Converting init Scripts to systemd Units

NYRHUG November 2016 Meeting

Patrick Ladd
Technical Account Manager
pladd@redhat.com

Slides available at http://people.redhat.com/pladd/systemd-to-init.pdf

Topics

- Unit Types
- Unit Files
    - Structure
    - Syntax
    - Sections
        - [Unit]
        - [Install]
        - Custom
- Templates
- Converting SysV Init Scripts
- Converting inetd & xinetd

Units and Unit Files

Types of Units

Common types

Naming convention: myunit.type (myunit.service, myunit.socket, etc)

.service	Daemon or application on server
.swap	System swap space
.target	Synchronization point or grouping of other units

Types of Units

Triggers for others

New and replacement methods of launching processes

.socket	Network / IPC socket or FIFO buffer
.device	Device needing management by `udev` or `sysfs`
.mount	Filesystem mountpoint – alternate for `/etc/fstab`
.automount	Filesystem auto-mounting
.path	Path-based activation using `inotify()`
.timer	cron / at equivalent plus extras

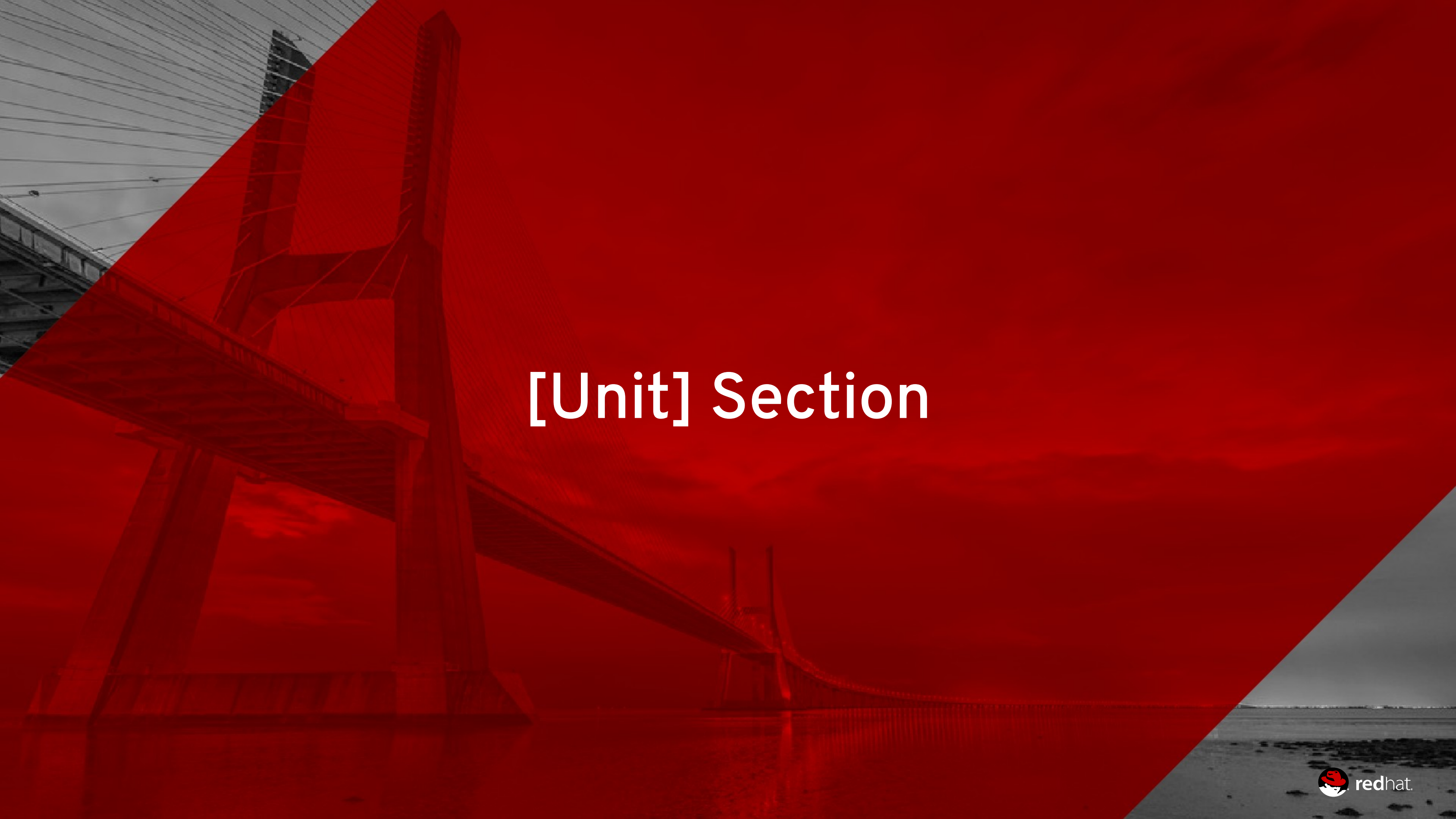
Types of Units

Less common / automatic

.snapshot	"`systemctl snapshot`" result – note: non-persistent
.slice	cgroup control of units
.scope	Automatically created by systemd to manage external processes

Unit File Structure & Syntax

- Section Names
  - Enclosed in `[ ]` brackets
  - Case sensitive
  - Use X- prefix for non-standard sections
- Directives
  - `Key=Value` pairs
  - Override default with empty string: `Key=`
- In all unit files
  - `[Unit]`
  - `[Install]`
- Full documentation: `man systemd.unit`



[Unit] Section

[Unit] Section

General Directives

- Commonly at the top (not required)
- General Directives:

Description=	Describe name & function
Documentation=	List of URIs / man pages

[Unit] Section

Ordering & Dependency

- **Dependency directives (prefer [Install] section however):**

Requires=	Units explicitly required to operate – fails if any of these fail
Wants=	Similar to Requires, less strict – continues to function if others fail/not found
BindsTo=	Similar to Requires, causes unit to stop when other unit terminates
Conflicts=	Units that cannot run at the same time as this unit

- **Ordering Directives**

Before=	Units listed will not start until current unit starts
After=	Units listed started before the current unit starts

[Unit] Section

Conditionals

- Directives:

Condition…=	Test conditions prior to unit start – skipped if any fail
Assert…=	Similar to Condition… skipped if any negative result

- Tests (… part):

Architecture	Machine architecture (x86, x86_64, arm, s390x, …)
Virtualization	vm / container -or- specific virt env (qemu, kvm, vmware,…)
Host	Specific host name or host ID
KernelCommandLine	Specific kernel command line option set
Security	selinux / apparmor / ima /smack / audit enabled
Capability	Specific capability enabled
ACPower	System has AC power
FirstBoot	Boolean indicating unpopulated /etc directory
Path / Directory / File	Collection of file / dir tests: PathExists / PathExistsGlob / PathIsDirectory / PathIsSymbolicLink / PathIsMountPoint / PathIsReadWrite / DirectoryNotEmpty / FileNotEmpty / FileIsExecutable

[Install] Section

[Install] Section

- Commonly at the bottom (not required)
- Directives:

WantedBy=	Places symlink to unit in /etc/systemd/system/xxx.wants/ directory
RequiredBy=	Places symlink to unit in /etc/systemd/system/xxx.requires/ directory
Alias=	Specifies alternate names for the unit
Also=	Units to automatically install/uninstall with this unit
DefaultInstance=	Used in template files

[Install] Section

Tips & Tricks for waiting on network

- **Wants / Requires network.target does not guarantee that network will be up, just that it will be activated**
- **Enable special service to wait for network up:**
  - **For network manager: `systemctl enable NetworkManager-wait-online.service`**
  - **For networkd: `systemctl enable systemd-networkd-wait-online.service`**
  - **Timeout of 90 seconds – could delay startup significantly**
- **Or add both:**
  - **After=network-online.target**
  - **Wants=network-online.target**

Unit Type Specific Sections

Unit specific sections

- Between [Unit] and [Install] sections
- Each unit type has a specifically named section
- `man systemd.`*`unitType`* for full documentation

[Service] Section

[Service] Section

Service type

• Type=	Characterizes process and daemonizing behavior
• simple:	Main process specified in start line
• forking:	Forks a child and then immediately exits
• oneshot:	Short-lived – wait for process to exit
• dbus:	Takes a name on D-Bus bus
• notify:	Issues a notification when finished starting up
• idle:	Service will not be run until all jobs are dispatched

[Service] Section

Service type supplements

- **Additional directives for some service types:**

RemainAfterExit=	**oneshot: indicates to consider active even after exit**
PIDFile=	**forking: path of file containing PID of main child**
BusName=	**dbus: D-Bus bus name service will attempt to acquire**
NotifyAccess=	**notify: [none|main|all] sockets to listen for status updates from sd_notify()**

[Service] Section

Service Management

- Actual directives to start / stop / reload service

ExecStart=	Full path and arguments of command (preceding '–' will ignore return code)
ExecStartPre=	Additional commands to be executed before process start
ExecStartPost=	Additional commands to be executed after process start
ExecReload=	Command to reload configuration (optional)
ExecStop=	Command to stop (optional – process killed if omitted)
ExecStopPost=	Command to execute following stop

[Service] Section

Timing directives

• RestartSec=	Amount of time to wait before attempting restart
• Restart=	Circumstances to automatically restart: [always|on-success|on-failure|on-abnormal|on-abort|on-watchdog]
• TimeoutSec=	Time to wait when starting / stopping before forcefully killing
• TimeoutStartSec=	
• TimeoutStopSec=	

[Socket] Section

[Socket] Section

Triggered – most common items

ListenStream=	TCP based service address
ListenDatagram=	UDP based service address
ListenSequentialPacket=	UNIX socket based service
ListenFIFO=	FIFO buffer based service

- Spec:

Starts with /	File system socket
Starts with @	Abstract namespace socket
Single number	IPV6 port number
v.w.x.y:z	IPV4 address/port
[x]:y	IPV6 address / port

[Socket] Section

Additional directives

• Accept=	Spawn additional instances of service for each request (default: false)
• SocketUser=	UNIX socket userid owner (default: root)
• SocketGroup=	UNIX socket group owner (default: root or matching group for SocketUser=)
• SocketMode=	POSIX permissions for UNIX socket / FIFO buffers
• Service=	Name of corresponding .service unit if not same as this unit
• BindIPv6Only=	Bind IPV6 and/or IPV4

[Mount] Section

[Mount] Section

Filesystem mounts without /etc/fstab

What=	Absolute path to resource to mount
Where=	Absolute path to mount point (should be same as unit file name)
Type=	Filesystem type
Options=	Mount options (comma separated list)
SloppyOptions=	Boolean – fail if unrecognized option encountered
DirectoryMode=	Permission mode of parent directories of mount point (if being created)
TimeoutSec=	Amount of time to wait before marking mount failed

[Automount] Section

[Automount] Section

Filesystem automount points

- Must be named the same as an associated [Mount] unit
  - /home/pladd must have a home-pladd.mount file
- Directives:

Where=	Absolute path to mount point (should be same as unit file name)
DirectoryMode=	Permission mode of parent directories of mount point (if being created)

[Swap] Section

[Swap] Section

Specify system swap space

• What=	Absolute path to swap space
• Priority=	Integer indicating priority of swap
• Options=	Mount options (comma separated list)
• TimeoutSec=	Amount of time to wait before marking as failed

[Path] Section

[Path] Section

Path to be monitored for changes

- Configuration

Unit=	Unit to activate when path tests are met
MakeDirectory=	Create the path prior to watching?
DirectoryMode=	Permission mode of any created elements when MakeDirectory=1

- Tests

PathExists=	
PathExistsGlob=	Check if path/path glob exists
PathChanged=	Change to file when closed
PathModified=	Activates on file writes as well as closes
DirectoryNotEmpty=	Activates when directory no longer empty

[Timer] Section

[Timer] Section

Replacement / supplement for cron & at

- Configuration

Unit=	Unit to activate when timer activated (default: *unitname*.service)
AccuracySec=	Upper limit to accuracy of timer (default: 1 minute)
Persistent=	Trigger when timer is active if would have trigger when inactive
WakeSystem=	Wake from suspend if system timer reached during suspend

- Timers

OnActiveSec=	Amount of time since timer activated
OnBootSec=	Amount of time after system boot
OnStartupSec=	Amount of time after systemd startup
OnUnitActiveSec=	Timer relative to last activation
OnUnitInactiveSec=	Timer relative to last time unit marked inactive
OnCalendar=	Absolute timer

[Timer] time syntax

See `man systemd.time`

- Space separated list of numbers followed by unit
- No unit specified – seconds are assumed (with some exceptions)
- Units understood:
  - usec, us
  - msec, ms
  - seconds, second, sec, s
  - minutes, minute, min, m
  - hours, hour, hr, h
  - days, day, d
  - weeks, week, w
  - months, month
  - years, year, y

Templates

Template Unit Files & Unit Names

- Unit file name & Unit name contain @ symbol
    - After base name
    - Before unit suffix
        - example@.service
- Specific instances have identifier inserted after @ symbol
    - example@inst1.service
- Template instance files generally created as symlinks to template

Template Directives

%n	Anywhere where this appears in a template file, the full resulting unit name will be inserted.
%N	Same as the above, but any escaping, such as those present in file path patterns, will be reversed.
%p	Unit name prefix. This is the portion of the unit name that comes before the @ symbol.
%P	This is the same as above, but with any escaping reversed.
%i	This references the instance name, which is the identifier following the @ in the instance unit.
%I	This specifier is the same as the above, but with any escaping reversed.

Template Directives

%f	This will be replaced with the unescaped instance name or the prefix name, prepended with a /.
%c	This will indicate the control group of the unit, with the standard parent hierarchy of /sys/fs/cgroup/ssytemd
%u	The name of the user configured to run the unit.
%U	The same as above, but as a numeric UID instead of name.
%H	The host name of the system that is running the unit.
%%	This is used to insert a literal percentage sign.

Converting SysV Init Script

SysV init script: abrtd

```
#!/bin/bash
# Start the ABRT daemon
#
# chkconfig: 35 82 16
# description: Saves segfault data, kernel oopses, fatal exceptions
# processname: abrtd
# pidfile: /var/run/abrtd.pid
### BEGIN INIT INFO
# Provides: abrt
# Required-Start: $syslog $local_fs messagebus
# Required-Stop: $syslog $local_fs
# Default-Stop: 0 1 2 6
# Default-Start: 3 5
# Short-Description: Saves segfault data, kernel oopses, fatal exceptions
# Description: Saves segfault data, kernel oopses, fatal exceptions
### END INIT INFO

# Source function library.
. /etc/rc.d/init.d/functions
ABRT_BIN="/usr/sbin/abrtd"
LOCK="/var/lock/subsys/abrtd"
RETVAL=0

#
# Set these variables if you are behind proxy
#
#export http_proxy=
#export https_proxy=

check() {
          # Check that we're a privileged user
          [ "`id -u`" = 0 ] || exit 4

          # Check if abrt is executable
          test -x "$ABRT_BIN" || exit 5
}

start() {
          check

          # Check if it is already running
          if [ ! -f "$LOCK" ]; then
                    echo -n $"Starting abrt daemon: "
                    daemon "$ABRT_BIN"
                    RETVAL=$?
                    [ $RETVAL -eq 0 ] && touch $LOCK
                    echo
          fi
          return $RETVAL
}

stop() {
          check

          echo -n $"Stopping abrt daemon: "
          killproc "$ABRT_BIN"
          RETVAL=$?
          [ $RETVAL -eq 0 ] && rm -f "$LOCK"
          echo
          return $RETVAL
}

restart() {
          stop
          start
}

reload() {
          restart
}

case "$1" in
start)
          start
          ;;
stop)
          stop
          ;;
reload)reload
          ;;
force-reload)
          echo "$0: Unimplemented feature."
          RETVAL=3
          ;;
restart)
          restart
          ;;
condrestart)
          if [ -f "$LOCK" ]; then
                    restart
          fi
          ;;
status)
          status abrtd
          RETVAL=$?
          ;;
*)
          echo $"Usage: $0 {start|stop|status|restart
|condrestart|reload|force-reload}"
          RETVAL=2
esac

exit $RETVAL
```

Converted script: abrtd.service

```
[Unit]
Description=Daemon to detect crashing apps
After=syslog.target

[Service]
ExecStart=/usr/sbin/abrtd
Type=forking

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

Shipping script: abrtd.service

```
[Unit]
Description=ABRT Automated Bug Reporting Tool
After=syslog.target

[Service]
Type=dbus
BusName=com.redhat.abrt
ExecStart=/usr/sbin/abrtd -d -s

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

Converting inetd specification

ssh inetd / xinetd

inetd:

```
ssh stream tcp nowait root /usr/sbin/sshd sshd -i
```

xinetd:

```
service ssh {
        socket_type = stream
        protocol = tcp
        wait = no
        user = root
        server = /usr/sbin/sshd
        server_args = -i
}
```

Systemd sshd.socket

```
[Unit]
Description=SSH Socket for Per-Connection Servers

[Socket]
ListenStream=22
Accept=yes

[Install]
WantedBy=sockets.target
```

Systemd sshd.service

```
[Unit]
Description=SSH Per-Connection Server

[Service]
ExecStart=-/usr/sbin/sshd -i
StandardInput=socket
```

redhat.

THANK YOU

plus.google.com/+RedHat

facebook.com/redhatinc

linkedin.com/company/red-hat

twitter.com/RedHatNews

youtube.com/user/RedHatVideos