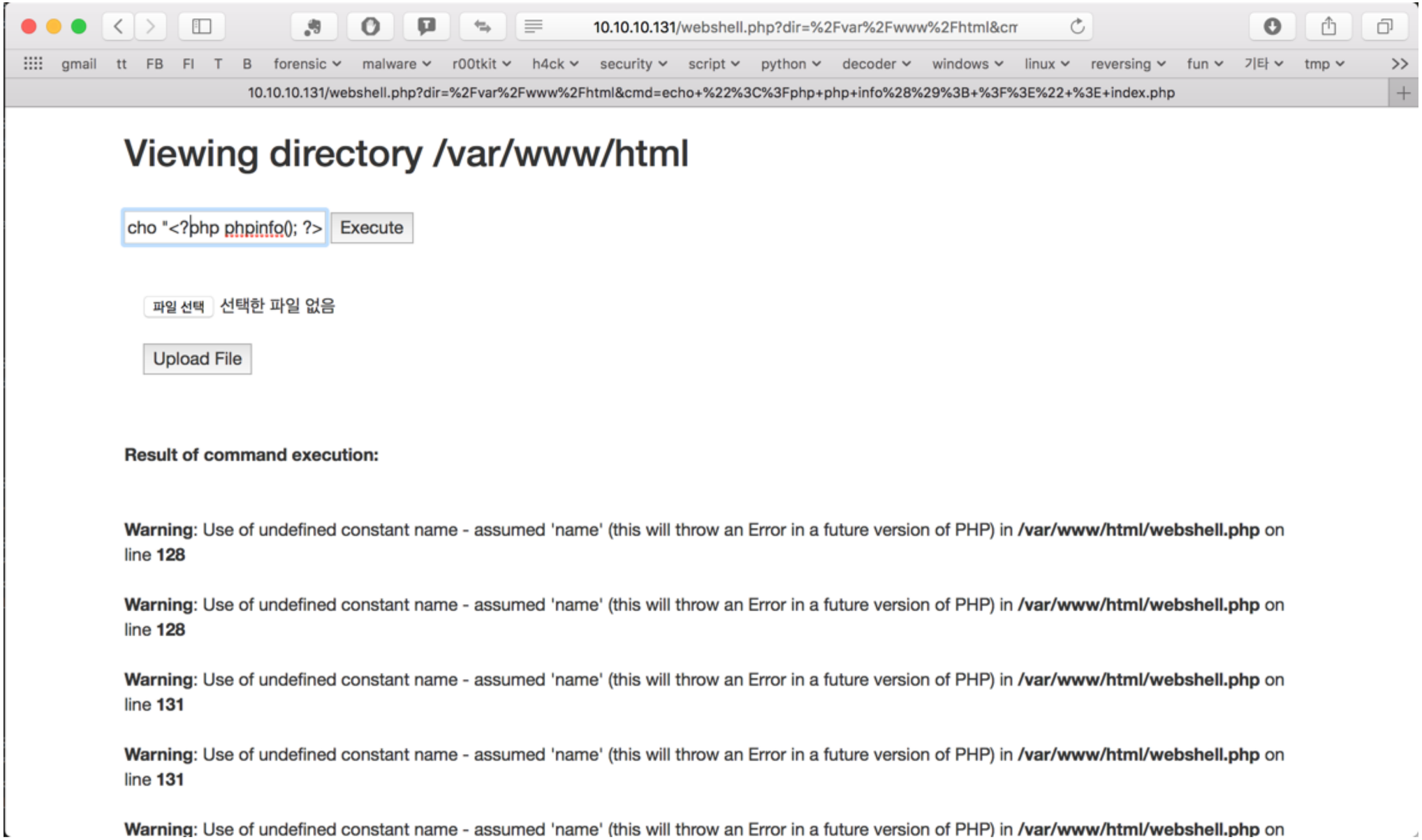


Docker container
(AUFS storage-driver demonstrating whiteout file)

delete Image layer file and try to recover test…



delete Image layer file and try to recover test…



delete Image layer file and try to recover test…

```
[root@arkham ~]# find /var/lib/docker/overlay2/86e5e059b940727d7a0f88bdbdd281887bb46e0780d21be12e04a8d3981ceae1/ -type c -ls
51606633    0 c---------   1 root     root       0,   0  6월  15 01:50 /var/lib/docker/overlay2/86e5e059b940727d7a0f88bdbdd281887bb46e0780d21be12e04a8d3981ceae1/diff/var/www/html/index.php
[root@arkham ~]#
[root@arkham ~]# stat /var/lib/docker/overlay2/86e5e059b940727d7a0f88bdbdd281887bb46e0780d21be12e04a8d3981ceae1/diff/var/www/html/index.php
  File: `/var/lib/docker/overlay2/86e5e059b940727d7a0f88bdbdd281887bb46e0780d21be12e04a8d3981ceae1/diff/var/www/html/index.php'
  Size: 0               Blocks: 0          IO Block: 4096   문자 특수 파일
Device: fd00h/64768d    Inode: 51606633    Links: 1     Device type: 0,0
Access: (0000/c---------)  Uid: (    0/    root)   Gid: (    0/    root)
Context: system_u:object_r:container_share_t:s0
Access: 2018-06-15 01:50:00.001433717 +0900
Modify: 2018-06-15 01:50:00.001433717 +0900
Change: 2018-06-15 01:50:00.003433768 +0900
 Birth: -
[root@arkham ~]# stat /var/lib/docker/overlay2/86e5e059b940727d7a0f88bdbdd281887bb46e0780d21be12e04a8d3981ceae1/diff/var/www/html/index.php
  File: `/var/lib/docker/overlay2/86e5e059b940727d7a0f88bdbdd281887bb46e0780d21be12e04a8d3981ceae1/diff/var/www/html/index.php'
  Size: 21              Blocks: 8          IO Block: 4096   일반 파일
Device: fd00h/64768d    Inode: 51606628    Links: 1
Access: (0644/-rw-r--r--)  Uid: (  100/ UNKNOWN)   Gid: (  101/ UNKNOWN)
Context: system_u:object_r:container_file_t:s0:c71,c122
Access: 2018-06-15 01:51:19.138493870 +0900
Modify: 2018-06-15 01:51:19.138493870 +0900
Change: 2018-06-15 01:51:19.138493870 +0900
 Birth: -
[root@arkham ~]#
```

root@arkham:~ (ssh) ⌘1

```
[root@arkham ~]#
[root@arkham ~]# ls -al /var/lib/docker/overlay2/86e5e059b940727d7a0f88bdbdd281887bb46e0780d21be12e04a8d3981ceae1/diff/var/www/html
합 계  40
drwxr-xr-x. 2  100  101    75  6월  15 02:00 .
drwxr-xr-x. 3 root root    18  6월  15 01:06 ..
c---------. 1 root root  0, 0  6월  15 02:00 index.php
-rw-r--r--. 1 root root   695  6월  15 01:11 upload.php
-rw-r--r--. 1  100  101  4704  6월  15 01:11 webshell.php
-rw-r--r--. 1  100  101 27976  6월  15 01:11 xw.php
[root@arkham ~]# ls -al /var/lib/docker/overlay2/86e5e059b940727d7a0f88bdbdd281887bb46e0780d21be12e04a8d3981ceae1/merged/var/www/html
합 계  40
drwxr-xr-x. 1  100  101    75  6월  15 02:00 .
drwxr-xr-x. 1 root root    18  6월  15 01:06 ..
-rw-r--r--. 1 root root   695  6월  15 01:11 upload.php
-rw-r--r--. 1  100  101  4704  6월  15 01:11 webshell.php
-rw-r--r--. 1  100  101 27976  6월  15 01:11 xw.php
[root@arkham ~]#
```

root@arkham:~ (ssh) ⌘1

R/W layer

Recovery of Files of the Image Layer

- 삭제된 파일에 대한 레퍼런스는 R/W 레이어에

do u remember??

Forensic analysis on Docker hosts

- **Recovery of Files of the Image Layer**
  - 삭제된 파일에 대한 레퍼런스는 R/W 레이어에 있으나, 파일은 이미지 레이어에 남아 있음
  - [redacted] overlay2 스토리지 드라이버를 사용하는 도커라면 삭제된 파일 이름을 가진 inode를 R/W 레이어에 할당
  - 삭제된 파일은 캐릭터 디바이스(character device)로 파일 시스템 플래그를 변경
    - ✓ 즉, overlay2 환경에서 삭제된 파일을 찾으려면 캐릭터 디바이스 파일을 찾아라!!

delete Image layer file and try to recover test…

```
[root@arkham ~]# docker images
REPOSITORY          TAG                 IMAGE ID            CREATED             SIZE
fios-nginx-php-fpm  0.1                 5d419ca9cbba        About an hour ago   303 MB
docker.io/nginx     latest              cd5739a0906a        8 days ago          109 MB
docker.io/redis     latest              bfcb1f6df2db        6 weeks ago         107 MB
docker.io/ubuntu    14.04               8cef1fa16c77        6 weeks ago         223 MB
docker.io/php       7.2.4-fpm-alpine    8bf48293eebf        7 weeks ago         77.8 MB
[root@arkham ~]#
root@arkham:~ (ssh) ⌘1
```

```
# docker inspect 5d419ca9cbba
```

```
        "GraphDriver": {
            "Name": "overlay2",
            "Data": {
                "LowerDir": "/var/lib/docker/overlay2/4f1aead3a3b3cc6dccaab895cf3b5442008eb2d532c113bc6dd4d75f83d4f72e/diff:/var/lib/docker/overlay2/54876d07c7975c96c834e
1acfc/diff:/var/lib/docker/overlay2/b12d352b2d953890dddfb9ce113b04a5f96e4dd56051a876c7d4a15992d7ce81/diff:/var/lib/docker/overlay2/329141488a5aa2ec80337a8ff1d8d3680d68bd9
b/docker/overlay2/016da25f51da5085497ae6cd6fc079b8b225c562b7821d288849b30e9a45308d/diff:/var/lib/docker/overlay2/a9ea974c3399a4f75d9b7cd975294ea930a73eb2138a5abd5464259698
2aa33ae1b248e889a5a255f69cf162a0505495e379f208cff37f9b2b46c66b35/diff:/var/lib/docker/overlay2/6213ab8b764cf92a74987382311d04bd431c717dd60f2edf06575b9c42fcfbbc/diff:/var/
2af209c9f15ab845ab004cd6113301417b0e5e7acbaecd/diff:/var/lib/docker/overlay2/48e36bef94e7f9d5e4651b8334aff0fca1d9168e1784d31e7826e68130b9c33d/diff:/var/lib/docker/overlay
dcc708be58b1c0a3870d31ee30ee/diff:/var/lib/docker/overlay2/25efba09bc57cbf33dfb02d6fba5232bf392116a29c67c9a466761f9a719ea59/diff:/var/lib/docker/overlay2/13652837bbe582a4
d07722a0c0/diff:/var/lib/docker/overlay2/ccc2b1fd50c2cdd054596931bb9c3dee108d54ecb7bff5246a0672df654a7c02/diff:/var/lib/docker/overlay2/f6896190f067f621fa155283691f0f7468
ar/lib/docker/overlay2/b6783d0b935ca98877d864eb8607a06334fb62ca98f24e8257f95d054a4253eb/diff:/var/lib/docker/overlay2/06df6dd586b10d97b852f03fdc9e13cce8f781d70eba55f5a607
lay2/afd47f040246a951ba2371611c6c3eb16ea1653e8b11a7cd48fb8a7c611e1413/diff:/var/lib/docker/overlay2/2d3d9aa38dbee120a14de67579c7b452fb99dadb490a6aa17df4df14bdc59483/diff:
6a23b7708a75e712a269779656293aa3351365e834b90069bfe/diff:/var/lib/docker/overlay2/f26d4b296cd2c1235e894ab3ca0db30dd56d120c4db8f58e04ec53f1baa8cf89/diff:/var/lib/docker/ov
f810256b09626bd2f5394f56066abab07/diff:/var/lib/docker/overlay2/b30ef82621dbad396b3135ca91acc9a66916f4dea3ea7e075665fa52ffec0eb5/diff:/var/lib/docker/overlay2/e187c38a53a
c6f14b67ef7e5e3/diff:/var/lib/docker/overlay2/855f16b5fe067912fa820e112d3aa706f69484a3fdeedf0f3cad540d9c697224/diff:/var/lib/docker/overlay2/8eccf04d03c79933d9286c0f89c82
ff:/var/lib/docker/overlay2/e7e1f7d9732db4347d0d424f9de5656eacc65e3e353f75854439088b0d8ed0da/diff:/var/lib/docker/overlay2/0119e037faa8a090845c6937efd299a24bf19e2ab264c5e
/overlay2/369a06704e1dfb132c48633a8c6e795d51b3b1ab482a36a8db5bfeec34dc21cf/diff",
                "MergedDir": "/var/lib/docker/overlay2/ccc1ba09ddcbe03df1e8e16b92819cbd822789e57aecdb406865fe5fc9092f3e/merged",
                "UpperDir": "/var/lib/docker/overlay2/ccc1ba09ddcbe03df1e8e16b92819cbd822789e57aecdb406865fe5fc9092f3e/diff",
                "WorkDir": "/var/lib/docker/overlay2/ccc1ba09ddcbe03df1e8e16b92819cbd822789e57aecdb406865fe5fc9092f3e/work"
            }
        },
```

delete Image layer file and try to recover test…

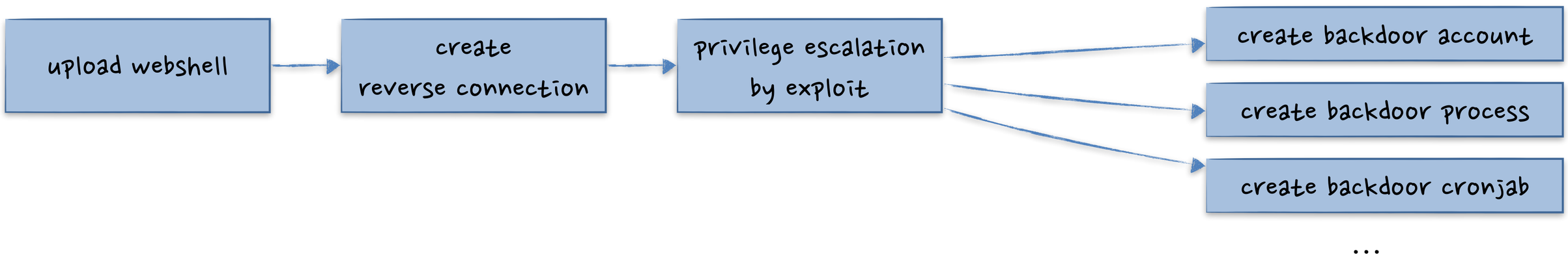
```
>>>
>>> lowerdir=['/var/lib/docker/overlay2/4f1aead3a3b3cc6dccaab895cf3b5442008eb2d532c113bc6dd4d75f83d4f72e/diff', '/var/lib/docker/overlay2/5
4876d07c7975c96c834ec9697e9f7bbe6fe5ecd6d8e8c5506b0767357e1acfc/diff', '/var/lib/docker/overlay2/b12d352b2d953890dddfb9ce113b04a5f96e4dd560
51a876c7d4a15992d7ce81/diff', '/var/lib/docker/overlay2/329141488a5aa2ec80337a8ff1d8d3680d68bd97487fac766b029d8e0168bfde/diff', '/var/lib/d
ocker/overlay2/016da25f51da5085497ae6cd6fc079b8b225c562b7821d288849b30e9a45308d/diff', '/var/lib/docker/overlay2/a9ea974c3399a4f75d9b7cd975
294ea930a73eb2138a5abd54642596980710bb/diff', '/var/lib/docker/overlay2/2aa33ae1b248e889a5a255f69cf162a0505495e379f208cff37f9b2b46c66b35/di
ff', '/var/lib/docker/overlay2/6213ab8b764cf92a74987382311d04bd431c717dd60f2edf06575b9c42fcfbbc/diff', '/var/lib/docker/overlay2/ac95180b50
14948d8b2af209c9f15ab845ab004cd6113301417b0e5e7acbaecd/diff', '/var/lib/docker/overlay2/48e36bef94e7f9d5e4651b8334aff0fca1d9168e1784d31e782
6e68130b9c33d/diff', '/var/lib/docker/overlay2/f2f1a2c4cf69fc738ac11d730f2c3cff30d9dcc708be58b1c0a3870d31ee30ee/diff', '/var/lib/docker/ove
rlay2/25efba09bc57cbf33dfb02d6fba5232bf392116a29c67c9a466761f9a719ea59/diff', '/var/lib/docker/overlay2/13652837bbe582a43274ba98b287bda15f9
9cc8743ff2a61c27b3ad07722a0c0/diff', '/var/lib/docker/overlay2/ccc2b1fd50c2cdd054596931bb9c3dee108d54ecb7bff5246a0672df654a7c02/diff', '/va
r/lib/docker/overlay2/f6896190f067f621fa155283691f0f7468461382963ef57092a63a707227d8db/diff', '/var/lib/docker/overlay2/b6783d0b935ca98877d
864eb8607a06334fb62ca98f24e8257f95d054a4253eb/diff', '/var/lib/docker/overlay2/06df6dd586b10d97b852f03fdc9e13cce8f781d70eba55f5a60707198852
6a70/diff', '/var/lib/docker/overlay2/afd47f040246a951ba2371611c6c3eb16ea1653e8b11a7cd48fb8a7c611e1413/diff', '/var/lib/docker/overlay2/2d3
d9aa38dbee120a14de67579c7b452fb99dadb490a6aa17df4df14bdc59483/diff', '/var/lib/docker/overlay2/568a944f252496a23b7708a75e712a269779656293aa
3351365e834b90069bfe/diff', '/var/lib/docker/overlay2/f26d4b296cd2c1235e894ab3ca0db30dd56d120c4db8f58e04ec53f1baa8cf89/diff', '/var/lib/doc
ker/overlay2/ee3cac9abef155ff10333847ead3a2df810256b09626bd2f5394f56066abab07/diff', '/var/lib/docker/overlay2/b30ef82621dbad396b3135ca91ac
c9a66916f4dea3ea7e075665fa52ffec0eb5/diff', '/var/lib/docker/overlay2/e187c38a53a58bafe4ef11bc18df5730ecba59058af3923b5c6f14b67ef7e5e3/diff
', '/var/lib/docker/overlay2/855f16b5fe067912fa820e112d3aa706f69484a3fdeedf0f3cad540d9c697224/diff', '/var/lib/docker/overlay2/8eccf04d03c7
9933d9286c0f89c828f9855abbd3af9bb9522523d10cd5128b22/diff', '/var/lib/docker/overlay2/e7e1f7d9732db4347d0d424f9de5656eacc65e3e353f758544390
88b0d8ed0da/diff', '/var/lib/docker/overlay2/0119e037faa8a090845c6937efd299a24bf19e2ab264c5ed55766cad9058a416/diff', '/var/lib/docker/overl
ay2/369a06704e1dfb132c48633a8c6e795d51b3b1ab482a36a8db5bfeec34dc21cf/diff']
>>>
>>> for x in lowerdir:
...     file = os.system('find {0} -name index.php'.format(x))
...
/var/lib/docker/overlay2/4f1aead3a3b3cc6dccaab895cf3b5442008eb2d532c113bc6dd4d75f83d4f72e/diff/var/www/html/index.php
>>>
>>> f = open('/var/lib/docker/overlay2/4f1aead3a3b3cc6dccaab895cf3b5442008eb2d532c113bc6dd4d75f83d4f72e/diff/var/www/html/index.php')
>>> f.read()
'<?php\n\necho phpinfo();\n'
>>>
>>>
```

- **Namespaces**
  - UTS namespace
    - ✓ 컨테이너별로 시간대(time zone)를 별도로 설정 가능
    - ✓ 라이브 분석 시에는 시간대를 고려해서 분석해야 함
    - ✓ Overlay2는 파일을 수정할 때 항상 호스트 시스템 시간을 사용하고 각 컨테이너의 타임 스탬프를 동적으로 조정
  - PID namespace
    - ✓ PID를 포함하는 로그 파일
  - user namespace
    - ✓ UID를 포함하는 로그 파일
    - ✓ container uid/gid to host uid/gid -> /etc/subuid, /etc/subgid

Considreations (a.k.a Painfull things)

- **나의 도커가 공격을 받는다면?**
  - what service?
  - what application?
- **외부에서 하는 공격이라면 뭔가 파일을 새로 만들꺼야..**
  - R/W layer (not Image layer)
- **새로운 파일을 만들고 원래 있던 파일을 수정한다면?**
- **새로 만든 파일을 이용해서 시스템 파일을 변경하거나 삭제하면?**
  - Image layer
  - 삭제하면 눈에 띄는데 변경만 하면 쉽게 눈에 안보임

Considreations (a.k.a Painfull things)



Conclusion

- **로카르트 법칙을 잘 되새겨보자.**
  - 공격자는 공격이 성공하기 전까지 다양한 행위들을 할 것이고, 이 과정에서 또 다른 흔적이 남을 가능성이 있다.
- **근데, inode 값을 알아도 소용이 없다...**
  - icat does not work… why?
- **게다가, xfs 파일시스템은 sleuthkit으로 분석이 불가능하다.**
  - xfs 파일 시스템 분석해서 툴을 만들던가...
  - 그냥 포기하든가...

- **To Do**
  - do icat working… or find the reason
  - xfs file system analysis ~~and write python script~~
  - forensicating with ELK

Reference

- https://medium.freecodecamp.org/a-beginner-friendly-introduction-to-containers-vms-and-docker-79a9e3e119b
- https://subicura.com/2017/01/19/docker-guide-for-beginners-1.html
- https://docs.docker.com
- https://tech.ssut.me/2017/08/15/what-even-is-a-container/
- https://docs.docker.com/storage/storagedriver/aufs-driver/
- https://docs.docker.com/storage/storagedriver/overlayfs-driver/

Question and Answer