Durcissement système à l'aide de systemd

Timothée Ravier

Agence nationale de la sécurité des systèmes d'information

SSTIC 2017

Objectif

Objectif

- Tirer profit des fonctionnalités de sécurité du noyau Linux ;
- S'aider de l'intégration dans systemd ;
- Pour simplifier le durcissement et la maintenance d'un système.

systemd en trois slides

systemd ?

- Remplaçant de SysVinit intégré dans la plupart des distributions ;
- Chargé du démarrage et de la gestion des services système ;
- Remplace les scripts d'init par des fichiers de configuration déclaratifs :
  - les **units**.

Unit ?

Afficher la configuration actuelle d'un service :

```
# systemctl cat php-fpm.service

# /usr/lib/systemd/system/php-fpm.service

[Unit]
Description=The PHP FastCGI Process Manager
After=network.target

[Service]
Type=notify
PIDFile=/run/php-fpm/php-fpm.pid
ExecStart=/usr/bin/php-fpm --nodaemonize
PrivateTmp=true

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

Commande

Unit ?

Afficher la configuration actuelle d'un service :

```
# systemctl cat php-fpm.service

# /usr/lib/systemd/system/php-fpm.service

[Unit]
Description=The PHP FastCGI Process Manager
After=network.target

[Service]
Type=notify
PIDFile=/run/php-fpm/php-fpm.pid
ExecStart=/usr/bin/php-fpm --nodaemonize
PrivateTmp=true

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

Fichier correspondant

Unit ?

Afficher la configuration actuelle d'un service :

```
# systemctl cat php-fpm.service

# /usr/lib/systemd/system/php-fpm.service

[Unit]
Description=The PHP FastCGI Process Manager
After=network.target

[Service]
Type=notify
PIDFile=/run/php-fpm/php-fpm.pid
ExecStart=/usr/bin/php-fpm --nodaemonize
PrivateTmp=true

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

Qui ?
Quand ?

Unit ?

Afficher la configuration actuelle d'un service :

```
# systemctl cat php-fpm.service

# /usr/lib/systemd/system/php-fpm.service

[Unit]
Description=The PHP FastCGI Process Manager
After=network.target

[Service]
Type=notify
PIDFile=/run/php-fpm/php-fpm.pid
ExecStart=/usr/bin/php-fpm --nodaemonize
PrivateTmp=true

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

Quoi ?
Comment ?

Unit ?

Afficher la configuration actuelle d'un service :

```
# systemctl cat php-fpm.service

# /usr/lib/systemd/system/php-fpm.service

[Unit]
Description=The PHP FastCGI Process Manager
After=network.target

[Service]
Type=notify
PIDFile=/run/php-fpm/php-fpm.pid
ExecStart=/usr/bin/php-fpm --nodaemonize
PrivateTmp=true

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

Pourquoi ?

Exemple : Utilisateur et groupe non privilégiés

Éditer la configuration d'un service :

```
# systemctl edit php-fpm.service
```

Exemple : Utilisateur et groupe non privilégiés

Éditer la configuration d'un service :

```
# systemctl edit php-fpm.service
```

pour ajouter :

```
[Service]
User=http
Group=www
```

Exemple : Utilisateur et groupe non privilégiés

Éditer la configuration d'un service :

```
# systemctl edit php-fpm.service
```

pour ajouter :

```
[Service]
User=http
Group=www
```

et rendre les modifications effectives :

```
# systemctl daemon-reload
# systemctl restart php-fpm.service
```

Tirer profit des nouvelles fonctions de sécurité du noyau Linux

Filtrage d'appels système avec *seccomp-bpf*

Concept

- Filtrage des appels système disponibles pour un processus ;
- S'applique aussi aux processus fils.

Filtrage d'appels système avec *seccomp-bpf*

Concept

- Filtrage des appels système disponibles pour un processus ;
- S'applique aussi aux processus fils.

Exemple

```
[Service]
SystemCallFilter=~chroot
SystemCallFilter=~@obsolete
```

Filtrage d'appels système avec *seccomp-bpf*

Concept

- Filtrage des appels système disponibles pour un processus ;
- S'applique aussi aux processus fils.

Exemple

```
[Service]
SystemCallFilter=~chroot
SystemCallFilter=~@obsolete
```

À noter

- Contournable sur les noyaux < 4.8 avec ptrace ;
- Solution : filtrer l'appel système ptrace :

```
[Service]
SystemCallFilter=~ptrace
```

Capacités Linux

Concept

- Restriction des droits accordés à un processus (potentiellement) root ;
- Ajout de permissions à un processus non root.

Capacités Linux

Concept

- Restriction des droits accordés à un processus (potentiellement) root ;
- Ajout de permissions à un processus non root.

Exemple

```
[Service]
CapabilityBoundingSet=CAP_NET_BIND_SERVICE
AmbientCapabilities=CAP_NET_BIND_SERVICE
```

Capacités Linux

Concept

- Restriction des droits accordés à un processus (potentiellement) root ;
- Ajout de permissions à un processus non root.

Exemple

```
[Service]
CapabilityBoundingSet=CAP_NET_BIND_SERVICE
AmbientCapabilities=CAP_NET_BIND_SERVICE
```

À noter

- Certaines capacités sont équivalentes à root ;
- Préférer une liste blanche des capacités réellement nécessaires.

Espace de nom de points de montage

Concept

- Arborescence du système de fichier distincte pour chaque service.

Espace de nom de points de montage

Concept

- Arborescence du système de fichier distincte pour chaque service.

Exemple

```
[Service]
InaccessiblePaths=/etc/secrets
ProtectSystem=full
```

Espace de nom de points de montage

Concept

- Arborescence du système de fichier distincte pour chaque service.

Exemple

```
[Service]
InaccessiblePaths=/etc/secrets
ProtectSystem=full
```

À noter

- Réversible si CAP_SYS_ADMIN :

```
[Service]
CapabilityBoundingSet=~CAP_SYS_ADMIN
SystemCallFilter=~@mount
```

Getting your hands dirty (cow ?)

Mise en pratique avec Dirty CoW

- Vulnérabilité CVE-2016-5195 ;
- *Local root* rendue publique en octobre 2016 ;
- Présente depuis la version 2.6.22 du noyau, publiée en 2007 ;
- Situation de compétition dans le mécanisme de *copy-on-write*.

Mise en pratique avec Dirty CoW

Vulnérabilité

- Situation de compétition déclenchée avec l'appel système `madvise`.

Options pour réduire l'impact

- Bloquer l'appel système `madvise`.

Configuration

```
[Service]
SystemCallFilter=~madvise
```

Mise en pratique avec Dirty CoW

Vulnérabilité

- Utilisation de l'appel système `ptrace` et de `/proc/self/mem`.

Options pour réduire l'impact

- Bloquer l'appel système `ptrace` ;
- Supprimer l'accès au système de fichiers virtuel `/proc`.

Configuration

```
[Service]
SystemCallFilter=~ptrace
InaccessiblePaths=/proc
```

Mise en pratique avec Dirty CoW

Vulnérabilité

- Certains drivers de périphériques matériels potentiellement concernés.

Options pour réduire l'impact

- Supprimer l'accès aux périphériques matériels exposés dans `/dev`.

Configuration

```
[Service]
PrivateDevices=yes
```

Conclusion

Conclusion

- Interface simplifiant l'utilisation des fonctionnalités de sécurité du noyau ;
- Ne remplace pas l'application des mises à jour ;
- Durcissement appliqué uniquement aux services système ;
- Plus de détails et d'exemples dans l'article publié dans les actes.

Merci pour votre attention

Contact :

- ✉ timothee.ravier@ssi.gouv.fr / tim@siosm.fr
- @siosm