



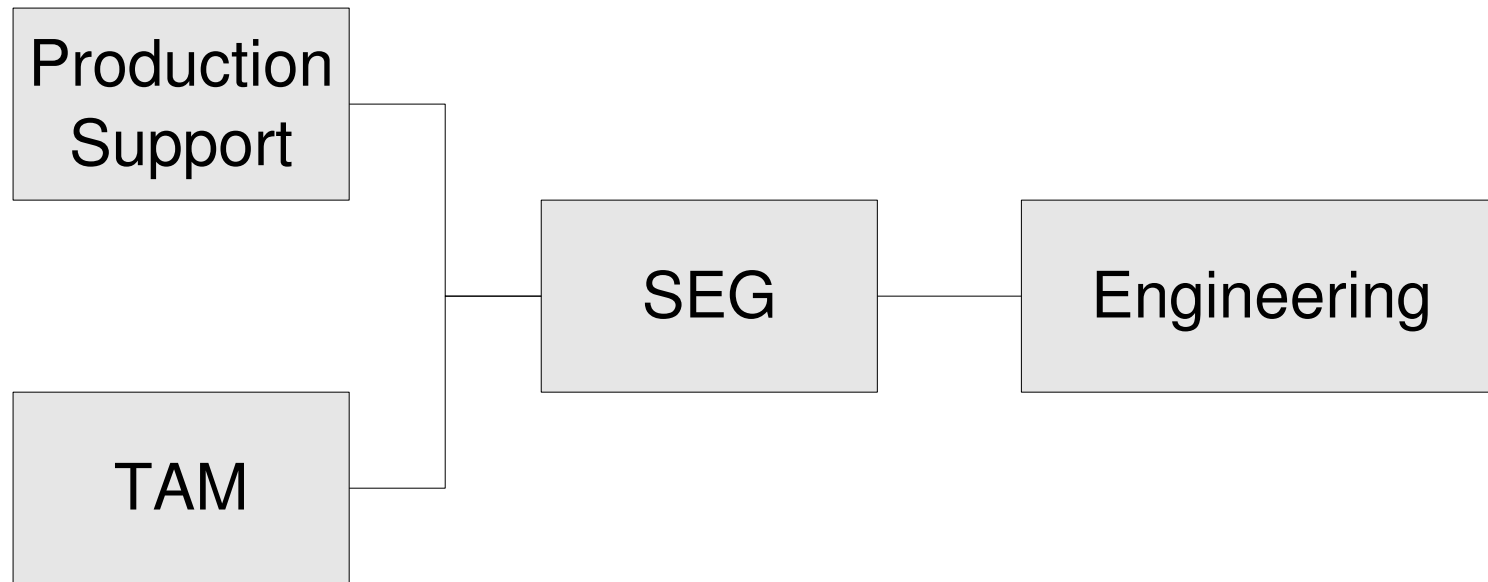
# Getting in the Fast Lane with Red Hat Support

Jeff Bastian & Guy Streeter, Red Hat

# Agenda

- Introductions
- Red Hat Support Structure
- Issues and Bugs
- Working with Red Hat Support
- Q&A

# Red Hat Support Structure



TAM = Technical Account Manager

SEG = Support Engineering Group

# Issues and Bugs

- *Or* Issues versus Bugs
- Multiple ticket tools: Customer-facing-tools and Bugzilla
- Issues
  - Questions on how to do something
  - Documentation request (clarification, new docs, etc.)
  - Problem with a configuration file
  - Request For Enhancement (a.k.a. Feature requests)
  - Bug
- Bugs
  - A defect

# Lifecycle of a Bug

- Open a ticket with Red Hat Support
- Escalated to Support Engineering Group
- Escalated to Engineering via Bugzilla
- Patch developed or found upstream to fix the problem
- Test packages generated
  - Tested by customer to verify the patch fixes the problem
- Fixed package scheduled for a future .Y release
- Hot-fix package may be released
- Quality Assurance testing
- FasTrack or async errata package may be released
- Released with .Y

# Accelerated Fixes for Packages

- Async Errata
- FasTrack
- Development Hot-Fix

# Working with Red Hat Support

- Opening a ticket
- Interacting with Support
- Uploading files to Red Hat (both large and small)
- Testing fixes

# A “Bad” Ticket

Problem Description: firefox is broken

# Opening a Ticket

## ■ Bugzilla Template

- Description of problem: clear and detailed
- Version-Release number of selected component:  
pkg-1.0-1.i386
- How reproducible? Every time? Occasionally? Rarely?  
Only once (so far)?
- Steps to reproduce: Short programs to trigger the bug are *very* helpful!
- Actual results: What went wrong
- Expected results: What *should* it have done
- Additional info: patches, references to upstream bugs, etc.

# Opening a Ticket (continued)

- Always include a sosreport (or sysreport)
  - sos = son of sysreport (RHEL 4.6, 5.0 and newer)
  - Tip: to speed up sosreport generation:  
`sosreport -k rpm.rpmva=off`
  - We may ask for the `rpm -va` output later...
- Method to reproduce the problem
  - Manual steps: start program xyz, open attached file, click here, watch xyz crash
  - Short program to demonstrate or trigger the problem
- Include a screenshot if there's a GUI problem
  - `import -frame -pause 2 screenshot.png`

# Example program: execvp memory leak

```
#define _GNU_SOURCE
#include <stdio.h>
#include <unistd.h>

int main (int argc, char* argv[])
{
    char *cmd[3] = { (char *) "true", (char *) "1", NULL };
    pid_t child;

    while(1) {
        child = vfork();
        if (child == 0) {
            execvp(*cmd, cmd);
        }
        usleep(500000);
    }
    return 0;
}
```

# Debugging Tools

- *command* - -debug + syslog
- strace
- gdb + debuginfo
- Oprofile
- The “Magic” SysRq Button
- tcpdump and Wireshark
- SystemTap [Frank Eigler, Wed, 11:30am]
- Frysk [Andrew Cagney, Thur, 2:45pm]

# Syslog and Debug Messages

- Many programs, e.g., autofs, send debug output to syslog
- By default, debug output is not logged
  - Enable it by adding this line to `/etc/syslog.conf`  
`daemon.* /var/log/debug`
  - And restart syslogd: `service syslog restart`
- Enable debugging for the program
  - e.g., edit `/etc/sysconfig/autofs` and set  
`DAEMONOPTIONS="--debug" (autofs-4.x)`  
`DEFAULT_LOGGING="debug" (autofs-5.x)`
  - And restart autofs: `service autofs restart`

# Rotate Debug Logs

- If you need to capture debug messages over many days or weeks, you may wish to rotate the debug logs to save disk space
- Create an `/etc/logrotate.d/debug` file:

```
/var/log/debug {
    sharedscripts
    compress
    rotate 5
    weekly
    missingok
    postrotate
        /bin/kill -HUP ...
        /bin/kill -HUP ...
    endscript
}
```



see `/etc/logrotate.d/syslog`

# Tracing System Calls

- Capture everything:  
`strace -v -f -tt -o /tmp/cmd-trace.txt command`
- Limit tracing to individual syscalls, or groups of calls:  
`strace -e trace=open,close,read,write ...`  
`strace -e trace=file ...`  
`strace -e trace=network ...`
- Attach to a running process:  
`strace -p pid ...`
- ```
$ strace -v -f -tt -o /tmp/host.out host srvr.xyz.com
;; connection timed out; no servers could be reached
$ cat /tmp/host.out
...
28209 11:44:26.572951 open("/etc/resolv.conf",
0_RDONLY) = -1 ENOENT (No such file or directory)
```

# **gdb (GNU Debugger)**

- Used for stepping through programs, watching variables, and all the other things a debugger can do
- Can tell us *how* and *where* a program crashed
- For long running programs or daemons, use a batch file of commands; create a gdb-cmds.txt file, for example:

```
set args "-d"  
set height 0  
run  
thread apply all backtrace  
generate-core-file
```

- Then launch the daemon through gdb:  

```
gdb -quiet -batch -x gdb-cmds.txt \  
/usr/bin/nsd > nsd_bt.out
```

# debuginfo packages

- Debugging data is stripped from RPMs to make them smaller and saved into debuginfo RPMs
- [ftp://ftp.redhat.com/pub/redhat/linux/enterprise/\\$releasever/en/os/\\$basearch/Debuginfo/](ftp://ftp.redhat.com/pub/redhat/linux/enterprise/$releasever/en/os/$basearch/Debuginfo/)
- RHEL5 (or RHEL4+yum):  
`yum --enablerepo=rhel-debuginfo install xyz-debuginfo`
- Make sure the debuginfo version matches the package!  

```
# rpm -q glibc glibc-debuginfo
glibc-2.5-24.x86_64
glibc-debuginfo-2.5-24.x86_64
```

# Example: Bad Backtrace

```
$ rpm -q glibc-debuginfo
package glibc-debuginfo is not installed
$ gcc -o input input.c
$ gdb -quiet -batch -x gdb-cmds.txt ./input
(no debugging symbols found)
Using host libthread_db library "/lib64/libthread_db.so.1".
(no debugging symbols found)
(no debugging symbols found)
Enter a number: 1

Program received signal SIGSEGV, Segmentation fault.
0x0000003778258a1d in _IO_vfscanf_internal () from /lib64/libc.so.6
#0  0x0000003778258a1d in _IO_vfscanf_internal () from /lib64/libc.so.6
#1  0x000000377825ea0c in scanf () from /lib64/libc.so.6
#2  0x00000000000400509 in ?? ()
#3  0x00000000000400523 in ?? ()
#4  0x00000000000400544 in ?? ()
#5  0x000000377821d8b4 in __libc_start_main () from /lib64/libc.so.6
#6  0x00000000000400429 in ?? ()
#7  0x00007ffff49f71748 in ?? ()
#8  0x00000000000000000 in ?? ()
Saved corefile core.21438
```

# Example: Good Backtrace

```
$ sudo rpm -i glibc-debuginfo-2.5-24.x86_64.rpm
$ gcc -g -o input input.c
$ gdb -quiet -batch -x gdb-cmds.txt ./input
Using host libthread_db library "/lib64/libthread_db.so.1".
Enter a number: 1
Program received signal SIGSEGV, Segmentation fault.
0x0000003778258a1d in _IO_vfscanf_internal (s=0x377854d680,
    format=<value optimized out>, argptr=0x7fffa39cbf70, errp=0x0)
    at vfscanf.c:1582
1582             *ARG (int *) = (int) num.1;
#0 0x0000003778258a1d in _IO_vfscanf_internal (s=0x377854d680,
    format=<value optimized out>, argptr=0x7fffa39cbf70, errp=0x0)
    at vfscanf.c:1582
#1 0x000000377825ea0c in __scanf (format=0x31 <Address 0x31 out of bounds>)
    at scanf.c:35
#2 0x0000000000400509 in readInput (prompt=0x40065e "Enter a number: ")
    at input.c:9
#3 0x0000000000400523 in processData (prompt=0x40065e "Enter a number: ")
    at input.c:21
#4 0x0000000000400544 in main (argc=2, argv=0x7fffa39cc1b8) at input.c:30
Saved corefile core.21314
```

# Oprofile

- System-wide tool to find performance bottle-necks
- Unlike other profiling tools, e.g., gprof, no profiling data needed from the compiler
  - but debuginfo packages can provide more detailed reports
- Works with CPU counters and other hardware features
- Profile the kernel, libraries, and user-space applications
- Reports can include time spent in functions, cache misses, branch mis-prediction, code annotation, and more
- For more info:
  - <http://people.redhat.com/wcohen/Oprofile.pdf>
  - <http://oprofile.sourceforge.net/>

# The “Magic” SysRq Button

- Have you ever noticed the SysRq on your Print Screen button and wondered what it does?
- The Linux kernel uses it for debugging
  - Print kernel data structures, e.g., task list
  - Send commands to kernel, e.g., reboot immediately
- By default, the key is disabled
- Enable it by
  - Edit `/etc/sysctl.conf`  
`kernel.sysrq=1`
  - Load the new settings  
`sysctl -p`

# The “Magic” SysRq Button (continued)

- If running X11, switch to a text console with CTRL-ALT-F1
- Hit ALT-SysRq-X to print data or send commands
  - ALT-SysRq-h                      brief help
  - ALT-SysRq-b                      reboot immediately
  - ALT-SysRq-p                      print CPU registers and flags
  - ALT-SysRq-m                      print memory information
  - ALT-SysRq-t                      print task (process) list
  - ALT-SysRq-w                      show CPUs (Red Hat specific)
  - More keys at [http://en.wikipedia.org/wiki/Magic\\_SysRq\\_key](http://en.wikipedia.org/wiki/Magic_SysRq_key)
- If remote, send a command to /proc/sysrq-trigger
  - `echo b > /proc/sysrq-trigger`

# tcpdump and Wireshark

- Ethereal renamed to Wireshark because of trademark issues
- Capturing network traffic may be necessary to debug a network service (NIS, http, etc)
- Be sure to snarf enough data with tcpdump
  - 68 bytes is too small for some protocols, e.g., NFS
  - `tcpdump -s 0 ...`
- Intermittent problems
  - May require running packet sniffer for hours or days
  - Use circular buffer to keep only recent data
  - `tcpdump -s 0 -C 10 -W 5 -w dump.pcap -i eth0 ...`
  - `tshark -b filesize:10 -b files:5 -w test.pcap \`  
`-i eth0 ...`

# SystemTap and Frysk

- SystemTap lets you probe system events and kernel internals and safely on a live system and generate reports
- Frysk is an always on system monitoring and debugging tool
- For more info:
  - <http://sourceware.org/systemtap/>
  - <http://sourceware.org/frysk/>

# Kernel Core Dumps

- RHEL3 & RHEL4
  - Diskdump
    - Limited support for certain SCSI and SATA controllers
  - Netdump
    - Supports most network cards
    - Requires netdump server to capture the core
- RHEL5
  - Kdump/Kexec
    - Keeps 2nd copy of kernel in memory to generate the core dump if the primary kernel panics
    - Supports all H/W, but uses more memory
- Test with ALT-SysRq-c (this will reboot your system)

# Uploading Large Files

- Ticket and Bugzilla attachments are limited to 50MB
- Kernel core dumps, debug logs, network captures, etc. can easily grow larger than 50MB
- Compress files with gzip or bzip2 first
  - bzip2 creates much smaller files, but it's slower
- Upload files to `ftp://dropbox.redhat.com/incoming/`
  - Send file name and md5sum to your support contact
  - Enable passive mode
- sos includes a script – `/usr/bin/rh-upload-core` – to ease uploading large vmcore files to Red Hat's ftp dropbox
- See [http://kbase.redhat.com/faq/FAQ\\_80\\_11089.shtml](http://kbase.redhat.com/faq/FAQ_80_11089.shtml) for more info

# RPM Tips and Tricks

- To resolve circular dependencies, install at the same time:
  - `rpm -ivh abc-1.0-1.i386.rpm xyz-1.0-1.i386.rpm`
- `rpm -qp --changelog package.rpm`
- `cd /tmp; rpm2cpio package.rpm | cpio -ivd`
- Custom query format
  - `rpm --querytags`
  - `rpm -q --queryformat '%{name}\n' package`
- `$HOME/.rpmmacros`
  - `%_query_all_fmt %%{name}-%%{version}-%%{release}.%%{arch}`
  - `$ rpm -q firefox`  
`firefox-3.0-0.beta5.3.el5.x86_64`  
`firefox-3.0-0.beta5.3.el5.i386`
  - See `/usr/lib/rpm/macros` for defaults

# RPM Tips and Tricks (continued)

- libpopt (pronounced p-opt, not pop-t) can be used to define custom command line flags

- \$HOME/.popt

- rpm alias --bydate \  
    --qf '%{installtime}\t%{installtime:date}           ▪  
          %{name}-%{version}-%{release}.%{arch}\n' \  
    --pipe 'sort -n | cut -f 2-' \  
    --POPTdesc="\$sort by and print install time"  
rpm exec --qa rpmq -qa --nodigest --nosignature
- \$ rpm -qa --bydate  
Fri 09 Nov 2007 03:50:18 AM CST    setup-2.6.10-1.fc8.noarch  
Fri 09 Nov 2007 03:50:20 AM CST    filesystem-2.4.11-1.fc8.i386  
Fri 09 Nov 2007 03:50:22 AM CST    basesystem-8.1-1.noarch  
...  
Tue 22 Apr 2008 10:22:09 AM CDT    pykickstart-1.29-1.fc8.noarch  
Tue 22 Apr 2008 10:22:11 AM CDT    kdemultimedia-libs-3.5.9-2.fc8.i386  
Tue 22 Apr 2008 10:22:25 AM CDT    kdemultimedia-3.5.9-2.fc8.i386

- Works for any app that uses libpopt, not just rpm

# Security: Firewalls and SELinux

- Occasionally suspected bugs turn out to be caused by security policies such as firewalls or SELinux (Security Enhanced Linux)
- Many people simply turn these features off permanently: this is BAD!
- Try to duplicate the problem on a secure, private network, and try toggling the firewall and/or SELinux on/off to see if the problem goes away
- Run SELinux in permissive mode
  - Allows everything, but logs what would have been denied
  - `setenforce Permissive`

# Q & A