Systemd

Systemd

Agenda

1. Verden før Systemd
2. Diskussionen om Systemd
3. Logging
4. Timers

Systemd

Agenda

1. **Verden før Systemd**
2. Diskussionen om Systemd
3. Logging
4. Timers

Boot processen

Lidt kerne-kode, hvor der ledes efter et init-program.

```
$ cat init/main.c
[...]
static noinline int init_post(void)
{
[...]
        if (execute_command) {
                run_init_process(execute_command);
                printk(KERN_WARNING "Failed to execute %s.  Attempting "
                                    "defaults...\n", execute_command);
        }
        run_init_process("/sbin/init");
        run_init_process("/etc/init");
        run_init_process("/bin/init");
        run_init_process("/bin/sh");

        panic("No init found.  Try passing init= option to kernel.");
}
```

Linux Standard Base Specification (LSB)

- Linux Standard Base Specification (LSB) defines a binary interface for application programs.
- Location: http://refspecs.linux-foundation.org/lsb.shtml
- LSB core-section, chapter 22 defines the so-called "System Initialization".

The init process

init is the parent of all processes. It reads the file /etc/inittab, and creates processes based on it.

A “runlevel” is a software configuration and init can be in one of the following runlevels:

- **Runlevel 0 (reserved)**: used to halt the system
- **Runlevel 1 (reserved)**: used to get the system in single user mode
- **Runlevel 2-5**: multiuser runlevels
- **Runlevel 6**: used to reboot the system
- **Runlevel s or S**: Single user mode, /etc/rcS.d/* are executed during boot of the system.
- **Runlevel A, B and C**: “on demand” runlevels, the actual runlevel is unchanged, but the actions associated with the level are executed.

Configuring /etc/inittab

- An entry in this file have the format:
  id:runlevels:action:process [paramters to process]
- A sample inittab:

```
id:2:initdefault
# First script to be executed
si::sysinit:/etc/init.d/rcS
l0:0:wait:/etc/init.d/rc 0
l1:1:wait:/etc/init.d/rc 1
l2:2:wait:/etc/init.d/rc 2
l3:3:wait:/etc/init.d/rc 3
l4:4:wait:/etc/init.d/rc 4
l5:5:wait:/etc/init.d/rc 5
l6:6:wait:/etc/init.d/rc 6
1:2345:respawn:/sbin/getty 38400 tty1
2:23:respawn:/sbin/getty 38400 tty2
```

/etc/init.d/rc

For each of the runlevels 0-6 there is an entry in:
/etc/inittab that executes:
/etc/init.d/rc ?

```
l2:2:wait:/etc/init.d/rc 2
```

So **/etc/init.d/rc** is called with the runlevel as a parameter

The rc-script now inspect the corresponding directory for processes to stop and start.

/etc/rc directories

The directory /etc contains several, runlevel specific, directories:

```
$ ls -d /etc/rc.d/rc*
        /etc/rc.d/rc0.d /etc/rc.d/rc3.d /etc/rc.d/rc6.d
        /etc/rc.d/rc1.d /etc/rc.d/rc4.d /etc/rc.d/rcS.d
        /etc/rc.d/rc2.d /etc/rc.d/rc5.d
```

Each of these contain runlevel specific links to scripts in /etc/init.d:

- K* Scripts are executed with “stop” as argument
- S* scripts are executed with “start” as argument
- The K* and S* scripts are named with a 2 digit number followed by a name in order to control the order of execution (eg. K20gpm).

chkconfig and update-rc.d

It is not recommended to manual create or remove links from /etc/init.d/ to the right runlevel under /etc/rc.d/

Instead one of the commands ***chkconfig*** or ***update-rc.d*** depending on the distribution.

- chkconfig: suse, redhat, centos, fedora, ...
- update-rc.d: ubuntu, debian, linux mint, ...

Systemd

Agenda

1. Verden før Systemd
2. **Diskussionen om Systemd**
3. Logging
4. Timers

tmux

- tmux killed upon exist
  https://github.com/tmux/tmux/issues/428

  “With systemd 230 we switched to a default in which user processes started as part of a login session are terminated when the session exists (KillUserProcesses=yes).

  Unfortunately this means starting tmux in the usual way is not effective, because it will be killed upon logout. There are a few option to avoid that, the best being:”

```
systemd-run --scope --user tmux
```

0day

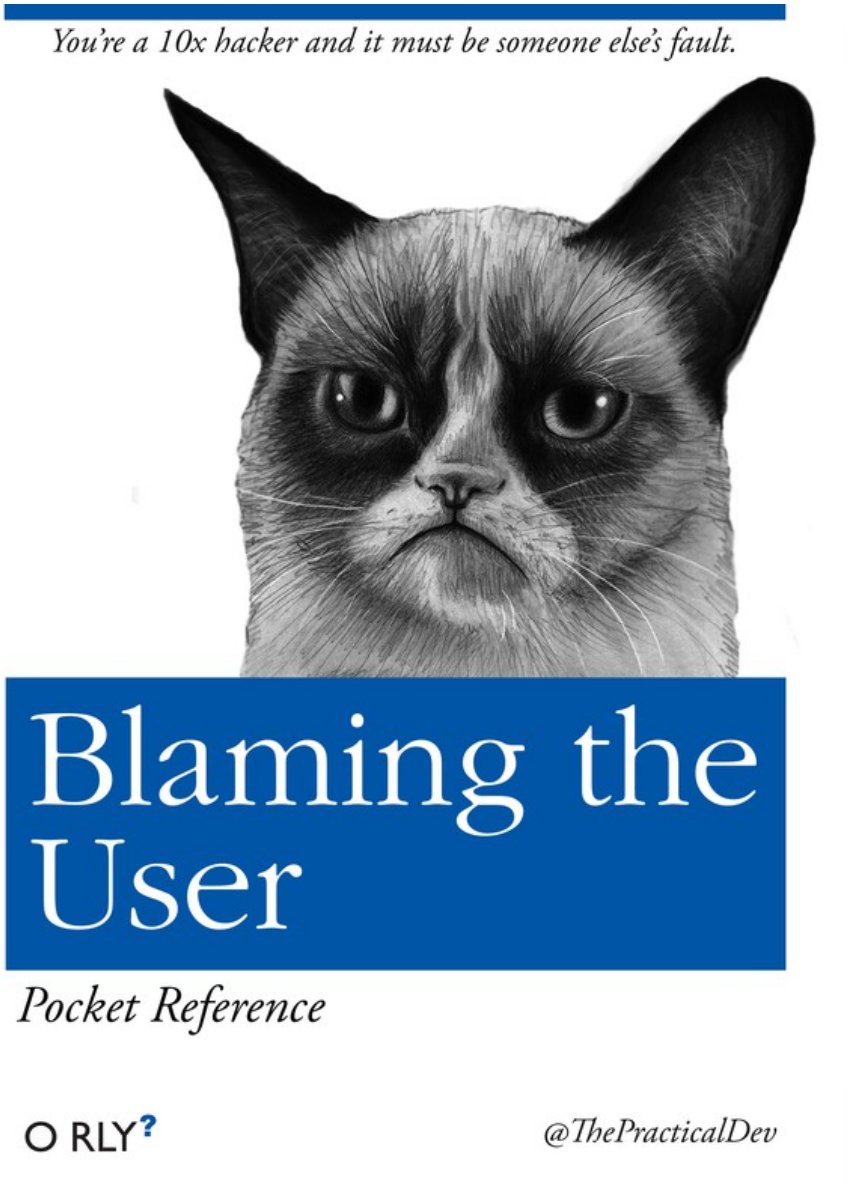
systemd can't handle the process privileges that belongs to a user name that starts with number, such as 0day (https://github.com/systemd/systemd/issues/6237)

```
[Unit]
Description=0day socat service
After=network.target

[Service]
User=0day
Restart=always
Type=simple
WorkingDirectory=/home/0day/
ExecStart=/usr/bin/socat TCP-LISTEN:18086,reuseaddr,fork EXEC:"/opt/run-elf"

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

Systemd


You're a 10x hacker and it must be someone else's fault.
Blaming the User
Pocket Reference
O RLY?
@ThePracticalDev

0day

Answer:

Yes, as you found out "0day" is not a valid username. I wonder which tool permitted you to create it in the first place.

Note that not permitting numeric first characters is done on purpose: to avoid ambiguities between numeric UID and textual user names.

systemd will validate all configuration data you drop at it, making it hard to generate invalid configuration.

Hence, yes, it's a feature that we don't permit invalid user names, and I'd consider it a limitation of xinetd that it doesn't refuse an invalid username.

Debian

- October 2014:
  A group of “Veteran Unix Admins” reckons too much input from GNOME devs is dumbing down Debian, and in response, is floating the idea of a fork.

- Spring of 2015: A target release will see users “be able to switch from Debian 7 to Devuan 1 smoothly, as if they would dist-upgrade to Jessie, and start using our package repositories.”

Debian

Debian / Devuan

- May 2013 - Debian 7
- April 2015 - Debian 8
- May 2017 - Devuan 1.0.0 (Debian 8)
- June 2017 - Debian 9

Større alternativer til Systemd

- Devuan
- Gentoo (OpenRC)
- Slackware
- *BSD

Systemd

Agenda

1. Verden før Systemd
2. Diskussionen om Systemd
3. **Logging**
4. Timers

Logging

Traditionalt bruger man syslog, syslog-ng eller rsyslog til at logge beskeder med i tekstformat og logrotate til at rydde op i logfilerne.

systemd-journald har overtaget den opgave.

Den gemmer det i et binært format.

Logging

Konfigurationen for *systemd-journald* befinder sig i filen: /etc/systemd/journald.conf

```
[Journal]
#Storage=auto
#Compress=ues
#Seal=yes
[...]
```

Normalt er ingen værdier sat og der bruges derfor de default værdier.

Logging

Storage=

- **volatile**
  log data will be stored only in memory
- **persistent**
  data will be stored under /var/log/journal/ (preferably) or /run/log/journal/
- **none**
  turns off all storage, all log data received will be dropped. Forwarding to other targets, such as the console, the kernel log buffer, or a syslog socket will still work however.
- **auto** (default)
  Same as persistent but /var/log/journal/ will not be created if needed

Logging

SystemMaxUse/RuntimeMaxUse

Control how much disk space/memory the journal may use up at most.

Default 10%, max 4GB.

Logging

SystemKeepFree/RuntimeKeepFree

Control how much disk space/memory systemd-journald shall leave free for other uses.

Default 15%.

Logging

SystemMaxFileSize/RuntimeMaxFileSize

Control how large individual journal files may grow at most. This influences the granularity in which disk space is made available through rotation, i.e. deletion of historic data.

Default: SystemMaxUse/8 or RuntimeMaxUse/8

Logging

ForwardToSyslog
ForwardToKMsg
ForwardToConsole
ForwardToWall

Control whether log messages received by the journal daemon shall be forwarded to:

- the traditional syslog daemon,
- the kernel log buffer (kmsg),
- the system console,
- or sent as wall messages to all logged-in users.

Default: forwarding to syslog and wall is enabled

Logging

ForwardToSyslog
ForwardToKMsg
ForwardToConsole
ForwardToWall

These settings may be overridden at boot time with the "*systemd.journald.forward_to_syslog*", ”*systemd.journald.forward_to_kmsg*", "*systemd.journald.forward_to_console*", "*systemd.journald.forward_to_wall*". If the option name is specified without "=" and the following argument, true is assumed.

Otherwise, the argument is parsed as a boolean. When forwarding to the console, the TTY to log to can be changed with *TTYPath=*

Logging

Checking the size and integrity of the journal data:

```
# journalctl --disk-usage
Archived and active journals take up 9.8M in the file system.
```

```
# journalctl --verify
PASS: /run/log/journal/506013/system.journal
PASS: /run/log/journal/506013e/system@f913150b5-000000001-00070.journal
```

Logging

Reducing the journal size while it's running.

Option 1:

Delete the oldest entries until the journal reaches the desired size.

```
# journalctl --vaccum-size=100M
```

Option 2:

Remove all entries before a certain time:

```
# journalctl --vaccum-time=1month
```

Logging

Reading all the logs. No need to be root.

```
$ journalctl
```

All logs since last boot

```
$ journalctl -b
```

Showing logs but limit the output with built-in tail

```
$ journalctl -b -n 10
```

Logging

Filter by component

```
$ journalctl -u apache2.service
$ journalctl -u snapd.service
```

Filter by pid

```
$ journalctl _PID=1003
```

Filter by uid

```
$ journalctl _UID=106
```

Logging

Display all boot events saved in the journal

```
$ journalctl --list-boots
0 7394f92cfc124089bd4a9 Thu 2017-07-27 10:15:24 CEST—Fri 2017-07-28 08:58:53 CEST
```

And use the number in front to display information on a specific boot

```
$ journalctl -b 0
```

Show only kernel messages since last boot

```
$ journalctl -k -b
```

Systemd

Agenda

1. Verden før Systemd
2. Diskussionen om Systemd
3. Logging
4. **Timers**

Timers

Timers can be used as an alternative to cron.

They have built-in support for calendar time events, monotonic time events, and can be run asynchronously.

Timers

Benefits:

- The main benefits of using timers come from each job having its own systemd service.
- Jobs can be easily started independently of their timers. This simplifies debugging.
- Each job can be configured to run in a specific environment.
- Jobs can be attached to cgroups.
- Jobs can be set up to depend on other systemd units.
- Jobs are logged in the systemd journal for easy debugging.

Timers

Caveats:

- Some things that are easy to do with cron are difficult to do with timer units alone.
- Complexity: to set up a timed job with systemd you create two files and run a couple systemctl commands. Compare that to adding a single line to a crontab.
- Emails: there is no built-in equivalent to cron's MAILTO for sending emails on job failure.

Timers

systemd-cron-next:

- A compatibility layer for crontab-to-systemd timers framework.
- It parses crontab and generating systemd timers and services.
- It's intended to be a drop-in replacement for all cron implementations.

Timers

```
$ systemctl list-timers
NEXT                         LEFT          LAST                         PASSED             UNIT
Fri 2017-07-28 13:31:09 CEST  3h 44min left Thu 2017-07-27 13:31:09 CEST  20h ago            systemd-tmpfiles-c
Sat 2017-07-29 00:00:00 CEST  14h left      Fri 2017-07-28 00:00:05 CEST  9h ago             logrotate.timer
Sat 2017-07-29 00:00:00 CEST  14h left      Fri 2017-07-28 00:00:05 CEST  9h ago             man-db.timer
Sat 2017-07-29 00:00:00 CEST  14h left      Fri 2017-07-28 00:00:05 CEST  9h ago             shadow.timer
Sat 2017-07-29 00:00:00 CEST  14h left      Fri 2017-07-28 00:00:05 CEST  9h ago             updatedb.timer
Tue 2017-08-01 00:00:00 CEST  3 days left   Sat 2017-07-01 00:00:20 CEST  3 weeks 6 days ago pamac-cleancache.t
Tue 2017-08-01 00:00:00 CEST  3 days left   Sat 2017-07-01 00:00:20 CEST  3 weeks 6 days ago pamac-mirrorlist.t

7 timers listed.
Pass --all to see loaded but inactive timers, too.
```

```
$ systemctl --user list-timers
NEXT                         LEFT          LAST                         PASSED             UNIT
Fri 2017-07-28 10:00:00 CEST  3min 9s left Fri 2017-07-28 09:55:52 CEST  58s ago btc-stats.timer btc-stats.serv

1 timers listed.
Pass --all to see loaded but inactive timers, too.
```

Timers

~/.config/systemd/user/

```
$ ls -F
btc-stats.service  btc-stats.timer
```

Timers

btc-stats.service

```
[Unit]
Description=Please run my btc stat job

[Service]
Type=simple
ExecStart=/home/mm/bin/btc-stats.sh

[Install]
WantedBy=default.target
```

Timers

btc-stats.timer

```
[Unit]
Description=Run my job every 5 minutes
RefuseManualStart=no
RefuseManualStop=no

[Timer]
Persistent=false
OnBootSec=80
OnCalendar=*:0/5
Unit=btc-stats.service

[Install]
WantedBy=timers.target
```

Timers

Test that the service is working:

```
$ systemctl --user start btc-stats.service
```

Enable and start the timer

```
$ systemctl --user start btc-stats.timer
$ systemctl --user enable btc-stats.timer
```

Timers

~/.config/systemd/user/

```
$ ls -F
btc-stats.service  btc-stats.timer  default.target.wants/  timers.target.wants/
```