

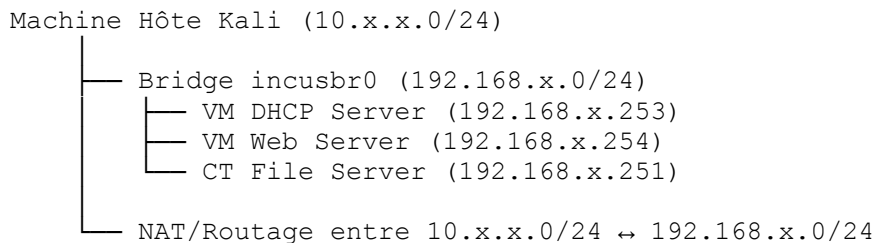
TP Incus - Virtualisation et Administration Système

Objectifs du TP

À l'issue de ce TP, vous serez capable de :

- Installer et configurer Incus (système de conteneurs et machines virtuelles)
- Créer et gérer des réseaux virtuels avec NAT et routage
- Déployer une infrastructure complète avec VM et conteneurs
- Administrer des services réseaux (DHCP, Web, Fichiers)
- Diagnostiquer et résoudre des problèmes de connectivité

Architecture cible



Planification

- **Séance 1 (2,5h)** : Installation, configuration réseau, VM DHCP
- **Séance 2 (2,5h)** : VM Web, CT Fichiers, interconnexions
- **Séance 3 (1h)** : Tests globaux, optimisation, troubleshooting

Séance 1 : Installation et Configuration de Base

Étape 1.1 : Vérification des prérequis

Q1.1 Vérifiez que votre système supporte la virtualisation :

```
# Commande à compléter pour vérifier les extensions CPU
grep -E '(___) ' /proc/____
```

```
# Vérifier que KVM est disponible
ls /dev/____
```

Q1.2 Vérifiez l'état du réseau hôte :

```
# Afficher la configuration IP actuelle
ip ____ show
```

```
# Identifier l'interface connectée au réseau 10.x.x.0/24
```

Étape 1.2 : Installation d'Incus

Q1.3 Correction :

```
# Mise à jour complète
sudo apt update && sudo apt upgrade -y

# Installation complète avec dépendances (Kali Linux)
sudo apt install incus incus-extra qemu-system-x86 qemu-utils ovmf bridge-
utils iptables -y
```

Q1.4 Configuration post-installation :

```
# Ajouter votre utilisateur au groupe incus-admin
sudo usermod -aG _____-admin $USER

# Activer le service
sudo systemctl _____ --now incus

# Recharger les groupes (sans redémarrer)
newgrp _____-admin
```

Étape 1.3 : Initialisation d'Incus

Q1.5 Lancez l'initialisation interactive d'Incus :

```
sudo incus admin _____
```

Paramètres recommandés (séquence réelle Incus 6.0.4) :

- Would you like to use clustering? → **no** (Entrée)
- Do you want to configure a new storage pool? → **yes** (Entrée)
- Name of the new storage pool [default=default]: → **default** (Entrée)
- Where should this storage pool store its data? → **(Entrée pour accepter le défaut)**
- Would you like to create a new local network bridge? → **yes** (Entrée)
- What should the new bridge be called? [default=incusbr0]: → **incusbr0** (Entrée)
- What IPv4 address should be used? → **192.168.x.1/24**
- Would you like to NAT IPv4 traffic on your bridge? → **yes** (Entrée)
- What IPv6 address should be used? → **none**
- Would you like the server to be available over the network? → **no** (Entrée)
- Would you like stale cached images to be updated automatically? → **yes** (Entrée)
- Would you like a YAML "init" preseed to be printed? → **no** (Entrée)

Q1.6 Vérifiez l'installation :

```
# Version d'Incus
incus _____

# Informations système
incus _____

# État du bridge créé
ip addr _____
```

Étape 1.4 : Configuration réseau avancée

Q1.7 Examinez la configuration réseau par défaut :

```
# Lister les réseaux Incus
incus network _____
```

```
# Détails du bridge
incus ____ show _____
```

Q1.8 IMPORTANT - Désactivation du DHCP intégré d'Incus :

```
# Désactiver le DHCP d'Incus pour éviter les conflits
incus ____ set incusbr0 ____=false
```

```
# Vérifier la modification (DHCP désactivé, NAT toujours actif)
incus network show incusbr0 | grep -E "(dhcp|nat)"
```

💡 Pourquoi cette étape ?

Attention : Ne pas désactiver le NAT lors de l'init.

Q1.8 Configuration du NAT entre les réseaux (approche iptables) :

💡 **Note pédagogique** : Nous utilisons iptables pour comprendre le mécanisme. En production, les bridges managés par Incus sont plus simples.

```
# Activer le forwarding IP
echo 1 | sudo tee /proc/____/net/ipv4/____

# Rendre permanent
echo 'net.ipv4.ip_forward=1' | sudo tee -a /etc/____.conf

# Règle NAT pour communication 10.x.x.0/24 → 192.168.x.0/24
sudo iptables -t nat -A POSTROUTING -s 10.x.x.0/24 -d 192.168.x.0/24 -j
_____

# Autoriser le forward entre les réseaux
sudo iptables -A ____ -s 10.x.x.0/24 -d 192.168.x.0/24 -j _____
sudo iptables -A ____ -s 192.168.x.0/24 -d 10.x.x.0/24 -j _____
```

Q1.9 Alternative avec bridge managé Incus (à titre informatif) :

```
# Cette méthode serait plus simple en production
incus ____ set ____ ipv4.routes="10.x.x.0/24"
```

Étape 1.5 : Création de la VM serveur DHCP

Q1.10 Listez les images disponibles :

```
# Images Ubuntu/Debian disponibles
incus image ____ images: | grep -E "(ubuntu|debian)"
# créer une VM (debian/12) et un CT (ubuntu/24). Puis familiarisez vous avec
les commandes Incus qui se trouvent dans le glossaire (shell, exec,
console, list, start, stop, delete, etc.).
```

Q1.11 Créez et configurez la VM DHCP :

```
# Création de la VM (complétez la commande)
incus launch ____:debian/12 dhcp-server --____ --config
limits.memory=____GiB --config limits.cpu=____

# Attendre le démarrage complet
incus list
```

Q1.12 Configuration de l'IP statique pour le serveur DHCP :

```
# Assigner l'IP 192.168.x.253
incus config device ____ dhcp-server eth0 ipv4. ____=192.168.x.____

# Redémarrer pour appliquer
incus ____ dhcp-server
```

Étape 1.6 : Installation et configuration du service DHCP

Q1.13 Installez le serveur DHCP dans la VM :

```
# Connexion à la VM
incus shell ____-server

# Dans la VM : mise à jour et installation
apt update
apt install -y isc-____-server
```

Q1.14 Configurez le service DHCP :

```
# Toujours dans la VM
cat > /etc/dhcp/dhcpd.conf << 'EOF'
authoritative;
default-lease-time 600;
max-lease-time 7200;

subnet 192.168.x.0 netmask 255.255.255.0 {
    range 192.168.x.100 192.168.x.200;
    option routers 192.168.x.1;
    option domain-name-servers 8.8.8.8, 8.8.4.4;
    option domain-name "incus.local";
}

# Réservations statiques
host web-server {
    hardware ethernet __:__:__:__:__:__; # À compléter après création
    fixed-address 192.168.x.254;
}

host file-server {
    hardware ethernet __:__:__:__:__:__; # À compléter après création
    fixed-address 192.168.x.251;
}
EOF
```

Q1.15 Démarrez le service DHCP :

```
# Activer et démarrer le service
systemctl ____ isc-dhcp-server
systemctl ____ isc-dhcp-server

# Vérifier le statut
systemctl ____ isc-dhcp-server

# Sortir de la VM
exit
```

Tests de la séance 1

Q1.16 Testez la connectivité réseau :

```
# Depuis l'hôte, pinguer la VM DHCP
ping -c 3 192.168.x.253

# Vérifier que la VM peut accéder à Internet
incus exec dhcp-server -- ____ -c 3 8.8.8.8
```

Q1.17 Créez un snapshot de sauvegarde :

```
# Snapshot de la VM configurée
incus snapshot ____ dhcp-server ____-config
```

Séance 2 : Serveurs Web et Fichiers

Étape 2.1 : Création de la VM serveur Web

Q2.1 Créez la VM serveur web :

```
# VM Ubuntu pour diversifier
incus ____ images:ubuntu/24.04 web-server --vm \
  --config limits.memory=____GiB \
  --config limits.cpu=____
```

Q2.2 Récupérez l'adresse MAC de la VM web :

```
# Afficher les détails de la VM
incus ____ show web-server | grep -A5 eth0

# Noter l'adresse MAC pour la réservation DHCP
```

Q2.3 Mettez à jour la configuration DHCP :

```
# Modifier la configuration DHCP avec la bonne MAC
incus ____ dhcp-server -- nano /etc/____/dhcpd.conf

# Redémarrer le service DHCP
incus exec dhcp-server -- ____ restart isc-dhcp-server
```

Étape 2.2 : Configuration du serveur Web

Q2.4 Installez Apache dans la VM web :

```
# Connexion à la VM
incus _____ web-server

# Installation d'Apache et PHP
apt update
apt install -y apache2 _____

# Activer les modules nécessaires
a2enmod rewrite
systemctl ____ apache2
```

Q2.5 Créez un site web de test :

```
# Site personnalisé
cat > /var/www/html/index.php << 'EOF'
<!DOCTYPE html>
<html>
<head>
  <title>Serveur Web Incus</title>
  <style>
    body { font-family: Arial; margin: 40px; background: #f0f0f0; }
    .container { background: white; padding: 20px; border-radius: 8px;
  }
    .info { background: #e7f3ff; padding: 10px; margin: 10px 0; }
  </style>
</head>
<body>
  <div class="container">
    <h1>🌐 Serveur Web Incus - BUT RT</h1>
    <div class="info">
      <strong>Serveur:</strong> <?php echo gethostname(); ?><br>
      <strong>IP:</strong> <?php echo $_SERVER['SERVER_ADDR']; ?><br>
      <strong>Date:</strong> <?php echo date('Y-m-d H:i:s'); ?><br>
      <strong>OS:</strong> <?php echo php_uname('s r'); ?>
    </div>

    <h2>🇫🇷 Informations Système</h2>
    <pre><?php echo shell_exec('free -h'); ?></pre>

    <h2>🔌 Connectivité</h2>
    <pre><?php echo shell_exec('ip addr show eth0'); ?></pre>
  </div>
</body>
</html>
EOF

# Permissions correctes
chown www-data:www-data /var/www/html/index.php

# Sortir de la VM
exit
```

Q2.6 Configurez l'accès externe au serveur web :

```
# Proxy device pour accès depuis l'hôte
incus config ____ add web-server http-proxy proxy \
  listen=tcp:0.0.0.0:____ \
```

```
connect=tcp:127.0.0.1:_____
```

Étape 2.3 : Création du conteneur serveur de fichiers

Q2.7 Créez un conteneur (plus léger qu'une VM) :

```
# Conteneur Alpine Linux (très léger)
incus launch ____:alpine/_____ file-server \
  --config limits.memory=_____MiB \
  --config limits.cpu=_____
```

Q2.8 Récupérez la MAC du conteneur et mettez à jour DHCP :

```
# MAC du conteneur
incus config ____ file-server | grep -A5 eth0

# Mise à jour configuration DHCP (même processus que Q2.3)
```

Étape 2.4 : Configuration du serveur de fichiers

Q2.9 Installez et configurez Samba :

```
# Connexion au conteneur
incus shell _____-server

# Installation Samba (Alpine utilise apk)
apk update
apk add samba samba-common-tools

# Créer les répertoires partagés
mkdir -p /srv/samba/{public,secure}
```

Q2.10 Configuration Samba :

```
# Configuration principale
cat > /etc/samba/smb.conf << 'EOF'
[global]
    workgroup = WORKGROUP
    server string = Incus File Server
    security = user
    map to guest = bad password
    dns proxy = no

[public]
    comment = Partage Public
    path = /srv/samba/public
    browsable = yes
    writable = yes
    guest ok = yes
    public = yes
    create mask = 0644
    directory mask = 0755

[secure]
    comment = Partage Sécurisé
    path = /srv/samba/secure
    valid users = user1
```

```
browsable = yes
writable = yes
create mask = 0600
directory mask = 0700
EOF
```

Q2.11 Configuration des utilisateurs et services :

```
# Créer un utilisateur système
adduser user1

# Ajouter à Samba (même mot de passe)
____ -a user1

# Permissions des répertoires
chown -R user1:user1 /srv/samba/secure
____ 755 /srv/samba/public

# Démarrer les services
rc-update add samba
service samba start

# Vérifier
service samba status

# Sortir du conteneur
exit
```

Étape 2.5 : Configuration de l'accès aux fichiers

Q2.12 Configurez l'accès SMB depuis l'hôte :

```
# Proxy pour SMB (port 445)
incus ____ device add file-server smb-proxy proxy \
  listen=tcp:0.0.0.0:____ \
  connect=tcp:127.0.0.1:____
```

Q2.13 Alternative : Serveur FTP (plus simple) :

```
# Si Samba pose problème, utiliser FTP
incus exec file-server -- apk add vsftpd

# Configuration FTP basique
incus exec file-server -- cat > /etc/vsftpd/vsftpd.conf << 'EOF'
anonymous_enable=YES
local_enable=YES
write_enable=YES
anon_upload_enable=YES
anon_mkdir_write_enable=YES
anon_root=/srv/samba/public
EOF

# Démarrer FTP
incus exec file-server -- service vsftpd start
```



Tests de la séance 2

Q2.14 Tests de connectivité globale :

```
# Test 1 : Accès web depuis l'hôte
curl http://localhost: ____

# Test 2 : Connexion entre VMs
incus exec web-server -- ____ -c 3 192.168.x.253
incus exec web-server -- ____ -c 3 192.168.x.251

# Test 3 : Résolution DNS via DHCP
incus exec web-server -- ____ google.com
```

Q2.15 Créez des snapshots de toutes les instances :

```
# Snapshots de sauvegarde
incus ____ create web-server ____-complete
incus ____ create file-server ____-complete
```



Séance 3 : Tests, Optimisation et Troubleshooting

Étape 3.1 : Tests d'infrastructure globale

Q3.1 Testez l'architecture complète depuis l'hôte :

```
# Test 1 : Accès web avec informations système
curl -s http://localhost: ____ | grep -____ "(Serveur|IP|Date) "

# Test 2 : Test de charge basique
for i in {1..10}; do curl -s http://localhost:8080 > /dev/null && echo "Req
$i: OK"; done
```

Q3.2 Testez l'accès aux fichiers :

```
# Installation client SMB sur l'hôte
sudo apt install cifs-utils ____

# Lister les partages disponibles
smbclient -L localhost -p 4445 -N

# Montage du partage public
sudo mkdir -p /____/incus-public
sudo mount -t cifs //localhost/public /____/incus-public -o
port=4445,guest,uid=$USER
```

Étape 3.2 : Monitoring et métriques

Q3.3 Surveillez les performances :

```
# Ressources utilisées par les instances
incus info dhcp-server
incus info web-server
incus info file-server

# Monitoring en temps réel
```

```
incus ____ --pretty
```

Q3.4 Analysez l'utilisation réseau :

```
# Statistiques du bridge
ip -s ____ show incusbr0
```

```
# Connexions actives
sudo netstat -____ | grep -E "(8080|4445|67)"
```

Étape 3.3 : Optimisation et scripts d'automatisation

Q3.5 Créez un script de déploiement automatique :

```
cat > ~/deploy-incus-infra.sh << 'EOF'
#!/bin/bash

echo "🚀 Déploiement automatique infrastructure Incus"

# Fonction de vérification
check_instance() {
    local name=$1
    local expected_ip=$2

    echo -n "Vérification $name ($expected_ip)... "
    if incus ____ | grep -q "$name.*RUNNING"; then
        if incus exec $name -- ping -c 1 -W 2 8.8.8.8 > /dev/null 2>&1;
    then
        echo "✅ OK"
        return 0
    fi
    fi
    echo "❌ ERREUR"
    return 1
}

# Tests de l'infrastructure
echo "🔍 Tests de l'infrastructure..."
check_instance "dhcp-server" "192.168.x.253"
check_instance "web-server" "192.168.x.254"
check_instance "file-server" "192.168.x.251"

# Test des services
echo "🌐 Test du serveur web..."
if curl -s http://localhost:8080 > /dev/null; then
    echo "✅ Serveur web accessible"
else
    echo "❌ Serveur web inaccessible"
fi

echo "📁 Test du serveur de fichiers..."
if smbclient -L localhost -p 4445 -N > /dev/null 2>&1; then
    echo "✅ Serveur de fichiers accessible"
else
    echo "❌ Serveur de fichiers inaccessible"
fi
```

```
echo "🌟 Tests terminés"
EOF

chmod +x ~/deploy-incus-infra.sh
```

Étape 3.4 : Troubleshooting courant

Q3.6 Diagnostiquez les problèmes réseau courants :

```
# Vérifier les routes
ip ____ show

# Vérifier iptables
sudo iptables -L -n -v

# Logs des services
incus ____ dhcp-server --show-log
incus ____ dhcp-server -- systemctl status isc-dhcp-server
```

Q3.7 Script de diagnostic automatique :

```
cat > ~/diagnostic-incus.sh << 'EOF'
#!/bin/bash

echo "🔧 DIAGNOSTIC INCUS - Infrastructure"
echo "===== "

echo "🇫🇷 État général:"
echo "- Version Incus: $(incus version)"
echo "- Service actif: $(systemctl is-active incus)"
echo "- Bridge configuré: $(ip addr show incusbr0 | grep -q inet && echo "Oui" || echo "Non")"

echo -e "\n🖥️ Instances:"
incus list --format table

echo -e "\n🌐 Réseau:"
echo "- Forwarding ____ : $(cat /proc/sys/net/ipv4/ip_forward)"
echo "- Bridge incusbr0:"
ip addr show incusbr0 | grep inet

echo -e "\n🔍 Tests de connectivité:"
for instance in dhcp-server web-server file-server; do
    if incus list | grep -q "$instance.*RUNNING"; then
        echo -n "- $instance: "
        if incus exec $instance -- ping -c 1 -W 2 8.8.8.8 > /dev/null 2>&1;
        then
            echo "✅ Connecté"
        else
            echo "❌ Non connecté"
        fi
    fi
done

echo -e "\n🚀 Services:"
echo -n "- Web (port 8080): "
curl -s http://localhost:8080 > /dev/null && echo "✅ OK" || echo "❌ KO"
```

```
echo -n "- Fichiers (port 4445): "
smbclient -L localhost -p 4445 -N > /dev/null 2>&1 && echo "✅ OK" || echo
"❌ KO"
EOF

chmod +x ~/diagnostic-incus.sh
./diagnostic-incus.sh
```

Étape 3.5 : Sauvegarde et restauration

Q3.8 Exportez votre infrastructure :

```
# Export des instances en images
incus ____ dhcp-server --alias dhcp-server-final
incus ____ web-server --alias web-server-final
incus ____ file-server --alias file-server-final

# Lister les images créées
incus image list local:
```

Q3.9 Test de restauration :

```
# Simuler un problème
incus stop web-server
incus delete web-server

# Restauration depuis snapshot
incus ____ dhcp-server backup-complete

# Ou création depuis image
incus launch ____:web-server-final web-server-restored --vm
```

🎯 Tests finaux et validation

Q3.10 Validation complète de l'infrastructure :

```
# Exécutez vos scripts de test
~/diagnostic-incus.sh
~/deploy-incus-infra.sh

# Test de charge sur le serveur web
ab -n 100 -c 10 http://localhost:8080/

# Test transfert de fichiers
echo "Test de transfert" > /tmp/test.txt
smbclient //localhost/public -p 4445 -N -c "put /tmp/test.txt"
```

Questions de réflexion et rapport

Analyse comparative

Q4.1 Comparez les technologies utilisées :

Aspect	Machine Virtuelle	Conteneur	Justification
Performance			
Isolation			
Démarrage			
Ressources			
Cas d'usage			

Q4.2 Analysez les choix réseau :

- Pourquoi utiliser un bridge plutôt que du bridge host ?
- Avantages/inconvénients d'iptables vs bridge managé ?
- Impact du NAT sur les performances ?

Optimisations possibles

Q4.3 Proposez 3 améliorations de l'architecture :

1. _____
2. _____
3. _____

Q4.4 Sécurisation de l'infrastructure :

- Quelles mesures de sécurité pourriez-vous ajouter ?
- Comment monitorer les accès aux services ?
- Stratégie de sauvegarde en production ?

Questions avancées

Q4.5 Dimensionnement :

- Comment cette architecture se comporterait avec 50 clients ?
- Ressources nécessaires pour un environnement de production ?

Q4.6 Haute disponibilité :

- Comment assurer la continuité de service ?
- Quelles sont les SPOF (Single Point of Failure) ?



Compétences acquises

- ✓ **AC11.03** - Configurer les fonctions de base du réseau local
- ✓ **AC11.04** - Maîtriser les systèmes d'exploitation pour l'administration
- ✓ **AC11.05** - Identifier les dysfonctionnements et signaler
- ✓ **AC13.01** - Utiliser un système informatique et ses outils

Technologies maîtrisées

- Incus (conteneurs et VMs)
- Configuration réseau avancée
- Services DHCP, Web, Fichiers
- Scripts d'automatisation
- Monitoring et diagnostic



Ressources et documentation

- [Documentation officielle Incus](#)
 - [Guide réseau Incus](#)
 - [Comparaison LXD vs Incus](#)
-