NTFY (Self-Hosted)

Le Concept (La Théorie)

Dans un monde dominé par les GAFAM, recevoir une notification sur son téléphone implique généralement de passer par les serveurs de Google (FCM) ou Apple (APNs). Cela signifie : métadonnées partagées, dépendance, et quotas.

Ntfy (prononcer "Notify") est une solution Pub-Sub (Publication-Subscription) HTTP.

Le site : https://ntfy.sh/

- **Le Principe :** Vous envoyez une requête HTTP (PUT/POST) à une URL. Ntfy la transforme en notification sur votre téléphone.
- **Pourquoi l'héberger ?**
    1. **Confidentialité Totale :** Les messages ne quittent jamais votre serveur VPS jusqu'à votre téléphone.
    2. **Zéro Quota :** Vous n'êtes pas limité à 5 messages par jour.
    3. **Temps Réel :** Utilisation des WebSockets pour une livraison instantanée sans tuer la batterie.
    4. **UnifiedPush :** Peut servir de "tuyau" pour d'autres applis (comme Element/Matrix) à l'avenir.

Installation Serveur (Le Bunker)

Nous déployons Ntfy derrière un Reverse Proxy (Traefik) pour gérer le SSL. La configuration est durcie par défaut : personne ne lit, personne n'écrit sans badge.

```
/opt/docker/ntfy/
├── docker-compose.yml
├── cache/            # Persistance des messages (SQLite)
├── etc/              # Configuration users
└── logs/             # Sortie logs (pour Fail2Ban)
```

```
admin@lab-debian-64:~$ sudo mkdir -p /opt/docker/ntfy
admin@lab-debian-64:~$ cd /opt/docker/ntfy/
admin@lab-debian-64:/opt/docker/ntfy$ sudo nano docker-compose.yml
```

```
services:
  ntfy:
    image: binwiederhier/ntfy
    container_name: ntfy
    command:
      - serve
      - --log-level=INFO
      - --log-format=text
    environment:
      - NTFY_BASE_URL=https://ntfy.domaine.ext
      - NTFY_CACHE_FILE=/var/cache/ntfy/cache.db
      - NTFY_AUTH_DEFAULT_ACCESS=deny-all          <-- VERROUILLAGE PAR DÉFAUT
      - NTFY_AUTH_FILE=/etc/ntfy/user.db
      - NTFY_BEHIND_PROXY=true
    volumes:
      - ./cache:/var/cache/ntfy
      - ./etc:/etc/ntfy
      - ./logs:/var/log/ntfy
    networks:
      - proxy
    labels:
      - "traefik.enable=true"
      - "traefik.http.routers.ntfy.rule=Host(`ntfy.domaine.ext`)"
      - "traefik.http.routers.ntfy.entrypoints=websecure"
      - "traefik.http.routers.ntfy.tls.certresolver=myresolver"
      - "traefik.http.services.ntfy.loadbalancer.server.port=80"
      - "traefik.docker.network=proxy"

networks:
  proxy:
    external: true
```

```
admin@lab-debian-64:/opt/docker/ntfy$ docker compose up -d
[+] up 2/4
[+] up 4/4inwiederhier/ntfy [⣿⣿⣿] 32.06MB / 32.06MB Pulling
3.1s
 ✔ Image binwiederhier/ntfy Pulled
3.1s
   ✔ 2d35ebdb57d9           Pull complete
0.8s
   ✔ 656054084276           Pull complete
1.0s
   ✔ 8d5d9816f615           Pull complete
1.5s
 ✔ Container ntfy Created
```

https://ntfy.domaine.ext

Sécurisation & Gestion des Accès

Par défaut, l'accès est deny-all. Nous n'utilisons pas d'utilisateur générique admin (trop évident), mais un utilisateur dédié au service (ex: warder).

Création de la Warder

Dans le terminal du serveur :

Créer l'utilisateur (Mot de passe fort requis)

```
admin@lab-debian-64:~$ docker exec -it ntfy ntfy user add sentinel
```

Lui donner les droits d'écriture/lecture sur TOUS les topics (*)

```
admin@lab-debian-64:~$ docker exec -it ntfy ntfy access sentinel '*' read-write
```

Vérifier les utilisateurs

```
admin@lab-debian-64:~$ docker exec -it ntfy ntfy user list
```

Génération de Token (Pour les Scripts)

Pour éviter de stocker le mot de passe dans les scripts Bash (fail2ban-ntfy.sh), nous générons un jeton qui n'expire pas.

```
admin@lab-debian-64:~$ docker exec -it ntfy ntfy token add --label "Script Token" sentinel
```

Ce token est le Sésame à utiliser dans les en-têtes Authorization.

Configuration Client (Android)

L'application officielle **Ntfy** (disponible sur F-Droid/Play Store) permet de recevoir les alertes.

Paramétrage Initial

1. **Server :** Aller dans Settings > General > Default server.
    - Entrer : https://ntfy.unsafe.fr
2. **Authentification :** Le serveur va rejeter la connexion (401).
    - Aller dans Settings > Manage users.
    - Ajouter l'utilisateur sentinel + Mot de passe.
3. **Optimisation Batterie (Critique) :**
    - Aller dans Settings > Advanced > Connection protocol.
    - Sélectionner **WebSockets**.
    - *Note : Cela évite de drainer la batterie avec une connexion HTTP ouverte en permanence.*

S'abonner à un Topic

- Cliquer sur +
- Nom du Topic : Alertes_Server_X (Respecter la casse ! A ≠ a)
- S'abonner. Si le cercle est vert, le tuyau est ouvert.

Usage (L'Envoi d'Alerte)

Ntfy s'utilise avec une simple commande curl. C'est l'outil parfait pour les scripts de monitoring (Fail2Ban, Backups, Uptime).

Syntaxe Standard

```
admin@lab-debian-64:~$ curl -u sentinel:PASSWORD -d "Message simple" https://ntfy.domaine.ext/MonTopic
```

Syntaxe Avancée (Avec Token & Mise en forme)

Utilisée dans nos scripts de production.

```
# Variables
NTFY_SERVER="https://ntfy.domaine.ext"
NTFY_TOPIC="MonTopic"
NTFY_TOKEN="tk_votre_token_recupere"

curl -s \
     -H "Authorization: Bearer $NTFY_TOKEN" \
     -H "Title: 🚨 ALERTE SECURITE" \
     -H "Priority: 5" \
     -H "Tags: warning,skull" \
     -d "Intrusion détectée sur le serveur. IP bannie." \
     "$NTFY_SERVER/$NTFY_TOPIC"
```

FAIL2BAN x NTFY

Type : Monitoring / Automatisation

Objectif : Alerter en temps réel lors d'un bannissement IP.

Dépendances : curl, fail2ban

Le Script de Notification (Le Cerveau)

Ce script est le pont entre Fail2Ban et Ntfy. Il récupère les paramètres de l'attaque, formate un message lisible (avec date et durée) et l'envoie via une requête sécurisée par Token.

Fichier : /usr/local/bin/fail2ban-ntfy.sh

Permissions : chmod +x (755)

```
#!/bin/bash
# --------------------------------------------------------------------------------
# SCRIPT D'ALERTE FAIL2BAN -> NTFY
# Description : Envoie une notif push via Ntfy lors d'un ban.
# --------------------------------------------------------------------------------

# --- 1. CONFIGURATION ---
NTFY_SERVER="https://ntfy.domaine.ext"
NTFY_TOPIC="Alertes_hostname_fail2ban_00"  # Attention à la Casse (A != a) !
# Token généré via : docker exec -it ntfy ntfy token add --label "Script" sentinel
NTFY_TOKEN="tk_xxxxxxxxxxxxxxxxxxxxxxxxxxx"

# --- 2. RÉCUPÉRATION DES ARGUMENTS FAIL2BAN ---
# $1 = Nom de la Jail (ex: nginx-html)
# $2 = IP Bannie
# $3 = Nombre de tentatives (Failures)
# $4 = Durée du bannissement (en secondes)
JAIL_NAME="${1:-Unknown Jail}"
BANNED_IP="${2:-Unknown IP}"
FAIL_COUNT="${3:-N/A}"
BAN_DURATION_SEC="${4:-0}"
HOSTNAME=$(hostname)

# --- 3. TRAITEMENT DES DONNÉES ---
# Date actuelle formatée (Ex: 19/01 16:45)
DATE_BAN=$(date '+%d/%m %H:%M')

# Conversion Durée (Secondes -> Format lisible)
if [ "$BAN_DURATION_SEC" -ge 3600 ]; then
    # Si > 1h, conversion en Heures
    BAN_PRETTY="$((BAN_DURATION_SEC / 3600)) h"
elif [ "$BAN_DURATION_SEC" -ge 60 ]; then
    # Si > 1min, conversion en Minutes
    BAN_PRETTY="$((BAN_DURATION_SEC / 60)) min"
else
    # Sinon secondes brutes
    BAN_PRETTY="$BAN_DURATION_SEC sec"
fi

# --- 4. ENVOI DE LA NOTIFICATION ---
# Priority 5 = Max (Sonne même en silencieux si configuré sur le tel)
# Tags : skull (banni), clock (durée), calendar (date)
curl -s -o /dev/null \
     -H "Authorization: Bearer $NTFY_TOKEN" \
     -H "Title: 🚨 ALERTE POLLUX [$JAIL_NAME]" \
     -H "Priority: 5" \
     -H "Tags: warning,skull,clock" \
     -d "Cible : $BANNED_IP
Durée : $BAN_PRETTY
Date  : $DATE_BAN
Fail  : $FAIL_COUNT tentatives" \
     "$NTFY_SERVER/$NTFY_TOPIC"
```

L'Action Fail2Ban (Le Déclencheur)

Fail2Ban a besoin d'un fichier de définition pour savoir comment utiliser le script ci-dessus. Nous créons une "Action" personnalisée.

Fichier : /etc/fail2ban/action.d/ntfy.conf

```
[Definition]
# Description :
# Appelle le script fail2ban-ntfy.sh avec les arguments :
# <name> (Jail), <ip> (Cible), <failures> (Tentatives), <bantime> (Durée)

actionban = /usr/local/bin/fail2ban-ntfy.sh "<name>" "<ip>" "<failures>" "<bantime>"

# Optionnel : Notification au débannissement (On envoie juste "DEBAN" en durée)
actionunban = /usr/local/bin/fail2ban-ntfy.sh "<name>" "<ip>" "DEBAN" "0"

[Init]
# Rien de spécifique
```

L'Activation dans la Jail (L'Ordre)

Enfin, nous devons dire à nos prisons (Jails) d'utiliser cette action.

Fichier : /etc/fail2ban/jail.d/docker-nginx.conf (ou votre fichier de jail habituel)

```
[nginx-nosec]
enabled = true
filter = nginx-nosec
logpath = /opt/docker/www/logs/access.log

# --- ACTIONS ---
# On combine l'action Docker (pare-feu) ET l'action Ntfy (notification)
action = docker-action[name=nginx-nosec]
         ntfy[name=nginx-nosec]

# --- REGLAGES BAN ---
maxretry = 3
findtime = 600
bantime = 1h
bantime.increment = true
```

Validation & Test

Une fois les fichiers en place, il est impératif de recharger la configuration et de tester la chaîne complète.

Recharger Fail2Ban

```
admin@lab-debian-64:~$ sudo systemctl reload fail2ban
```

Vérifier les erreurs de config

```
admin@lab-debian-64:~$ sudo fail2ban-client -d | grep ntfy
# Doit retourner la configuration chargée sans erreur
```

Tir de Test Manuel (Simulation Totale)

Pour vérifier que le script fonctionne AVEC l'utilisateur sentinel et le bon Token :

```
admin@lab-debian-64:~$ /usr/local/bin/fail2ban-ntfy.sh "TEST_INTEGRATION" "8.8.8.8" "99" "3600"
```

Si le téléphone vibre, l'intégration est validée.

Test ultime

Tape cette commande pour bannir une fausse IP (1.2.3.4) dans la prison

```
admin@lab-debian-64:~$ sudo fail2ban-client set nginx-nosec banip 1.2.3.4
```

Une fois que tu as confirmé le "Bip", tu peux débannir ce pauvre fantôme

```
admin@lab-debian-64:~$ sudo fail2ban-client set nginx-nosec unbanip 1.2.3.4
```