Tuned: helper for system tuning

Jaroslav Škarvada <jskarvad@redhat.com>

System tuning

- sysctl
- sysfs
- various configs (usually in /etc)
- various tools (ethtool, hdparm, taskset, …)
- boot parameters (elevator, nohz, isolcpus, …)
- services / systemd units
- hotplug (udev events processing)

Usually handled by ad-hoc scripting

Problems

- Maintainability
  - Various SW / kernels, HW / architectures.
  - Changes in API / interface of tools / helpers.
  - Maintainers leaving.
- Verification
  - Is the tuning correctly applied?
  - No interference with other SW over time?
- Roll back
  - How to return back without reinstall / reboot?

Tuned

- Plug-in architecture
- Tuning is centralized in profiles
  - Inheritance support, tree like hierarchy
  - Factory / user profiles
- Roll back support
- Hotplug support
- Verification
- HW / system detection for auto configuration
- CLI, D-Bus control for integration (Cockpit)
- **Installed and enabled in RHEL**

Basic control

- tuned-adm

```
# systemctl start tuned
# systemctl enable tuned
# tuned-adm list
# tuned-adm profile throughput-performance
# tuned-adm active
# tuned-adm verify [-i]
```

- D-Bus control

```
# dbus-send --system --print-reply
--type=method_call
--dest='com.redhat.tuned' '/Tuned'
com.redhat.tuned.control.active_profile
```

Factory / upstream profiles

- For general goals:
  - throughput-performance
  - latency-performance, realtime
  - powersave
  - balanced
- For various products:
  - SAP (sap-hana, sap-netweaver, ...)
  - MS SQL Server (mssql)
  - Oracle RDBMS (oracle)

  ...

Recommended tuning

Knowledge base articles

Profiles location

- **Factory / system profiles**
  - *`/usr/lib/tuned/PROFILE_1`*
    `/usr/lib/tuned/PROFILE_2`
    `...`
  - Do not directly edit
    - Copy or override
    - Can have user editable config in *`/etc/tuned`*
  - Provided by distro or 3rd party packages
- **Custom / user profiles**
  - *`/etc/tuned/PROFILE_1`*
    `/etc/tuned/PROFILE_2`
    `...`
  - User editable
  - Takes precedence

Profile structure

PROFILE_NAME/tuned.conf :

```
[main]
summary=My profile for testing
description= My profile is cool :) ...
```

More verbose form

```
[disk]
readahead=4096
```

```
[sysctl]
vm.swappiness=5
```

```
[disk]
type=disk
devices=*
readahead=4096
```

Glob

Plugins

Multiple instances of the plugin


```
[main]

[disk-system]
type=disk
devices=sda1
readahead=>8192

[disk-data]
type=disk
devices=sda2
readahead=4096

[disk-other]
type=disk
devices=!sda1, !sda2
readahead=2048
```

Instance name - arbitrary string

Plugin to use

Device name(s) from udev for plugin instance to handle

Tuning

Conditional change operator

Udev regex matching


```
[main]

[disks-samsung]
type=disk
devices_udev_regex=ID_MODEL=SAMSUNG.*
readahead=8192 sectors
elevator=deadline

[disks-ssd]
type=disk
devices_udev_regex=ID_ATA_ROTATION_RATE_RPM=0
readahead=4096 sectors
```

Regex matching arbitrary string

It can match anything from the:

```
# udevadm info --query=property -n /dev/sda
```

Override / chain profiles


```
[main]
summary=My overridden profile
include=throughput-performance

[cpu]
governor=userspace

[disk]
enabled=0

[sysctl]
replace=1
vm.dirty_ratio=20
```

Take this profile

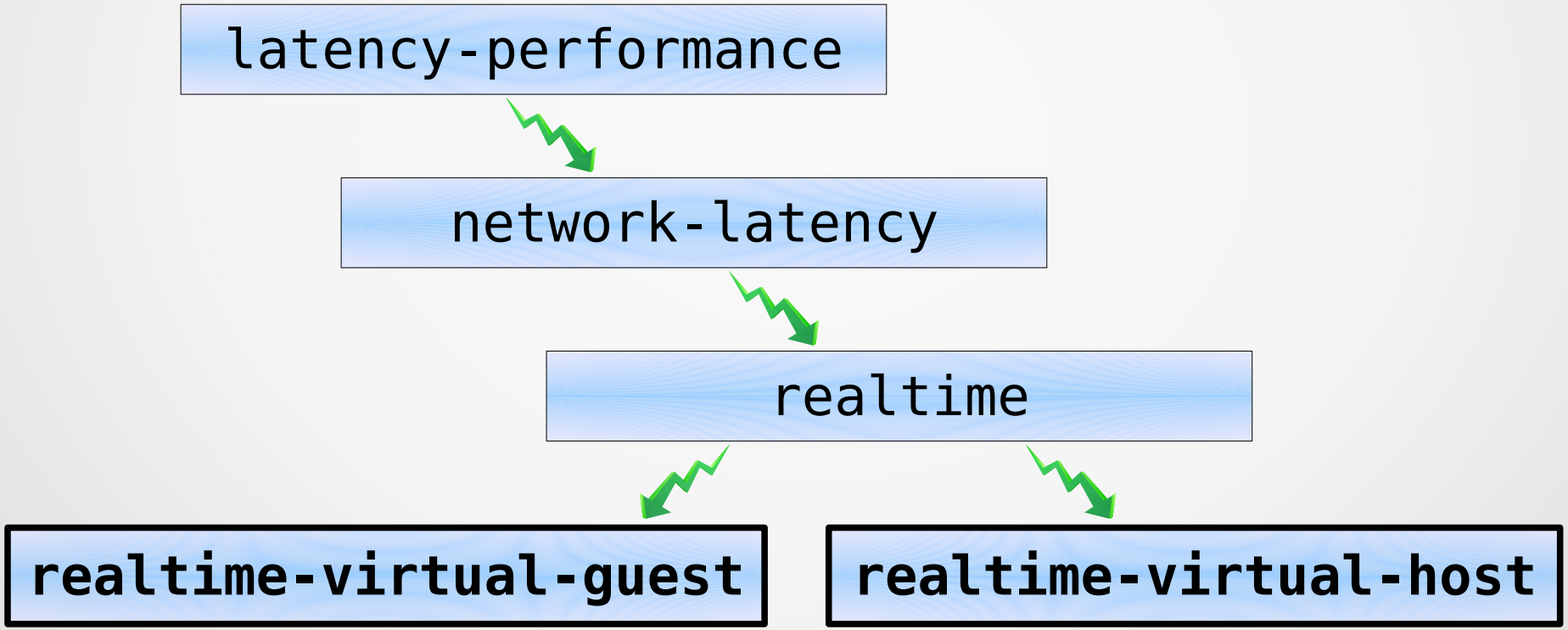
Change just governor, all other previously defined properties remains

Disable disk plugin

Clear all previously defined properties, use just mine definition

Profiles chaining

- Can create new specialized profiles from generic
- Example RPM packages *tuned-profiles-nfv*:*



System profiles customization

- Copy & edit, example for *powersave* profile:
  - Can miss system profile change / update
  - */etc/tuned* takes precedence if same name

```
# cp -r /usr/lib/tuned/powersave /etc/tuned
# vim /etc/tuned/powersave/tuned.conf
```

- or create new profile & override:

```
# mkdir /etc/tuned/my-powersave
# vim /etc/tuned/my-powersave/tuned.conf
```

Unique name not needed, but better

```
[main]
include=powersave

# customize by overrides
...
```

Preferred way

Abstraction layer – example 1

- Tuned profile:

```
[vm]
transparent_hugepage=always
```

- Red Hat Enterprise Linux 6:

```
echo "always" > /sys/kernel/mm/
redhat_transparent_hugepage/enabled
```

- Red Hat Enterprise Linux 7:

```
echo "always" > /sys/kernel/mm/
transparent_hugepage/enabled
```

Abstraction layer – example 2

- **Tuned profile:**

```
[bootloader]
cmdline=isolcpus=1
```

- **Manually:**
  - BLS?
    - Edit grubenv / patch entries in */boot/loader/entries*
  - GRUB2?
    - Patch GRUB_CMDLINE_LINUX in */etc/default/grub*
    - EFI? Legacy?
      - patch */etc/grub2[-efi].cfg* or
      - `grub2-mkconfig -o /etc/grub2[-efi].cfg`

Variables / parameters

- **Profile:**

Tuned profile

```
[main]

[variables]
include=/etc/tuned/my-profile-variables.conf

[bootloader]
cmdline=isolcpus=${isolated_cores}
```

- **User editable variables:**

/etc/tuned/my-profile.conf

```
# Cores excluded
# from the kernel load
# balancing
isolated_cores=1
```

Functions

- Pluginable, some examples:

```
[variables]
include=/etc/tuned/my-profile-variables.conf
cores=${isolated_cores}

[bootloader]
cmdline=isolcpus=${f:cpulist_invert:${cores}}

[disk]
readahead=${f:exec:/usr/libexec/calc-ra}
```

Complement:
online CPUs - cores

Execute external command,
substitute result

Dynamic tuning

- Some plugins can do dynamic tuning:
  - Monitor various performance counters at runtime (CPU load, disk load, network load, ...)
  - Change various system settings accordingly
  - Experimental feature
- Disabled in Red Hat Enterprise Linux
  - To have predictable performance

```
/etc/tuned/tuned-main.conf

dynamic_tuning = 0
```

Auto configuration

- Check system and preset Tuned profile according to predefined rules:

```
# tuned-adm auto_profile
```

- Just show what's recommended:

```
# tuned-adm recommend
```

- Drop your rules into */etc/tuned/recommend.d/*

FILENAME.conf

```
[throughput-performance]
virt=
system=.*(computenode|server).*

[virtual-guest]
virt=.+

[balanced]
```

No-daemon mode

- Consumes no CPU / RAM
- One shot – starts, do it's job, exits
- Preferred for embedded & low resources systems

```
/etc/tuned/tuned-main.conf

daemon = 0
```

- Less functionality
  - No D-Bus control
  - No hotplug support yet
  - No tuning of newly created processes
  - No dynamic tuning
  - No roll-back yet

Todo

- Better documentation
  - Reference manual (auto-generated)
- Better no-daemon mode
  - More functions supported in this mode
- Simulation mode
  - Show what will be set by the profile
- Support for containers
- And much much more :)

Conclusion

- Give Tuned a try
  - **Installed and enabled by default in Red Hat Enterprise Linux 7**
  - Available in other distros, e.g. Fedora, CentOS, openSUSE, Arch Linux, Debian, ...
- If your project needs specific tuning, consider using Tuned and writing profile
  - If Tuned miss function you need, report upstream
- Post useful profiles upstream
  - We can maintain it for you
- Post patches & PRs, report bugs :)

Contacts

https://tuned-project.org/

power-management@lists.fedoraproject.org

jskarvad@redhat.com

olysonek@redhat.com

Thank you.

PowerTOP integration

- Install:

```
# dnf install tuned-utils powertop
```

- Create profile from PowerTOP recommendations & merge with your current profile:

```
# powertop2tuned my-profile
```

- Enable what you need by uncommenting lines:

```
# vim /etc/tuned/my-profile/tuned.conf
```

- Activate:

```
# tuned-adm profile my-profile
```

PM QoS

- Taken into account by cpuidle kernel driver
- It can be used to limit CPU C states transition
  - Deeper C state → higher latency

```
# cat /sys/devices/system/cpu/cpuX/cpuidle/stateY/latency
```

```
[main]

[cpu]
governor=performance
force_latency=10
```

No more than 10µs

Initrd overlays

- It can be used to add / edit content of initrd
- No need to regenerate existing initrd

```
[main]

[bootloader]
initrd_add_dir=${i:PROFILE_DIR}/initrd
```

```
# mkdir -p initrd/etc
# echo "hello world" > initrd/etc/test
```

Processes tuning

- Tuning of newly created processes through the kernel perf infrastructure
- Match process name by regex, tune:
  - scheduler policy
  - sheduler priority
  - core affinity
- Syntax inspired by the *rtctl* tool:

```
[scheduler]
group.ksoftirqd=0:f:2:*:ksoftirqd.*
group.rcub=0:f:4:*:rcub.*
```

rule priority FIFO policy Run everywhere