Documentation de déploiement - GLIMPS OneDrive SharePoint Connector

Vue d'ensemble

Le GLIMPS OneDrive SharePoint Connector est une solution qui surveille vos drives OneDrive/SharePoint pour détecter et traiter les malwares via GLIMPS Malware Detect. Cette documentation détaille le déploiement via Docker Compose.

Prérequis

Environnement technique

- Docker Engine version > 28.0.0
- Docker Compose plugin version > v2.33.0

Documentation d'installation :

- Docker Engine
- Docker Compose

Prérequis Microsoft 365

Informations obtenues lors des étapes précédentes de déploiement :

- Nom du tenant Microsoft
- ID du tenant Microsoft
- Secret du tenant Microsoft

GLIMPS Malware Detect

Informations fournies par l'équipe support GLIMPS :

- URL pour soumettre à l'API GLIMPS Malware Detect/Expert
- Token de connexion à GLIMPS Malware Detect (avec permission "search")

DNS (pour mode Real-time)

Le système de notification nécessite l'envoi d'emails par Microsoft. Si la plateforme n'est pas gérée par GLIMPS, il est nécessaire d'avoir un enregistrement MX pointant vers l'adresse IP publique.

La plateforme nécessite un enregistrement de domaine pour les notifications webhooks HTTPS.

Configuration préalable

1. Enregistrement de l'application Azure

Pour cette étape critique, consultez la documentation dédiée :

📋 **Documentation : Enregistrement de l'application "Microsoft ODSP Connector GLIMPS"**

Cette documentation détaille :

- La création de l'application Azure AD
- La configuration des permissions Microsoft Graph requises
- La génération et gestion des secrets client
- Les bonnes pratiques de sécurité

2. Création du site de quarantaine

Créez un site SharePoint dédié à la quarantaine :

```
quarantine:
  url: https://votre-sharepoint.com/sites/Quarantine
  lib-name: Quarantine-lib  # optionnel
```

Structure du déploiement

```
glimps-sharepoint-connector/
├── docker-compose.yml              # Fichier de base (ne pas modifier)
├── docker-compose.override.yml     # Configuration spécifique (copie
d'un des override)
├── config/
│   ├── letsencrypts/
│   │   └── acme.json               # Certificats Let's Encrypt
│   ├── m365/                       # Configuration M365
│   ├── sharepoint/
│   │   └── sharepoint.yml          # Configuration du connector
│   └── traefik/                    # Configuration Traefik (mode Real-time)
│       ├── providers.yml
│       └── traefik.yml
└── .env                            # Variables d'environnement (optionnel)
```

Configuration

Fichier docker-compose.yml (de base)

⚠️ **Important :** Ce fichier de base ne doit pas être modifié. Il fournit une configuration commune pour tous les services. Il peut être mis à jour lors de futures releases, les configurations personnalisées doivent être définies dans les fichiers override.

```
services:
  onedrive-connector:
    image: glimpsre/onedrive-sharepoint-connector:feat-v1
    volumes:
      - ./config/sharepoint:/etc/glimps_connector:ro
      - ./config/m365:/etc/m365
    restart: on-failure:3
    healthcheck:
      test: ["CMD", "wget", "-qO", "/dev/null", "http://localhost/healthz"]
```

```
    interval: 3s
    timeout: 3s
    start_period: 5s
    retries: 3
```

Fichiers Override

Les fichiers override permettent de fournir des configurations spécifiques :

- **docker-compose.override-standalone.yml** : Déploiement autonome (mode Periodic uniquement)
- **docker-compose.override-with-traefik.yml** : Déploiement avec Traefik et TLS (mode Real-time)

L'un de ces fichiers doit être copié sous le nom `docker-compose.override.yml` dans le même répertoire que le fichier de base.

Déploiements

Option 1 : Déploiement Autonome (Periodic uniquement)

Configuration

```
# Copier le fichier override autonome
cp docker-compose.override-standalone.yml docker-compose.override.yml
```

Fichier docker-compose.override-standalone.yml

```
services:
  onedrive-connector:
    environment:
      # Variables d'environnement pour un client nommé: test1
      SHAREPOINT_CLIENTS_TEST1_M365_CLIENT_SECRET: secret
      SHAREPOINT_CLIENTS_TEST1_DETECT_CLIENT_API_TOKEN: 00000000-0
0000000-00000000-00000000-00000000
      SHAREPOINT_CLIENTS_TEST1_ADMIN_TOKEN: 00000000-00000000-00
```

```
000000-00000000-00000000
      SHAREPOINT_CLIENTS_TEST1_ALERT_CONFIG_SMTP_PASSWORD: pwd
```

Caractéristiques :

- ✅ Configuration simple
- ✅ Surveillance périodique uniquement
- ✅ Pas d'infrastructure SSL requise
- ❌ Pas de notifications temps réel

Option 2 : Déploiement avec Traefik (Real-time + Periodic)

Configuration

```
# Copier le fichier override Traefik
cp docker-compose.override-with-traefik.yml docker-compose.override.yml

# Configurer le hostname pour les notifications
export NOTIFICATION_HOSTNAME=votre.domaine.com
sed -i "s@\`public-api.com\`@\`${NOTIFICATION_HOSTNAME:-public-api.co
m}\`@" config/traefik/providers.yml
```

Fichier docker-compose.override-with-traefik.yml

```
services:
  onedrive-connector:
    environment:
      # Variables d'environnement pour un client nommé: test1
      SHAREPOINT_CLIENTS_TEST1_M365_CLIENT_SECRET: secret
      SHAREPOINT_CLIENTS_TEST1_DETECT_CLIENT_API_TOKEN: 00000000-0
0000000-00000000-00000000-00000000
      SHAREPOINT_CLIENTS_TEST1_ADMIN_TOKEN: 00000000-00000000-00
000000-00000000-00000000
      SHAREPOINT_CLIENTS_TEST1_ALERT_CONFIG_SMTP_PASSWORD: pwd
    depends_on:
```

```
      traefik:
        condition: service_healthy
    # Pas d'exposition de ports (géré par le proxy)
    ports: !reset []

  traefik:
    image: traefik:v3.5
    restart: on-failure:3
    healthcheck:
      test: ["CMD", "traefik", "healthcheck"]
      interval: 3s
      timeout: 3s
      start_period: 3s
      retries: 3
    ports:
      - "443:443"
    volumes:
      - ./config/letsencrypts:/letsencrypts
      - ./config/traefik:/etc/traefik:ro
```

Caractéristiques :

- ✅ Notifications temps réel (< 1 minute)
- ✅ Surveillance périodique de backup
- ✅ Certificats SSL automatiques (Let's Encrypt)
- ✅ Infrastructure complète
- ⚠️ Nécessite un domaine public

Fichier de configuration (config/sharepoint/sharepoint.yml)

```
listen: ":8094"
debug: true
log-level-reset-tick: 10s
worker-count: 3
```

```
clients:
  my-client:
    m365-client:
      id: "00000000-0000-0000-0000-000000000000"     # ID de l'App Azure
      tenant: "00000000-0000-0000-0000-000000000000" # Tenant ID
      # secret chargé via variable d'environnement
      api-url: "https://votre-domaine.com"           # URL publique pour webhooks
(optionnel)

    detect-client:
      api-url: "https://poc.gmalware.glimps.re/"
      # api-token chargé via variable d'environnement (doit avoir la permission
"search")
      api-insecure: false
      analysis-timeout: "3m"
      max-upload-size-allowed: "100MB"

    retry:
      period: "1h"
      nb: 2

    monitoring-period: "20m"
    delete-malware: false  # Utilisez quarantine à la place

    quarantine:
      url: "https://votre-sharepoint.com/sites/quarantine"
      delete-malware-on-microsoft-detection: false

    monitored-items:
      sites:
        - url: "https://votre-sharepoint.com/sites/test"
          initial-scan: false

    alert-config:
      smtp:
        host: "smtp.office365.com"
```

```
    port: 587
    login: "sharepoint.connector@votre-tenant.onmicrosoft.com"
    # password chargé via variable d'environnement
    skip-cert-check: false

  from: "sharepoint.connector@votre-tenant.onmicrosoft.com"
  to-admins: ["admin@votre-domaine.com"]
  to-clients: ["client@votre-domaine.com"]
  send-warning-alerts-to-admins: true

  admin-mail:
    subject: "[ADMIN SHAREPOINT ALERT] Fichier non traité"
    body: |
      <h2>Alerte : un fichier n'a pas pu être traité</h2>
      <b>Détails :</b>
      <p>Type d'alerte: {{.AlertType}}
      <br>Client: {{.ClientName}}
      <br>Drive ID: {{.DriveID}}
      <br>Fichier: {{.FileName}}
      <br>Raison: {{.Reason}}
      <br>Erreur: {{.Error}}
      </p>

 db-path: "/etc/glimps_connector/bdd/my-client.db"
 admin-token: "aaaaaaaa-bbbbbbbb-cccccccc-dddddddd-eeeeeeee"  # To
ken pour opérations admin (restauration)
 # Format obligatoire : XXXXXXXX-XXXXXXXX-XXXXXXXX-XXXXXXXX-XXX
XXXXX (caractères hexadécimaux)
 # Si non configuré, les opérations de restauration ne sont pas disponibles
```

Variables d'environnement

Les variables d'environnement sont définies dans les fichiers override selon le client configuré.

Format des variables (exemple pour un client nommé "test1") :

```
SHAREPOINT_CLIENTS_TEST1_M365_CLIENT_SECRET=secret
SHAREPOINT_CLIENTS_TEST1_DETECT_CLIENT_API_TOKEN=00000000-000
00000-00000000-00000000-00000000  # Doit avoir permission "search"
SHAREPOINT_CLIENTS_TEST1_ADMIN_TOKEN=00000000-00000000-00000
000-00000000-00000000
SHAREPOINT_CLIENTS_TEST1_ALERT_CONFIG_SMTP_PASSWORD=pwd
```

⚠️ **Important :**

- Remplacez "TEST1" par le nom de votre client dans la configuration YAML et les variables d'environnement
- Le token GLIMPS Detect doit avoir la permission "search" pour fonctionner

Gestion des conteneurs

Démarrage

```
docker compose up -d
```

Arrêt

```
docker compose down
```

Consultation des logs

```
# Logs en temps réel avec timestamps
docker compose logs -f --timestamps

# Logs d'un service spécifique
docker compose logs -f onedrive-connector
```

Vérification de la configuration

```
# Afficher la configuration finale (avec merge des overrides)
docker compose config
```

Statut des conteneurs

```
# Statut détaillé
docker compose ps

# Exemple de sortie
NAME                    IMAGE                                          COMMAND                  SE
RVICE           CREATED         STATUS         PORTS
odshp-onedrive-connector-1  glimpsre/onedrive-sharepoint-connector:feat-v1
"/srv/onedrive-share..."  onedrive-connector  7 minutes ago   Up 6 minutes
(healthy)
odshp-traefik-1             traefik:v3.5                                   "/entrypoint.sh trae..."
traefik           7 minutes ago   Up 7 minutes    (healthy)  80/tcp, 0.0.0.0:443->4
43/tcp
```

Vérification du service

```
# Test de l'endpoint de santé
wget -qO /dev/null http://localhost/healthz && echo "Service OK" || echo "Ser
vice KO"

# Vérification des bases de données générées automatiquement
ls -la config/sharepoint/bdd/

# Test de l'endpoint de restauration (si port exposé)
curl -X POST http://localhost:PORT/api/v1/restore \
  --data '{"id":"<quarantineID>"}' \
  --header "Content-Type: application/json" \
  --header "X-Auth-Token: <adminToken>"
```

Personnalisation des conteneurs

Configuration Proxy

Ajoutez dans votre `docker-compose.override.yml` :

```
services:
  onedrive-connector:
    environment:
      # Variables existantes...
      http_proxy: "http://proxy.exemple.lan:3128"
      https_proxy: "https://proxy.exemple.lan:3128"
      no_proxy: "localhost, traefik"

  traefik: # Si utilisé
    environment:
      http_proxy: "http://proxy.exemple.lan:3128"
      https_proxy: "https://proxy.exemple.lan:3128"
      no_proxy: "localhost, onedrive-connector"
```

Configuration DNS personnalisé

```
services:
  onedrive-connector:
    dns: 1.1.1.1

  traefik: # Si utilisé
    dns: 1.1.1.1
```

Tests et validation

Test automatique des alertes

Si configuré, un email de test est envoyé au démarrage via `test-mail-sending`.

Tests manuels

1. **Test taille de fichier** :
   - Configurez `max-upload-size-allowed: "1kB"`
   - Uploadez un fichier plus volumineux
2. **Test quotas atteints** :
   - Utilisez un token avec quotas épuisés
   - Ajoutez de nouveaux fichiers

Monitoring et logs

```
# Logs en temps réel
docker compose logs -f

# Logs spécifiques
docker compose logs onedrive-connector

# Métriques container
docker stats onedrive-connector
```

Maintenance

Mise à jour

```
docker compose pull
docker compose up -d
```

Sauvegarde des données

```
# Sauvegarde des bases SQLite et configuration
tar -czf backup-$(date +%Y%m%d).tar.gz config/
```

Résolution de problèmes courants

1. **Webhook non accessible (mode Traefik)** :
   - Vérifiez la configuration DNS et l'accessibilité du domaine
   - Contrôlez les certificats Let's Encrypt dans `config/letsencrypts/acme.json`
2. **Permissions manquantes** : Vérifiez les permissions Azure AD
3. **Secret expiré** : Régénérez le secret dans Azure AD et mettez à jour la variable d'environnement
4. **Quotas Detect** : Vérifiez votre consommation API
5. **Conteneur unhealthy** :

```
# Vérifiez les logs détaillés
docker compose logs onedrive-connector

# Testez manuellement le health check
docker compose exec onedrive-connector wget -qO /dev/null http://localh
ost/healthz
```

Comparaison des modes de déploiement

Aspect	Mode Autonome	Mode avec Traefik
Complexité	Simple	Modérée
Infrastructure	Minimale	Traefik + DNS requis
Détection	Periodic uniquement	Real-time + Periodic
Latence	20 min (configurable)	< 1 minute
Sécurité	Basique	TLS automatique
Cas d'usage	PME, tests, POC	Production, grandes org
Maintenance	Faible	Modérée

Sécurité

- Utilisez des secrets forts pour les tokens
- Renouvelez régulièrement les secrets Azure AD

- Surveillez les accès via les logs
- Limitez les permissions aux minimum requis
- Utilisez HTTPS pour les webhooks
- Assurez-vous que le token GLIMPS Detect a la permission "search"