MLB

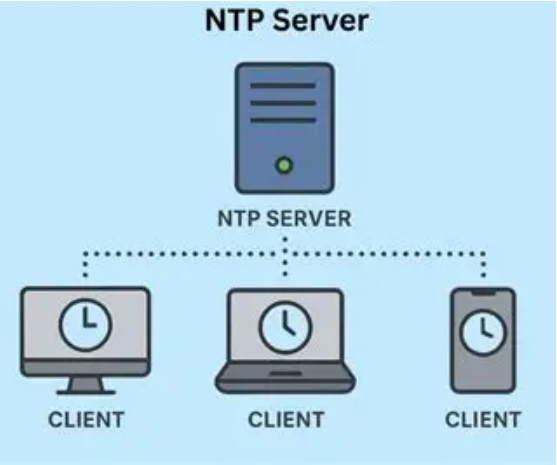
Auteur : LE BRAS Maxime

Date : 04/05/2026

Version : 1.2

Nombre de pages : 9

Synchronisation temporelle du serveur Prometheus



Suivi des versions

Version	Date	Rédacteur	Modification
1.0	04/05/2026	LE BRAS Maxime	Création du document
1.1	05/05/2026	LE BRAS Maxime	Réorganisation concernant la structure du document
1.2	07/05/2026	LE BRAS Maxime	Supervision Zabbix

Liste de diffusion

Service / Fonction	Nom	Objet de la diffusion
IT Service Desk	/	Interne

SOMMAIRE

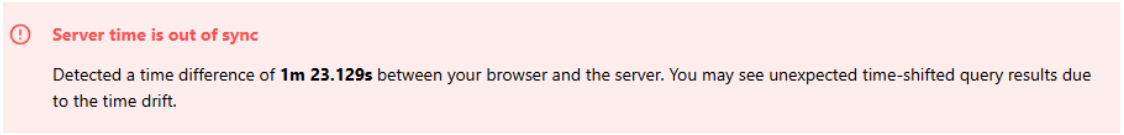
1 – Définitions.....4
2 – Contexte et solution apportée.....4
3 – Configuration du service Chrony.....4
4 – Supervision..... 6
5 – Corrections.....9

1 – Définitions

Signification des protocoles utilisés :

- **UDP** => **U**ser **D**atagram **P**rotocol
- **NTP** => **N**etwork **T**ime **P**rotocol

2 – Contexte et solution apportée



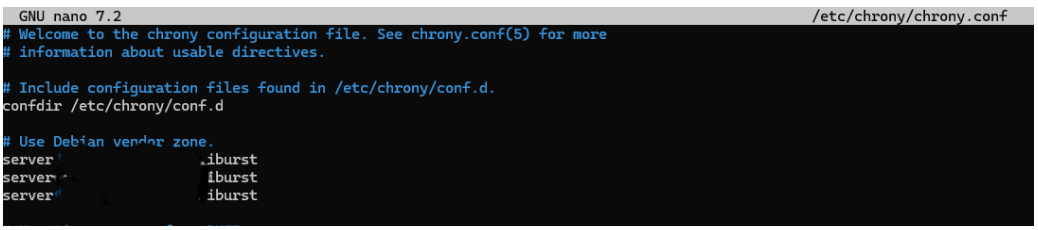
Le serveur **Prometheus** a une dérive de temps trop importante ce qui veut dire qu'il doit être synchronisé via un service **NTP** interne.

Ce type de service doit avoir une horloge système stable, car il surveille les machines du parc informatique.

La solution est de mettre en place le client Chrony sur le serveur **Prometheus** afin qu'il se synchronise avec un des contrôleurs de domaine **Active Directory** utilisant le protocole **NTP**.

3 – Configuration du service Chrony





Le service **Chrony** est lié au protocole **NTP** qui travaille sur le port 123 travaillant en **UDP**.

Il fonctionne en mode client ou en mode serveur selon les besoins.

Ce client est lié aux trois contrôleurs de domaine afin d'assurer une continuité de service concernant la synchronisation temporelle du serveur **Prometheus**.

```
root@C-PROMETHEUS:~# systemctl restart chronyd
```

```
root@C-PROMETHEUS:~# chronyc sources
MS Name/IP address         Stratum Poll Reach LastRx Last sample
===============================================================================
^*                               4   6     7     1   -879us[ -616ms] +/-   44ms
^+                               5   6     7     1   +592us[ -615ms] +/-   77ms
^+                               5   6     7     1   +151us[ -615ms] +/-   65ms
root@C-PROMETHEUS:~# chronyc tracking
Reference ID    : AC13F609
Stratum         : 5
Ref time (UTC)  : Mon May 04 12:01:08 2026
System time     : 0.000000516 seconds slow of NTP time
Last offset     : -0.000104823 seconds
RMS offset      : 0.000104823 seconds
Frequency       : 0.000 ppm slow
Residual freq   : -17.566 ppm
Skew            : 1000000.000 ppm
Root delay      : 0.023898233 seconds
Root dispersion : 28.178119659 seconds
Update interval : 2.0 seconds
Leap status     : Normal
```

```
root@C-PROMETHEUS:~# timedatectl
               Local time: lun. 2026-05-04 14:04:14 CEST
           Universal time: lun. 2026-05-04 12:04:14 UTC
                 RTC time: lun. 2026-05-04 12:04:13
                Time zone: Europe/Paris (CEST, +0200)
System clock synchronized: yes
              NTP service: active
          RTC in local TZ: no
```

On redémarre **Chrony** pour prendre en compte les modifications effectuées sur sa configuration.

On voit que la communication **NTP** entre le serveur **Prometheus** et les contrôleurs de domaine, est fonctionnelle ce qui veut dire que l'horloge système est bien synchronisée.

Pour conclure, le système de synchronisation temporelle est opérationnel.

4 – Supervision

Pour garantir la synchronisation temporelle du serveur **Prometheus**, nous mettons en place une supervision basée sur un script **Bash**, intégrée au sein de la solution **Zabbix**.

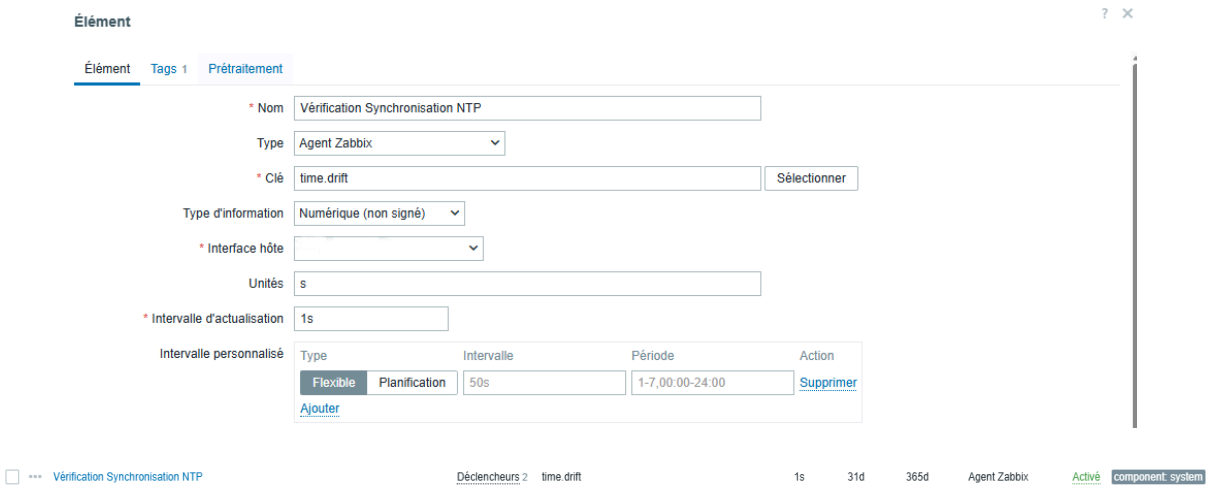
```
  GNU nano 7.2                                    verification_sync_temp.sh
#!/bin/bash

# Auteur : LE BRAS Maxime
# Date : 05/05/2026
# Titre : Vérification de la synchronisation temporelle du serveur Prometheus
# Version : 1.0

# Version 1.0 - 05/05/2026 - Mise en place du script

# Calcul du temps de décalage entre le serveur Prometheus et le serveur de référence

DATE_PROMETHEUS_SERVER=$(date '+%Y-%m-%d %H:%M:%S')
DATE_REMOTE_SERVER=$(curl -s http://                                    )
ADAPTED_DATE_REMOTE_SERVER="${DATE_REMOTE_SERVER//\//-}"

TIME_SECONDS_PROMETHEUS_SERVER=$(date -d "$DATE_PROMETHEUS_SERVER" +%s)
TIME_SECONDS_REMOTE_SERVER=$(date -d "$ADAPTED_DATE_REMOTE_SERVER" +%s)
TIME_DRIFT=$(( TIME_SECONDS_PROMETHEUS_SERVER - TIME_SECONDS_REMOTE_SERVER ))
TIME_DRIFT=${TIME_DRIFT#-}

echo "$TIME_DRIFT"
```

```
-rwx------ 1 zabbix       zabbix            696  5 mai   12:11 verification_sync_temp.sh
```

Tout d'abord, ce script permet de calculer la dérive de temps du serveur **Prometheus** et il est affecté à l'utilisateur « zabbix » afin de pouvoir l'utiliser dans **Zabbix**.

```
UserParameter=time.drift,/usr/local/bin/verification_sync_temp.sh
```

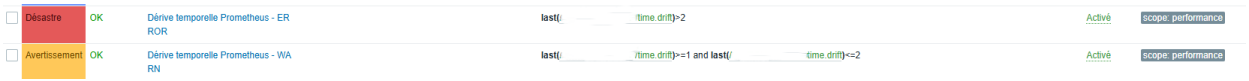
Ensuite, il faut adapter la configuration du client **Zabbix** (/etc/zabbix/zabbix_agent2.conf) du serveur **Prometheus** en mettant en place une commande shell liée à une clé personnalisée (time.drift).



Puis, il faut configurer un élément utilisant la clé configurée précédemment.

Cette mise en place va permettre de mettre en relation l'agent **Zabbix** et le serveur **Zabbix** ce qui va entraîner l'exécution du programme de vérification concernant la synchronisation temporelle.

Il est nécessaire d'exécuter ce script toutes les secondes, car la synchronisation temporelle est primordiale pour le bon fonctionnement du serveur **Prometheus**.



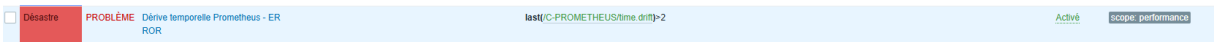
Pour le premier déclencheur, il est en sévérité « Avertissement » puisqu'il ne doit pas avoir une dérive temporelle supérieure à 2 secondes.

Pour le deuxième déclencheur, il est en sévérité « Désastre » puisqu'il ne doit pas avoir une dérive temporelle supérieure à 2 secondes.

Si le service a un temps de décalage trop important, il risque de mal fonctionner et de devenir indisponible.

```
root@C-PROMETHEUS:/usr/local/bin# systemctl stop chrony && timedatectl set-time "2026-05-05 14:45:20"
root@C-PROMETHEUS:/usr/local/bin# timedatectl
               Local time: mar. 2026-05-05 14:45:25 CEST
           Universal time: mar. 2026-05-05 12:45:25 UTC
                 RTC time: mar. 2026-05-05 12:45:26
                Time zone: Europe/Paris (CEST, +0200)
System clock synchronized: no
              NTP service: inactive
          RTC in local TZ: no
```

```
root@C-PROMETHEUS:/usr/local/bin# ./verification_sync_temp.sh
6
```



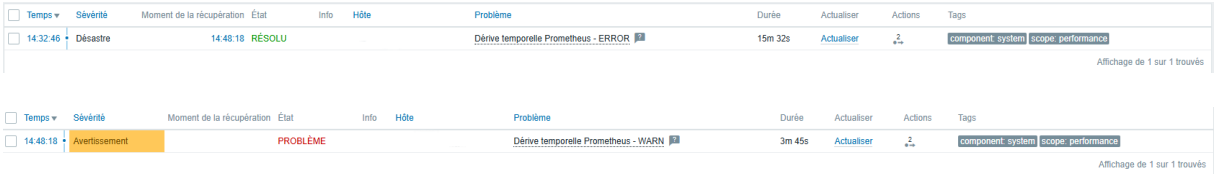
Pour la phase de tests, il faut désactiver la synchronisation temporelle via le service **Chrony** et mettre en place une horloge système décalée.

Après exécution du script en local, il y a une dérive de temps de 6 secondes sur le serveur **Prometheus**.

On remarque que le deuxième déclencheur s'active ce qui est parfaitement cohérent avec le test effectué.

```
root@C-PROMETHEUS:/usr/local/bin# timedatectl set-time "2026-05-05 14:48:20"
```

```
root@C-PROMETHEUS:/usr/local/bin# ./verification_sync_temp.sh
2
```



Après exécution du script en local, il y a une dérive de temps de 2 secondes sur le serveur **Prometheus**.

On constate que le premier déclencheur se lance et le deuxième déclencheur se désactive ce qui est en accord avec le test réalisé.

```
root@C-PROMETHEUS:~# systemctl restart chrony
root@C-PROMETHEUS:~# chronyc sources
MS Name/IP address         Stratum Poll Reach LastRx Last sample
===============================================================================
^*                               4   6    17     7  +1157us[+1162us] +/-   55ms
^+                               5   6    17     7   -986us[ -981us] +/-   87ms
^+                               5   6    17     7   -893us[ -888us] +/-   87ms
root@C-PROMETHEUS:~# chronyc tracking
Reference ID    : AC13F609
Stratum         : 5
Ref time (UTC)  : Tue May 05 12:54:47 2026
System time     : 0.000004739 seconds fast of NTP time
Last offset     : +0.000004880 seconds
RMS offset      : 0.000004880 seconds
Frequency       : 12.878 ppm slow
Residual freq   : +1.706 ppm
Skew            : 0.287 ppm
Root delay      : 0.024556037 seconds
Root dispersion : 0.042533740 seconds
Update interval : 2.0 seconds
Leap status     : Normal
root@C-PROMETHEUS:~# timedatectl
               Local time: mar. 2026-05-05 14:55:25 CEST
           Universal time: mar. 2026-05-05 12:55:25 UTC
                 RTC time: mar. 2026-05-05 12:55:25
                Time zone: Europe/Paris (CEST, +0200)
System clock synchronized: yes
              NTP service: active
          RTC in local TZ: no
```



En remettant la synchronisation temporelle via **Chrony**, tous les déclencheurs sont inactifs.



Lorsqu'un déclencheur se met en route, un mail va être envoyé au service DESK et au responsable DESK.

Pour finir, le système de supervision est fonctionnel.

5 – Corrections

Nous avons constaté que l'intervalle d'actualisation du script est trop court puisque les déclencheurs se mettent en route ce qui peut créer un déni de service sur le serveur **Zabbix**.

Pour corriger cette défaillance technique, nous avons allongé cet intervalle à 86400 secondes (= 24 heures) par rapport au décalage de temps potentiel du serveur **Prometheus**.