Story: https://issues.redhat.com/browse/CNF-2344

PTP-chronyd interoperability

Context

DU nodes are synchronized to a high precision time source using PTP protocol.
PTP protocol is implemented by a set of daemons deployed and controlled by the Openshift PTP operator on the designated nodes. PTP operator is installed on the cluster after it is deployed (AKA “Day 2”)
All Openshift nodes are deployed by default with the “chronyd” daemon, which provides NTP time synchronization.
Node time synchronization is essential to facilitate TLS handshakes, timestamping of fault / performance information, events and logs.
NTP and PTP don’t work together smoothly, since they both update the same system time reference. If operated together, intermittent jumps occur to the system time.
There are solutions that allow PTP and NTP to work together, but they haven't been verified to work with the Telco PTP profile and provide the performance required for DU workloads.
To overcome PTP-NTP interoperability issue, current practice is to disable NTP daemon (chronyd) on the nodes that PTP daemon is deployed to. This workaround has a number of caveats:

- Chronyd is disabled on “Day 2” using "Machineconfig object. This implies the node reboot - not acceptable from the deployment time perspective.
- The chronyd daemon can’t be disabled on the node installation phase (“Day 0”), since this would leave the node without any time reference (and without TLS).
- If PTP sync is lost for a long time, the clock would drift (~50 ppm) and eventually leave the node not suitable for TLS communications.

Requirements

1. The chronyd daemon must be configured on “Day 0” to avoid node rebooting
2. The configuration of “chronyd” must be dynamic and take into account actual PTP synchronization condition, so the node is always synced
    a. To PTP by default
    b. To NTP, if PTP is not synced

Design

Logic

Every SYNC_CHECK_PERIOD check:

- If PTP_NOT_SYNCHRONIZED condition is met for a time period SYNC_AGE or since reboot, ensure chronyd is running. Else, ensure chronyd is not running.

Definitions

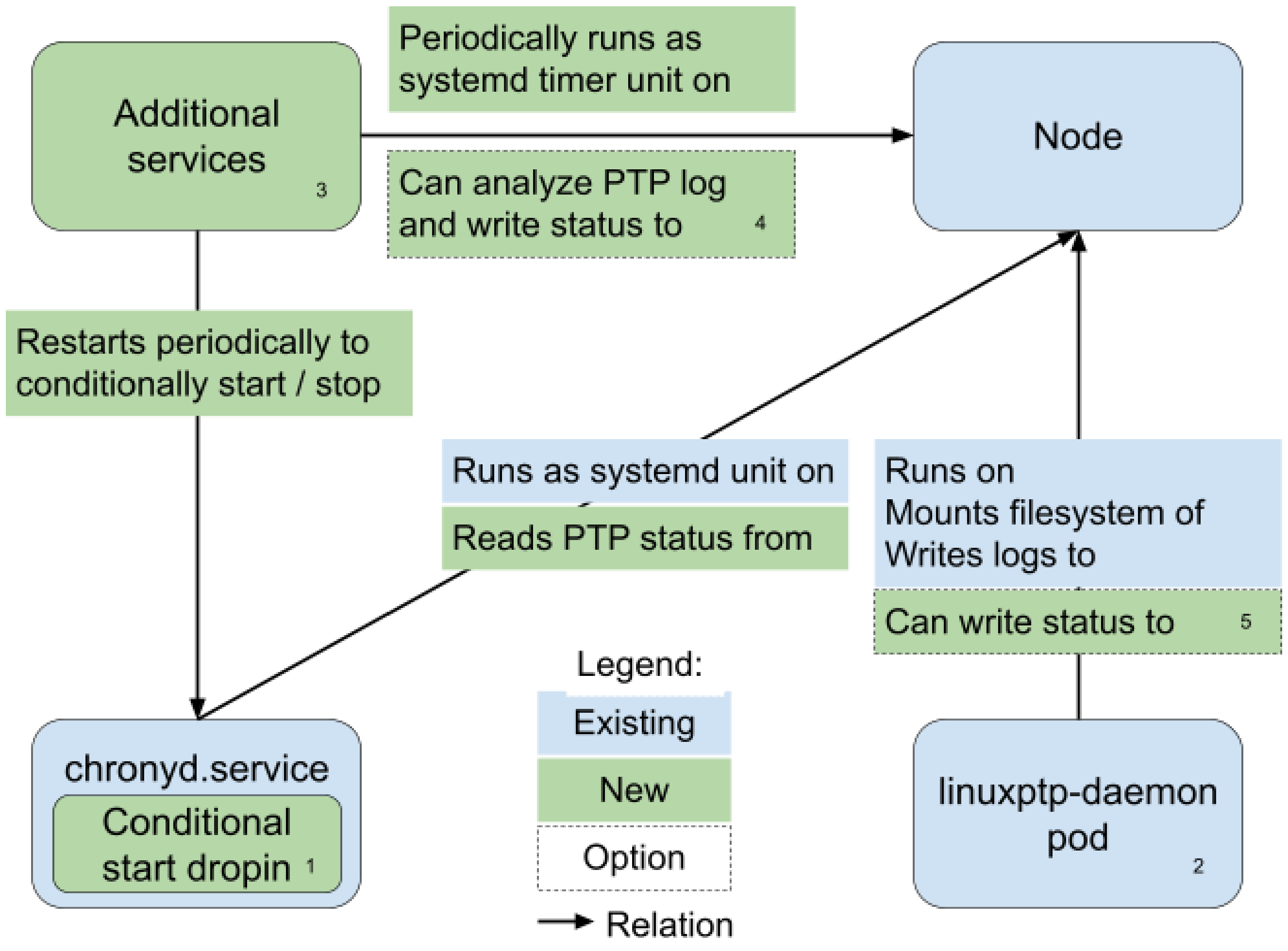
SYNC_CHECK_PERIOD	5 minutes	If too long, might cause long PTP stabilization time (PTP and chronyd working together) If too short - waste of resources
PTP_NOT_SYNCHRONIZED	ptp4l offset is greater than 100 microsecond, or there is no known ptp4l offset available (e.g. when PTP daemon is not installed, misconfigured or in servo state s0)	We don't want to jump into NTP when the PTP network is poorly performing, so it is 5000 times more than normal sync threshold.
PTP_OFFSET_THRESHOLD	20% of PTP_NOT_SYNCHRONIZED	Prevents oscillating around PTP_NOT_SYNCHRONIZED threshold
SYNC_AGE	10 minutes Difference between the current time and the time last PTP sync was achieved	To prevent bouncing into NTP

The above parameters together will ensure chronyd is turned off fast if there is PTP sync, but will be turned on only if PTP sync is absent for a long time.

Assumptions

1. The chronyd daemon can be restarted every SYNC_CHECK_PERIOD without negative impact on NTP synchronization quality

Entities and relationships



1. chronyd.service - existing systemd unit responsible for the NTP daemon. Can be made conditional using `ConditionPathExists=` directive, for example:

```
[Unit]
Description=NTP client/server
...
ConditionPathExists=!/var/run/ptp/insync
```

   Note: the above insert is for illustration only. In production, unit file additions are better provided by drop-ins.
   The directive above will allow chronyd start only if the `/var/run/ptp/insync` file does not exist. However, there are two constraints of systemd condition checking mechanism:
   - Conditions are checked only when the unit starts, i.e. if the unit is already running, it will not stop if the above file suddenly appears in the defined location.
   - If not started due to the failed condition check, the service won't try starting again.

   To address the above constraints, “Additional services” entity (3) is introduced.
2. linuxptp-daemon pod is deployed on the designated nodes by openshift-ptp operator. It is responsible for running the ptp4l and phc2sys daemons and providing them with the runtime parameters configured through the `ptpconfigs` custom resource. The

`linuxptp-daemon-container` mounts several node directories, most notable [for option (5)] is `/var/run/ptp` used to create socket units. In addition, linuxptp writes all daemon logs to `/var/log/pods/<autogenerated-names-with-specific-prefix>` using standard `glog` facilities.
These properties of linuxptp pod allow two options for a program running on the node to obtain the PTP synchronization status:
   - Through PTP log scraping and analysis (4)
   - Through logic implemented in the linuxptp-daemon itself (5)
3. Additional services are required to implement the periodical chronyd restarts due to the constraints outlined in (1). One of the possible implementations - a combination of systemd timer unit and one-shot service unit:

   chronyd-restart.timer

   ```
   [Unit]
   Description=Restart chronyd periodically

   [Timer]
   OnCalendar=*-*-* *:*/5:00  # every 5 minutes

   [Install]
   WantedBy=timers.target
   ```

   chronyd-restart.service

   ```
   [Unit]
   Description=Restart chronyd

   [Service]
   Type=oneshot
   ExecStart=/usr/bin/systemctl restart chronyd.service
   ```
   Note: the above inserts are for illustration only. In production, time constants and other variables are better provided by systemd environment drop-ins.

4. This option allows the required functionality to be provided without any modification to PTP operator or linuxptp-daemon. Here the **chronyd-restart.service** `ExecStart` directive should be replaced by a **script** that can
   - attain the ptp sync status from PTP log
   - create or delete the file chronyd ConditionPathExists directive is checking
   - restart chronyd.service
5. This option implies that the linuxptp-daemon-container code is modified to
   - Receive PTP_NOT_SYNCHRONIZED constant through the pod environment variables (that means ptp-operator repository should also be modified)
   - Periodically check ptp synchronization status
   - Create or delete the that file chronyd ConditionPathExists directive is checking

Implementation options comparison

Both options below provide the same set of capabilities.

Aspects	Option 4 - scraping PTP log	Option 5 - changes to linuxptp-daemon
Simplicity	Simple ++	Simple -
Parameters	All parameters are defined through the machineconfig	Some parameters must be defined through the operator code, i.e definitions of synchronized VS unsynchronized
Locality	All changes can be done in one PR in one repo (ztp)	The changes must be done in three repos 1. ptp-operator - to provide pod parameters through the environment variables 2. linuxptp-daemon - to implement the evaluation of sync and status file touch / delete 3. ztp - to implement chronyd conditions and restart logic
Estimated Time-to-market	1week	1month