

Politechnika Wrocławska

Wydział Elektroniki

Kierunek: Cyberbezpieczeństwo

Ochrona Centrów Danych

Raport 4

Aneta Prządka 227164

1. Cel ćwiczenia

W ćwiczeniu czwartym należało zainstalować dwa programy symulujące TPM: firmy IBM (SWTPM) oraz firmy Microsoft (MSSIM). Należało uruchomić programy i udokumentować to na screenach w celu potwierdzenia poprawności instalacji.

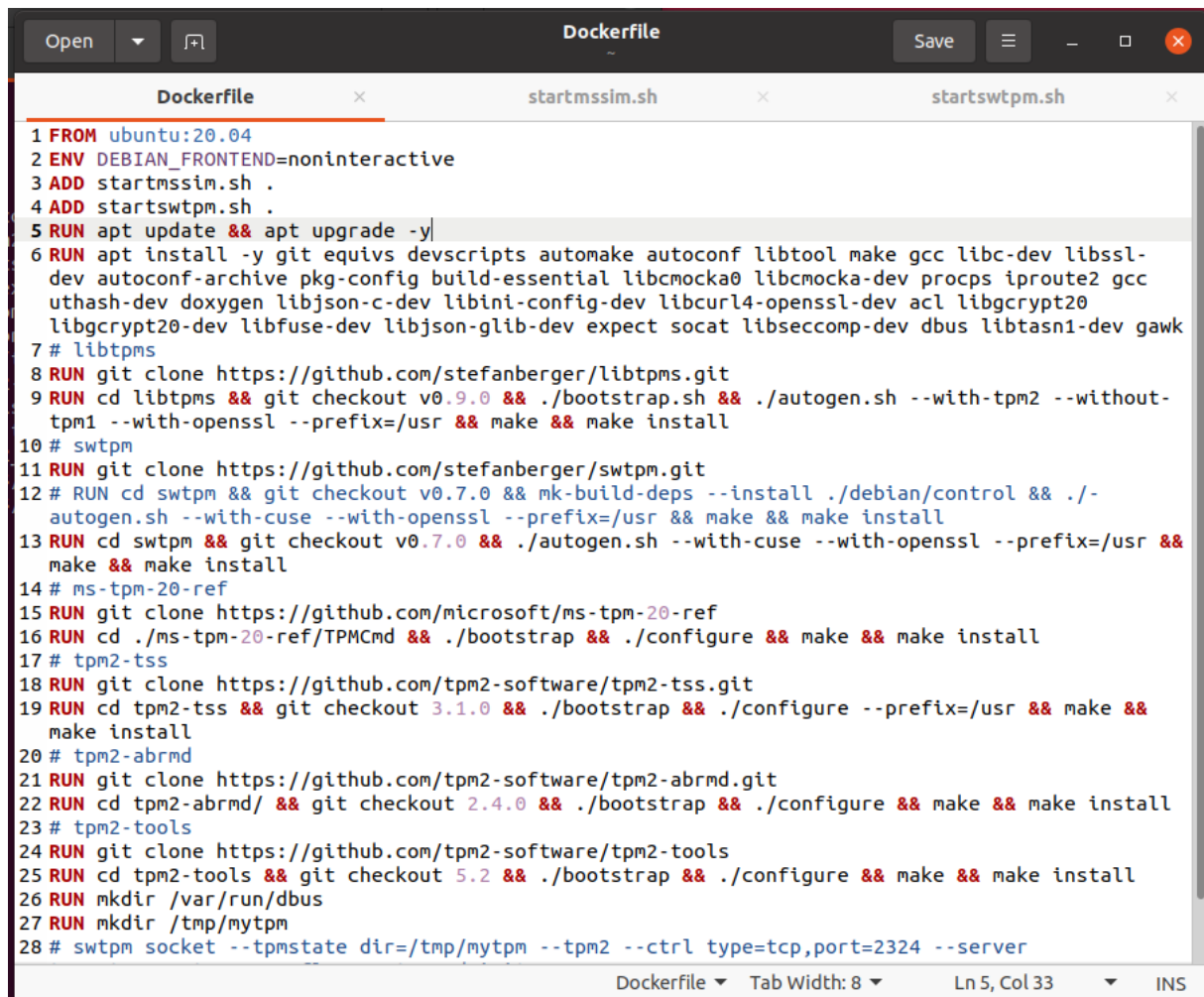
2. Przebieg ćwiczenia

Na maszynie z zainstalowanym dockerem należało uruchomić go z zainstalowanym obrazem Ubuntu w wersji 20.0.4. Następnie należało zainstalować podane niżej narzędzia:

- libtpms v0.9.0
- swtpm v0.7.0
- mssim (master)
- tss 3.1.0
- tabrmd 2.4.0
- tools 5.2

Na początku instalacji każdego narzędzia z osobna należało za pomocą komendy git clone pobrać podane repozytorium, a następnie wchodząc w folder pobranego repo zainstalować wymagane pakiety.

Zdecydowano się stworzyć dockerfile, którego zawartość widać poniżej:



```
1 FROM ubuntu:20.04
2 ENV DEBIAN_FRONTEND=noninteractive
3 ADD startmssim.sh .
4 ADD startswtpm.sh .
5 RUN apt update && apt upgrade -y
6 RUN apt install -y git equivs devscripts automake autoconf libtool make gcc libc-dev libssl-
  dev autoconf-archive pkg-config build-essential libcmocka0 libcmocka-dev procs iproute2 gcc
  uthash-dev doxygen libjson-c-dev libini-config-dev libcurl4-openssl-dev acl libgcrypt20
  libgcrypt20-dev libfuse-dev libjson-glib-dev expect socat libseccomp-dev dbus libtasn1-dev gawk
7 # libtpms
8 RUN git clone https://github.com/stefanberger/libtpms.git
9 RUN cd libtpms && git checkout v0.9.0 && ./bootstrap.sh && ./autogen.sh --with-tpm2 --without-
  tpm1 --with-openssl --prefix=/usr && make && make install
10 # swtpm
11 RUN git clone https://github.com/stefanberger/swtpm.git
12 # RUN cd swtpm && git checkout v0.7.0 && mk-build-deps --install ./debian/control && ./-
  autogen.sh --with-cuse --with-openssl --prefix=/usr && make && make install
13 RUN cd swtpm && git checkout v0.7.0 && ./autogen.sh --with-cuse --with-openssl --prefix=/usr &&
  make && make install
14 # ms-tpm-20-ref
15 RUN git clone https://github.com/microsoft/ms-tpm-20-ref
16 RUN cd ./ms-tpm-20-ref/TPMcmd && ./bootstrap && ./configure && make && make install
17 # tpm2-tss
18 RUN git clone https://github.com/tpm2-software/tpm2-tss.git
19 RUN cd tpm2-tss && git checkout 3.1.0 && ./bootstrap && ./configure --prefix=/usr && make &&
  make install
20 # tpm2-abrmd
21 RUN git clone https://github.com/tpm2-software/tpm2-abrmd.git
22 RUN cd tpm2-abrmd/ && git checkout 2.4.0 && ./bootstrap && ./configure && make && make install
23 # tpm2-tools
24 RUN git clone https://github.com/tpm2-software/tpm2-tools
25 RUN cd tpm2-tools && git checkout 5.2 && ./bootstrap && ./configure && make && make install
26 RUN mkdir /var/run/dbus
27 RUN mkdir /tmp/mytpm
28 # swtpm socket --tpmstate dir=/tmp/mytpm --tpm2 --ctrl type=tcp,port=2324 --server
```

Rysunek 1. Zawartość Dockerfile.

```
FROM ubuntu:20.04

ENV DEBIAN_FRONTEND=noninteractive

ADD startmssim.sh .

ADD startswtpm.sh .

RUN apt update && apt upgrade -y

RUN apt install -y git equivs devscripts automake autoconf libtool make gcc libc-dev libssl-
dev autoconf-archive pkg-config build-essential libcmocka0 libcmocka-dev procs iproute2 gcc
uthash-dev doxygen libjson-c-dev libini-config-dev libcurl4-openssl-dev acl libgcrypt20
libgcrypt20-dev libfuse-dev libjson-glib-dev expect socat libseccomp-dev dbus libtasn1-dev
gawk

# libtpms

RUN git clone https://github.com/stefanberger/libtpms.git

RUN cd libtpms && git checkout v0.9.0 && ./bootstrap.sh && ./autogen.sh --with-tpm2 --without-
tpm1 --with-openssl --prefix=/usr && make && make install

# swtpm

RUN git clone https://github.com/stefanberger/swtpm.git

# RUN cd swtpm && git checkout v0.7.0 && mk-build-deps --install ./debian/control &&
./autogen.sh --with-cuse --with-openssl --prefix=/usr && make && make install

RUN cd swtpm && git checkout v0.7.0 && ./autogen.sh --with-cuse --with-openssl --prefix=/usr
&& make && make install
```

```
# ms-tpm-20-ref
RUN git clone https://github.com/microsoft/ms-tpm-20-ref
RUN cd ./ms-tpm-20-ref/TPMCmd && ./bootstrap && ./configure && make && make install

# tpm2-tss
RUN git clone https://github.com/tpm2-software/tpm2-tss.git
RUN cd tpm2-tss && git checkout 3.1.0 && ./bootstrap && ./configure --prefix=/usr && make && make install

# tpm2-abrmd
RUN git clone https://github.com/tpm2-software/tpm2-abrmd.git
RUN cd tpm2-abrmd/ && git checkout 2.4.0 && ./bootstrap && ./configure && make && make install

# tpm2-tools
RUN git clone https://github.com/tpm2-software/tpm2-tools
RUN cd tpm2-tools && git checkout 5.2 && ./bootstrap && ./configure && make && make install
RUN mkdir /var/run/dbus
RUN mkdir /tmp/mytpm

# swtpm socket --tpmstate dir=/tmp/mytpm --tpm2 --ctrl type=tcp,port=2324 --server
type=tcp,port=2323 --flags not-need-init &

# tpm2-abrmd --allow-root -t swtpm:port=2323 &
RUN cp /usr/local/etc/dbus-1/system.d/tpm2-abrmd.conf /etc/dbus-1/system.d/
```

Oraz zawartość dodatkowych skryptów:

- **startwtpm.sh:**

```
dbus-daemon --system

swtpm socket --tpmstate dir=tmp/mytpm --tpm2 --ctrl type=tcp,port=2324 --server
type=tcp,port=2323 --flags not-need-init &

tpm2-abrmd --allow-root -t mssim:port=2325 &

tpm2_startup -c
```

- **startmssim.sh:**

```
dbus-daemon --system

tpm2-simulator 2325 -m &

tpm2-abrmd --allow-root -t mssim:port=2325 &
```

Dodatkowe komendy do użycia w terminalu, aby przetestować działanie Dockerfile:

```
docker build . -f Dockerfile -t tpmsimulator
docker run -it tpmsimulator bash

docker container rm lucid_jepsen funny_turing dazzling_newton recursing_bhabha
condescending_visvesvaraya elated_pasteur
```

Próba działania symulatora tpm2_pcrread:

