# strace

(not quite so) short overview

Eugene Syromiatnikov

November 15, 2017

# Table of Contents

## What is strace?

- A diagnostic, debugging and instructional userspace utility for Linux.
- It is used to monitor and tamper with interactions between processes and the Linux kernel, which include system calls, signal deliveries, and changes of process state.
- CLI and multiple filtering capabilities make it a powerful yet easy to use tracing tool.
- The operation of strace is made possible by the kernel feature known as ptrace.

## Table of Contents

## Before revision control, 1991 – 1999

**1991 – 1992, Paul Kranenburg**

Wrote strace for SunOS

**1993, Branko Lankester**

| | |
|---|---|
| 1.5 | ported to Linux x86 |

**1993 – 1996, Richard Sladkey**

| | |
|---|---|
| 3.0 | merged 2.5 for SunOS and 2nd release for Linux |
| 3.1 | ported to SVR4, Solaris, Irix;<br>Linux 2.0 (x86, alpha, m68k) |

**1996 – 1999, Wichert Akkerman**

Debian packages: 3.1-1 – 3.1.0.1-12

## Since revision control, 1999 – 2002, Wichert Akkerman

**3.99 – 4.4**

| | |
|---|---|
| 19.02.1999 | introduced revision control (CVS) |
| 18.03.1999 | **3.99** |
| 09.06.1999 | **3.99.1** |
| 09.07.1999 | **4.0**: Linux powerpc, sparc, arm |
| 26.11.1999 | **4.1**: Linux mips |
| 24.12.1999 | **4.2**: Linux s390 |
| 01.03.2001 | **4.3**: Linux ia64 and hppa, FreeBSD i386 |
| 19.08.2001 | **4.4**: Linux ioctl parser |

## Since revision control, 2002 – 2009, Roland McGrath

### 4.4.90 – 4.5.18

| | |
|---|---|
| 10.01.2003 – 17.07.2003 | **4.4.90 – 4.4.99** |
| 24.09.2003 | **4.5**: Linux x86-64 biarch, Linux s390x, sh and sh64 |
| 13.11.2003 | **4.5.1**: display multiple ioctl name matches on Linux |
| 01.03.2004 | **4.5.2**: Linux syscalls enhancements |
| 16.04.2004 | **4.5.3**: Linux syscalls: mq_* |
| 03.06.2004 | **4.5.4**: Linux ioctl update |
| 27.06.2004 | **4.5.5**: bug fixes |
| 12.07.2004 | **4.5.6**: Linux sparc64, Linux ioctl updates |
| 31.08.2004 | **4.5.7**: Linux *xattr and clock_* enhancements |
| 19.10.2004 | **4.5.8**: Linux syscalls: fadvise64, fadvise64_64, epoll_*, mbind, set_mempolicy, get_mempolicy, waitid |
| 04.02.2005 | **4.5.9**: Linux ioctl and syscalls enhancements |

## Since revision control, 2002 – 2009, Roland McGrath

### 4.4.90 – 4.5.18

| | |
|---|---|
| 14.03.2005 | **4.5.10**: Linux signal decoding enhancements |
| 22.03.2005 | **4.5.11**: build fix |
| 09.06.2005 | **4.5.12**: mm fixes, x86-64 biarch enhancements, ppc updates, Linux aio enhancements |
| 03.08.2005 | **4.5.13**: “-e trace=desc” syntax |
| 16.01.2006 | **4.5.14**: accept numeric system calls in -e |
| 11.01.2007 | **4.5.15**: Linux syscalls: *at, inotify*, pselect6, ppoll, unshare |
| 03.08.2007 | **4.5.16**: Linux syscalls: move_pages, utimensat, signalfd, timerfd, eventfd, getcpu, epoll_pwait |
| 21.07.2008 | **4.5.17**: Linux arm improvements |
| 28.08.2008 | **4.5.18**: Linux syscalls enhancements |
| 02.06.2009 | CVS $\rightarrow$ GIT |

## Since revision control, 2009 – 2017, Dmitry Levin

| | |
|---|---|
| 21.10.2009 | **4.5.19**: exit status transparency; Linux bfin, avr32, and cris; new Linux syscalls; Linux arm improvements; Linux syscalls enhancements; Linux socket type flags decoding |
| 13.04.2010 | **4.5.20**: new -C option; Linux tile; new Linux syscalls; Linux syscalls enhancements; Linux ioctls update |
| 15.03.2011 | **4.6**: new Linux method of tracing clone family syscalls; Linux microblaze; new Linux syscalls; Linux syscalls enhancements; Linux ioctls enhancements; Linux biarch enhancements |
| 02.05.2012 | **4.7**: new options: -y, -P, and -I; process monitoring enhancements; x86-32; x86-64 multiarch; multiarch enhancements; new syscalls; syscalls enhancements; ioctls enhancements; speed improvements; non-Linux code finally removed |
| 03.06.2013 | **4.8**: PTRACE_SEIZE support; PTRACE_GETREGSET support; "-e trace=desc" syntax; +arch: AArch64, Meta, OpenRISC 1000, TileGx, Xtensa; multiarch enhancements; new syscalls; syscalls enhancements; ioctls enhancements. |

## Since revision control, 2009 – 2017, Dmitry Levin

| | |
|---|---|
| 15.08.2014 | **4.9**: -k option (stack traces); -w option (stats on syscall latency). +arch: ARC; multiarch enhancements; new syscalls; syscalls enhancements. |
| 06.03.2015 | **4.10**: -yy option (socket protocol and address information); full 32-bit decoding of ioctl commands; getrandom and seccomp syscalls decoding; decoding of 64-bit capability sets; decoding of all prctl commands; evdev, v4l, SG_IO v4, FIFREEZE/FITHAW/FITRIM ioctl decoding; syscalls enhancements; ioctls enhancements; Linux >= 2.5.46 is required (PTRACE_SETOPTIONS). |
| 21.12.2015 | **4.11**: Enhanced and extended test suite; reliable mpers support; optimized decoding of indirect socket syscalls; +arch: Nios II; new syscalls; syscalls enhancements; ioctls enhancements. |
| 31.05.2016 | **4.12**: Simultaneous -p option and command tracing; caching of netlink conversations; -yy for netlink socket descriptors; BTRFS_* and UFFDIO_* ioctl decoding. new syscalls; syscalls enhancements; ioctls enhancements. |
| 26.07.2016 | **4.13**: general netlink socket parser; syscalls enhancements. |

## Since revision control, 2009 – 2017, Dmitry Levin

| | |
|---|---|
| 04.10.2016 | **4.14**: +arch: RISC-V; syscalls enhancements. |
| 14.12.2016 | **4.15**: Time stamps in ISO 8601; syscall fault injection; DM_* ioctl decoding; decoding of attr parameter of perf_event_open syscall; new syscalls; syscalls enhancements. |
| 14.02.2017 | **4.16**: syscall return value injection; signal injection; all SG_* ioctl commands are now decoded; Enhanced decoding of kernel long type on x32 and mips n32; ustat syscall decoding; syscalls enhancements. |
| 24.05.2017 | **4.17**: % prefix for syscall classes; %*stat* syscall classes; -e trace=/regex; -e trace=?; statx syscall decoding; NS_* ioctls decoding; syscalls enhancements; ioctls enhancements. |
| 05.07.2017 | **4.18**: netlink decoding enhancements. |
| 05.09.2017 | **4.19**: netlink decoding enhancements; syscalls enhancements; ioctls enhancements. |
| 13.11.2017 | **4.20**: netlink decoding enhancements; syscalls enhancements. |

# strace.git commit count summary: 4.5.19 – 4.20

```
$ git shortlog -s v4.5.19..v4.20 |sort -rn
  3141 Dmitry V. Levin
   424 Denys Vlasenko
   319 Eugene Syromyatnikov
   197 JingPiao Chen
    92 Mike Frysinger
    60 Fei Jie
    54 Elvira Khabirova
    37 Andreas Schwab
    32 Masatake YAMATO
    21 Gleb Fotengauer-Malinovskiy
    16 Elliott Hughes
    15 Fabien Siron
    13 Jeff Mahoney
    11 H.J. Lu
    10 Chris Metcalf
     9 Wang Chao
     9 Nikolay Marchuk
     9 Edgar Kaziakhmedov
     8 Victor Krapivensky
     8 Gabriel Laskar
     7 James Hogan
     7 Anton Blanchard
     5 Steve McIntyre
     5 Keith Owens
     5 Bart Van Assche
     4 Philippe De Muyter
     4 JayRJoshi
     4 Dr. David Alan Gilbert
     4 Carmelo Amoroso
     3 Stefan Sørensen
     3 Sebastian Pipping
     3 James Cowgill
     3 Holger Hans Peter Freyther
     3 Heiko Carstens
     3 Frederik Schüler
     3 Bernhard Reutner-Fischer
     3 Abhishek Tiwari
     2 Zev Weiss
     2 Thomas De Schampheleire
     2 Stanislav Brabec
     2 Seraphime Kirkovski
     2 Roland McGrath
     2 Quentin Monnet
```

## Release stats: 4.5.20 – 4.13

Commits and authors per release

| Release | | commits | | authors | | | authors of |
|---|---|---|---|---|---|---|---|
| date | version | total | per year | recurring | total | per year | 80% of commits |
| 13.04.2010 | 4.5.20 | | | | | | |
| 15.03.2011 | 4.6 | 112 | 122 | 5 | 12 | 13 | 4 |
| 02.05.2012 | 4.7 | 400 | 353 | 3 | 11 | 10 | 2 |
| 03.06.2013 | 4.8 | 237 | 218 | 4 | 17 | 16 | 3 |
| Total | | 749 | 238 | | 33 | 11 | 2 |
| 15.08.2014 | 4.9 | 247 | 206 | 4 | 22 | 18 | 3 |
| 06.03.2015 | 4.10 | 400 | 719 | 4 | 15 | 27 | 1 |
| 21.12.2015 | 4.11 | 586 | 737 | 5 | 15 | 19 | 1 |
| 31.05.2016 | 4.12 | 799 | 1804 | 5 | 16 | 36 | 1 |
| 26.07.2016 | 4.13 | 182 | 1182 | 4 | 6 | 39 | 1 |
| Total | | 2214 | 703 | 8 | 51 | 16 | 1 |

## Table of Contents

## strace usage examples: -P, -e trace=%file

```
$ strace -e %file ls /var/empty
execve("/bin/ls", ["ls", "/var/empty"], [/* 32 vars */]) = 0
access("/etc/ld.so.preload", R_OK)      = -1 ENOENT (No such file or directory)
open("/etc/ld.so.cache", O_RDONLY)      = 3
open("/lib64/libtinfo.so.5", O_RDONLY)  = 3
open("/lib64/libselinux.so.1", O_RDONLY) = 3
open("/lib64/librt.so.1", O_RDONLY)     = 3
open("/lib64/libcap.so.2", O_RDONLY)    = 3
open("/lib64/libacl.so.1", O_RDONLY)    = 3
open("/lib64/libc.so.6", O_RDONLY)      = 3
open("/lib64/libdl.so.2", O_RDONLY)     = 3
open("/lib64/libpthread.so.0", O_RDONLY) = 3
open("/lib64/libattr.so.1", O_RDONLY)   = 3
stat("/var/empty", {st_mode=S_IFDIR|0555, st_size=4096, ...}) = 0
open("/var/empty", O_RDONLY|O_NONBLOCK|O_DIRECTORY|O_CLOEXEC) = 3
+++ exited with 0 +++

$ strace -P /etc/ld.so.cache ls /var/empty
open("/etc/ld.so.cache", O_RDONLY)      = 3
fstat(3, {st_mode=S_IFREG|0644, st_size=22446, ...}) = 0
mmap(NULL, 22446, PROT_READ, MAP_PRIVATE, 3, 0) = 0x2b7ac2ba9000
close(3)                                = 0
+++ exited with 0 +++
```

## strace usage examples: -P, -v

```
$ strace -P /var/empty ls /var/empty
stat("/var/empty", {st_mode=S_IFDIR|0555, st_size=4096, ...}) = 0
open("/var/empty", O_RDONLY|O_NONBLOCK|O_DIRECTORY|O_CLOEXEC) = 3
fcntl(3, F_GETFD)                       = 0x1 (flags FD_CLOEXEC)
getdents(3, /* 2 entries */, 32768)     = 48
getdents(3, /* 0 entries */, 32768)     = 0
close(3)                                = 0
+++ exited with 0 +++
```

```
stat("/var/empty", {st_dev=makedev(253, 1), st_ino=415895, st_mode=S_IFDIR|0555,
  st_nlink=2, st_uid=0, st_gid=0, st_blksize=4096, st_blocks=8, st_size=4096,
  st_atime=1510683874 /* 2017-11-14T19:24:34.004085550+0100 */, st_atime_nsec=4085550,
  st_mtime=1510683874 /* 2017-11-14T19:24:34.004085550+0100 */, st_mtime_nsec=4085550,
  st_ctime=1510683874 /* 2017-11-14T19:24:34.004085550+0100 */, st_ctime_nsec=4085550}) = 0
openat(AT_FDCWD, "/var/empty", O_RDONLY|O_NONBLOCK|O_DIRECTORY|O_CLOEXEC) = 3
getdents(3, [{d_ino=415895, d_off=5602493789890385846, d_reclen=24, d_name=".",
  d_type=DT_DIR}, {d_ino=390917, d_off=9223372036854775807, d_reclen=24, d_name="..",
  d_type=DT_DIR}], 32768) = 48
getdents(3, [], 32768)                  = 0
close(3)                                = 0
+++ exited with 0 +++
```

## strace usage examples: -y, -e trace=

```
$ strace -e fstat,close -y ls /var/empty >/dev/null
fstat(3</etc/ld.so.cache>, {st_mode=S_IFREG|0644, st_size=22446, ...}) = 0
close(3</etc/ld.so.cache>)              = 0
fstat(3</lib64/libtinfo.so.5.7>, {st_mode=S_IFREG|0644, st_size=135352, ...}) = 0
close(3</lib64/libtinfo.so.5.7>)        = 0
fstat(3</lib64/libselinux.so.1>, {st_mode=S_IFREG|0644, st_size=121992, ...}) = 0
close(3</lib64/libselinux.so.1>)        = 0
fstat(3</lib64/librt-2.11.3.so>, {st_mode=S_IFREG|0755, st_size=31792, ...}) = 0
close(3</lib64/librt-2.11.3.so>)        = 0
fstat(3</lib64/libcap.so.2.16>, {st_mode=S_IFREG|0644, st_size=23048, ...}) = 0
close(3</lib64/libcap.so.2.16>)         = 0
fstat(3</lib64/libacl.so.1.1.0>, {st_mode=S_IFREG|0644, st_size=35376, ...}) = 0
close(3</lib64/libacl.so.1.1.0>)        = 0
fstat(3</lib64/libc-2.11.3.so>, {st_mode=S_IFREG|0755, st_size=1452024, ...}) = 0
close(3</lib64/libc-2.11.3.so>)         = 0
fstat(3</lib64/libdl-2.11.3.so>, {st_mode=S_IFREG|0755, st_size=14776, ...}) = 0
close(3</lib64/libdl-2.11.3.so>)        = 0
fstat(3</lib64/libpthread-2.11.3.so>, {st_mode=S_IFREG|0755, st_size=138064, ...}) = 0
close(3</lib64/libpthread-2.11.3.so>)   = 0
fstat(3</lib64/libattr.so.1.1.0>, {st_mode=S_IFREG|0644, st_size=18704, ...}) = 0
close(3</lib64/libattr.so.1.1.0>)       = 0
close(3</var/empty>)                    = 0
close(1</dev/null>)                     = 0
close(2</dev/pts/0>)                    = 0
+++ exited with 0 +++
```

## strace usage examples: -y, -e trace=, -e read=

```
$ strace -e trace=read -e read=3 -y ls /var/empty
read(3</lib64/libtinfo.so.5.7>, "\177ELF\2\1\1\0\0\0\0\0\0\0\0\0\3\0>\0\1\0\0\0\300\315\0\0\0\0\0\0"..., 832) = 832
 | 00000  7f 45 4c 46 02 01 01 00  00 00 00 00 00 00 00 00  .ELF.... ........ |
 | 00010  03 00 3e 00 01 00 00 00  c0 cd 00 00 00 00 00 00  ..>..... ........ |
 | 00320  00 00 00 00 00 00 00 00  00 00 00 00 4d 00 00 00  ........ ....M... |
 | 00330  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ........ ........ |
read(3</lib64/libselinux.so.1>, "\177ELF\2\1\1\0\0\0\0\0\0\0\0\0\3\0>\0\1\0\0\0\240W\0\0\0\0\0\0"..., 832) = 832
 | 00000  7f 45 4c 46 02 01 01 00  00 00 00 00 00 00 00 00  .ELF.... ........ |
 | 00010  03 00 3e 00 01 00 00 00  a0 57 00 00 00 00 00 00  ..>..... .W...... |
 | 00320  40 20 00 00 20 00 00 80  c8 84 e2 00 00 12 00 03  @ .. ... ........ |
 | 00330  20 40 02 21 80 50 02 21  70 00 00 00 71 00 00 00   @.!.P.! p...q... |
read(3</lib64/librt-2.11.3.so>, "\177ELF\2\1\1\0\0\0\0\0\0\0\0\0\3\0>\0\1\0\0\0\200!\0\0\0\0\0\0"..., 832) = 832
 | 00000  7f 45 4c 46 02 01 01 00  00 00 00 00 00 00 00 00  .ELF.... ........ |
 | 00010  03 00 3e 00 01 00 00 00  80 21 00 00 00 00 00 00  ..>..... .!...... |
 | 00320  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ........ ........ |
 | 00330  00 00 00 00 48 00 00 00  00 00 00 00 49 00 00 00  ....H... ....I... |
read(3</lib64/libcap.so.2.16>, "\177ELF\2\1\1\0\0\0\0\0\0\0\0\0\3\0>\0\1\0\0\0@\30\0\0\0\0\0\0"..., 832) = 832
 | 00000  7f 45 4c 46 02 01 01 00  00 00 00 00 00 00 00 00  .ELF.... ........ |
 | 00010  03 00 3e 00 01 00 00 00  40 18 00 00 00 00 00 00  ..>..... @....... |
 | 00320  89 71 ee b2 ee 3e 3c d4  dd e7 a8 99 18 bf 5b 17  .q...><. ......[. |
 | 00330  7d dd 81 63 ed 16 0b 88  4d a7 3a ea f5 3e 3c d4  }..c.... M.:..><. |
read(3</lib64/libacl.so.1.1.0>, "\177ELF\2\1\1\0\0\0\0\0\0\0\0\0\3\0>\0\1\0\0\0\240\37\0\0\0\0\0\0"..., 832) = 832
 | 00000  7f 45 4c 46 02 01 01 00  00 00 00 00 00 00 00 00  .ELF.... ........ |
 | 00010  03 00 3e 00 01 00 00 00  a0 1f 00 00 00 00 00 00  ..>..... ........ |
 | 00320  47 00 00 00 00 00 00 00  48 00 00 00 4a 00 00 00  G....... H...J... |
 | 00330  00 00 00 00 4b 00 00 00  00 00 00 00 00 00 00 00  ....K... ........ |
read(3</lib64/libc-2.11.3.so>, "\177ELF\2\1\1\3\0\0\0\0\0\0\0\0\3\0>\0\1\0\0\0\360\354\1\0\0\0\0\0"..., 832) = 832
 | 00000  7f 45 4c 46 02 01 01 03  00 00 00 00 00 00 00 00  .ELF.... ........ |
 | 00010  03 00 3e 00 01 00 00 00  f0 ec 01 00 00 00 00 00  ..>..... ........ |
 | 00320  80 ca 44 42 28 00 06 80  10 18 42 00 20 40 80 00  ..DB(... ..B. @.. |
 | 00330  09 50 00 51 8a 40 10 00  00 00 00 08 00 00 11 10  .P.Q.@.. ........ |
read(3</lib64/libdl-2.11.3.so>, "\177ELF\2\1\1\0\0\0\0\0\0\0\0\0\3\0>\0\1\0\0\0\340\r\0\0\0\0\0\0"..., 832) = 832
 | 00000  7f 45 4c 46 02 01 01 00  00 00 00 00 00 00 00 00  .ELF.... ........ |
 | 00010  03 00 3e 00 01 00 00 00  e0 0d 00 00 00 00 00 00  ..>..... ........ |
 | 00320  91 21 fc f8 06 02 04 f9  fb 33 fb 0f f9 19 73 42  .!...... .3....sB |
 | 00330  fa 19 73 42 95 b3 5f 19  7f 9e d0 18 61 a2 92 06  ..sB.._. ....a... |
read(3</lib64/libpthread-2.11.3.so>, "\177ELF\2\1\1\0\0\0\0\0\0\0\0\0\3\0>\0\1\0\0\0\360Y\0\0\0\0\0\0"..., 832) = 832
 | 00000  7f 45 4c 46 02 01 01 00  00 00 00 00 00 00 00 00  .ELF.... ........ |
 | 00010  03 00 3e 00 01 00 00 00  f0 59 00 00 00 00 00 00  ..>..... .Y...... |
 | 00320  01 05 00 50 20 a9 02 07  28 00 00 82 04 98 40 04  ...P ... (.....@. |
 | 00330  00 10 e0 54 00 02 40 02  02 10 c1 30 44 02 80 00  ...T..@. ...0D... |
read(3</lib64/libattr.so.1.1.0>, "\177ELF\2\1\1\0\0\0\0\0\0\0\0\0\3\0>\0\1\0\0\0\260\23\0\0\0\0\0\0"..., 832) = 832
 | 00000  7f 45 4c 46 02 01 01 00  00 00 00 00 00 00 00 00  .ELF.... ........ |
 | 00010  03 00 3e 00 01 00 00 00  b0 13 00 00 00 00 00 00  ..>..... ........ |
 | 00320  bf a8 e3 f8 db 0c 16 89  bb e3 92 7c c5 e8 1b 9b  ........ ...|.... |
```

## strace usage examples: -r, -e trace=%process

```
$ strace -r /bin/true
     0.000000 execve("/bin/true", ["/bin/true"], [/* 32 vars */]) = 0
     0.000281 exit_group(0)                  = ?
     0.000063 +++ exited with 0 +++

# strace -r -e %process unshare -i /bin/true
     0.000000 execve("/usr/bin/unshare",
       ["/usr/bin/unshare", "-i", "/bin/true"], [/* 32 vars */]) = 0
     0.000899 arch_prctl(ARCH_SET_FS, 0x7f4e537cd700) = 0
     0.000398 unshare(CLONE_NEWIPC)      = 0
     0.000190 execve("/bin/true", ["/bin/true"], [/* 32 vars */]) = 0
     0.000186 exit_group(0)                  = ?
     0.028931 +++ exited with 0 +++
```

## strace usage examples: -r, -T, -F, -e trace=%process

```
$ strace -e %process -r -T sh -c 'kill $$'
     0.000000 execve("/bin/sh", ["sh", "-c", "kill $$"], [/* 32 vars */]) = 0 <0.000361>
     0.001185 arch_prctl(ARCH_SET_FS, 0x2b0c3236b020) = 0 <0.000008>
     0.002239 --- SIGTERM {si_signo=SIGTERM, si_code=SI_USER, si_pid=12345, si_uid=500} ---
     0.000218 +++ killed by SIGTERM +++
```

```
$ strace -e %process -f -q sh -c 'sleep 1 & pid=$!; sleep 0.1; kill $pid; wait'
execve("/bin/sh", ["sh", "-c", "sleep 1 & pid=$!; sleep 0.1; kil"...], [/* 32 vars */]) = 0
arch_prctl(ARCH_SET_FS, 0x2ae37beef020) = 0
clone(child_stack=0, flags=CLONE_CHILD_CLEARTID|CLONE_CHILD_SETTID|SIGCHLD, child_tidptr=0x2ae37beef2f0) = 10001
[pid 10000] clone(child_stack=0, flags=CLONE_CHILD_CLEARTID|CLONE_CHILD_SETTID|SIGCHLD, child_tidptr=0x2ae37beef2f0) = 10002
[pid 10001] execve("/bin/sleep", ["sleep", "1"], [/* 32 vars */] <unfinished ...>
[pid 10002] execve("/bin/sleep", ["sleep", "0.1"], [/* 32 vars */] <unfinished ...>
[pid 10001] <... execve resumed> )      = 0
[pid 10002] <... execve resumed> )      = 0
[pid 10000] wait4(-1,  <unfinished ...>
[pid 10001] arch_prctl(ARCH_SET_FS, 0x2b8cf7d49b20) = 0
[pid 10002] arch_prctl(ARCH_SET_FS, 0x2ada74416b20) = 0
[pid 10002] exit_group(0)               = ?
[pid 10002] +++ exited with 0 +++
[pid 10000] <... wait4 resumed> [{WIFEXITED(s) && WEXITSTATUS(s) == 0}], 0, NULL) = 10002
[pid 10000] --- SIGCHLD {si_signo=SIGCHLD, si_code=CLD_EXITED, si_pid=10002, si_status=0, si_utime=0, si_stime=0} ---
[pid 10000] wait4(-1, 0x7fff7560e53c, WNOHANG, NULL) = 0
[pid 10001] --- SIGTERM {si_signo=SIGTERM, si_code=SI_USER, si_pid=10000, si_uid=600} ---
[pid 10001] +++ killed by SIGTERM +++
--- SIGCHLD {si_signo=SIGCHLD, si_code=CLD_KILLED, si_pid=10001, si_status=SIGTERM, si_utime=1, si_stime=0} ---
wait4(-1, [{WIFSIGNALED(s) && WTERMSIG(s) == SIGTERM}], WNOHANG, NULL) = 10001
wait4(-1, 0x7fff7560e58c, WNOHANG, NULL) = -1 ECHILD (No child processes)
exit_group(0)                           = ?
+++ exited with 0 +++
```

# strace usage examples: -ff, -ttt, -o, strace-log-merge

```
$ strace -e process -ff -ttt -o log sh -c 'sleep 1 & pid=$!; sleep 0.1; kill $pid; wait'
sh: line 1: 10001 Terminated              sleep 1
```

```
$ head -1 log.*
==> log.10000 <==
1342993484.722384 execve("/bin/sh", ["sh", "-c", "sleep 1 & pid=$!; sleep 0.1; kil"...], [/* 32 vars */]) = 0
==> log.10001 <==
1342993484.727498 execve("/bin/sleep", ["sleep", "1"], [/* 32 vars */]) = 0
==> log.10002 <==
1342993484.727422 execve("/bin/sleep", ["sleep", "0.1"], [/* 32 vars */]) = 0
```

```
$ strace-log-merge log
10000 1342993484.722384 execve("/bin/sh", ["sh", "-c", "sleep 1 & pid=$!; sleep 0.1; kil"...], [/* 33 vars */]) = 0
10000 1342993484.723369 arch_prctl(ARCH_SET_FS, 0x2ad5cc1fa020) = 0
10000 1342993484.725378 clone(child_stack=0, flags=CLONE_CHILD_CLEARTID|CLONE_CHILD_SETTID|SIGCHLD,
  child_tidptr=0x2ad5cc1fa2f0) = 10001
10000 1342993484.726783 clone(child_stack=0, flags=CLONE_CHILD_CLEARTID|CLONE_CHILD_SETTID|SIGCHLD,
  child_tidptr=0x2ad5cc1fa2f0) = 10002
10000 1342993484.727188 wait4(-1, [{WIFEXITED(s) && WEXITSTATUS(s) == 0}], 0, NULL) = 10002
10002 1342993484.727422 execve("/bin/sleep", ["sleep", "0.1"], [/* 32 vars */]) = 0
10001 1342993484.727498 execve("/bin/sleep", ["sleep", "1"], [/* 32 vars */]) = 0
10002 1342993484.769744 arch_prctl(ARCH_SET_FS, 0x2acee796db20) = 0
10001 1342993484.769845 arch_prctl(ARCH_SET_FS, 0x2b2bd019cb20) = 0
10002 1342993484.872233 exit_group(0)         = ?
10002 1342993484.872389 +++ exited with 0 +++
10000 1342993484.872492 --- SIGCHLD {si_signo=SIGCHLD, si_code=CLD_EXITED, si_pid=10002, si_status=0, si_utime=0,
  si_stime=0} ---
10000 1342993484.872519 wait4(-1, 0x7fffe27a860c, WNOHANG, NULL) = 0
10001 1342993484.872666 --- SIGTERM {si_signo=SIGTERM, si_code=SI_USER, si_pid=10000, si_uid=600} ---
10000 1342993484.872795 wait4(-1, [{WIFSIGNALED(s) && WTERMSIG(s) == SIGTERM}], 0, NULL) = 10001
10001 1342993484.872849 +++ killed by SIGTERM +++
10000 1342993484.873117 --- SIGCHLD {si_signo=SIGCHLD, si_code=CLD_KILLED, si_pid=10001, si_status=SIGTERM,
  si_utime=0, si_stime=0} ---
10000 1342993484.873140 wait4(-1, 0x7fffe27a81bc, WNOHANG, NULL) = -1 ECHILD (No child processes)
10000 1342993484.873339 exit_group(0)         = ?
10000 1342993484.873599 +++ exited with 0 +++
```

## strace usage examples: -o pipeline

```
$ strace -e open -o '|grep /lib' ls /var/empty
open("/lib64/libtinfo.so.5", O_RDONLY)  = 3
open("/lib64/libselinux.so.1", O_RDONLY) = 3
open("/lib64/librt.so.1", O_RDONLY)     = 3
open("/lib64/libcap.so.2", O_RDONLY)    = 3
open("/lib64/libacl.so.1", O_RDONLY)    = 3
open("/lib64/libc.so.6", O_RDONLY)      = 3
open("/lib64/libdl.so.2", O_RDONLY)     = 3
open("/lib64/libpthread.so.0", O_RDONLY) = 3
open("/lib64/libattr.so.1", O_RDONLY)   = 3

$ strace -e desc -y -o "|grep '</[^l]'" ls /var/empty
fstat(3</etc/ld.so.cache>, {st_mode=S_IFREG|0644, st_size=22446, ...}) = 0
mmap(NULL, 22446, PROT_READ, MAP_PRIVATE, 3</etc/ld.so.cache>, 0) = 0x2ab097dfb000
close(3</etc/ld.so.cache>)              = 0
ioctl(1</dev/pts/0>, SNDCTL_TMR_TIMEBASE or SNDRV_TIMER_IOCTL_NEXT_DEVICE or TCGETS, {B38400 opost is
ioctl(1</dev/pts/0>, TIOCGWINSZ, {ws_row=46, ws_col=128, ws_xpixel=1408, ws_ypixel=828}) = 0
fcntl(3</var/empty>, F_GETFD)           = 0x1 (flags FD_CLOEXEC)
getdents(3</var/empty>, /* 2 entries */, 32768) = 48
getdents(3</var/empty>, /* 0 entries */, 32768) = 0
close(3</var/empty>)                    = 0
close(1</dev/pts/0>)                    = 0
close(2</dev/pts/0>)                    = 0
```

## strace usage examples: -p

```
$ sleep 1 & sleep 1 & sleep 0.1 &&
  strace -e process -p "$(pidof sleep)"
[1] 10000
[2] 10001
Process 10001 attached
Process 10000 attached
[pid 10000] exit_group(0)                 = ?
[pid 10001] exit_group(0)                 = ?
[pid 10001] +++ exited with 0 +++
[2]+  Done                      sleep 1
+++ exited with 0 +++
[1]-  Done                      sleep 1
```

## strace usage examples: -c, -S

```
$ strace -c -S calls find /usr/share/doc/ > /dev/null
% time     seconds  usecs/call     calls    errors syscall
------ ----------- ----------- --------- --------- ----------------
  1.77    0.000023           0      6417         1 fcntl
  1.85    0.000024           0      1992           close
 93.83    0.001216           1       982           getdents
  1.70    0.000022           0       982           newfstatat
  0.00    0.000000           0       520           fstat
  0.85    0.000011           0       511           openat
  0.00    0.000000           0        60           write
  0.00    0.000000           0        23           mmap
  0.00    0.000000           0        14           mprotect
  0.00    0.000000           0         9           open
  0.00    0.000000           0         8           read
  0.00    0.000000           0         6           brk
  0.00    0.000000           0         6           fadvise64
  0.00    0.000000           0         3           munmap
  0.00    0.000000           0         3         2 ioctl
  0.00    0.000000           0         2           rt_sigaction
  0.00    0.000000           0         2         1 futex
  0.00    0.000000           0         1           rt_sigprocmask
  0.00    0.000000           0         1         1 access
  0.00    0.000000           0         1           execve
  0.00    0.000000           0         1           fchdir
  0.00    0.000000           0         1           gettimeofday
  0.00    0.000000           0         1           statfs
  0.00    0.000000           0         1           arch_prctl
  0.00    0.000000           0         1           set_tid_address
  0.00    0.000000           0         1           set_robust_list
------ ----------- ----------- --------- --------- ----------------
100.00    0.001296                 11549         5 total
```

## System call tampering and fault injection

**traditional syscall fault injection**

-e fault=*set*[:error=*errno*][:when=*expr*]

**strace -a28 -e trace=open**
**-e fault=*open*:when=*3*:error=EACCES cat /dev/null**

```
open("/etc/ld.so.cache", O_RDONLY|O_CLOEXEC) = 3
open("/lib64/libc.so.6", O_RDONLY|O_CLOEXEC) = 3
open("/dev/null", O_RDONLY) = -1 EACCES (Permission denied) (INJECTED)
cat: /dev/null: Permission denied
+++ exited with 1 +++
```

## System call tampering and fault injection

**syscall tampering improvements**

- return value injection
- signal injection

-e inject=*set*[:error=*errno*|:retval=*value*][:signal=*sig*][:when=*expr*]

**strace -e trace=open**
**-e fault=*open*:when=*3*:retval=42 cat /dev/null**

```
open("/etc/ld.so.cache", O_RDONLY|O_CLOEXEC) = 3
open("/lib64/libc.so.6", O_RDONLY|O_CLOEXEC) = 3
open("/dev/null", O_RDONLY)                 = 42 (INJECTED)
cat: /dev/null: Bad file descriptor
cat: /dev/null: Bad file descriptor
+++ exited with 1 +++
```

# System call tampering and fault injection

**syscall tampering improvements**

- return value injection
- signal injection

-e inject=*set*[:error=*errno*|:retval=*value*][:signal=*sig*][:when=*expr*]

**strace -a20 -P precious.txt
-e fault=unlink:error=EACCES:signal=ABRT
unlink precious.txt**

```
unlink("precious.txt") = -1 EACCES (Permission denied) (INJECTED)
--- SIGABRT si_signo=SIGABRT, si_code=SI_KERNEL ---
+++ killed by SIGABRT (core dumped) +++
```

## System call specification improvements

### syscall classes now have % prefix

strace -e trace=%*class*

### added new syscall classes

%stat, %lstat, %fstat, %%stat, %statfs, %fstatfs, %%statfs

### strace -y -e %%stat ls /var/empty

```
fstat(3</etc/ld.so.cache>, st_mode=S_IFREG|0644, st_size=30341, ...) = 0
...
fstat(3</proc/filesystems>, st_mode=S_IFREG|0444, st_size=0, ...) = 0
stat("/var/empty", st_mode=S_IFDIR|0555, st_size=40, ...) = 0
fstat(3</var/empty>, st_mode=S_IFDIR|0555, st_size=40, ...) = 0
+++ exited with 0 +++
```

## System call specification improvements

**added support of regular expressions**

strace -e trace=/*regexp*

**strace -e 'trace=/^(.*_)?statv?fs' df / >/dev/null**

```
statfs("/", f_type=TMPFS_MAGIC, f_bsize=4096, f_blocks=29042980, f_bfree=19704764
  f_bavail=19704764, f_files=16501693, f_ffree=15397338, f_fsid=val=[0, 0],
  f_namelen=255, f_frsize=4096, f_flags=ST_VALID|ST_NOSUID|ST_RELATIME) = 0
+++ exited with 0 +++
```

**strace -e %statfs df / >/dev/null**

```
statfs("/", f_type=TMPFS_MAGIC, f_bsize=4096, f_blocks=29042980, f_bfree=19704764
  f_bavail=19704764, f_files=16501693, f_ffree=15397338, f_fsid=val=[0, 0],
  f_namelen=255, f_frsize=4096, f_flags=ST_VALID|ST_NOSUID|ST_RELATIME) = 0
+++ exited with 0 +++
```

## System call specification improvements

**added support of conditional descriptions**

strace -e trace=?*set*

**strace -e trace=?statx tests/statx**

```
statx(AT_FDCWD, "/dev/full", AT_STATX_SYNC_AS_STAT,
STATX_ALL, stx_mask=STATX_BASIC_STATS, stx_attributes=0,
stx_mode=S_IFCHR|0666, stx_size=0, ...) = 0
```

**strace -e trace=/statx does not work**

```
strace: invalid system call '/statx'
```

## before: strace -e trace=sendto,recvmsg ip route list

```
sendto(3, {{len=40, type=0x1a /* NLMSG_??? */, flags=NLM_F_REQUEST|0x300, seq=1507842992, pid=0},
"\2\0\0\0\0\0\0\0\0\0\0\0\0\0\0\0\10\0\35\0\1\0\0\0" }, 40, 0, NULL, 0) = 40

recvmsg(3, {msg_name={sa_family=AF_NETLINK, nl_pid=0, nl_groups=00000000}, msg_namelen=12, msg_iov=[{iov_base=[
{{len=60, type=0x18 /* NLMSG_??? */, flags=NLM_F_MULTI, seq=1507842992, pid=23852},
"\2 \0\0\377\2\375\3\0\0\0\0\10\0\17\0\377\0\0\0\10\0\1\0\177\0\0\0\10\0\7\0"...}, {{len=60, type=0x18 /* NLMSG_???
*/, flags=NLM_F_MULTI, seq=1507842992, pid=23852},
"\2\10\0\0\377\2\376\2\0\0\0\0\10\0\17\0\377\0\0\0\10\0\1\0\177\0\0\0\10\0\7\0"...}, {{len=60, type=0x18 /*
NLMSG_??? */, flags=NLM_F_MULTI, seq=1507842992, pid=23852},
"\2 \0\0\377\2\376\2\0\0\0\0\10\0\17\0\377\0\0\0\10\0\1\0\177\0\0\1\10\0\7\0"...}, {{len=60, type=0x18 /* NLMSG_???
*/, flags=NLM_F_MULTI, seq=1507842992, pid=23852},
"\2 \0\0\377\2\375\3\0\0\0\0\10\0\17\0\377\0\0\0\10\0\1\0\177\377\377\377\10\0\7\0"...} ], iov_len=32768}],
msg_iovlen=1, msg_controllen=0, msg_flags=0}, 0) = 240

...
```

## after: strace -e trace=sendto,recvmsg ip route list

```
sendto(3, {{len=40, type=RTM_GETROUTE, flags=NLM_F_REQUEST|NLM_F_DUMP, seq=1357924680, pid=0},
{rtm_family=AF_UNSPEC, rtm_dst_len=0, rtm_src_len=0, rtm_tos=0, rtm_table=RT_TABLE_UNSPEC,
rtm_protocol=RTPROT_UNSPEC, rtm_scope=RT_SCOPE_UNIVERSE, rtm_type=RTN_UNSPEC, rtm_flags=0}, {nla_len=0,
nla_type=RTA_UNSPEC}}, 40, 0, NULL, 0) = 40

recvmsg(3, {msg_name={sa_family=AF_NETLINK, nl_pid=0, nl_groups=00000000}, msg_namelen=12, msg_iov=[{iov_base=[ {{len=60,
type=RTM_NEWROUTE, flags=NLM_F_MULTI, seq=1357924680, pid=12345}, {rtm_family=AF_INET, rtm_dst_len=32, rtm_src_len=0,
rtm_tos=0, rtm_table=RT_TABLE_LOCAL, rtm_protocol=RTPROT_KERNEL, rtm_scope=RT_SCOPE_LINK,
rtm_type=RTN_BROADCAST, rtm_flags=0}, [{{nla_len=8, nla_type=RTA_TABLE}, RT_TABLE_LOCAL}, {{nla_len=8,
nla_type=RTA_DST}, 127.0.0.0}, {{nla_len=8, nla_type=RTA_PREFSRC}, 127.0.0.1}, {{nla_len=8, nla_type=RTA_OIF},
if_nametoindex("lo")}]}, {{len=60, type=RTM_NEWROUTE, flags=NLM_F_MULTI, seq=1357924680, pid=12345},
{rtm_family=AF_INET, rtm_dst_len=8, rtm_src_len=0, rtm_tos=0, rtm_table=RT_TABLE_LOCAL, rtm_protocol=RTPROT_KERNEL,
rtm_scope=RT_SCOPE_HOST, rtm_type=RTN_LOCAL, rtm_flags=0}, [{{nla_len=8, nla_type=RTA_TABLE}, RT_TABLE_LOCAL},
{{nla_len=8, nla_type=RTA_DST}, 127.0.0.0}, {{nla_len=8, nla_type=RTA_PREFSRC}, 127.0.0.1}, {{nla_len=8, nla_type=RTA_OIF},
if_nametoindex("lo")}]}, {{len=60, type=RTM_NEWROUTE, flags=NLM_F_MULTI, seq=1357924680, pid=12345},
{rtm_family=AF_INET, rtm_dst_len=32, rtm_src_len=0, rtm_tos=0, rtm_table=RT_TABLE_LOCAL,
rtm_protocol=RTPROT_KERNEL, rtm_scope=RT_SCOPE_HOST, rtm_type=RTN_LOCAL, rtm_flags=0}, [{{nla_len=8,
nla_type=RTA_TABLE}, RT_TABLE_LOCAL}, {{nla_len=8, nla_type=RTA_DST}, 127.0.0.1}, {{nla_len=8,
nla_type=RTA_PREFSRC}, 127.0.0.1}, {{nla_len=8, nla_type=RTA_OIF}, if_nametoindex("lo")}]}, {{len=60, type=RTM_NEWROUTE,
flags=NLM_F_MULTI, seq=1357924680, pid=12345}, {rtm_family=AF_INET, rtm_dst_len=32, rtm_src_len=0, rtm_tos=0,
rtm_table=RT_TABLE_LOCAL, rtm_protocol=RTPROT_KERNEL, rtm_scope=RT_SCOPE_LINK, rtm_type=RTN_BROADCAST,
rtm_flags=0}, [{{nla_len=8, nla_type=RTA_TABLE}, RT_TABLE_LOCAL}, {{nla_len=8, nla_type=RTA_DST}, 127.255.255.255},
{{nla_len=8, nla_type=RTA_PREFSRC}, 127.0.0.1}, {{nla_len=8, nla_type=RTA_OIF}, if_nametoindex("lo")}]} ], iov_len=32768}],
msg_iovlen=1, msg_controllen=0, msg_flags=0}, 0) = 240

...
```

## Test suite

- Contains 491 tests in 4.20 (up from 5 in 4.8, in 2013)
- Revealed several bugs in Linux, glibc, musl, grep

## Table of Contents

## Tracing

- ptrace(2) API (ab)uses standard Unix parent/child signaling over waitpid.
- Tracer runs in a loop, waiting for children (strace.c:next_event)
- Tracer receives ptrace events via waitpid API, parses them (there is a lot of peculiarities, most of them related to the old kernels support; but having `PTRACE_SEIZE` and `PTRACE_O_TRACESYSGOOD` available relieves from most of them) and returns new tracing state
- Tracer performs an action on tracee using ptrace(PTRACE_name, tid, ...) based on the current stop (in strace.c:dispatch)

## Syscall decoding

- On receive of ptrace-syscall-stop signal from tracee, strace has to decide whether it is a syscall of interest and decode it, if needed.
- It uses `get_regs()` and `arch_get_scno()` functions that are implemented for each supported architecture and have knowledge how to obtain syscall number and arguments[1].
- strace also has table of syscalls (`syscallent.h`) that contains information about each system call for each supported architecture, indexed by syscall number. Each record contains its "normalised name", unique number (so we can perform syscall-type filtering, path tracing and stuff based on this ID), and pointer to the decoder function.
- While most of the decoder code is more or less architecture independent, there are still a lot of peculiarities present (like, the handling of `mask` argument of compat `fanotify_mark` on HP PA-RISC, or adidtional padding for `offset` argument of `pread` on SH, or the shuffled arguments of `fadvise64_64` syscall on ppc64, xtensa, ARM64, and ARM EABI, and so on, and so on)

[1] Except when they don't, see `int_0x80` test, for example

## Table of Contents

## Current projects

- Structured output (GSoC 2016)
- Advanced syscall filtering syntax (GSoC 2017)
- Advanced syscall tampering and filtering with Lua (GSoC 2017)
- Advanced syscall information tool (GSoC 2017)
- gdbserver backend support (Red Hat)

# Structured output

- Separate parsing and printing (duh)
  - Syscall decoders should store all the information in internal representation
  - Printers should output internal representation to the desired format
- Parsers should have only one job and be as simple as possible
- Printers should be as local as possible
- The middle layer is introduced, which backs up parsers in their workings, and calls appropriate formatter methods.

```
SYS_FUNC(swapon)
{
    unsigned int flags = tcp->u_arg[1];
    unsigned int prio = flags & SWAP_FLAG_PRIO_MASK;
    flags &= ~SWAP_FLAG_PRIO_MASK;

    printpath(tcp, tcp->u_arg[0]);
    tprints(", ");
    if (flags) {
        printflags(swap_flags, flags, "SWAP_FLAG_???");
        if (prio)
            tprintf("|%u", prio);
       } else {
           tprintf("%u", prio);
       }

    return RVAL_DECODED;
}
```

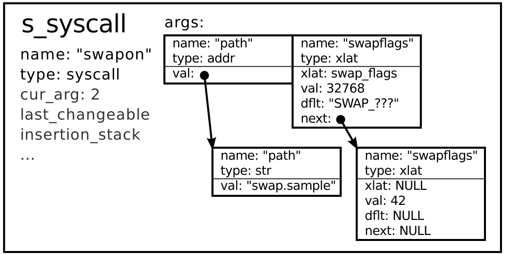
```
SYS_FUNC(swapon)
{
    s_push_path("path");
    s_push_xlat_flags_int("swapflags", swap_flags,
        NULL, SWAP_FLAG_PRIO_MASK,
        "SWAP_FLAG_???", NULL, false, 0);

    return RVAL_DECODED;
}
```

## Structured output: illustration

```
SYS_FUNC(swapon)
{
    s_push_path("path");
    s_push_xlat_flags_int("swapflags", swap_flags,
        NULL, SWAP_FLAG_PRIO_MASK,
        "SWAP_FLAG_???", NULL, false, 0);

    return RVAL_DECODED;
}
```



```
s_syscall
name: "swapon"
type: syscall
cur_arg: 2
last_changeable
insertion_stack
...
```

```
args:
name: "path"        name: "swapflags"
type: addr          type: xlat
val: ●              xlat: swap_flags
                    val: 32768
                    dflt: "SWAP_???"
                    next: ●

name: "path"        name: "swapflags"
type: str           type: xlat
val: "swap.sample"  xlat: NULL
                    val: 42
                    dflt: NULL
                    next: NULL
```

Legacy:

```
swapon("swap.sample", SWAP_FLAG_PREFER|42) =
    -1 EPERM (Operation not permitted)
```

JSON:

```
{
    "name": "swapon", "type": "syscall",
    "return": -1, "errno": 1, "errnostr": "EPERM",
    "retstring": "Operation not permitted",
    "args": [ {
            "name": "path", "type": "address",
            "addr": 4199168,
            "value": { "name": "path", "type": "str",
                "value": "swap.sample" }
        }, {
            "name": "swapflags", "type": "xlat",
            "value": [
                { "default": false, "value": 32768,
                    "str": "SWAP_FLAG_PREFER" },
                { "default": false, "value": 42 }
] } ] }
```

# Structured output: JSON output example

```
{
    "name": "getrandom",
    "type": "syscall",
    "args": [
        {
            "name": "buf",
            "type": "changeable",
            "entering_value": null,
            "exiting_value": {
                "name": "buf",
                "type": "address",
                "addr": 140722827922048,
                "value": {
                    "name": "buf",
                    "type": "str",
                    "value": "\\x26\\x4d\\x4e",
                    "size": 3,
                    "truncated": true
                }
            }
        },
        {
            "name": "count",
            "value": 3
        },
        {
            "name": "flags",
            "type": "xlat",
            "value": [
                {
                    "default": true,
                    "value": 0
                }
            ]
        }
    ],
    "return": 3
}
```

## Advanced syscall filtering syntax

**new syntax**

[*action(*]*filter_ expression*[;*arg1*[;*arg2*...]][)]

| | |
|---|---|
| *action* | is one of **trace**, **abbrev**, **verbose**, **raw**, **read**, **write**, **fault**, **inject**, or **stacktrace**; |
| *arnN* | are arguments of ***action***; |
| *filter_ expression* | is a combination of filters. |

**supported filters**

| | | |
|---|---|---|
| syscall *set* | : | set of syscalls described by ***set***; |
| fd *fd1* . . . | : | set of syscalls operating with descriptor numbers described by ***fd1*** . . . ; |
| path *path* | : | set of syscalls operating with paths described by ***path***. |

## Advanced syscall filtering syntax

echo -n foo | strace -e 'fd 1' cat >/dev/null

```
fstat(1, st_mode=S_IFCHR|0666, st_rdev=makedev(1, 3), ...) = 0
write(1, "foo", 3)                      = 3
close(1)                                = 0
+++ exited with 0 +++
```

strace -y -s4 -e 'syscall read' -e 'read(path /dev/zero)' head -c5 /dev/zero

```
read(3</lib64/libc-2.26.so>, "\177ELF"..., 832) = 832
read(3</dev/zero>, "\0\0\0\0"..., 5)    = 5
 | 00000  00 00 00 00 00                         .....            |
+++ exited with 0 +++
```

## Advanced syscall filtering syntax

strace -ve 'syscall %file and not syscall %desc' cat /dev/null

```
execve("/bin/cat", ["/bin/cat", "/dev/null"], []) = 0
access("/etc/ld.so.preload", R_OK) = -1 ENOENT (No such file or directory)
+++ exited with 0 +++
```

strace -ve 'syscall %file and !(syscall %desc || path /usr/bin/cat)' /bin/cat /dev/null

```
strace: Requested path '/usr/bin/cat' resolved into '/bin/cat'
access("/etc/ld.so.preload", R_OK) = -1 ENOENT (No such file or directory)
+++ exited with 0 +++
```

## Advanced syscall filtering syntax

**strace -k -e 'fd 1' cat /dev/null**

```
fstat(1, st_mode=S_IFCHR|0620, st_rdev=makedev(136, 5), ...) = 0
 > /lib64/libc-2.24.so(__fxstat64+0x14) [0xdab54]
 > /bin/cat() [0x1bb9]
 > /lib64/libc-2.24.so(__libc_start_main+0xf0) [0x20400]
 > /bin/cat() [0x258b]
close(1)                                = 0
 > /lib64/libc-2.24.so(_IO_file_close+0xb) [0x7195b]
 > /lib64/libc-2.24.so(_IO_file_close_it+0x13c) [0x7302c]
 > /lib64/libc-2.24.so(fclose+0x1a3) [0x669a3]
 > /bin/cat() [0x5daa]
 > /bin/cat() [0x2a92]
 > /lib64/libc-2.24.so(__locale_getenv+0x140) [0x35c60]
 > /lib64/libc-2.24.so(exit+0x1a) [0x35cba]
 > /lib64/libc-2.24.so(__libc_start_main+0xf7) [0x20407]
 > /bin/cat() [0x258b]
+++ exited with 0 +++
```

## Lua scripting

**Advanced syscall tampering and filtering with Lua**

| | |
|---|---|
| student | : Victor Krapivensky |
| mentors | : Eugene Syromyatnikov, Dmitry Levin |

**Features available with Lua scripting**

- proper system call success injection
- without breaking system call semantics
- writing into tracee's memory

## strace GSoC 2017 projects: Lua scripting

```
$ uname
Linux
$ strace -l pretend.lua -qq -etrace=none uname
NeverMindOS

$ cat pretend.lua
ffi = require 'ffi'
f = assert(io.popen([[cpp - <<EOF | grep -v '^#'
#include <sys/utsname.h>
EOF]], 'r'))
ffi.cdef(f:read('*a'))
f:close()
assert(strace.hook('uname', 'exiting', function(tcp)
if tcp.u_rval == -1 then return end
local u = assert(strace.read_obj(tcp.u_arg[0], 'struct utsname'))
ffi.copy(u.sysname, 'NeverMindOS')
assert(strace.write_obj(tcp.u_arg[0], u))
end))
```

## strace GSoC 2017 projects: syscall information tool

### Advanced syscall information tool

| | |
|---|---|
| student | : Edgar Kaziakhmedov |
| mentors | : Eugene Syromyatnikov, Dmitry Levin |

### asinfo features

- lists syscalls by various selection parameters, e.g. number, name, group, and regex
- processes any subset of many architectures supported by strace
- prints in human-readable or script-friendly format

## strace GSoC 2017 projects: syscall information tool

```
$ asinfo --set-arch x86_64 --set-abi all --get-sname %%stat
|    |              | x86_64 | x86_64 | x86_64 |
|  N | Syscall name |  64bit |    x32 |  32bit |
|  1 |        fstat |      5 |      5 |    108 |
|  2 |      fstat64 |      - |      - |    197 |
|  3 |    fstatat64 |      - |      - |    300 |
|  4 |        lstat |      6 |      6 |    107 |
|  5 |      lstat64 |      - |      - |    196 |
|  6 |   newfstatat |    262 |    262 |      - |
|  7 |     oldfstat |      - |      - |     28 |
|  8 |     oldlstat |      - |      - |     84 |
|  9 |      oldstat |      - |      - |     18 |
| 10 |         stat |      4 |      4 |    106 |
| 11 |       stat64 |      - |      - |    195 |
| 12 |        statx |    332 |    332 |    383 |
```

# strace GSoC 2017 projects: syscall information tool

```
$ asinfo --set-arch x86_64 --set-abi all --get-sname /read
|    |                 | x86_64 | x86_64 | x86_64 |
|  N |    Syscall name |  64bit |    x32 |  32bit |
|  1 | get_thread_area |    211 |      - |    244 |
|  2 |         pread64 |     17 |     17 |    180 |
|  3 |          preadv |    295 |    534 |    333 |
|  4 |         preadv2 |    327 |    546 |    378 |
|  5 | process_vm_readv |    310 |    539 |    347 |
|  6 |            read |      0 |      0 |      3 |
|  7 |       readahead |    187 |    187 |    225 |
|  8 |         readdir |      - |      - |     89 |
|  9 |        readlink |     89 |     89 |     85 |
| 10 |      readlinkat |    267 |    267 |    305 |
| 11 |           readv |     19 |    515 |    145 |
| 12 | set_thread_area |    205 |      - |    243 |
```

## Future ideas

- More elaborate parsers (especially for ioctls)
- More output formats (pcap-ng, CTF)
- Declarative syscall syntax description
- More tracing backends (EBPF)
- Better support for argument decoding in Lua (handling of structures in mpers)
- PID namespace translation
- Support for different target architectures (for gdbserver backend)

## Links

**Project page**

https://strace.io/

**strace.git**

git://git.code.sf.net/p/strace/code.git
https://github.com/strace/strace.git

**mailing list**

strace-devel@lists.sourceforge.net

**IRC channel**

#strace@freenode