

Fail2ban

En bref

Niveau : avancé

Durée : 45 minutes

Prérequis : Traefik et au moins une application en place

Niveau de test : reconstitué

Fail2ban, c'est quoi ?

Fail2ban lit des fichiers de journaux, repère les adresses qui multiplient les échecs (mots de passe erronés, pages interdites) et les bloque quelques instants dans le pare-feu. Il ne remplace pas les mots de passe solides : il réduit le bruit des attaques automatiques et ralentit les tentatives de devinette.

Élément	Rôle
Filtre (<code>filter.d</code>)	Expression régulière qui reconnaît un échec dans une ligne de journal
Jail (<code>jail.d</code>)	Associe un filtre, un fichier de journal, un seuil et une durée de blocage
Action de blocage	Ajoute une règle dans le pare-feu pour l'adresse fautive
Chaîne <code>DOCKER-USER</code>	Point où les règles s'appliquent aux conteneurs

Visiteur fautif → journal Traefik ou Vaultwarden → filtre → jail → règle dans `DOCKER-USER` → adresse bloquée

Concrètement, le rôle installe Fail2ban sur la VM et y dépose deux filtres et deux jails : un pour les erreurs d'authentification vues par Traefik, un pour les échecs de connexion à Vaultwarden. Les jails portent le préfixe `nebulouse-`.

Les principes de Fail2ban (filtres, jails, commandes de base) sont détaillés dans le livre **Fail2Ban** de la base de connaissance. Cette page ne traite que l'intégration au socle Nébuleuse.

Prérequis

1. Le socle est en place : Docker a créé la chaîne `DOCKER-USER`.
2. Traefik écrit son journal d'accès en JSON dans `/var/log/traefik/access.log` (page Traefik).
3. Pour la jail Vaultwarden : le journal `/opt/nebuleuse/vaultwarden/data/vaultwarden.log` existe (page Vaultwarden).

Pourquoi `DOCKER-USER` ?

Docker publie les ports des conteneurs en les redirigeant avant le pare-feu de la machine. Une règle de blocage placée dans la chaîne habituelle (`INPUT`) ne verrait jamais le trafic destiné à Traefik. La chaîne `DOCKER-USER`, elle, est traversée avant les règles de Docker, et Docker n'y touche jamais : c'est le bon endroit pour bloquer une adresse.

Arborescence du rôle

```
roles/fail2ban/  
├─ defaults/main.yml  
├─ handlers/main.yml  
├─ tasks/main.yml  
└─ templates/  
    ├─ filter-nebuleuse-traefik.conf.j2  
    ├─ filter-nebuleuse-vaultwarden.conf.j2  
    └─ jail-nebuleuse.conf.j2
```

Variables

```
# roles/fail2ban/defaults/main.yml  
  
# Adresses jamais bloquées : la VM elle-même et le réseau d'administration.  
# Ajouter ici l'adresse fixe du client si elle est connue.  
fail2ban_ignore_ips:  
  - 127.0.0.1/8  
  - "{{ admin_network_cidr }}"
```

```
# Jail Traefik : trop d'erreurs 401 ou 403 en peu de temps.
```

```
fail2ban_traefik_maxretry: 30
```

```
fail2ban_traefik_findtime: 10m
```

```
fail2ban_traefik_bantime: 1h
```

```
# Jail Vaultwarden : échecs de connexion.
```

```
fail2ban_vaultwarden_maxretry: 5
```

```
fail2ban_vaultwarden_findtime: 10m
```

```
fail2ban_vaultwarden_bantime: 1h
```

```
# Chaîne du pare-feu où les blocages sont ajoutés.
```

```
fail2ban_chain: DOCKER-USER
```

Les filtres

Le journal d'accès de Traefik contient une ligne JSON par requête. Les champs sont triés par ordre alphabétique : `ClientHost` (l'adresse du visiteur) précède donc toujours `DownstreamStatus` (le code de réponse).

```
# roles/fail2ban/templates/filter-nebuleuse-traefik.conf.j2
```

```
# Fichier géré par Ansible.
```

```
[Definition]
```

```
# Une ligne est un échec quand la réponse est 401 (non authentifié) ou 403 (interdit).
```

```
failregex = ^.*"ClientHost": "<HOST>".*"DownstreamStatus":(?:401|403).*$
```

```
# Format de la date dans le journal JSON.
```

```
datepattern = "time": "%Y-%m-%dT%H:%M:%S
```

```
ignoreregex =
```

```
# roles/fail2ban/templates/filter-nebuleuse-vaultwarden.conf.j2
```

```
# Fichier géré par Ansible.
```

```
[Definition]
```

```
# Mot de passe erroné, ou jeton d'administration erroné. <ADDR> désigne l'adresse du visiteur.
```

```
failregex = ^.*Username or password is incorrect\. Try again\. IP: <ADDR>\. Username:.*$
```

```
^.*Invalid admin token\. IP: <ADDR>.*$
```

```
ignoreregex =
```

Les jails

```
# roles/fail2ban/templates/jail-nebuleuse.conf.j2
# Fichier g  r   par Ansible.

[DEFAULT]
ignoreip = {{ fail2ban_ignore_ips | join(' ') }}
# Blocage par iptables, dans la cha  ne travers  e par le trafic des conteneurs.
banaction = iptables-multiport
chain = {{ fail2ban_chain }}
port = http,https

[nebuleuse-traefik]
enabled = true
filter = nebuleuse-traefik
logpath = /var/log/traefik/access.log
maxretry = {{ fail2ban_traefik_maxretry }}
findtime = {{ fail2ban_traefik_findtime }}
bantime = {{ fail2ban_traefik_bantime }}
{% if 'vaultwarden' in nebuleuse_apps %}

[nebuleuse-vaultwarden]
enabled = true
filter = nebuleuse-vaultwarden
logpath = /opt/nebuleuse/vaultwarden/data/vaultwarden.log
maxretry = {{ fail2ban_vaultwarden_maxretry }}
findtime = {{ fail2ban_vaultwarden_findtime }}
bantime = {{ fail2ban_vaultwarden_bantime }}
{% endif %}
```

T  ches

```
# roles/fail2ban/tasks/main.yml

- name: Installer Fail2ban et iptables
  ansible.builtin.apt:
    name:
```

```

- fail2ban
- iptables          # absent d'une Debian 12 minimale
state: present

- name: Déposer les filtres
  ansible.builtin.template:
    src: "filter-{{ item }}.conf.j2"
    dest: "/etc/fail2ban/filter.d/{{ item }}.conf"
    mode: "0644"
  loop:
    - nebuleuse-traefik
    - nebuleuse-vaultwarden
  notify: Recharger fail2ban

- name: Déposer les jails
  ansible.builtin.template:
    src: jail-nebuleuse.conf.j2
    dest: /etc/fail2ban/jail.d/nebuleuse.conf
    mode: "0644"
  notify: Recharger fail2ban

- name: Activer et démarrer Fail2ban
  ansible.builtin.service:
    name: fail2ban
    state: started
    enabled: true

```

```

# roles/fail2ban/handlers/main.yml

- name: Recharger fail2ban
  ansible.builtin.service:
    name: fail2ban
    state: restarted

```

Vérification

```

# Contrôle que le filtre reconnaît bien des lignes du journal (aucune erreur attendue).
sudo fail2ban-regex /var/log/traefik/access.log /etc/fail2ban/filter.d/nebuleuse-traefik.conf

```

```
# Les jails sont actives.
sudo fail2ban-client status

# Détail d'une jail : compteurs d'échecs et adresses bloquées.
sudo fail2ban-client status nebuleuse-traefik

# La chaîne DOCKER-USER contient les règles de Fail2ban.
sudo iptables -L DOCKER-USER -n
```

Pour provoquer un blocage, depuis une machine qui n'est pas dans `ignoreip`, envoyer plus de requêtes refusées que le seuil :

```
# 35 requêtes avec un mauvais mot de passe : la réponse est 401 à chaque fois.
for i in $(seq 35); do
    curl -s -o /dev/null -u test:test https://cloud.penhoat.example/remote.php/dav
done

# L'adresse de la machine de test doit apparaître dans "Banned IP list".
sudo fail2ban-client status nebuleuse-traefik

# Débloquer l'adresse après le test.
sudo fail2ban-client set nebuleuse-traefik unbanip <adresse>
```

Pièges

1. **Fichier de journal absent** : une jail dont le fichier n'existe pas ne démarre pas. D'où la place de Fail2ban en dernier dans `site.yml`.
2. **Blocage sans effet** : une action qui insère les règles dans `INPUT` ne voit pas le trafic des conteneurs. L'action doit viser la chaîne `DOCKER-USER`.
3. **Adresse de Traefik bloquée à la place du visiteur** : si l'adresse lue dans le journal est celle du proxy et non celle du visiteur, toute la VM est coupée. Le champ `ClientHost` doit contenir l'adresse réelle ; l'adresse du réseau d'administration est exclue par `ignoreip`.
4. **Clients de synchronisation bloqués** : une application de synchronisation (Nextcloud, par exemple) provoque normalement quelques réponses 401. Le seuil de 30 en dix minutes laisse de la marge. Un seuil trop bas bloque un utilisateur légitime.
5. **Blocages perdus après un redémarrage de Docker** : Docker recrée la chaîne `DOCKER-USER` et efface les règles. Fail2ban doit être redémarré après Docker pour réinstaller les blocages en cours.
6. **Expression régulière qui ne reconnaît rien** : `fail2ban-regex` est l'outil de contrôle. Si la date n'est pas reconnue, la ligne est ignorée, même si l'expression est bonne.

7. **Se bloquer soi-même** : en cas de test depuis le poste d'administration, l'adresse peut être bloquée à son tour. `fail2ban-client set <jail> unbanip <adresse>` la libère.

Page suivante : **Déploiement d'un client.**

Revision #1

Created 2026-10-17 13:54:44 UTC by clement-derouet

Updated 2026-10-17 13:54:44 UTC by clement-derouet