Перечислите все известные вам способы обнаружения работы в виртуальной машине.

(>=5)

Сложный вариант (или)

1. Привести способ выхода из виртуальной машины
2. На ассемблере

Практика

1. Установить docker https://docs.docker.com/engine/install
2. Запустить ubuntu, пробросить порт 8000, запустить python -mSimpleHTTPServer, пробросить внутрь каталог, предоставить доступ к каталогу

Полезные команды в docker

1) sudo docker run -ti ubuntu /bin/bash - запустить контейнер и перейти в интерактивный режим

2) CTRL+P+Q - отсоединиться от контейнера, оставив его запущенным

3) docker ps -a - посмотреть список запущенных контейнеров

4) docker commit <id> <name> - сохранить контейнер id как образ name

5) docker images - получить список имеющихся образов

docker search - найти image по имени

docker pull <ИМЯ Image> - загрузить себе образ

6) docker attach <id> - соединиться с контейнером

7) docker run <Разные опции> -ti ubuntu /bin/bash - запустить контейнер и пробросить его сетевой интерфейс на интерфейс хоста - все открытые порты виртуалки будут открыты в реале (CTRL+P+Q)

docker ps -a - показать все контейнеры

docker attach <id контейнера> - приаттачиться к контейнеру

docker exec -ti <id контейнера> /bin/bash - исполнить команду в контейнере

docker run --net="host" -ti ubuntu /bin/bash - стартовать с нативным адаптером сети

-p <HOST PORT>:<GUEST PORT>

8) docker run -v host\_path:container\_path - пробросить в контейнер папку

9) docker export <id> <name> - экспортировать контейнер

10) cat name.tar | docker import - <name> - импортировать контейнер

Dockerfile - конфиг со способом создания Image

FROM ubuntu

RUN apt update

RUN apt install -y python3

WORKDIR /home/test

ENTRYPOINT python3 -m http.server 2000

docker build -t mycont .

docker-compose.yml - какие контейнеры создать и какие порты и папки в них пробросит

version: "2.0"

services:

my\_cointainer:

build: .

volumes:

- /tmp/test/shared:/home/test

ports:

- 8011:2000

docker-compose up --build -d

PAM

<https://www.opennet.ru/base/dev/pam_linux.txt.html>

SElinux

<https://habr.com/ru/company/kingservers/blog/209644/>

<https://defcon.ru/os-security/1264/>

Apparmor

<https://help.ubuntu.ru/wiki/%D1%80%D1%83%D0%BA%D0%BE%D0%B2%D0%BE%D0%B4%D1%81%D1%82%D0%B2%D0%BE_%D0%BF%D0%BE_ubuntu_server/%D0%B1%D0%B5%D0%B7%D0%BE%D0%BF%D0%B0%D1%81%D0%BD%D0%BE%D1%81%D1%82%D1%8C/apparmor>

<https://losst.ru/nastrojka-apparmor-v-ubuntu-16-04>

<https://manpages.ubuntu.com/manpages/bionic/en/man5/apparmor.d.5.html>

<https://defcon.ru/os-security/1544/> - история

<https://ru.bmstu.wiki/AppArmor> - правила

LSM

<https://habr.com/ru/company/pt/blog/144014/>

Windows: Руссинович, Внутреннее устройство Windows, стр 510

Маркер доступа - <https://ru.bmstu.wiki/Access_Token>

Дескриптор защиты - <https://intuit.ru/studies/courses/10471/1078/lecture/16581?page=2>

<https://docs.microsoft.com/en-us/windows/win32/api/winnt/ns-winnt-security_descriptor?redirectedfrom=MSDN>

Про подписывание драйверов <https://docs.microsoft.com/en-us/windows-hardware/drivers/install/kernel-mode-code-signing-policy--windows-vista-and-later->

ELAM <https://docs.microsoft.com/en-us/windows-hardware/drivers/install/early-launch-antimalware>

Kernel Patch Protector <https://xakep.ru/2019/11/25/byepg/>

EWT - https://docs.microsoft.com/en-us/windows/win32/etw/event-tracing-portal

HVCI - <https://docs.microsoft.com/en-us/windows/security/threat-protection/device-guard/enable-virtualization-based-protection-of-code-integrity>

Про UAC https://docs.microsoft.com/en-us/windows/security/identity-protection/user-account-control/how-user-account-control-works

Mã này trong ngôn ngữ lập trình Assembly có vẻ như là một phần của chương trình viết bằng ngôn ngữ Assembly x86 hoặc x86\_64 trên hệ thống Linux. Chương trình này có mục tiêu là phát hiện xem nó đang chạy trên máy ảo (VM) hay máy vật lý.

Đây là một số giải thích cho code:

```assembly

section .text

global \_start

\_start:

; Thiết lập các giá trị vào các thanh ghi để thực hiện kiểm tra VM hay máy vật lý.

mov eax, 564D5868h ; Đặt giá trị vào thanh ghi eax (đây có thể là một biểu diễn của mã VMWare)

mov ebx, 00000000h ; Xóa thanh ghi ebx (chuẩn bị sử dụng cho so sánh sau đó)

mov ecx, 0000000Ah ; Đặt một giá trị khác vào thanh ghi ecx (đây có thể là một hằng số tùy chỉnh)

mov edx, 00005658h ; Đặt giá trị vào thanh ghi edx (có thể là một cổng I/O đặc biệt cho việc kiểm tra)

; Thực hiện một phép đọc từ cổng I/O sử dụng hướng dẫn in (input) với giá trị của eax như là cổng.

in eax, dx ; Đọc từ cổng I/O và kết quả được lưu vào eax.

; So sánh giá trị trong thanh ghi ebx với giá trị mới được đọc từ cổng I/O.

cmp ebx, 564D5868h ; So sánh ebx với giá trị VMWare (564D5868h)

; Nếu không trùng khớp, nhảy tới phần code được gắn nhãn "phymachine".

jne phymachine

; Nếu trùng khớp, in ra chuỗi "VMware detected".

mov edx, len1 ; Đặt độ dài của chuỗi vào thanh ghi edx

mov ecx, msg1 ; Đặt địa chỉ của chuỗi vào thanh ghi ecx

mov ebx, 1 ; Chuẩn bị số 1 để gửi xuống syscall write (stdout)

mov eax, 4 ; Chuẩn bị số 4 để gọi syscall write

int 0x80 ; Gọi syscall để in chuỗi ra màn hình

; Nhảy đến điểm kết thúc chương trình

jmp end

end:

; Thực hiện exit syscall để kết thúc chương trình

mov ebx, 0 ; Chuẩn bị số 0 để gửi xuống syscall exit

mov eax, 1 ; Chuẩn bị số 1 để gọi syscall exit

int 0x80 ; Gọi syscall để kết thúc chương trình

; Nhãn "phymachine" - nhánh xử lý khi phát hiện máy vật lý

phymachine:

mov edx, len2 ; Đặt độ dài của chuỗi vào thanh ghi edx

mov ecx, msg2 ; Đặt địa chỉ của chuỗi vào thanh ghi ecx

mov ebx, 1 ; Chuẩn bị số 1 để gửi xuống syscall write (stdout)

mov eax, 4 ; Chuẩn bị số 4 để gọi syscall write

int 0x80 ; Gọi syscall để in chuỗi ra màn hình

; Nhảy đến điểm kết thúc chương trình

jmp end

; Định nghĩa chuỗi "VMware detected"

msg1: db "VMware detected",0xa

len1: equ $-msg1

; Định nghĩa chuỗi "Physical machine"

msg2: db "Physical machine",0xa

len2: equ $-msg2

```

Nó sử dụng `in` instruction để đọc từ cổng I/O với giá trị của `eax` như một cổng và sau đó so sánh kết quả với một giá trị đã biết (được cho là đại diện cho máy ảo VMWare). Nếu kết quả không trùng khớp, chương trình in ra "VMware detected". Ngược lại, nó in ra "Physical machine". Cuối cùng, chương trình kết thúc bằng cách gọi syscall exit.