DRAKVUF

Official Documentation version (https://drakvuf.com/)

Installation guide

Preparing the Host -

Hardware Requirements:

To run DRAKVUF Sandbox, your setup must fulfil all of the listed requirements:

- Processor: Intel processor with VT-x and EPT features. (how to check)
- Host system: Debian 10 Buster/Ubuntu 18.04 Bionic/Ubuntu 20.04 Focal with at least 6 core CPUs and 16 GB RAM, must be able to boot in “LEGACY MODE”.
- Guest system: Windows 7 (x64), Windows 10 (x64; experimental support.
- Cloud Architecture system not supported.
- VM-Ware, Virtual Box, or any virtual hypervisor are not supported; the sandbox has to be implemented on a physical machine with an Intel CPU with virtualisation capabilities.

Installation:

To build DRAKVUF® from source, the following packages are typically required on Debian/Ubuntu-based Linux distros:

```
sudo apt-get install wget git bcc bin86 gawk bridge-utils iproute2
libcurl4-openssl-dev bzip2 libpci-dev build-essential make gcc clang
libc6-dev linux-libc-dev zlib1g-dev libncurses5-dev patch libvncserver-dev
libssl-dev libsdl-dev iasl libbz2-dev e2fslibs-dev git-core uuid-dev ocaml
libx11-dev bison flex ocaml-findlib xz-utils gettext libyajl-dev
libpixman-1-dev libaio-dev libfdt-dev cabextract libglib2.0-dev autoconf
automake libtool libjson-c-dev libfuse-dev liblzma-dev autoconf-archive
kpartx python3-dev python3-pip golang python-dev libsystemd-dev nasm
ninja-build llvm lld
```

Some Python packages are unfortunately quite old in Debian, so we will install them with Python's pip3 tool:

```
sudo pip3 install pefile construct
```

Main Installation for Drakvuf and XEN:

```
cd ~
git clone https://github.com/tklengyel/drakvuf
cd drakvuf
git submodule update --init
cd xen
./configure --enable-githttp --enable-systemd --enable-ovmf
--disable-pvshim
make -j4 dist-xen
make -j4 dist-tools
make -j4 debball
```

Note- If "make -j4 dist-tools" command throws an error related to the OVMF build then, the possible solution is the installation/upgradation of the NASM module. (solution)

Xen, a popular open-source hypervisor, is often used to create and manage virtualised environments. In this context, "dom0" refers to the first domain, which is also known as the "control domain" or "privileged domain." Dom0 is a special VM with direct access to physical hardware and is responsible for managing other virtual machines (domU) and handling hardware resources.

```
sudo su
apt-get remove xen* libxen*
dpkg -i dist/xen*.deb
echo "GRUB_CMDLINE_XEN_DEFAULT=\"dom0_mem=4096M,max:4096M dom0_max_vcpus=4
dom0_vcpus_pin=1 force-ept=1 ept=ad=0 hap_1gb=0 hap_2mb=0 altp2m=1
hpet=legacy-replacement smt=0\"" >> /etc/default/grub
echo "/usr/local/lib" > /etc/ld.so.conf.d/xen.conf
ldconfig
echo "none /proc/xen xenfs defaults,nofail 0 0" >> /etc/fstab
echo "xen-evtchn" >> /etc/modules
echo "xen-privcmd" >> /etc/modules
echo "xen-gntdev" >> /etc/modules
systemctl enable xencommons.service
systemctl enable xen-qemu-dom0-disk-backend.service
systemctl enable xen-init-dom0.service
systemctl enable xenconsoled.service
```

```
update-grub
```

Also, ensure you are running a relatively recent kernel, 5.11+ at least. Older kernels in your dom0 will not work correctly!

```
uname -r
```

Once you are done with these steps, you can finalise your setup:

```
sudo reboot
```

Once you are booted into Xen, verify that everything works as such:

```
sudo xen-detect
```

The output should be:

```
Running in PV context on Xen
```

Then, to check whether all the XEN services are running :

```
xl list
```

The output should be something similar:

```
Name                                        ID   Mem VCPUs      State   Time(s)
Domain-0                                     0  4096     2      r-----  614.0
```

Use High speed internet only(no private network) via ethernet no wifi

Once Domain-0 is up and running the next step is to get a full-featured DRAKVUF GUI to set up as an automated analysis sandbox created by CERT-Polska:

https://github.com/CERT-Polska/drakvuf-sandbox