

## Serveur winget privé (rewinged)

Source de paquets winget auto-hébergée pour le parc géré Make IT Simple. Permet d'aligner les versions logicielles sur l'ensemble des parcs clients sans donner à l'agent de déploiement le droit de remplacer son propre binaire.

## 1. Architecture



| Composant     | Machine          | Rôle                                                    |
|---------------|------------------|---------------------------------------------------------|
| Forgejo       | existante        | Source de vérité : manifests YAML et installateurs MSI. |
| Caddy frontal | frontal existant | Terminaison TLS et routage. Aucun fichier local.        |
| rewinged      | VM dédiée        | API REST Microsoft. Rest consommée par winget.          |
| Caddy local   | VM dédiée        | Sert les installateurs synchronisés.                    |
| Wazuh         | existant         | FIM sur manifests, installateurs et binaires.           |

**Le dépôt Forgejo peut rester privé.** Les postes ne dialoguent qu'avec `repo.makeitsimple.be`. Aucun accès anonyme à Forgejo n'est requis, puisque la synchronisation se fait par clé de déploiement SSH.

**Chaîne de confiance.** L'intégrité des paquets ne repose pas sur le TLS mais sur le SHA256 déclaré dans chaque manifest. Un installateur substitué échoue à la vérification et winget refuse l'installation.

## 2. Prérequis

- VM Debian 12 minimale, 1 vCPU / 1 Go RAM / disque dimensionné au volume d'installateurs
- Enregistrement DNS public `repo.makeitsimple.be` vers le frontal Caddy
- Dépôt Forgejo mis/winget-manifests avec Git LFS activé
- Clé de déploiement Forgejo **en lecture seule**
- Paquets `git`, `git-lfs`, `caddy` sur la VM

## 3. Installation de rewinged

### 3.1 Utilisateur et arborescence

```
useradd --system --home /srv/winget --shell /usr/sbin/nologin rewinged
mkdir -p /srv/winget
chown -R rewinged:rewinged /srv/winget
```

Les répertoires manifests et installers seront créés par le clone Git (section 6).

### 3.2 Binaire

Télécharger la release depuis [GitHub](#).

**Épingler une version précise.** Le projet est en pré-1.0 : l'auteur indique que configuration, sortie et comportement peuvent changer à tout moment.

```
VERSION="0.15.0"
cd /tmp
# adapter le nom d'archive à celui de la release
tar xzf rewinged_${VERSION}_linux_amd64.tar.gz
install -m 0755 rewinged /usr/local/bin/rewinged
/usr/local/bin/rewinged --help
```

<WRAP center round important> Relever dans `--help` le nom exact du flag désignant le répertoire de manifests, et vérifier si `rewinged` recharge à chaud ou s'il faut un signal après un `git pull`. Adapter les sections 3.3 et 7 en conséquence. </WRAP>

### 3.3 Service systemd

[/etc/systemd/system/rewinged.service](#)

```
[Unit]
Description=rewinged winget package source
After=network.target

[Service]
```

```
User=rewinged
Group=rewinged
ExecStart=/usr/local/bin/rewinged -listen 10.0.0.20:8080
Restart=on-failure
RestartSec=5

# Durcissement
ProtectSystem=strict
ProtectHome=true
PrivateTmp=true
PrivateDevices=true
NoNewPrivileges=true
RestrictSUIDSGID=true
LockPersonality=true
MemoryDenyWriteExecute=true
ReadOnlyPaths=/srv/winget

[Install]
WantedBy=multi-user.target
```

```
systemctl daemon-reload
systemctl enable --now rewinged
systemctl status rewinged
```

## 4. Service de fichiers local

Les installeurs sont servis par un Caddy local sur la VM rewinged, en HTTP sur le réseau interne. Le TLS reste la responsabilité du frontal.

[/etc/caddy/Caddyfile](#)

```
http://10.0.0.20:8081 {
    root * /srv/winget/installers
    file_server {
        hide .git*
    }
    log {
        output file /var/log/caddy/installers.log
        format json
    }
}
```

```
systemctl enable --now caddy
curl -I http://10.0.0.20:8081/mis-agent-1.2.0.msi
```

Alternative à évaluer : la fonction d'internalisation de rewinged sert elle-même les installeurs, ce qui

supprimerait ce composant. Vérifier le chemin d'URL exposé avant de s'en remettre à elle.

## 4.1 Pare-feu

Seul le frontal doit joindre les deux ports.

```
CADDY_IP="10.0.0.10"
ufw default deny incoming
ufw allow from ${CADDY_IP} to any port 8080 proto tcp
ufw allow from ${CADDY_IP} to any port 8081 proto tcp
ufw allow from 100.64.0.0/10 to any port 22 proto tcp
ufw enable
```

## 5. Configuration du frontal Caddy

À ajouter au Caddyfile **de la machine frontale**.

</etc/caddy/Caddyfile>

```
repo.makeitsimple.be {
    tls vincent@makeitsimple.be

    # API winget
    reverse_proxy /api* 10.0.0.20:8080

    # Installeurs
    reverse_proxy /files/* 10.0.0.20:8081

    header {
        Strict-Transport-Security "max-age=31536000"
        X-Content-Type-Options "nosniff"
        -Server
    }

    log {
        output file /var/log/caddy/repo.log
        format json
    }
}
```

Winget exige HTTPS avec une chaîne validée par le magasin de confiance Windows. Let's Encrypt sur un nom public évite d'avoir à déployer une CA interne sur chaque poste.

<WRAP center round tip> Si le Caddy local sert la racine et non /files/, ajuster avec `handle_path` côté frontal pour retirer le préfixe avant transmission. </WRAP>

## 6. Dépôt Forgejo

Un seul dépôt contient manifests et installers. Les MSI passent par Git LFS pour ne pas alourdir l'historique.

### 6.1 Structure

```
winget-manifests/
├── .gitattributes
├── manifests/
│   └── m/MakeItSimple/
│       ├── Agent/1.2.0/MakeItSimple.Agent.yaml
│       └── RustDesk/1.4.3/MakeItSimple.RustDesk.yaml
└── installers/
    ├── mis-agent-1.2.0.msi
    └── rustdesk-1.4.3.msi
```

### 6.2 Activation de Git LFS

Côté poste de travail, une seule fois :

```
git lfs install
git lfs track "*.msi"
git lfs track "*.exe"
git add .gitattributes
git commit -m "Suivi LFS des installers"
git push
```

Vérifier que LFS est activé sur l'instance Forgejo et que le quota de stockage est suffisant. LFS ne fait pas de delta : chaque version d'un MSI est stockée intégralement.

### 6.3 Clone initial sur la VM

```
apt install git-lfs
sudo -u rewinged git lfs install --skip-repo

sudo -u rewinged git clone \
  git@forgejo.makeitsimple.be:mis/winget-manifests.git \
  /srv/winget

sudo -u rewinged git -C /srv/winget lfs pull
```

La clé de déploiement doit être en **lecture seule** : le serveur de paquets ne doit jamais pouvoir écrire dans le dépôt.

## 7. Synchronisation automatique

Pas de webhook : un timer évite d'exposer un endpoint entrant sur la machine qui décide de ce que le parc installe. Le flux part toujours du serveur vers Forgejo.

[/etc/systemd/system/winget-sync.service](#)

```
[Unit]
Description=Sync winget manifests and installers from Forgejo

[Service]
Type=oneshot
User=rewinged
WorkingDirectory=/srv/winget
ExecStart=/usr/bin/git pull --ff-only
ExecStart=/usr/bin/git lfs pull
```

[/etc/systemd/system/winget-sync.timer](#)

```
[Unit]
Description=Sync winget repository every minute

[Timer]
OnBootSec=1min
OnUnitActiveSec=1min

[Install]
WantedBy=timers.target
```

```
systemctl daemon-reload
systemctl enable --now winget-sync.timer
systemctl list-timers winget-sync.timer
journalctl -u winget-sync.service -f
```

Le `--ff-only` est délibéré : si l'historique local diverge, le pull échoue au lieu de fusionner silencieusement. Une divergence signifie une écriture locale, ce qui ne doit jamais arriver.

<WRAP center round important> `ReadOnlyPaths=/srv/winget` dans l'unité `rewinged` empêche le service de modifier le dépôt, mais n'affecte pas `winget-sync` qui est une unité distincte. Vérifier que le pull aboutit après le premier démarrage. </WRAP>

## 8. Format d'un manifest

[MakeltSimple.Agent.yaml](#)

```

PackageIdentifier: MakeItSimple.Agent
PackageVersion: 1.2.0
PackageLocale: en-US
Publisher: Make IT Simple SRL
PackageName: MIS Agent
License: Proprietary
ShortDescription: Agent d'inventaire et de mise a jour
Installers:
  - Architecture: x64
    InstallerType: msi
    InstallerUrl:
https://repo.makeitsimple.be/files/mis-agent-1.2.0.msi
    InstallerSha256: A1B2C3D4E5F6...
    Scope: machine
  InstallerSwitches:
    Silent: /quiet /norestart
  AppsAndFeaturesEntries:
    - ProductCode: '{XXXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXX}'
      UpgradeCode: '{YYYYYYYY-YYYY-YYYY-YYYY-YYYYYYYYYYYY}'
ManifestType: singleton
ManifestVersion: 1.6.0

```

Calcul du hash :

```
sha256sum mis-agent-1.2.0.msi | tr 'a-f' 'A-F'
```

**Types d'installateurs supportés** : msi, exe, inno, nullsoft, wix, burn, msix, appx, zip, portable. Ne pas utiliser portable pour un service Windows — il ne fait qu'un alias sur le PATH.

## 9. Extraction des codes MSI

Le ProductCode change à chaque version d'un MSI bien construit ; l'UpgradeCode reste stable et identifie le produit dans la durée. Le bloc AppsAndFeaturesEntries en dépend, notamment pour l'adoption des logiciels déjà installés depuis d'anciens paquets custom.

### 9.1 Depuis un fichier MSI

À exécuter sur la machine de build (Windows). Lecture seule, sans installation ni Orca.

[Get-MsiCodes.ps1](#)

```

function Get-MsiCodes {
    param([string]$Path)
    $installer = New-Object -ComObject WindowsInstaller.Installer
    $db =
    $installer.GetType().InvokeMember('OpenDatabase', 'InvokeMethod', $null, $

```

```

installer,@($Path,0))
    $props = @{}
    foreach ($p in
'ProductCode', 'UpgradeCode', 'ProductVersion', 'ProductName') {
        $view =
$db.Gettype().InvokeMember('OpenView', 'InvokeMethod', $null, $db,
        @("SELECT Value FROM Property WHERE Property='$p'"))
$view.Gettype().InvokeMember('Execute', 'InvokeMethod', $null, $view, $null
)
        $rec =
$view.Gettype().InvokeMember('Fetch', 'InvokeMethod', $null, $view, $null)
        if ($rec) {
            $props[$p] =
$rec.Gettype().InvokeMember('StringData', 'GetProperty', $null, $rec, 1)
        }
    }
[PSCustomObject]$props
}

Get-MsiCodes 'C:\build\mis-agent-1.2.0.msi'

```

Sortie type :

```

ProductName      : MIS Agent
ProductVersion   : 1.2.0
ProductCode      : {A1B2C3D4-1111-2222-3333-444455556666}
UpgradeCode      : {99887766-AAAA-BBBB-CCCC-DDDDEEEFFFFFF}

```

Le COM WindowsInstaller.Installer n'existe que sous Windows : ce script ne tourne pas sur la VM Debian.

## 9.2 Depuis un poste où le logiciel est installé

Pour recenser les GUID en circulation dans le parc :

```

Get-ItemProperty `
HKLM:\SOFTWARE\Microsoft\Windows\CurrentVersion\Uninstall\*, `
HKLM:\SOFTWARE\WOW6432Node\Microsoft\Windows\CurrentVersion\Uninstall\* |
Where-Object DisplayName -like '*RustDesk*' |
Select-Object DisplayName, DisplayVersion, PSChildName

```

PSChildName porte le ProductCode lorsque la clé est un GUID, ce qui est le cas des MSI.  
L'UpgradeCode n'y figure pas.

<WRAP center round important> Éviter Get-CimInstance Win32\_Product en production : la classe déclenche une réévaluation de tous les MSI installés, ce qui est lent et peut générer des événements de réparation. Acceptable ponctuellement sur un poste témoin. </WRAP>

## 9.3 Automatisation

L'agent d'inventaire devrait remonter le ProductCode de chaque logiciel installé. Cela donne côté serveur la liste exacte des GUID du parc, donc les AppsAndFeaturesEntries à renseigner, sans interroger les postes un par un.

## 10. Workflow de publication

1. Construire l'installateur
2. Relever les codes avec Get-MsiCodes (9.1)
3. Calculer le SHA256
4. Placer le MSI dans installers/ et le manifest dans manifests/
5. Commit sur une branche, ouvrir une PR, revue, merge sur main
6. Le timer applique dans la minute
7. Valider depuis un poste témoin avant diffusion

```
winget search -s mis-repo MakeItSimple
winget install --id MakeItSimple.Agent -s mis-repo --silent
```

## 11. Configuration des postes

### 11.1 Enregistrement de la source

```
winget source add -n mis-repo `
-a https://repo.makeitsimple.be/api `
-t "Microsoft.Rest"
```

À exécuter en **scope machine**, sinon la source n'existe que pour l'utilisateur courant et reste invisible à l'agent tournant en service.

### 11.2 Winget en contexte SYSTEM

winget n'est pas dans le PATH pour le compte SYSTEM. Résoudre le chemin versionné :

```
$winget = (Get-ChildItem `
"C:\Program
Files\WindowsApps\Microsoft.DesktopAppInstaller_*_x64__8wekyb3d8bbwe\winget.
exe" |
Select-Object -Last 1).FullName
```

Ajouter `--disable-interactivity` à tous les appels non interactifs, sinon un prompt bloque le processus indéfiniment.

## 11.3 Adoption du parc existant

Un paquet installé autrement que depuis la source privée n'est pas reconnu par winget upgrade tant qu'il n'a pas été installé une fois depuis cette source. Deux contournements :

- renseigner les AppsAndFeaturesEntries des anciens builds
- passer une réinstallation depuis mis - repo

À valider sur un poste témoin **avant** de dimensionner la campagne.

## 11.4 Compatibilité

| Système                       | App Installer                                        |
|-------------------------------|------------------------------------------------------|
| Windows 10 1809+ / 11         | Livré                                                |
| Windows Server 2025           | Livré                                                |
| Windows Server 2019 / 2022    | <b>Absent</b> — installation manuelle, non supportée |
| Éditions N et LTSC sans Store | Installation manuelle                                |

Inventorier la part de Server 2019/2022 du parc : elle décide si winget est le canal unique ou s'il faut un second canal.

# 12. Surveillance Wazuh

## 12.1 FIM

[ossec.conf](#)

```
<syscheck>
  <directories check_all="yes" realtime="yes" report_changes="yes">
    /srv/winget/manifests
  </directories>
  <directories check_all="yes" realtime="yes">
    /srv/winget/installers
  </directories>
  <directories check_all="yes" realtime="yes">
    /usr/local/bin
  </directories>
</syscheck>
```

## 12.2 Réduction du bruit

La FIM se déclenche à chaque synchronisation légitime. Pour distinguer le légitime de l'anormal :

- signer les commits en GPG et vérifier la signature dans winget -sync

avant application — un fichier modifié sans commit signé devient alors une

```
alerte non ambiguë
* à défaut, corrélér avec les logs de ''winget-sync.service''
```

## 12.3 Alertes complémentaires

- arrêt des services rewinged ou caddy
- échecs répétés de `winget-sync.timer` — un dépôt figé est un mode de

panne silencieux

- volumétrie anormale de téléchargements sur `/files/`
- saturation disque de la VM (croissance LFS)

## 13. Points ouverts

| Sujet            | Action                                                                                        |
|------------------|-----------------------------------------------------------------------------------------------|
| Flags rewinged   | Confirmer le nom du flag de répertoire et le rechargement à chaud via <code>—help</code>      |
| Internalisation  | Évaluer si rewinged peut remplacer le Caddy local (section 4)                                 |
| Bug de matching  | Tester sur un poste témoin avec un logiciel préinstallé                                       |
| Server 2019/2022 | Inventorier la proportion dans le parc                                                        |
| Upgrade RustDesk | Vérifier que la mise à jour préserve <code>RustDesk2.toml</code> et le mot de passe permanent |
| Rétention LFS    | Définir une politique de purge des anciennes versions d'installateurs                         |
| Quota Forgejo    | Dimensionner le stockage LFS et surveiller sa croissance                                      |

## 14. Références

- [rewinged](#) — dépôt GitHub
- [Documentation Windows Package Manager](#)
- [RustDesk Server Pro](#) — console et tokens API

From:

<https://wiki.makeitsimple.be/> - **makeITsimple wiki**

Permanent link:

<https://wiki.makeitsimple.be/doku.php?id=linux:rewinged&rev=1789197393>

Last update: **2026/09/12 07:16**

