BOSTON, MA
JUNE 23-26, 2015

Demystifying systemd

2015 Edition

Ben Breard
Senior Solutions Architect, Red Hat

Lennart Poettering
Senior Software Engineer, Red Hat

Agenda

- Concepts & Basic Usage
- Modifying Units
- Resource Management
- Converting init scripts
- The Journal
- Sneak peek at what's coming in RHEL 7.2

Life Beyond Init
Concepts & Basic Usage

systemd

- The default init system for most Linux distributions
- Controls “units” rather than just daemons
- Handles dependency between units.
- Tracks processes with service information
  - Services are owned by a cgroup.
  - Simple to configure “SLAs” for CPU, Memory, and IO
- Properly kill daemons
- Minimal boot times
- Debuggability – no early boot messages are lost
- Easy to learn and backwards compatible

systemd Units

```
foo.service
bar.socket
baz.device
qux.mount
waldo.automount
thud.swap
```

```
grunt.target
snork.timer
grault.path
garply.snapshot
pizza.slice
tele.scope
```

systemd Units: httpd.service

```
[Unit]
Description=The Apache HTTP Server
After=remote-fs.target nss-lookup.target

[Service]
Type=notify
EnvironmentFile=/etc/sysconfig/httpd
ExecStart=/usr/sbin/httpd $OPTIONS -DFOREGROUND
ExecReload=/usr/sbin/httpd $OPTIONS -k graceful
ExecStop=/usr/sbin/httpd $OPTIONS -k graceful-stop
PrivateTmp=true

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

*Comments were removed for readability

systemd Units: Locations

- Maintainer: /usr/lib/systemd/system
- Administrator: /etc/systemd/system
- Non-persistent, runtime: /run/systemd/system

Note: unit files in /etc take precedence over /usr

Managing Services: Start/Stop

Init

- `service httpd {start,stop,restart,reload}`

systemd

- `Systemctl {start,stop,restart,reload} httpd.service`

Managing Services: Start/Stop

- Glob units to work with multiple services
- `systemctl restart httpd mariadb`
- When the unit “type” isn't specified, .service is assumed.
  - `systemctl start httpd == systemctl start httpd.service`
- Make life easy and install shell completion
  - `yum install bash-completion`
  - `systemctl [tab] [tab]`
  - Add bash-completion to your SOE and minimal kickstarts
- Connect directly to remote hosts
  - `systemctl -H [hostname] restart httpd`

Managing Services: Status

Init

- `service httpd status`

systemd

- `systemctl status httpd`

Tip: pass -l if the logs are cutoff

Managing Services: Status

```
root@host158:~
File  Edit  View  Search  Terminal  Help
[root@host158 ~]# systemctl status httpd
httpd.service - The Apache HTTP Server
   Loaded: loaded (/usr/lib/systemd/system/httpd.service; enabled)
   Active: active (running) since Fri 2013-08-09 09:22:25 CDT; 12s ago
  Process: 890 ExecStop=/usr/sbin/httpd $OPTIONS -k graceful-stop (code=exited, status
=0/SUCCESS)
 Main PID: 893 (httpd)
   Status: "Total requests: 0; Current requests/sec: 0; Current traffic:   0 B/sec"
   CGroup: name=systemd:/system/httpd.service
           ├─893 /usr/sbin/httpd -DFOREGROUND
           ├─894 /usr/sbin/httpd -DFOREGROUND
           ├─895 /usr/sbin/httpd -DFOREGROUND
           ├─896 /usr/sbin/httpd -DFOREGROUND
           ├─897 /usr/sbin/httpd -DFOREGROUND
           └─898 /usr/sbin/httpd -DFOREGROUND

Aug 09 09:22:23 host158.local systemd[1]: Starting The Apache HTTP Server...
Aug 09 09:22:25 host158.local systemd[1]: Started The Apache HTTP Server.
[root@host158 ~]#
```

Why wasn't this in RHEL 5?!

Managing Services: Status

- List loaded services:
  - `systemctl -t service`
- List installed services:
  - `systemctl list-unit-files -t service` (like chkconfig –list)
- Check for services in failed state:
  - `systemctl --state failed`

Managing Services: Enable/Disable

Init

- `chkconfig httpd {on,off}`

systemd

- `systemctl {enable, disable} httpd`

Tip: Globing units will clean up your kickstarts

```
-systemctl enable httpd mariadb ntpd lm_sensors [etc]
```

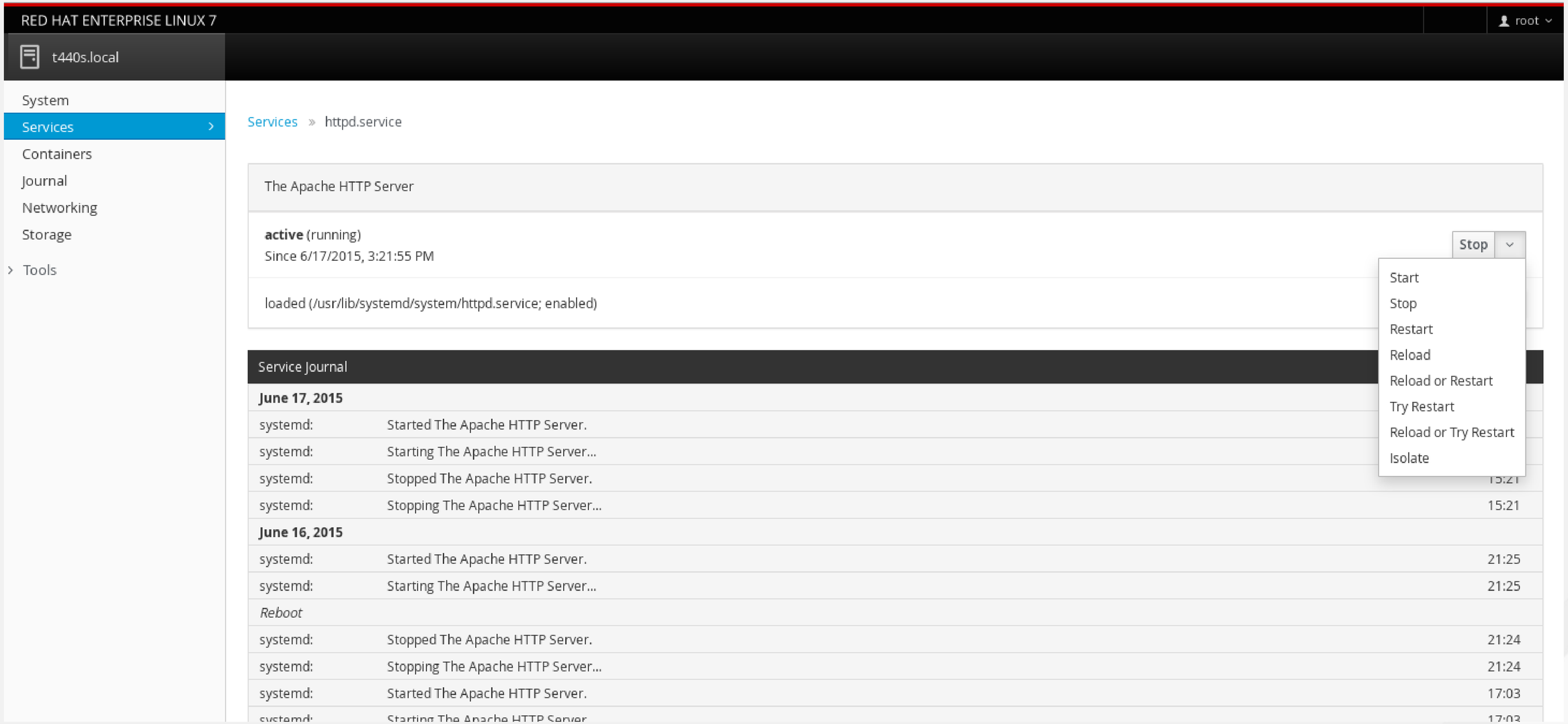
Targets == Runlevels

- “Runlevels” are exposed as target units
- More meaningful names:
  - multi-user.target vs. runlevel3
  - graphical.target vs. runlevel5
- View the default target: `systemctl get-default`
- Set the default target: `systemctl set-default [target]`
- Change at run-time: `systemctl isolate [target]`

Note: /etc/inittab is no longer used.

Too Many Commands!! I want GUI NOW!!!

Cockpit is now in the RHEL Extras repo



Sockets

tftp.socket

```
[Unit]
Description=Tftp Server
Activation Socket

[Socket]
ListenDatagram=69

[Install]
WantedBy=sockets.target
```

tftp.service

```
[Unit]
Description=Tftp Server

[Service]
ExecStart=/usr/sbin/in.tftpd -s
/var/lib/tftpboot
StandardInput=socket
```

```
man systemd.socket
```

Sockets

cockpit.socket

```
[Unit]
Description=Cockpit Web Server Socket
Documentation=man:cockpit-ws(8)

[Socket]
ListenStream=9090

[Install]
WantedBy=sockets.target
```

cockpit.service

```
[Unit]
Description=Cockpit Web Server
Documentation=man:cockpit-ws(8)

[Service]
ExecStartPre=/usr/sbin/remotectl cert --ensure --user=root --group=cockpit-ws
ExecStart=/usr/libexec/cockpit-ws
PermissionsStartOnly=true
User=cockpit-ws
Group=cockpit-ws
```

Timers

fstrim.timer

```
[Unit]
Description=Discard unused
blocks once a week

[Timer]
OnStartupSec=10min
OnCalendar=weekly
AccuracySec=1h
Persistent=true
[Install]
WantedBy=multi-user.target
```

fstrim.service

```
[Unit]
Description=Discard unused blocks

[Service]
Type=oneshot
ExecStart=/usr/sbin/fstrim /
```

```
man systemd.timer
```

Customizing Units

What's Available?

- List a unit's properties:
  - `systemctl show --all httpd`
- Query a single property:
  - `systemctl show -p Restart httpd`
  - `Restart=no`
- Helpful man files: systemd.exec and systemd.service
  - Restart, Nice, CPUAffinity, OOMScoreAdjust, LimitNOFILE, etc

Disclaimer: just because you **can** configure something doesn't mean you **should**!

Customizing Units: Drop-ins

1) Create directory

- `mkdir /etc/systemd/system/[name.type.d]/`

2) Create drop-in

- `vim /etc/systemd/system/httpd.service.d/50-httpd.conf`

```
[Service]
Restart=always
CPUAffinity=0 1 2 3
OOMScoreAdjust=-1000
```

Remember the 'S' is capitalized

3) Notify systemd of the changes

- `systemctl daemon-reload`

Customizing Units: Drop-ins

root@host243:/etc/systemd/system/httpd.service.d

File Edit View Search Terminal Help

```
[root@host243 httpd.service.d]# systemctl status httpd
httpd.service - The Apache HTTP Server
   Loaded: loaded (/usr/lib/systemd/system/httpd.service; enabled)
  Drop-In: /etc/systemd/system/httpd.service.d
           └─50-httpd.conf
   Active: active (running) since Sun 2014-03-16 14:31:08 CDT; 2min 6s ago
  Process: 686 ExecStop=/bin/kill -WINCH ${MAINPID} (code=exited, status=0/SUCCESS)
 Main PID: 689 (httpd)
   Status: "Total requests: 15884; Current requests/sec: 133; Current traffic:  60KB/sec"
   CGroup: /system.slice/httpd.service
           ├─689 /usr/sbin/httpd -DFOREGROUND
           ├─691 /usr/sbin/httpd -DFOREGROUND
           ├─692 /usr/sbin/httpd -DFOREGROUND
           ├─693 /usr/sbin/httpd -DFOREGROUND
           ├─694 /usr/sbin/httpd -DFOREGROUND
           ├─695 /usr/sbin/httpd -DFOREGROUND
           └─715 /usr/sbin/httpd -DFOREGROUND

Mar 16 14:31:08 host243.local systemd[1]: Started The Apache HTTP Server.
```

Customizing Units: Drop-ins

- Safe to apply on running services
  - Note: some options will require a service restart to take effect
- Use `systemd-delta` to see what's been altered on a system

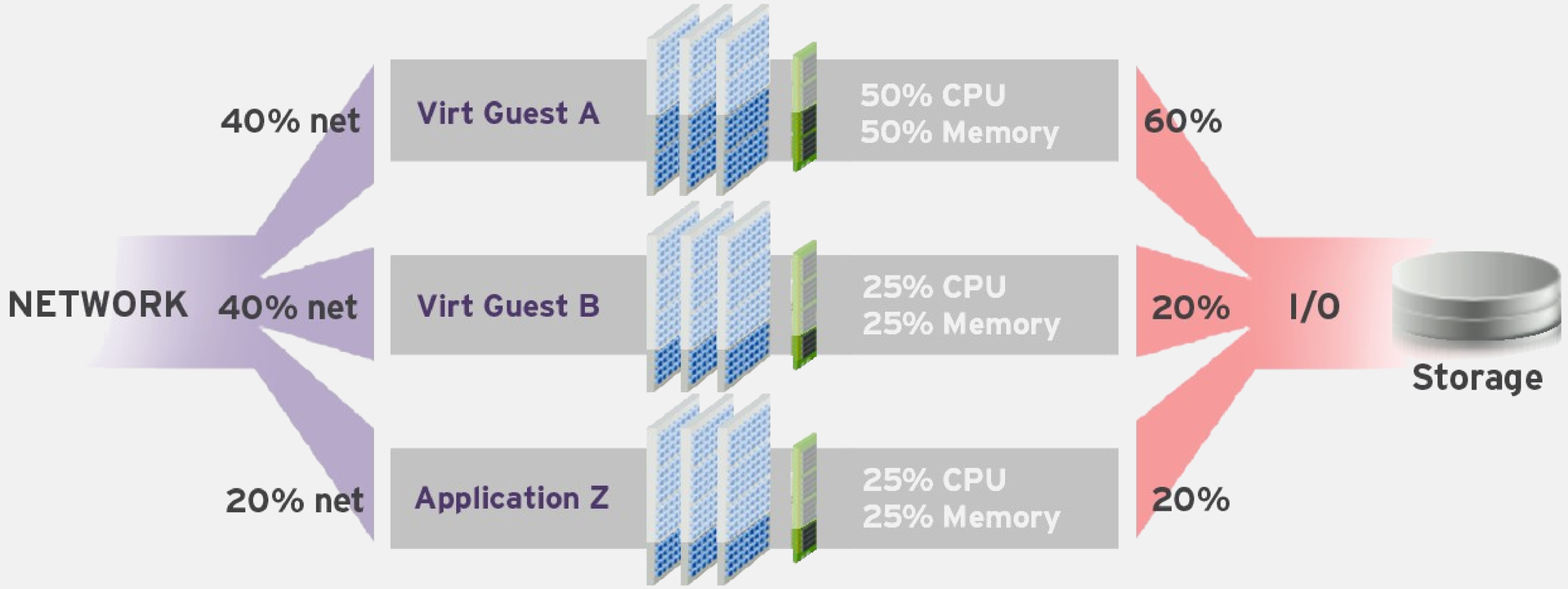
```
[EXTENDED]   /usr/lib/systemd/system/httpd.service → /etc/systemd/system/httpd.service.d/50-httpd.conf
[EXTENDED]   /usr/lib/systemd/system/httpd.service → /etc/systemd/system/httpd.service.d/90-CPUShares.conf
```

- Simple to use with configuration tools like Satellite, Puppet, etc.
- Simply delete the drop-in to revert to defaults.
- Don't forget `systemctl daemon-reload` when modifying units.

Resource Management
slices, scopes, services

Control Groups Made Simple

Resource Management with cgroups can reduce contention and improve throughput and predictability

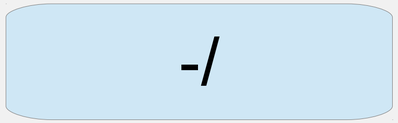


Ummm.... Yeah... [JAVA app], I'm going to need you to go ahead and use less Memory.

Slices, Scopes, Services

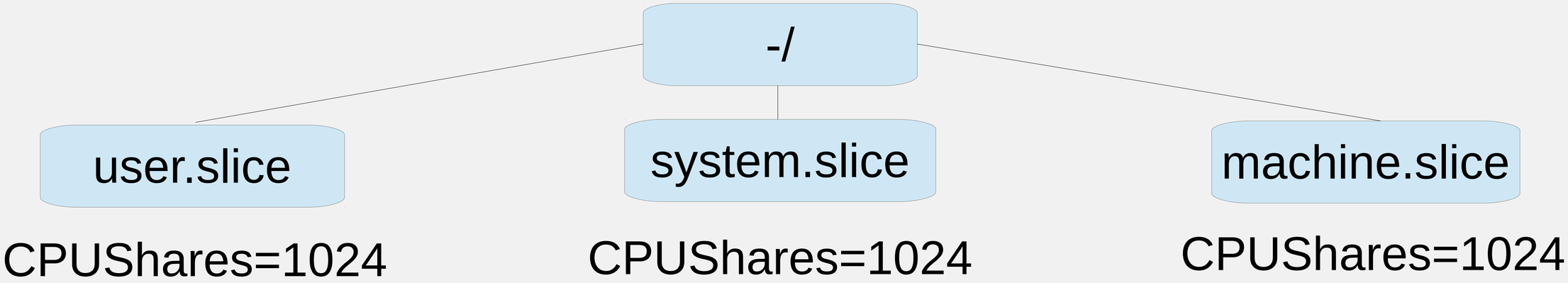
- **Slice** – Unit type for creating the cgroup hierarchy for resource management.
- **Scope** – Organizational unit that groups a services' worker processes.
- **Service** – Process or group of processes controlled by systemd

Understanding the Hierarchy



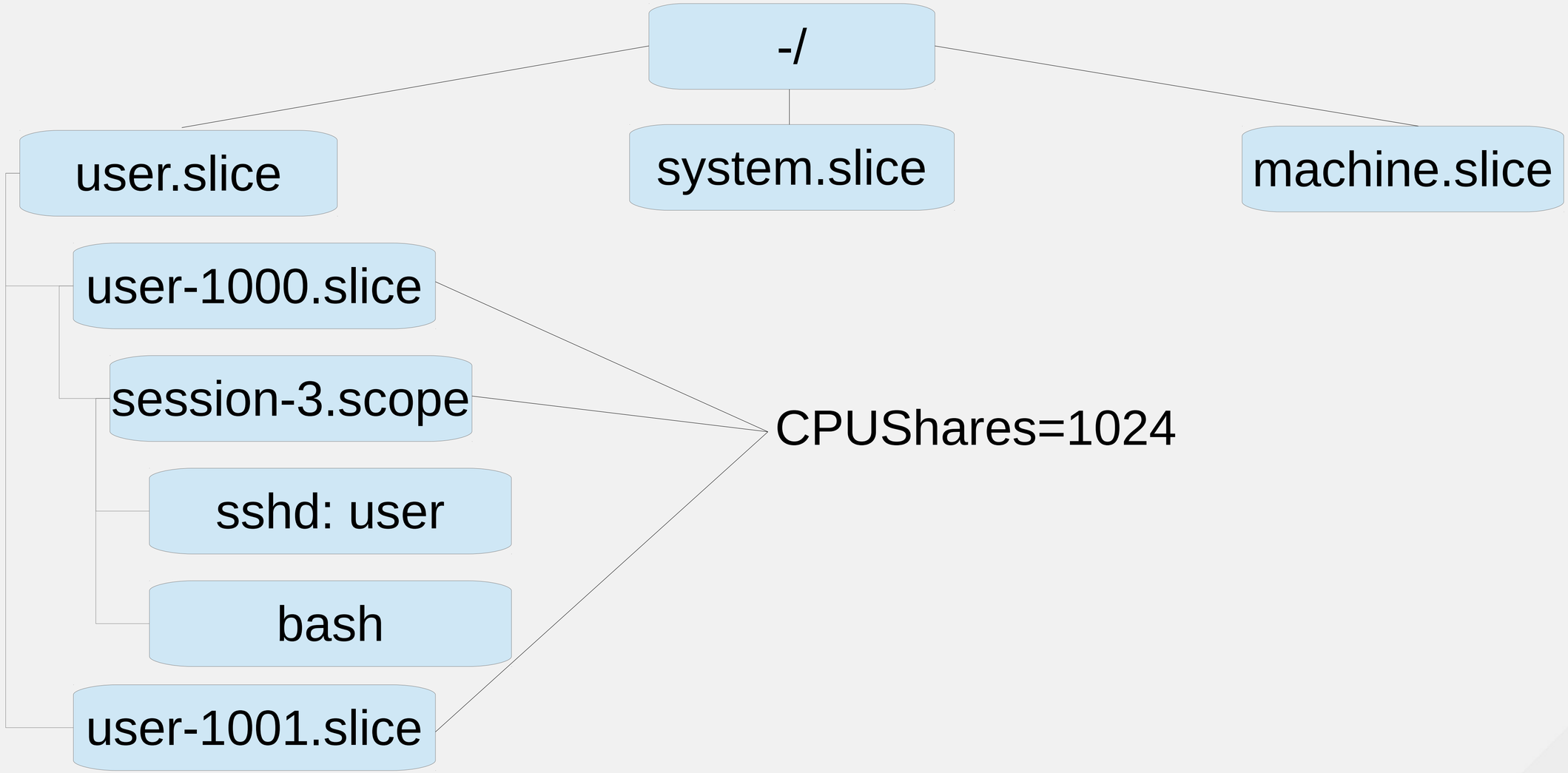
- systemd implements a standard, single-root hierarchy under /sys/fs/cgroup

Understanding the Hierarchy

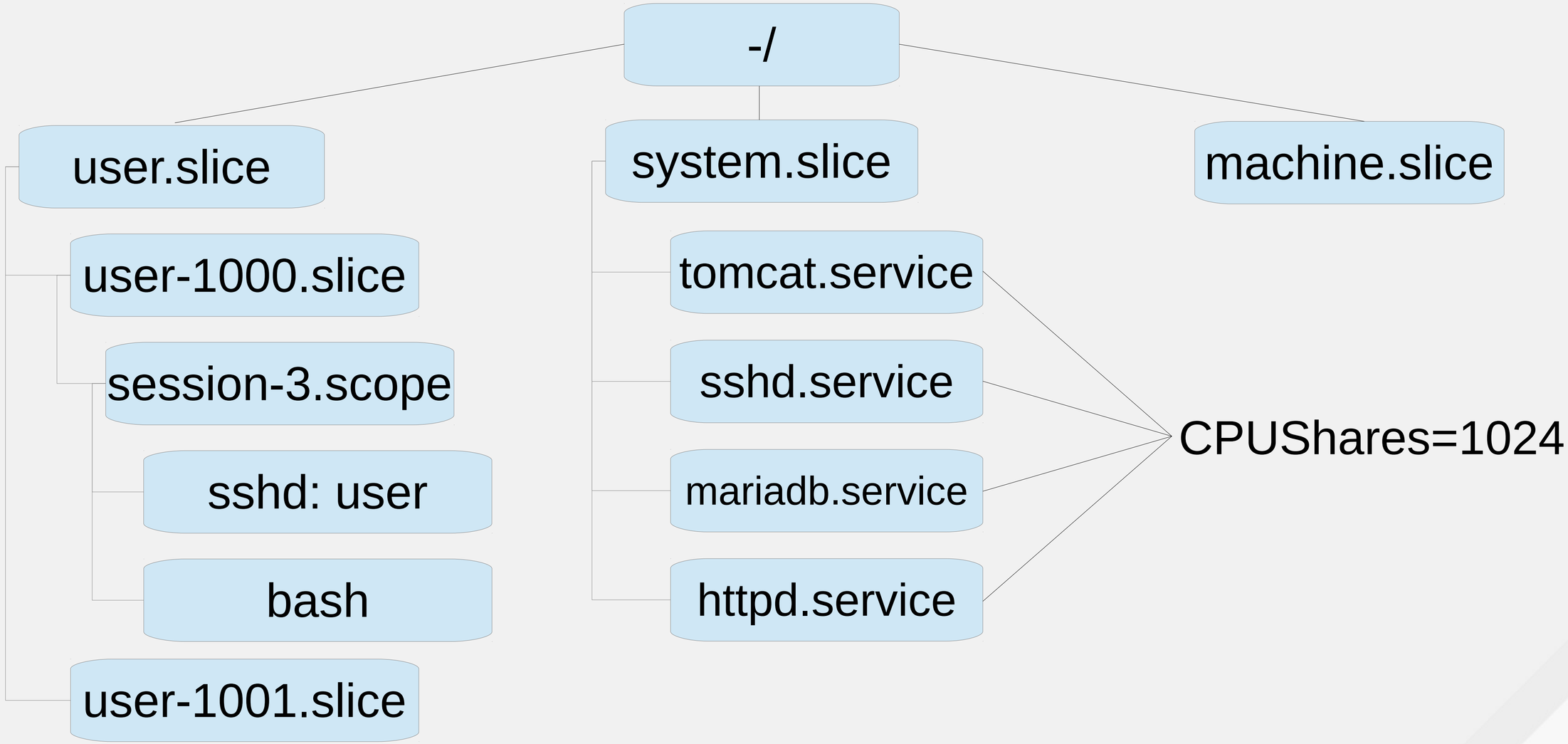


- Each slice gets equal CPU time on the scheduler.

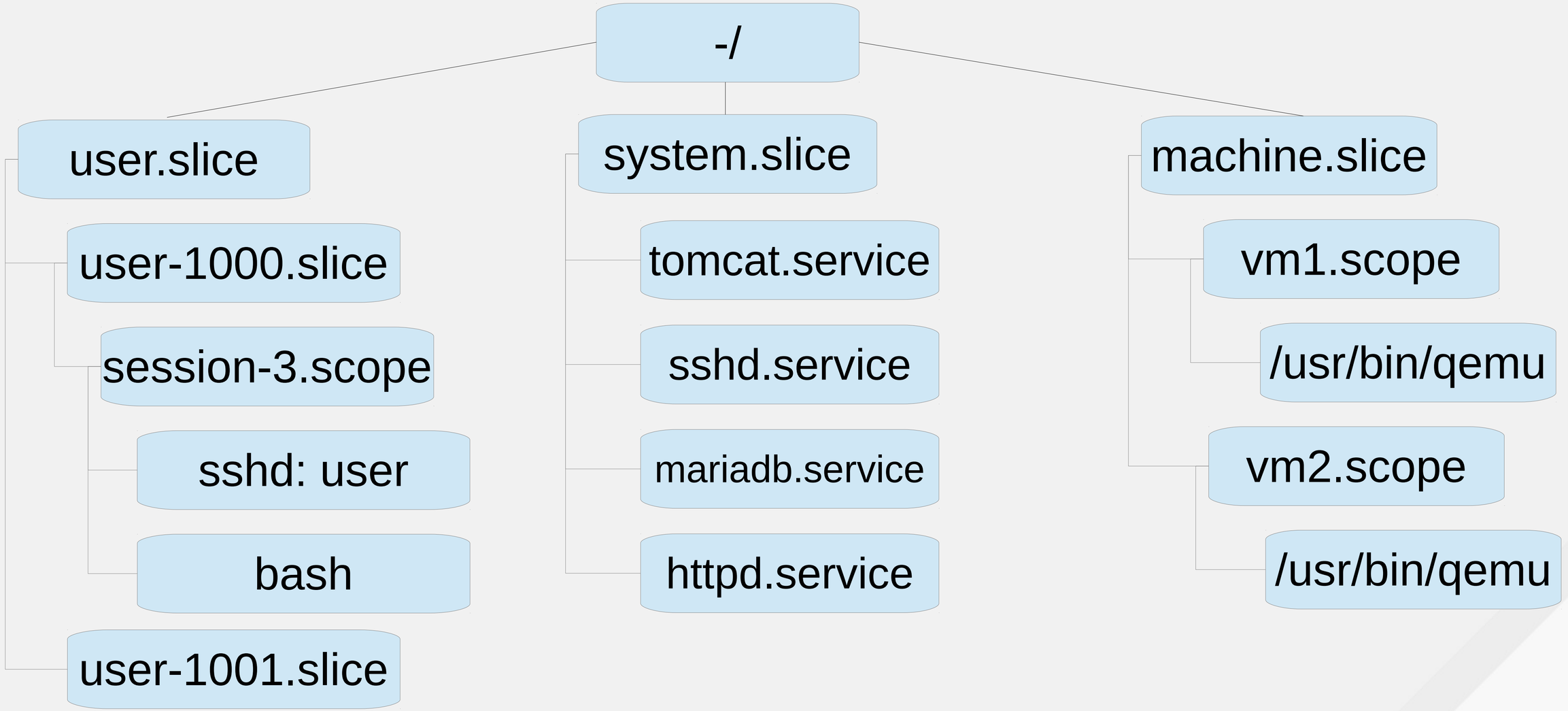
Understanding the Hierarchy



Understanding the Hierarchy



Understanding the Hierarchy



Resource Management – systemd-cgls

```
├─1 /usr/lib/systemd/systemd --switched-root --system --deserialize 22
├─machine.slice
│ ├─machine-qemu\x2drhel7.scope
│ │ └─17307 /usr/bin/qemu-system-x86_64 -machine accel=kvm -name rhel7 -S -machi
│ └─machine-qemu\x2dEAP6.scope
│   └─15290 /usr/bin/qemu-system-x86_64 -machine accel=kvm -name EAP6 -S -machin
├─user.slice
│ ├─user-0.slice
│ │ └─user@0.service
│ │   ├─3289 /usr/lib/systemd/systemd --user
│ │   └─3299 (sd-pam)
│ ├─user-1000.slice
│ │ ├─session-7.scope
│ │ │ ├─13655 gdm-session-worker [pam/gdm-password]
│ │ │ ├─13665 /usr/bin/gnome-keyring-daemon --daemonize --login
│ │ │ ├─13710 gnome-session
│ │ │ ├─13718 dbus-launch --sh-syntax --exit-with-session
│ │ │ ├─13719 /bin/dbus-daemon --fork --print-pid 4 --print-address 6 --session
│ │ │ ├─13784 /usr/libexec/gvfsd
│ │ │ ├─13788 /usr/libexec//gvfsd-fuse /run/user/1000/gvfs -f -o big_writes
│ │ │ ├─13879 /usr/libexec/at-spi-bus-launcher
│ │ │ ├─13883 /bin/dbus-daemon --config-file=/etc/at-spi2/accessibility.conf --n
│ │ │ ├─13887 /usr/libexec/at-spi2-registryd --use-gnome-session
lines 1-23
```

Resource Management – systemd-cgtop

Path	Tasks	%CPU	Memory	Input/s	Output/s
/	72	99.8	329.4M	-	-
/user.slice	20	49.1	-	-	-
/system.slice	16	49.1	287.2M	-	-
/system.slice/httpd.service	20	31.1	39.5M	-	-
/system.slice/mariadb.service	2	18.0	168.3M	0B	5.9M
/system.slice/NetworkManager.service	2	-	-	-	-
/system.slice/alsa-state.service	1	-	-	-	-
/system.slice/atd.service	1	-	-	-	-
/system.slice/auditd.service	1	-	-	-	-
/system.slice/chronyd.service	1	-	-	-	-
/system.slice/crond.service	1	-	-	-	-
/system.slice/dbus.service	1	-	-	-	-
/system.slice/libstoragemgmt.service	1	-	-	-	-
/system.slice/polkit.service	1	-	-	-	-
/system.slice/smartd.service	1	-	-	-	-
/system.slice/sshd.service	1	-	-	-	-
/system.slice/systemd-journald.service	1	-	-	-	-
/system.slice/systemd-logind.service	1	-	-	-	-
/system.slice/systemd-udevd.service	1	-	-	-	-
/user.slice/...0.slice/session-1.scope	2	-	-	-	-

Resource Management – Configuration

- Configure cgroup attributes:
  - `systemctl set-property --runtime httpd CPUShares=2048`
- Drop “--runtime” to persist:
  - `systemctl set-property httpd CPUShares=2048`
- Or place in the unit file:
  - `[Service]`
  - `CPUShares=2048`

http://0pointer.de/blog/projects/resources.html

Resource Management – CPU & MEM

- CPUAccounting=1 to enable
- CPUShares – default is 1024.
- Increase to assign more CPU to a service
  - e.g. CPUShares=1600

- MemoryAccounting=1 to enable
- MemoryLimit=
- Use K, M, G, T suffixes
  - MemoryLimit=1G

https://www.kernel.org/doc/Documentation/cgroups/memory.txt

https://www.kernel.org/doc/Documentation/scheduler/sched-design-CFS.txt

Resource Management - BlkIO

- BlockIOAccounting=1
- BlockIOWeight= assigns an IO weight to a specific service (requires CFQ)
  - Similar to CPU shares
  - Default is 1000
  - Range 10 – 1000
  - Can be defined per device (or mount point)
- BlockIOReadBandwidth & BlockIOWriteBandwidth
  - BlockIOWriteBandwith=/var/log 5M

https://www.kernel.org/doc/Documentation/cgroups/blkio-controller.txt

Converting Init Scripts
You can do it! It's easy!

Remember what init scripts look like?

/etc/init.d/httpd

```
. /etc/rc.d/init.d/functions
if [ -f /etc/sysconfig/httpd ]; then
        . /etc/sysconfig/httpd
fi
HTTPD_LANG=${HTTPD_LANG-"C"}
INITLOG_ARGS=""
apachectl=/usr/sbin/apachectl
httpd=${HTTPD-/usr/sbin/httpd}
prog=httpd
pidfile=${PIDFILE-/var/run/httpd/httpd.pid}
lockfile=${LOCKFILE-/var/lock/subsys/httpd}
RETVAL=0
STOP_TIMEOUT=${STOP_TIMEOUT-10}
start() {
        echo -n $"Starting $prog: "
        LANG=$HTTPD_LANG daemon --pidfile=${pidfile} $httpd $OPTIONS
        RETVAL=$?
        echo
        [ $RETVAL = 0 ] && touch ${lockfile}
        return $RETVAL
}
stop() {
        echo -n $"Stopping $prog: "
        killproc -p ${pidfile} -d ${STOP_TIMEOUT} $httpd
        RETVAL=$?
        echo
        [ $RETVAL = 0 ] && rm -f ${lockfile} ${pidfile}
}
```

From RHEL 6.4; comments removed

/etc/init.d/httpd – continued

```
reload() {
    echo -n $"Reloading $prog: "
    if ! LANG=$HTTPD_LANG $httpd $OPTIONS -t >&/dev/null; then
        RETVAL=6
        echo $"not reloading due to configuration syntax error"
        failure $"not reloading $httpd due to configuration syntax error"
    else
        LSB=1 killproc -p ${pidfile} $httpd -HUP
        RETVAL=$?
        if [ $RETVAL -eq 7 ]; then
            failure $"httpd shutdown"
        fi
    fi
    echo
}

case "$1" in
  start)
        start
        ;;
  stop)
        stop
        ;;
  status)
        status -p ${pidfile} $httpd
        RETVAL=$?
        ;;
```

/etc/init.d/httpd – continued

```
 restart)
        stop
        start
        ;;
  condrestart|try-restart)
        if status -p ${pidfile} $httpd >&/dev/null; then
                stop
                start
        fi
        ;;
  force-reload|reload)
        reload
        ;;
  graceful|help|configtest|fullstatus)
        $apachectl $@
        RETVAL=$?
        ;;
  *)
        echo $"Usage: $prog {start|stop|restart|condrestart|try-restart|force-reload|reload|status|fullstatus|graceful|help|configtest}"
        RETVAL=2
esac
exit $RETVAL
```

httpd.service

```
[Unit]
Description=The Apache HTTP Server
After=remote-fs.target nss-lookup.target

[Service]
Type=notify
EnvironmentFile=/etc/sysconfig/httpd
ExecStart=/usr/sbin/httpd $OPTIONS -DFOREGROUND
ExecReload=/usr/sbin/httpd $OPTIONS -k graceful
ExecStop=/usr/sbin/httpd $OPTIONS -k graceful-stop
PrivateTmp=true

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

*Comments were removed for readability

To be clear

- Systemd maintains 99% backwards compatibility with LSB compatible initscripts and the exceptions are well documented.
- While we do encourage everyone to convert legacy scripts to service unit files, it's not a requirement.
- Incompatibilities are listed here: http://www.freedesktop.org/wiki/Software/systemd/Incompatibilities/
- Converting SysV Init Scripts: http://0pointer.de/blog/projects/systemd-for-admins-3.html

Unit file layout – Custom application example

```
[Unit]
Description=Describe the daemon
After=network.target

[Service]
ExecStart=/usr/sbin/[myapp] -D
Type=forking
PIDFile=/var/run/myapp.pid

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

EAP Example

```
[Unit]
Description=JBoss Enterprise Application Platform
After=network.target
[Service]
User=jboss-as
Environment=JBOSS_USER=jboss-as
Environment=JBOSS_HOME=/usr/local/EAP-6.1.1/jboss-eap-6.1
Environment=JBOSS_CONSOLE_LOG=/var/log/jbossas/console.log
ExecStart=/usr/local/EAP-6.1.1/jboss-eap-6.1/bin/standalone.sh
PIDFile=/var/run/jboss-as/jboss-as-standalone.pid
SyslogIdentifier=jboss-as
LimitNOFILE=102642
Slice=jboss.slice
[Install]
WantedBy=multi-user.target
```

Note: If you don't define “Type=” it will be “simple” by default

http://0pointer.de/blog/projects/systemd-for-admins-3.html

The Journal

Journal

- Indexed
- Formatted
  - Errors in red
  - Warnings in bold
- Security
- Reliability
- Intelligently rotated

http://0pointer.de/blog/projects/journalctl.html

Journal

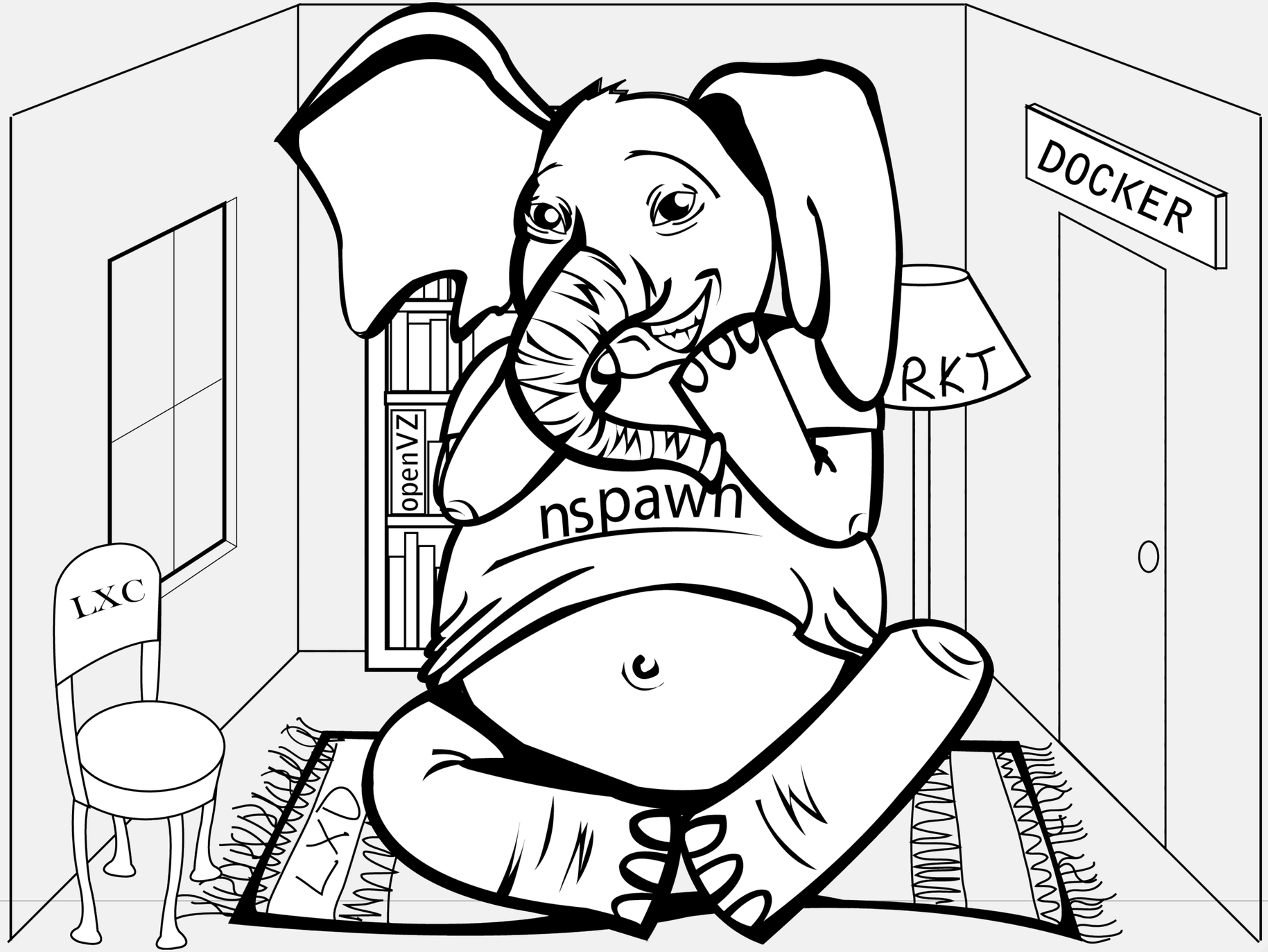
- Does not replace rsyslog in RHEL 7
  - rsyslog is enabled by default
- The journal is not persistent by default.
  - Enable persistence: `mkdir /var/log/journal`
- Stored in key-value pairs
  - `journalctl [tab] [tab]`
  - `Man 7 systemd.journal-fields`
- Collects event metadata along with the message
- Simple to filter
  - Interleave units, binaries, etc.

Journal – journalctl

root@host151:~

File Edit View Search Terminal Help

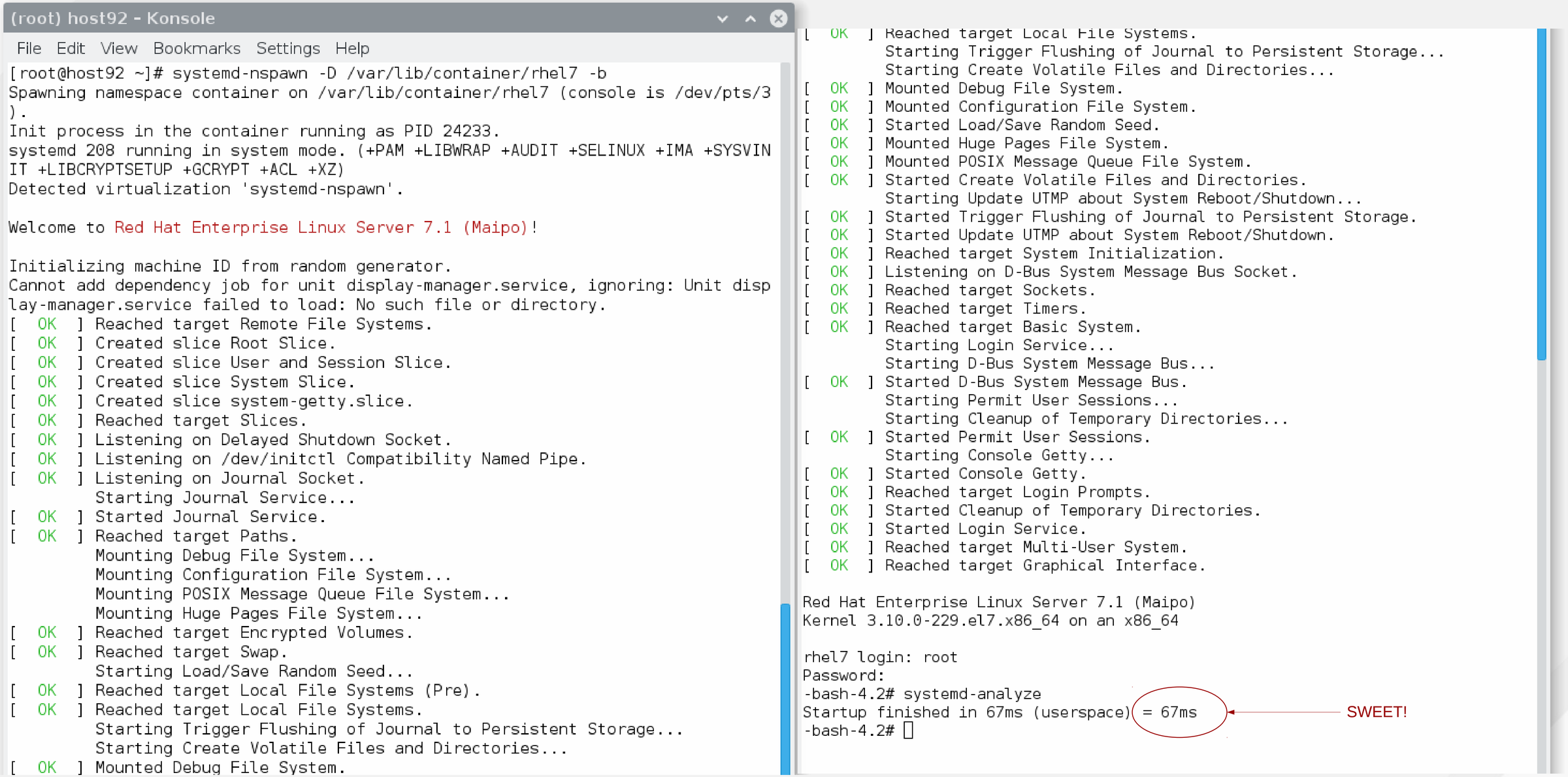
```
Oct 28 15:04:58 host151.local chronyd[329]: System clock wrong by -31.975399 seconds, adjustment
Oct 28 15:04:26 host151.local chronyd[329]: System clock was stepped by -31.975 seconds
Oct 28 15:04:26 host151.local systemd[1]: Time has been changed
Oct 28 15:04:52 host151.local systemd[1]: Starting Stop Read-Ahead Data Collection...
Oct 28 15:04:52 host151.local systemd[1]: Started Stop Read-Ahead Data Collection.
Oct 28 15:05:32 host151.local chronyd[329]: Selected source 174.133.168.194
Oct 28 15:06:08 host151.local sshd[2040]: Accepted password for root from 192.168.122.1 port 4512
Oct 28 15:06:08 host151.local systemd[1]: Starting user-0.slice.
Oct 28 15:06:08 host151.local systemd[1]: Created slice user-0.slice.
Oct 28 15:06:08 host151.local systemd[1]: Starting User Manager for 0...
Oct 28 15:06:08 host151.local systemd[1]: Starting Session 1 of user root.
Oct 28 15:06:08 host151.local systemd[1]: Started Session 1 of user root.
Oct 28 15:06:08 host151.local systemd-logind[322]: New session 1 of user root.
Oct 28 15:06:08 host151.local sshd[2040]: pam_unix(sshd:session): session opened for user root by
Oct 28 15:06:08 host151.local systemd[2044]: pam_unix(systemd-user:session): session opened for u
Oct 28 15:06:08 host151.local systemd[2044]: Failed to open private bus connection: Failed to con
Oct 28 15:06:08 host151.local systemd[2044]: Mounted /sys/kernel/config.
Oct 28 15:06:08 host151.local systemd[2044]: Stopped target Sound Card.
Oct 28 15:06:08 host151.local systemd[2044]: Starting Default.
Oct 28 15:06:08 host151.local systemd[2044]: Reached target Default.
Oct 28 15:06:08 host151.local systemd[2044]: Startup finished in 11ms.
Oct 28 15:06:08 host151.local systemd[1]: Started User Manager for 0.
lines 962-983/983 (END)
```



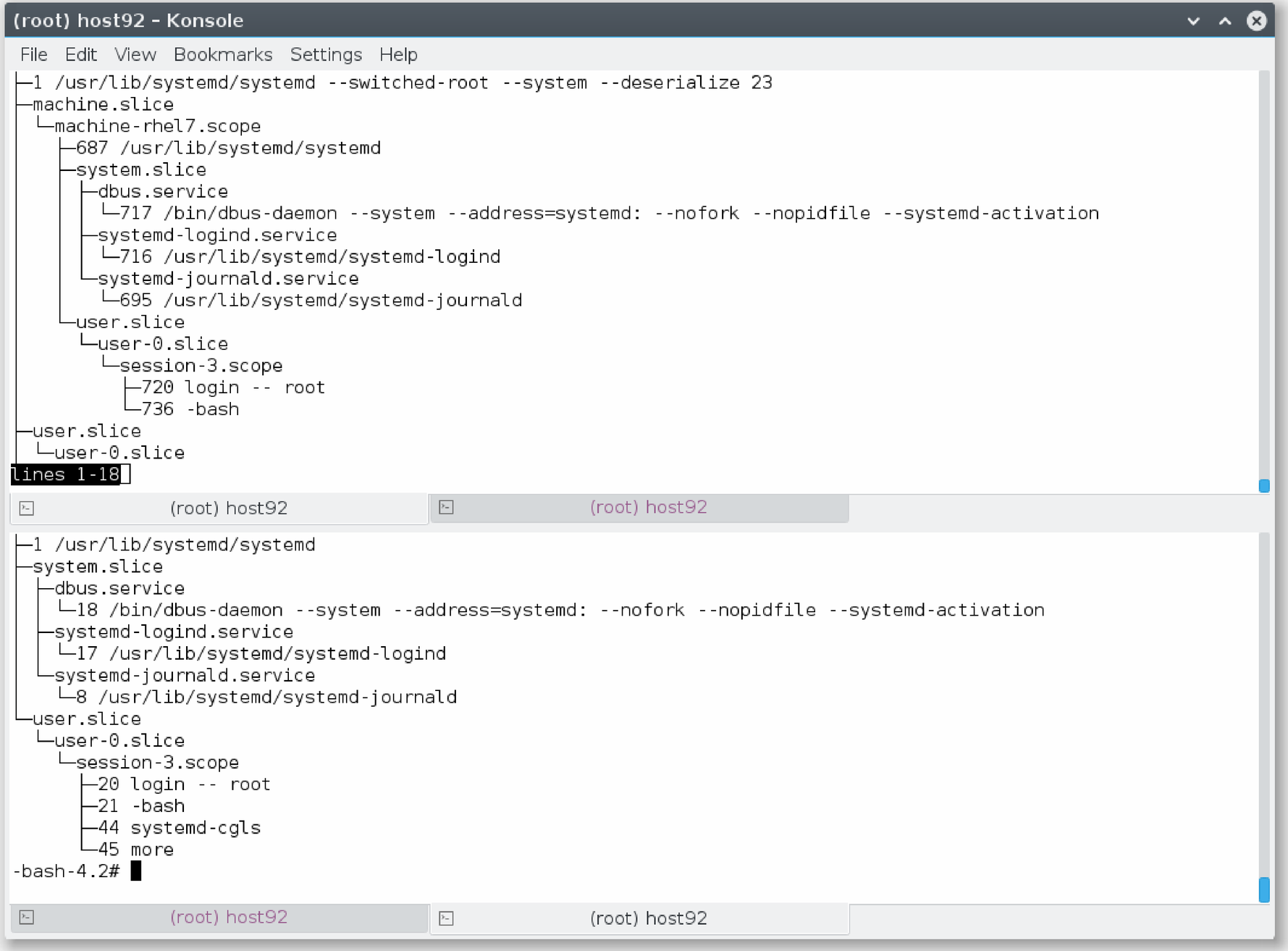
nspawn

- Store containers under /var/lib/container to align w/ @systemd-nspawn.service
  - `mkdir /var/lib/container`
- Install a minimal OS w/ yum; 306 rpms ~360MB on disk:
  - `yum -y --releasever=7Server --installroot=/var/lib/container/rhel7 install systemd passwd yum redhat-release vim-minimal`
- Point nspawn at the directory and go
  - `systemd-nspawn -D /var/lib/container/rhel7`
- To “boot with an init system”, we need to, start the container, set the root password, and configure the system if necessary, etc
  - `systemd-nspawn -D /var/lib/container/rhel7`
  - `passwd ; systemctl disable kdump postfix firewalld tuned`

`man systemd-nspawn`



This is where everyone can make fun of me for running KDE



Konsole is actually a really great terminal. You should try it.

nspawn - continued

- To start on boot in RHEL 7.0 & 7.1:
  - `cp /usr/lib/systemd/system/systemd-nspawn\@.service /etc/systemd/system/systemd-nspawn@rhel7.service`
  - systemctl enable systemd-nspawn@rhel7.service

RHEL 7.2 will likely rebase on systemd 219

219 Highlights

- systemctl – enhancements edit, cat, etc
- CPUQuota – “cap” CPU usage for services
- systemd-socket-proxy – add socket activation to daemons that don't support it natively
- systemd-nspawn
  - Improved networking
  - Ephemeral & template support
  - “-M [container]” option for systemctl, journalctl, etc
  - Import and run Docker containers & raw cloud images
- networkd – DHCP srv/clt, bridge, bond, vlan, vxlan, macvlan, tun

Additional Resources

- RHEL 7 documentation:
  https://access.redhat.com/site/documentation/Red_Hat_Enterprise_Linux/
- systemd project page:
  http://www.freedesktop.org/wiki/Software/systemd/
- Lennart Poettering's systemd blog entries: (read them all)
  http://0pointer.de/blog/projects/systemd-for-admins-1.html
- Red Hat System Administration II & III (RH134/RH254)
  http://redhat.com/training/
- systemd FAQ
- Tips & Tricks

LEARN. NETWORK.
EXPERIENCE OPEN SOURCE.

Tips & Troubleshooting

- Early boot shell on tty9
  - `systemctl enable debug-shell.service`
  - `ln -s /usr/lib/systemd/system/debug-shell.service \ /etc/systemd/system/sysinit.target.wants/`
- `systemctl list-jobs`
- Interactive boot append: `systemd.confirm_spawn=1`
- Enable debugging append:
  - `debug`
  - `debug systemd.log_target=kmsg log_buf_len=1M`
  - `debug systemd.log_target=console console=ttyS0`

http://freedesktop.org/wiki/Software/systemd/Debugging/

Tips & Troubleshooting

- `rc.local is supported, but no longer runs last`
  - `chmod +x /etc/rc.d/rc.local`
- `systemd-analyze`
  - Use 'blame', 'plot', or 'critical-chain' for more details
- Append systemd.unit=[target] to the kernel
  - Rescue mode: single, s, S, or 1
  - Emergency (similar to init=/bin/bash): -b or emergency