Annexe C

Infrastructure Docker OpenZiti N3

Cette annexe est une marche à suivre pour reproduire l'installation de la deuxième infrastructure OpenZiti déployée en partie sur des Raspberry Pi 4 ainsi que Docker (section 10.3).

Les Raspberry Pi 4 ont au préalables été préparé en installant Rasberry Pi OS grâce à l'utilitaire fournis sur le site https://www.raspberrypi.com/software/. La version 64bits a été installée et le service SSH activé.

Les différents fichiers sont disponibles à l'adresse suivante :

https://github.com/MehSalhi/TB-ZeroTrust-OpenZiti/tree/master/Network/N3

C.1 Composants

Contrôleur OpenZiti Installé sur le laptop dans un container Docker

Initialisateur du contrôleur Installé sur le laptop dans un container Docker

Edge router OpenZiti Cloud Installé sur un Raspberry

Edge router OpenZiti Laptop Installé sur le laptop dans un container Docker

Edge router OpenZiti GCS Installé sur un Raspberry

Console web OpenZiti Installé sur le laptop dans un container Docker

Drone Simulé par un Raspberry. Diffuse un flux vidéo, accessible via service SSH et 'echo'. L'application ziti-tunneller y est installée pour intercepter le trafic à destination du réseau OpenZiti

GCS Simulé par un Raspberry. L'application ziti-tunneller y est installée pour intercepter le trafic à destination du réseau OpenZiti

Base Simulé par un Raspberry

Police fédérale Identité créée depuis le contrôleur. Il est possible de s'inscrire en scannant un code QR via une application mobile

Serveur vidéo reçoit le flux et le retransmet à des clients. Installé sur le laptop dans un container Docker

C.2 Pré-requis

Les logiciels suivants doivent être installés :

- Docker
- Docker compose

C.3 Marche à suivre

C.3.1 Environnement

Créer un fichier suivant :

(Fichier disponible sur Github dans le répertoire 'N3/')

Code source C.3.1: docker-compose.yaml

```yaml
# Auteur      : Mehdi Salhi
# Auteur      : OpenZiti documentation
# Sujet       : Travail de Bachelor Zero Trust OpenZiti
# But         : Déploie une infrastructure OpenZiti
# No          : n3
# Description : 1 contrôleur, 1 routeur, 1 initialisteur, 1 serveur vidéo.
#         Tous les fichiers sont sauvegardés dans le répertoire
#         local "vol", et monté dans le répertoire "persistent"
#          sur les containers
version: '2.4'
services:
  # openziti controller
  # accessible depuis internet
  ziti-controller:
    hostname: ziti-edge-controller
    image: "${ZITI_IMAGE}:${ZITI_VERSION}"
    env_file:
      - ./.env
    ports:
      - ${ZITI_EDGE_CONTROLLER_PORT:-1280}:${ZITI_EDGE_CONTROLLER_PORT:-1280}
      - ${ZITI_CTRL_PORT:-6262}:${ZITI_CTRL_PORT:-6262}
    environment:
```

```
23      -
    ↪ ZITI_EDGE_IDENTITY_ENROLLMENT_DURATION=${ZITI_EDGE_IDENTITY_ENROLLMENT_DURATION}
24      - ZITI_EDGE_ROUTER_ENROLLMENT_DURATION=${ZITI_EDGE_ROUTER_ENROLLMENT_DURATION}
25    volumes:
26      - type: bind
27        source: ./vol
28        target: /persistent
29      - "/etc/timezone:/etc/timezone:ro"
30      - "/etc/localtime:/etc/localtime:ro"
31    entrypoint:
32      - "/var/openziti/scripts/run-controller.sh"
33    networks:
34      ziti:
35         aliases:
36           - ziti-edge-controller
37
38  # controller init
39  ziti-controller-init-container:
40    image: "${ZITI_IMAGE}:${ZITI_VERSION}"
41    depends_on:
42      - ziti-controller
43    environment:
44      - ZITI_CONTROLLER_RAWNAME="${ZITI_CONTROLLER_RAWNAME}"
45      - ZITI_EDGE_CONTROLLER_RAWNAME="${ZITI_EDGE_CONTROLLER_RAWNAME}"
46    env_file:
47      - ./.env
48    networks:
49      - ziti
50    volumes:
51      - type: bind
52        source: ./vol
53        target: /persistent
54      - "/etc/timezone:/etc/timezone:ro"
55      - "/etc/localtime:/etc/localtime:ro"
56    entrypoint:
57      - "/var/openziti/scripts/run-with-ziti-cli.sh"
58    command:
59      - "/var/openziti/scripts/access-control.sh"
60
61 ###################################
62 # Routers
63 ###################################
64
65  # edge router laptop
66  ziti-edge-router-laptop:
67    image: "${ZITI_IMAGE}:${ZITI_VERSION}"
68    hostname: ziti-edge-router-laptop
69    depends_on:
70      - ziti-controller
```

```
71      environment:
72        - ZITI_CONTROLLER_RAWNAME="${ZITI_CONTROLLER_RAWNAME}"
73        - ZITI_EDGE_CONTROLLER_RAWNAME="${ZITI_EDGE_CONTROLLER_RAWNAME}"
74          #-
    ↪   ZITI_EDGE_ROUTER_RAWNAME=${ZITI_EDGE_ROUTER_RAWNAME:-ziti-edge-router-laptop}
75        - ZITI_EDGE_ROUTER_RAWNAME="ziti-edge-router-laptop"
76        - ZITI_EDGE_ROUTER_ROLES=public
77      ports:
78        - ${ZITI_EDGE_ROUTER_PORT:-3022}:${ZITI_EDGE_ROUTER_PORT:-3022}
79        - 10080:10080
80      networks:
81        ziti:
82          aliases:
83            - ziti-edge-router-laptop
84      privileged: true
85      volumes:
86        - type: bind
87          source: ./vol
88          target: /persistent
89        - "/etc/timezone:/etc/timezone:ro"
90        - "/etc/localtime:/etc/localtime:ro"
91      entrypoint: ["bash", "-c", sleep 5 && /var/openziti/ziti-bin/ziti-router run
    ↪   /persistent/ziti-edge-router-laptop.yaml]
92      #stdin_open: true
93      #tty: true
94
95    # console
96    ziti-console:
97      image: openziti/zac
98      environment:
99        - ZAC_SERVER_CERT_CHAIN=/persistent/pki/${ZITI_EDGE_CONTROLLER_HOSTNAME:-ziti-⌋
    ↪   controller}-intermediate/certs/${ZITI_EDGE_CONTROLLER_HOSTNAME:-ziti-controller}⌋
    ↪   -server.cert
100       -
    ↪   ZAC_SERVER_KEY=/persistent/pki/${ZITI_EDGE_CONTROLLER_HOSTNAME:-ziti-controller}⌋
    ↪   -intermediate/keys/${ZITI_EDGE_CONTROLLER_HOSTNAME:-ziti-controller}-server.key
101       - PORTTLS=8443
102     ports:
103       - 1408:1408
104       - 8443:8443
105     working_dir: /usr/src/app
106     volumes:
107       - type: bind
108         source: ./vol
109         target: /persistent
110       - "/etc/timezone:/etc/timezone:ro"
111       - "/etc/localtime:/etc/localtime:ro"
112     networks:
113       - ziti
```

```yaml

    ##### machines
    # video server
  video-server:
    image: mehdi/rtmp-hls_server
    build:
      dockerfile: ./rtmp-hls-server/Dockerfile
      context: .
      network: host
    depends_on:
      - ziti-controller
      - ziti-edge-router-laptop
    cap_add:
      - NET_ADMIN
    ports:
      - 8080:8080
      - 1935:1935
    volumes:
      - type: bind
        source: ./vol
        target: /persistent
    healthcheck:
      test: curl --fail http://localhost:8080 || exit 1
    entrypoint:
      - "/persistent/tunnel-server.sh"
    networks:
      ziti:
        aliases:
          - video-server

# networks
networks:
  ziti:
    driver: bridge
```

Dans le même répertoire que le fichier docker-compose.yaml, mettre le fichier .env suivant : (Fichier disponible sur Github dans le répertoire 'N3/')

Code source C.3.2: .env

```
# OpenZiti Variables
ZITI_IMAGE=openziti/quickstart
ZITI_VERSION=latest

# The duration of the enrollment period (in minutes), default if not set
# shown - 7days
ZITI_EDGE_IDENTITY_ENROLLMENT_DURATION=10080
ZITI_EDGE_ROUTER_ENROLLMENT_DURATION=10080
```

```
9
10 # controller address/port information
11 ZITI_CONTROLLER_RAWNAME=ziti-controller
12 #ZITI_CONTROLLER_HOSTNAME=advertised.address
13 #ZITI_CTRL_PORT=8440
14
15 ZITI_EDGE_CONTROLLER_RAWNAME=ziti-edge-controller
16 #ZITI_EDGE_CONTROLLER_HOSTNAME=advertised.address
17 #ZITI_EDGE_CONTROLLER_PORT=8441
18 #ZITI_EDGE_CONTROLLER_IP_OVERRIDE=172.17.0.1
19
20 # router address/port information
21 #ZITI_EDGE_ROUTER_RAWNAME=advertised.address
22 #ZITI_EDGE_ROUTER_PORT=8442
23 #ZITI_EDGE_ROUTER_IP_OVERRIDE=172.17.0.1
```

C.4 Services, identités, configurations

Commande pour créer les différents services, configurations, accès, etc. :

```
####### Routeur laptop ########
# créer l'identité du routeur sur le laptop

ziti create config router edge --routerName  ziti-edge-router-laptop \
                               --output ziti-edge-router-laptop.yaml

# modifier le fichier ziti-edge-router-laptop.yaml et changer les
↪  occurences de
"ziti-edge-router" en "ziti-edge-router-laptop"

# créer le jeton pour le routeur laptop
iti edge create edge-router ziti-edge-router-laptop --jwt-output-file
↪  ziti-edge-router-laptop.jwt --tunneler-enabled -a public

# inscrire le routeur auprès du contrôleur depuis le routeur laptop
ziti-router enroll ziti-edge-router-laptop.yaml --jwt
↪  ziti-edge-router-laptop.jwt
```

```
####### Raspberry ########

# installer ziti-edge-tunnel sur les raspberry
sudo apt update
sudo curl -sSLf https://get.openziti.io/tun/package-repos.gpg \
| gpg --dearmor \
| sudo tee /usr/share/keyrings/openziti.gpg >/dev/null
sudo echo 'deb [signed-by=/usr/share/keyrings/openziti.gpg]
↪  https://packages.openziti.org/zitipax-openziti-deb-stable focal
↪  main' \
| sudo tee /etc/apt/sources.list.d/openziti.list >/dev/null
sudo apt update
sudo apt install -y ziti-edge-tunnel
```

```
####### Drone ########

# depuis contrôleur
ziti edge create identity user drone_rpi7.ziti -o drone_rpi7.ziti.jwt

# copier le jwt vers le raspberry via ssh
sudo scp drone_rpi7.ziti.jwt pi@192.168.1.203:/home/pi

# inscrire le raspberry drone auprès du contrôleur
sudo ziti-edge-tunnel enroll --jwt ./drone_rpi7.ziti.jwt --identity
↪  /opt/openziti/etc/identities/drone_rpi7.ziti.json

# transférer jwt sur rasp
sudo scp drone_rpi7.ziti.jwt pi@192.168.1.203:/home/pi

# depuis le drone, s'inscrire auprès du contrôleur
sudo ziti-edge-tunnel enroll --jwt ./drone_rpi7.ziti.jwt --identity
↪  /opt/openziti/etc/identities/drone_rpi7.ziti.json

# installer gstreamer sur rpi7 drone
sudo apt install -y libgstreamer1.0-dev
↪  libgstreamer-plugins-base1.0-dev libgstreamer-plugins-bad1.0-dev
↪  gstreamer1.0-plugins-base gstreamer1.0-plugins-good
↪  gstreamer1.0-plugins-bad gstreamer1.0-plugins-ugly
↪  gstreamer1.0-libav gstreamer1.0-tools gstreamer1.0-x
↪  gstreamer1.0-alsa gstreamer1.0-gl gstreamer1.0-gtk3
↪  gstreamer1.0-qt5 gstreamer1.0-pulseaudio
```

Pour que le tunnel aini que le stream se lance au démarrage, ajouter le script suivant dans : /etc/init.d/boot.sh

Code source C.4.1: /etc/init.d/boot.sh

```
1 #! /bin/bash
2
3 # boot script
4 # runs the ziti tunnel and router
5
6 ### BEGIN INIT INFO
7 # Provides:          boot.sh
8 # Required-Start:    $remote_fs $syslog
9 # Required-Stop:     $remote_fs $syslog
```

```
# Default-Start:     2 3 4 5
# Default-Stop:      0 1 6
# Short-Description: Start ziti tunnel and router
# Description:
### END INIT INFO

# ziti
ziti-edge-tunnel run -i /opt/openziti/etc/identities/gcs_rpi9.ziti.json &
/opt/openziti/ziti-router run /home/pi/ziti-edge-router-gcs.yaml &
# boot script
# runs the ziti tunnel, gstreamer, echo

# ziti
ziti-edge-tunnel run -i /opt/openziti/etc/identities/drone_rpi7.ziti.json &

ncat -t -l 9999 -e /bin/cat &

# gstreamer
while true
do
gst-launch-1.0 filesrc location=/home/pi/nautilus.mp4 ! qtdemux ! decodebin !
  ↪  x264enc tune=zerolatency key-int-max=60 ! flvmux ! rtmpsink
  ↪  location="rtmp://video-server.ziti/live/test live=1"

sleep 5
done
```

```
# mettre à jour init.d avec le nouveau script
sudo update-rc.d boot.sh defaults
```

```
####### OpenWRT ########

## mettre une entrée dns vers ziti-edge-controlleur sur le router
↪  openwrt

sous http://192.168.1.1/cgi-bin/luci/admin/network/hosts
hostname: ziti-edge-controller
ipv4-address : ip du laptop (92.168.1.145)

# de même avec le routeur laptop et le serveur vidéo

hostname: ziti-edge-router-laptop
ipv4-address : ip du laptop (92.168.1.145)

hostname: video-server
ipv4-address : ip du laptop (92.168.1.145)
```

```
####### GCS rpi9 ########
# depuis contrôler
ziti edge create identity user gcs_rpi9.ziti -o gcs_rpi9.jwt
sudo scp gcs_rpi9.ziti.jwt pi@192.168.1.170:/home/pi

# depuis rpi9
sudo ziti-edge-tunnel enroll --jwt ./gcs_rpi9.jwt --identity
↪  /opt/openziti/etc/identities/gcs_rpi9.ziti.json
sudo chmod +r /opt/openziti/etc/identities/gcs_rpi9.ziti.json
```

Pour que le tunnel ainsi que le stream se lance au démarrage, ajouter le script suivant dans : /etc/init.d/boot.sh

Code source C.4.2: /etc/init.d/boot.sh

```
1 #!/bin/bash
2
3 # boot script
4 # runs the ziti tunnel and router
5
6 ### BEGIN INIT INFO
7 # Provides:          boot.sh
8 # Required-Start:    $remote_fs $syslog
9 # Required-Stop:     $remote_fs $syslog
```

```
# Default-Start:     2 3 4 5
# Default-Stop:      0 1 6
# Short-Description: Start ziti tunnel and router
# Description:
### END INIT INFO

# ziti
ziti-edge-tunnel run -i /opt/openziti/etc/identities/gcs_rpi9.ziti.json &
/opt/openziti/bin/ziti-router run /home/pi/ziti-edge-router-gcs.yaml &
```

```
# mettre à jour init.d avec le nouveau script
sudo update-rc.d boot.sh defaults
```

Serveur vidéo

```
# création de l'identité depuis le contrôleur
ziti edge create identity user video-server.ziti -o
↪  video-server.ziti.jwt

# inscription depuis le serveur
ziti-edge-tunnel enroll --jwt video-server.ziti.jwt --identity
↪  video-server.ziti.json
```

Service vidéo

```
ziti edge create config video.host.v1 host.v1 '{"protocol":"tcp",
↪  "address":"'"video-server"'", "port":1935}'
ziti edge create config drone-video.intercept.v1 intercept.v1
↪  '{"protocols":["tcp"],"addresses":["video-server.ziti"],
↪  "portRanges":[{"low":1935, "high":1935}]}'
ziti edge create service video.svc --configs
↪  drone-video.intercept.v1,video.host.v1
ziti edge create service-policy drone-video.policy.dial Dial
↪  --service-roles "@video.svc" --identity-roles '#drone-video'
ziti edge create service-policy video.policy.bind Bind --service-roles
↪  '@video.svc' --identity-roles "@video-server.ziti"
#ajouer tag "drone-video" à drone
```

Service echo

```
# config echo host
ziti edge create config echo.host.v1 host.v1 '{"protocol":"tcp",
↪  "address":"'"localhost"'", "port":9999}'
# config echo intercept
ziti edge create config echo.intercept.v1 intercept.v1
↪  '{"protocols":["tcp"],"addresses":["drone.ziti"],
↪  "portRanges":[{"low":9999, "high":9999}]}'
# service echo
ziti edge create service echo.svc --configs
↪  echo.intercept.v1,echo.host.v1
# dial echo
ziti edge create service-policy echo.policy.dial Dial --service-roles
↪  "@echo.svc" --identity-roles '#echo'
# bind echo
ziti edge create service-policy echo.policy.bind Bind --service-roles
↪  '@echo.svc' --identity-roles "@drone_rpi7.ziti"

# ajouter attribut "echo" aux gcs
```

Service ssh

```
ziti edge create config drone.ssh.host.v1 host.v1 '{"protocol":"tcp",
↪  "address":"'"localhost"'", "port":22}'
ziti edge create config drone-ssh.intercept.v1 intercept.v1
↪  '{"protocols":["tcp"],"addresses":["drone.ziti"],
↪  "portRanges":[{"low":22, "high":22}]}'
ziti edge create service drone-ssh.svc --configs
↪  drone-ssh.intercept.v1,drone.ssh.host.v1
ziti edge create service-policy drone-ssh.policy.dial Dial
↪  --service-roles "@drone-ssh.svc" --identity-roles '#drone-ssh'
ziti edge create service-policy drone-ssh.policy.bind Bind
↪  --service-roles '@drone-ssh.svc' --identity-roles
↪  "@drone_rpi7.ziti"
```

Routeur et tunnel GCS

```
# créer router gcs puis enroll
ziti create config router edge --routerName  ziti-edge-router-gcs \
                               --output ziti-edge-router-gcs.yaml

# modif ziti-edge-router-gcs.yaml pour ajouer "gcs" dans les noms
"ziti-edge-router"

# changer également les repertoires ou seront sauvegardées les clés et
↪  les
# certificats tout en haut du fichier. Par exemple /home/pi

ziti edge create edge-router ziti-edge-router-gcs --jwt-output-file
↪  ziti-edge-router-gcs.jwt --tunneler-enabled -a public

sudo scp ziti-edge-router-gcs.jwt pi@192.168.1.170:/home/pi

# editer yaml pour changer repertoire ou sont stocké clé et co

ziti-router enroll ziti-edge-router-gcs.yaml --jwt
↪  ziti-edge-router-gcs.jwt
```

Pour que le routeur et le tunnel se lancent au démarrage, ajouter le script suivant dans : /etc/init.d/boot.sh

Code source C.4.3: /etc/init.d/boot.sh

```
1  #!/bin/bash
2
3  # boot script
4  # runs the ziti tunnel and router
5
6  ### BEGIN INIT INFO
7  # Provides:          boot.sh
8  # Required-Start:    $remote_fs $syslog
9  # Required-Stop:     $remote_fs $syslog
10 # Default-Start:     2 3 4 5
11 # Default-Stop:      0 1 6
12 # Short-Description: Start ziti tunnel and router
13 # Description:
14 ### END INIT INFO
15
16 # ziti
17 ziti-edge-tunnel run -i /opt/openziti/etc/identities/gcs_rpi9.ziti.json &
18 /opt/openziti/bin/ziti-router run /home/pi/ziti-edge-router-gcs.yaml &
```

```
# mettre à jour init.d avec le nouveau script
sudo update-rc.d boot.sh defaults
```

Routeur "cloud"

```
# créer router cloud puis enroll
ziti create config router edge --routerName  ziti-edge-router-cloud \
                               --output ziti-edge-router-cloud.yaml

# modif ziti-edge-router-cloud.yaml pour ajouer "cloud" dans les noms
"ziti-edge-router"

# changer également les repertoires ou seront sauvegardées les clés et
↪  les
# certificats tout en haut du fichier. Par exemple /home/pi


ziti edge create edge-router ziti-edge-router-cloud --jwt-output-file
↪  ziti-edge-router-cloud.jwt --tunneler-enabled -a public

sudo scp ziti-edge-router-cloud.jwt pi@192.168.1.224/home/pi

# editer yaml pour changer repertoire ou sont stocké clé et co

ziti-router enroll ziti-edge-router-cloud.yaml --jwt
↪  ziti-edge-router-cloud.jwt
```

Pour que le routeur et le tunnel se lancent au démarrage, ajouter le script suivant dans : /etc/init.d/boot.sh

Code source C.4.4: /etc/init.d/boot.sh

```
1  #!/bin/bash
2
3  # boot script
4  # runs the ziti tunnel and router
5
6  ### BEGIN INIT INFO
7  # Provides:          boot.sh
8  # Required-Start:    $remote_fs $syslog
9  # Required-Stop:     $remote_fs $syslog
10 # Default-Start:     2 3 4 5
```

```
# Default-Stop:      0 1 6
# Short-Description: Start ziti router
# Description:
### END INIT INFO

# ziti
/opt/openziti/bin/ziti-router run /home/pi/ziti-edge-router-cloud.yaml &
```

```
# mettre à jour init.d avec le nouveau script
sudo update-rc.d boot.sh defaults
```

Identité police fédérale

Pour cette identité, on choisit de s'y inscrire via un smartphone avec l'application android. Le réseau OpenZiti doit être atteignable depuis le résea ou se trouve le smartphone. Dans ce cas, un réseau wifi a été créé sur le routeur / point d'accès openWRT qui interconnecte les apparels. Une fois l'identité créée, il est possible de simplement scanner le jeton d'accès au format QR code depuis l'application android. On obtient alors directement accès aux réseau OpenZiti ainsi qu'aux services disponibles. Dans le cas d'utilisation de la Rega, cela est une bonne option pour donner rapidement accès à des intervenants tels que les secouristes, ambulanciers, police, etc.

```
ziti edge create identity user fp.ziti -a "echo"
```

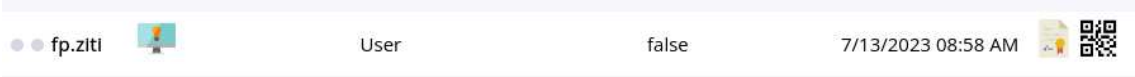

FIGURE C.1 – Console web - Inscription via certificat où QR code

FIGURE C.2 – Console web - Inscription via QR Code

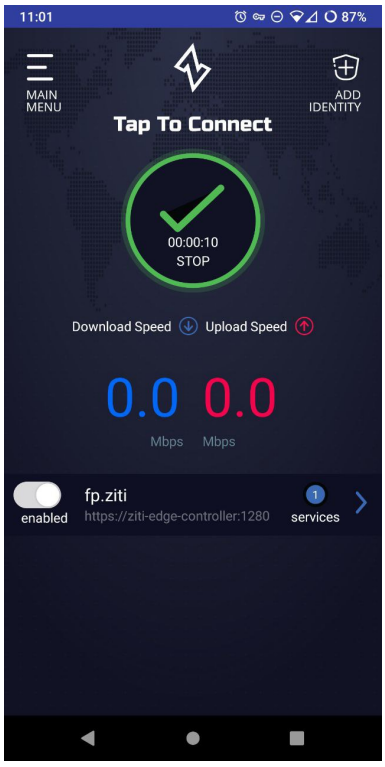

FIGURE C.3 – Inscription via app Android