

Politechnika Wrocławska
Wydział Elektroniki

Kierunek: Cyberbezpieczeństwo

Ochrona centrów danych

Lab 4



Politechnika Wrocławska

Krzysztof Bocian

Wrocław, 2021

1.Wstęp

Celem tego laboratorium było przygotowanie dwóch programów symulujących TPM. Jedno z nich pochodzi od firmy Microsoft, a drugie od IBM. Oba te programy wymagają dodatkowych bibliotek, które trzeba było doinstalować.

2.Przebieg laboratorium

Na wcześniej utworzonej maszynie wirtualnej, która służyła nam do wcześniejszych zajęć uruchomiliśmy ubuntu 20.04 za pomocą kontenera w Dockerze. W tym celu wykorzystaliśmy komendę “sudo docker run -i ubuntu:20.04”. Argument “-i” pozwala na utworzenie interaktywnej sesji po uruchomieniu kontenera. Na maszynie nie znajdował się obraz ubuntu w wersji 20.04 dlatego docker automatycznie go pobrał. Po uruchomieniu kontenera pierwsze co zrobiliśmy to uruchomienie komend “update” i “upgrade”.

```
chris@chris-VirtualBox:~$ sudo docker run -it ubuntu:20.04
[sudo] password for chris:
Unable to find image 'ubuntu:20.04' locally
20.04: Pulling from library/ubuntu
7b1a6ab2e44d: Pull complete
Digest: sha256:626ffe58f6e7566e00254b638eb7e0f3b11d4da9675088f4781a50ae288f3322
Status: Downloaded newer image for ubuntu:20.04
root@34a421a63656:/# apt update
```

Rys.1 Instalacja kontenera z obrazem ubuntu 20.04

Kolejnymi krokami była instalacja dodatkowych bibliotek oraz repozytorium “libtpms”. Wszystkie biblioteki i repozytorium udało się pobrać bez problemów. Następnie wykonaliśmy komendy “checkout”, “mk-build-deps --install ./debian/control” oraz “debuild -us -uc”.

```
Setting up gawk (1:5.0.1+dfsg-1) ...
Setting up dh-exec (0.23.2) ...
Setting up pkg-config (0.29.1-0ubuntu4) ...
Setting up libtpms-build-deps (0.9.0) ...
Processing triggers for man-db (2.9.1-1) ...
root@34a421a63656:/libtpms#
```

Rys.2 Wynik komendy debuild

Następnie zainstalowałem paczki, które znajdowały się w repozytorium.

```
root@34a421a63656:/# ls
bin      libtpms                                libtpms_0.9.0_amd64.changes
boot     libtpms-dev_0.9.0_amd64.deb           libx32
dev      libtpms0-dbgsym_0.9.0_amd64.ddeb     media
etc      libtpms0_0.9.0_amd64.deb             mnt
home     libtpms_0.9.0.dsc                   opt
lib      libtpms_0.9.0.tar.xz                proc
lib32    libtpms_0.9.0_amd64.build           root
lib64    libtpms_0.9.0_amd64.buildinfo       run
root@34a421a63656:/# apt install ./*.deb
Reading package lists... Done
Building dependency tree
Reading state information... Done
```

Rys.3 Wykonanie komendy `apt install ./*.deb`

W dalszym etapie zbudowaliśmy SWTPM. W tym celu pobraliśmy kolejne repozytorium i procedura instalacji wyglądała podobnie do poprzedniego repozytorium. Jedną różnicą było wykonanie komendy “echo libtpms0” > ./debian/shlibs.local”

```
root@34a421a63656:/swtpm# echo "libtpms0 libtpms" > ./debian/shlibs.local
```

Rys.4 Wykonanie komendy `echo libtpms0" > ./debian/shlibs.local`

```
root@34a421a63656:/# apt install ./swtpm*.deb
Reading package lists... Done
Building dependency tree
Reading state information... Done
Note, selecting 'swtpm-dev' instead of './swtpm-dev_0.7.0_amd64.deb'
Note, selecting 'swtpm-libs' instead of './swtpm-libs_0.7.0_amd64.deb'
Note, selecting 'swtpm-tools' instead of './swtpm-tools_0.7.0_amd64.deb'
Note, selecting 'swtpm' instead of './swtpm_0.7.0_amd64.deb'
```

Rys.5 Wykonanie komendy `./swtpm*.deb`

Kolejnym krokiem była budowa MSSIM. Po pobraniu repozytorium i instalacji dodatkowych pakietów w katalogu TPMCmd należało wykonać komendy “./bootstrap”, “./configure” i “make”.

```
root@34a421a63656:/ms-tpm-20-ref/TPMCmd# ./bootstrap
Generating file lists: src.mk
Setting up build
```

Rys.6 Wykonanie komendy `./bootstrap`

```

root@34a421a63656:/ms-tpm-20-ref/TPMcmd# ./configure
checking for gcc... gcc
checking whether the C compiler works... yes
checking for C compiler default output file name... a.out
checking for suffix of executables...
checking whether we are cross compiling... no
checking for suffix of object files... o

```

Rys.7 Wykonanie komendy ./configure

Następnym etapem była budowa TSS. W tym kroku nastąpiły błędy podczas próby instalacji. Pierwszym z nich był następujący błąd.

```

./configure: line 12754: `AM_PATH_LIBGCRYPT(1.6.0, , AC_MSG_ERROR([Missing required gcrypt library]))'
root@34a421a63656:/tpm2-tss# apt install libgcrypt-dev

```

Rys.8 Błąd związany z brakiem biblioteki libgcrypt-dev

Aby naprawić ten błąd wystarczyło zainstalować dodatkową paczkę “libgcrypt-dev”. Niestety w dalszym kroku tej części pojawił się kolejny błąd, który umieszczamy poniżej.

```

! -d /usr/local/var/lib/tpm2-tss/system/keystore/ ]; then mkdir -p /usr/local/var/lib/tpm2-tss/system/keys
i)) && (all set_fapi_permissions)) || true
/bin/bash: systemd-sysusers: command not found
102
101
/bin/bash: line 1: all: command not found
if [ ! -z "" ]; then \
    mv /usr/local/lib/udev/rules.d/tpm-udev.rules /usr/local/lib/udev/rules.d/tpm-udev.rules; \
fi
make[2]: Leaving directory '/tpm2-tss'
make[1]: Leaving directory '/tpm2-tss'

```

Rys.9 Błąd po wykonaniu komendy “make install”