Signals Overview

Advanced Embedded Linux Development

with **Dan Walkes**

University of Colorado **Boulder**

Learning objectives:

Introduce Signals

Understand default Signal handling

Introduce core files

Signals

- Software interrupts for handling asynchronous events:
  - Events outside the system (Ctrl->C).
  - Events from the program or kernel (divide by 0).
  - Interprocess Communication Method.
- Event is asynchronous and handler is asynchronous.

Signals

- Signal lifecycle:
  - raised (sent or generated)
  - stored (by kernel)
  - handled by kernel, based on process request

Signal Handling Options

- Ignore (except for SIGKILL and SIGSTOP)
- Catch and handle
  - Suspend execution of the process
    - Including execution of other signal handlers!
  - Jump to a previously registered function.
  - SIGINT and SIGTERM are two common examples.

Signal Handling Options continued

- Perform the default action
  - Often terminates the process, possibly with core dump (capture of running process memory)

Signal Default Actions

Term - SIGTERM
Ign - Ignore
Core - SIGTERM and dump core
Stop - SIGSTOP

Signal	Value	Action	Comment
SIGHUP	1	Term	Hangup detected on controlling terminal or death of controlling process
SIGINT	2	Term	Interrupt from keyboard
SIGQUIT	3	Core	Quit from keyboard
SIGILL	4	Core	Illegal Instruction
SIGABRT	6	Core	Abort signal from **abort**(3)
SIGFPE	8	Core	Floating-point exception
SIGKILL	9	Term	Kill signal
SIGSEGV	11	Core	Invalid memory reference
SIGPIPE	13	Term	Broken pipe: write to pipe with no readers; see **pipe**(7)
SIGALRM	14	Term	Timer signal from **alarm**(2)
SIGTERM	15	Term	Termination signal
SIGUSR1	30,10,16	Term	User-defined signal 1
SIGUSR2	31,12,17	Term	User-defined signal 2
SIGCHLD	20,17,18	Ign	Child stopped or terminated
SIGCONT	19,18,25	Cont	Continue if stopped
SIGSTOP	17,19,23	Stop	Stop process
SIGTSTP	18,20,24	Stop	Stop typed at terminal
SIGTTIN	21,21,26	Stop	Terminal input for background process
SIGTTOU	22,22,27	Stop	Terminal output for background process

http://man7.org/linux/man-pages/man7/signal.7.html

Common Signals

- SIGABRT - assert() - terminates & generates core file
- SIGHUP - May be used to reread config files
- SIGINT - Ctrl->C
- SIGKILL - cannot be ignored, unconditionally terminates the process

Common Signals continued

- SIGSEGV - Segmentation fault (null pointer, etc)
  - terminates and generates core file as default action
- SIGTERM - gracefully terminate a process
  - Process can catch and cleanup
- SIGSTOP - Unconditionally pause (can’t be ignored)

Core Dump Files and SIGSEGV

- Capture the state of a failing program at the point it terminates
- Use gdb to analyze

```
SIGKILL      9      Term    Kill signal
SIGSEGV     11      Core    Invalid memory reference
SIGPIPE     13      Term    Broken pipe: write to pipe with no
```

```
aesd@aesd-VirtualBox:~/aesd-lectures/lecture9$ cat /proc/sys/kernel/core_pattern
|/usr/share/apport/apport %p %s %c %d %P
      mkdir -p ~/.config/apport && printf "[main]\nunpackaged=true" > ~/.config/apport/settings
```

```
/**
* @author Dan Walkes
* segfaulting file
*/

int main( int argc, char **argv )
{
    int *mem=0x0;
    // oops!  segfault here
    *mem = *mem + 1;
    return 0;
}
```

```
aesd@aesd-VirtualBox:~/aesd-lectures/lecture9$ ./segfaulter
Segmentation fault (core dumped)

aesd@aesd-VirtualBox:~/aesd-lectures/lecture9$ tail -n 1 /var/log/apport.log
ERROR: apport (pid 30208) Fri Feb  7 12:49:37 2020: wrote report /var/crash/_home_aesd_a
esd-lectures_lecture9_segfaulter.1000.crash

aesd@aesd-VirtualBox:~/aesd-lectures/lecture9$ apport-unpack /var/crash/_home_aesd_aesd-lectures_
ecture9_segfaulter.1000.crash  core

aesd@aesd-VirtualBox:~/aesd-lectures/lecture9$ gdb segfaulter core/CoreDump
Program terminated with signal SIGSEGV, Segmentation fault.
#0  0x000055ad1acad611 in main (argc=1, argv=0x7ffcf7303c98) at segfaulter.c:10
10          *mem = *mem + 1;
```

https://stackoverflow.com/questions/14204961/how-to-change-apport-default-behaviour-for-non-packaged-application-crashes