Současný strace

Eugene Syromiatnikov

Brno, 2024

Co to je `strace`?

- `strace` je ladící program pro Linux, umožňující sledovaní (a případně i pozměňování) interakci komunikaci mezi procesy a linuxovým jádrem, včetně systémových volání, signálů a změn procesového stavu.
- `strace` funguje na základě systémového volání `ptrace`.

Co to je strace?

- strace je ladící program pro Linux, umožňující sledovaní (a případně i pozměňování) interakci komunikaci mezi procesy a linuxovým jádrem, včetně systémových volání, signálů a změn procesového stavu.
- strace funguje na základě systémového volání ptrace.

Příklad použití

```
$ strace echo "Hello world"
execve("/usr/bin/echo", ["echo", "Hello world"], 0x7ffd5f11c288 /* 12 vars */) = 0
brk(NULL)                               = 0x56514d3ff000
mmap(NULL, 8192, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1, 0) = 0x7fccc36c0000
access("/etc/ld.so.preload", R_OK)      = -1 ENOENT (No such file or directory)
openat(AT_FDCWD, "/etc/ld.so.cache", O_RDONLY|O_CLOEXEC) = 3
fstat(3, {st_mode=S_IFREG|0644, st_size=99278, ...}) = 0
mmap(NULL, 99278, PROT_READ, MAP_PRIVATE, 3, 0) = 0x7fccc36a7000
close(3)                                = 0
...
```

Tradiční funkce strace

Řízení trasování

- Připojení k již běžícímu procesu: **-p *pid***
- Trasovaní potomků procesů: **-f**

Filtrace systémových volání

- Trasovaní jen uvedené sady systémových volání: **-e trace=*set***
  - Podporuje použití několika speciálních jmen pro některé vymezené skupiny voláni: `process`, `network`, `signal`, `ipc`, `desc`, `memory`

Statistiky volání

- Počítání času, volání, a chyb pro každé volání: **-c**
- Seřazení tabulky volání: **-S *sortby***

Výpis instrukčního ukazatele and časového razítka

- Výpis instrukčního ukazatele: **-i**
- Výpis časového razítka: **-r**, **-t**, **-tt**, **-ttt** a **-T**

Maximální velikost a formát řetězců

- Velikost řetězce: **-s**
- Formát řetězce: **-x**, **-xx**

Podrobnost výpisu systémových volání

- Zkrácení výpisu: **-e abbrev=*set***, **-v**
- Dereference struktur: **-e verbose=*set***
- Nezpracovaný výpis volání: **-e raw=*set***

Výpis signálů

- Výpis signálů: **-e signal=*set***

Výpis I/O

- Výpis dat přečtených z uvedených deskriptorů: **-e read=*set***
- Výpis dat zapsaných do uvedených deskriptorů: **-e write=*set***

Přesměrování výstupů do souboru nebo roury

- Vystupování trasy do souboru: **-o *filename***
- Vystupování trasy do roury: **-o |*command***
- Výpis tras procesů do samostatných souborů: **-ff -o *prefix***

Příklad použití tradičních funkci strace

strace -fttTv -s 1024 echo "Hello world"

```
10:59:14.044687 execve("/usr/bin/echo", ["echo", "Hello world"], ["SHELL=/bin/bash", ↩
⟶ "LANGUAGE=en_US:en", "PWD=/esur", "LOGNAME=esyr", "HOME=/esyr", "LANG=en_US.UTF-8", ↩
⟶ "TERM=rxvt-unicode", "USER=esyr", "SHLVL=1", ↩
⟶ "PATH=/usr/local/sbin:/usr/local/bin:/usr/bin:/bin", "MAIL=/var/mail/esyr", ↩
⟶ "_=/usr/bin/strace"]) = 0 <0.000449>
10:59:14.045522 brk(NULL)               = 0x559a55a90000 <0.000137>
10:59:14.045917 mmap(NULL, 8192, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1, ↩
⟶ 0) = 0x7f40113d0000 <0.000166>
10:59:14.046322 access("/etc/ld.so.preload", R_OK) = -1 ENOENT (No such file or directory) ↩
⟶ <0.000136>
10:59:14.046869 openat(AT_FDCWD, "/etc/ld.so.cache", O_RDONLY|O_CLOEXEC) = 3 <0.000099>
10:59:14.047090 fstat(3, {st_dev=makedev(0xfe, 0x1), st_ino=2095255, st_mode=S_IFREG|0644, ↩
⟶ st_nlink=1, st_uid=0, st_gid=0, st_blksize=4096, st_blocks=200, st_size=99278, ↩
⟶ st_atime=1730518958 /* 2024-11-02T04:42:38.145457664+0100 */, st_atime_nsec=145457664, ↩
⟶ st_mtime=1730518958 /* 2024-11-02T04:42:38.137457664+0100 */, st_mtime_nsec=137457664, ↩
⟶ st_ctime=1730518958 /* 2024-11-02T04:42:38.137457664+0100 */, st_ctime_nsec=137457664}) ↩
⟶ = 0 <0.000033>
10:59:14.047301 mmap(NULL, 99278, PROT_READ, MAP_PRIVATE, 3, 0) = 0x7f40113b7000 <0.000025>
10:59:14.047377 close(3)                = 0 <0.000017>
...
```

Odkazy na přednášky o použití strace

Michael Kerrisk, "System Call Tracing with strace"

- NDC TechTown 2018; 29. srpna 2018, Kongsberg
- PDF:
  fhttps://www.man7.org/conf/ndctechtown2018/system_call_tracing_with_strace-NDC-TechTown-Kerrisk.pdf
- Youtube: https://www.youtube.com/watch?v=oFt6V56BOlo

Renaud Metrich, "Using strace to troubleshoot issues"

- Devconf.cz 2023; 17. června 2023, Brno
- PDF:
  https://static.sched.com/hosted_files/devconfcz2023/f3/Devconf%20CZ%202023%20-%20Using%20STRACE%20to%20troubleshoot%20issues.pdf
- Youtube: https://www.youtube.com/watch?v=H4BiBn-7GT8

Podpora dlouhých voleb přikazovacího řádku (od v5.6, duben 2020)

-e trace : --trace
-c : --summary-only
-f : --follow-forks
-ff : --follow-forks --output-separately
-r : --relative-timestamps[=*precision*]
-t : --absolute-timestamps
-tt : --absolute-timestamps=precision:us
-ttt : --absolute-timestamps=format:unix,precision:us
-T : --syscall-times
-v : --no-abbrev
-x : --strings-in-hex=non-ascii
-xx : --strings-in-hex[=all]

Výstup trasy [1/2]

- Rozvoj rozboru systémových volání, příkazů ioctl, a protokolů netlink
- Výpis cest k souborům u souborových deskriptorů: **-y**/**--decode-fds** (v4.7, květen 2012)
- Výpis doplňkové informace o souborových deskriptorech: **--decode-fds=[*set*]**
  - síťový protokol a adresy (**socket**, v4.12, květen 2016)
  - major/minor čísla zařízení (**dev**, v4.22, duben 2018)
  - číslo procesu u pidfd (**pidfd**, v5.6, duben 2020)
  - sada signálů u signalfd (**signalfd**, v6.3, květen 2023)
  - detaily události u eventfd (**eventfd**, v6.10, červenec 2024)
- Výpis doplňkové informace o PID: **--decode-pids=[*set*]**
  - Výpis PID procesu v jmenném prostoru PID strace: **--pidns-translation**/**--decode-pids=pidns** (v5.9, září 2019)
  - Výpis `comm` procesů, spojených s PID: **-Y**/**--decode-pids=comm** (v5.15, prosinec 2021)

Výstup trasy [2/2]

- Výpis zásobníku volání funkci: **-k**/--**stack-trace** (v4.9, srpen 2014)
  - Výpis informace o souboru a řádku v zdrojovém kódu: **-kk**/--**stack-trace=source** (v6.7, leden 2024)
- Výpis kontextů SELinux: --**secontext** (v5.12, duben 2021)
  - Výpis nesouladů mezi aktuálním kontextem a databází SELinux: --**secontext=mismatch** (v5.16, leden 2022)
- Výpis čísla systémového volání: **-n**/--**syscall-number** (v5.9, duben 2020)
- Nucení výpisu PID procesu v trase: --**always-show-pid** (v6.9, květen 2024)
- Formát výpisu jmenovaných konstant a příznaku: **-X**/--**const-print-style** (v4.23, červen 2018)
- Otevírání výstupních souboru na doplnění: **-A**/--**output-append-mode** (v4.22, duben 2018)

Filtrace systémových volání

- Souborové cesty, přístup k nímž se realizuje přes jméno nebo deskriptor: **-P** (v4.7, květen 2012)
- Návratový stav: **--status**=*set* (v5.2, červenec 2019)
- Používané deskriptory: **--trace-fds**=*set* (v6.3, květen 2023)
- Zadání systémových volání regulárními výrazy: **-e trace**=/***regexp*** (v4.17, květen 2017)
- Nepovinné specifikace systémových volání: **-e trace**=?***spec*** (v4.17, květen 2017)
- Nové skupiny systémových volání: **%clock** (v5.7, červen 2020), **%creds** (v5.5, únor 2020), **%pure** (v4.21, únor 2018), **%stat**, **%lstat**, **%fstat**, **%statfs**, **%fstatfs**, **%%stat**, **%%statfs** (v4.17, květen 2017)

Spuštění programu

- Zadaní `argv[0]` prováděného programu: **--argv0=*name*** (v6.3, květen 2023)

Řízení trasování

- Přípojení k několika procesům: **-p *pid_set*** (v4.7, květen 2012)
- Spuštění jako oddělené vnouče: **-D/--daemonize** (v4.5.19, září 2009)
  - Další možnosti oddělení: **--daemonize={pgroup|session}** (v5.4, listopad 2019)
- Odpojení po `execve`: **-b execve/--detach-on=execve** (v4.7, květen 2012)
- Odpojení po dosahu specifikovaného počtů vypsaných systémových volání: **--syscall-limit=*number*** (v6.3, květen 2023)
- Nastavení signálů, jež strace ignoruje: **-I/--interruptible** (v4.7, květen 2012)
- Ukončení tracee s ukončením strace: **--kill-on-exit** (v6.6, říjen 2023)
- Filtrace systémových volání přes `seccomp`: **--seccomp-bpf** (v5.3, září 2019)

Manipulovaní systémovými voláními

- Injekce chyb (v4.15, prosinec 2016):
  -e **inject**=*set*:**error**=*errno*[:**when**=*expr*][:**syscall**=*syscall*]
- Injekce návratných hodnot (v4.16, únor 2017):
  -e **inject**=*set*:**retval**=*value*[:**when**=*expr*][:**syscall**=*syscall*]
- Injekce signálů (v4.16, únor 2017):
  -e **inject**=*set*:**signal**=*set*
- Injekce zdržení (v4.22, duben 2018):
  -e **inject**=*set*:**delay_enter**=*usecs*
  -e **inject**=*set*:**delay_exit**=*usecs*
- Změna obsahu paměti (v5.11, únor 2021):
  -e **inject**=*set*:**poke_enter**=@*argN*=*dataN*]
  -e **inject**=*set*:**poke_exit**=@*argN*=*dataN*]

Statistiky volání

- Reálný čas ("nástěnných hodin"), strávený v systémovém voláním: **-w**/--**summary-wall-clock** (v4.9, srpen 2014)
- Kombinovaní regulárního výstupu a výpisu souhrnu volání: **-C**/--**summary** (v4.5.20, duben 2010)
- Nastavení sady sloupců tabulky souhrnu volání: **-U**/--**summary-columns**=***set*** (v5.6, duben 2020)

Různé

- Výpis rad a vychytávek: --**tips** (v5.18, červen 2022)

Některé skupiny operaci ioctl, jejichž rozbor podporuje strace

- BLK*
- BTRFS_*
- DM_*
- EV* (evdev)
- GPIO_*
- HDIO_*
- KD*
- LIRC_*
- LOOP_*
- MEM* (MTD)
- NBD_*
- NS_*
- PERF_EVENT_IOC_*
- PTP_*
- RANDOM_*
- RTC_*
- SECCOMP_*
- SIOC_*
- SG_*
- TEE_*
- UBI_*
- UFFDIO_*
- VIDIOC_*
- WDIOC_*

Rozbor operaci ioctl [2/2]

strace -v --trace=ioctl dmsetup ls

```
ioctl(3, DM_VERSION, {version=4.0.0, data_size=16384, flags=DM_EXISTS_FLAG}
⟶ => {version=4.43.0, data_size=16384, flags=DM_EXISTS_FLAG}) = 0
ioctl(3, DM_LIST_DEVICES, {version=4.0.0, data_size=16384, data_start=312,
⟶ flags=DM_EXISTS_FLAG} => {version=4.43.0, data_size=472, data_start=312,
⟶ flags=DM_EXISTS_FLAG, {dev=makedev(0xfe, 0), name="nvme0n1p3_crypt", event_nr=0},
⟶ {dev=makedev(0xfe, 0x3), name="nurgle-vg-home", event_nr=0},
⟶ {dev=makedev(0xfe, 0x1), name="nurgle-vg-root", event_nr=0},
⟶ {dev=makedev(0xfe, 0x2), name="nurgle-vg-swap_1"}}) = 0
```

https://git.kernel.org/pub/scm/linux/kernel/git/legion/kbd.git/commit/?id=1002d3b56b72

```
Author: Alexey Gladkov <gladkov.alexey@gmail.com>
Date:   Tue Apr 25 17:32:11 2023 +0200

    tests: Use strace to track syscalls

    Now strace is powerful enough to show ioctls specific to console
    configuration. Additional benefit of using strace is that we can
    enable end-to-end tests for statically linked utilities.

    Signed-off-by: Alexey Gladkov <gladkov.alexey@gmail.com>

 37 files changed, 15955 insertions(+), 105164 deletions(-)
```

Podporované v současnosti protokoly netlink

- NETLINK_AUDIT
- NETLINK_CRYPTO
- NETLINK_KOBJECT_UEVENT
- NETLINK_NETFILTER
- NETLINK_ROUTE
- NETLINK_SELINUX
- NETLINK_SOCK_DIAG
- NETLINK_XFRM
- NETLINK_GENERIC

NETLINK_ROUTE: ip route list table all

```
local 127.0.0.0/8 dev lo table local proto kernel scope host src 127.0.0.1
local 127.0.0.1 dev lo table local proto kernel scope host src 127.0.0.1
broadcast 127.255.255.255 dev lo table local proto kernel scope link src 127.0.0.1
local ::1 dev lo table local proto kernel metric 0 pref medium
```

Rozbor protokolů netlink: NETLINK_ROUTE

strace --trace=sendto,recvmsg ip route list

```
sendto(3, [[{nlmsg_len=28, nlmsg_type=RTM_GETROUTE, nlmsg_flags=NLM_F_REQUEST|NLM_F_DUMP, nlmsg_seq=135792468, nlmsg_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}], ↩
⟶ {nlmsg_len=0, nlmsg_type=0, nlmsg_flags=0, nlmsg_seq=0, nlmsg_pid=0}], 156, 0, NULL, 0) = 156

recvmsg(3, {msg_name={sa_family=AF_NETLINK, nl_pid=0, nl_groups=00000000}, msg_namelen=12, ↩
⟶ msg_iov=[{iov_base=[[{nlmsg_len=60, nlmsg_type=RTM_NEWROUTE, ↩
⟶ nlmsg_flags=NLM_F_MULTI|NLM_F_DUMP_FILTERED, nlmsg_seq=135792468, nlmsg_pid=1234567}, ↩
⟶ {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}, inet_addr("127.0.0.0")], ↩
⟶ [{nla_len=8, nla_type=RTA_PREFSRC}, inet_addr("127.0.0.1")], [{nla_len=8, nla_type=RTA_OIF}, if_nametoindex("lo")]]], ↩
⟶ [{nlmsg_len=60, nlmsg_type=RTM_NEWROUTE, nlmsg_flags=NLM_F_MULTI|NLM_F_DUMP_FILTERED, ↩
⟶ nlmsg_seq=135792468, nlmsg_pid=1234567}, ↩
⟶ {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}, inet_addr("127.0.0.1")], ↩
⟶ [{nla_len=8, nla_type=RTA_PREFSRC}, inet_addr("127.0.0.1")], [{nla_len=8, nla_type=RTA_OIF}, if_nametoindex("lo")]]], ↩
⟶ [{nlmsg_len=60, nlmsg_type=RTM_NEWROUTE, nlmsg_flags=NLM_F_MULTI|NLM_F_DUMP_FILTERED, ↩
⟶ nlmsg_seq=135792468, nlmsg_pid=1234567}, ↩
⟶ {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}, inet_addr("127.255.255.255")], ↩
⟶ [{nla_len=8, nla_type=RTA_PREFSRC}, inet_addr("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) = 180

...
```

Výpis informace o souborových deskriptorech

--decode-fds=*set*

path	: výpis souborové cesty spojené s deskriptorem a `AT_FDCWD`
socket	: výpis informace o protokolu a adresách spojené s sokety
dev	: výpis čísel znakových/blokových zařízení
pidfd	: výpis PID spojených s deskriptorem pidfd
signalfd	: výpis signálové masky spojené s deskriptorem signalfd
eventfd	: výpis informace o události spojené s deskriptorem eventfd

Implicitní hodnoty a pseudonymy

- Standardně je **--decode-fds=none**
- **-y** je pseudonym pro **--decode-fds=path**
- **-yy** je pseudonym pro **--decode-fds=all**

Výpis informace spojené s zařízeními and sokety

strace -yy -e %desc,%network nc 127.0.0.1 22 < /dev/null

```
...
socket(AF_INET, SOCK_STREAM, IPPROTO_TCP) = 3<TCP:[518663]>
connect(3<TCP:[518663]>, sa_family=AF_INET, sin_port=htons(22), ↩
⟶ sin_addr=inet_addr("127.0.0.1"), 16) = 0
poll([fd=3<TCP:[127.0.0.1:45678->127.0.0.1:22]>, events=POLLIN, ↩
⟶ fd=0</dev/null<char 1:3>>, events=POLLIN], 2, -1) = 1 ([fd=0, revents=POLLIN])
read(0</dev/null<char 1:3>>, "", 2048)  = 0
shutdown(3<TCP:[127.0.0.1:45678->127.0.0.1:22]>, SHUT_WR) = 0
poll([fd=3<TCP:[127.0.0.1:45678->127.0.0.1:22]>, events=POLLIN, fd=-1], 2, -1) ↩
⟶ = 1 ([fd=3, revents=POLLIN])
read(3<TCP:[127.0.0.1:45678->127.0.0.1:22]>, "SSH-2.0-OpenSSH_9.4\r\n", 2048) = 21
write(1</dev/pts/1<char 136:1>>, "SSH-2.0-OpenSSH_9.4\r\n", 21) = 21
poll([fd=3<TCP:[127.0.0.1:45678->127.0.0.1:22]>, events=POLLIN, fd=-1], 2, -1) ↩
⟶ = 1 ([fd=3, revents=POLLIN|POLLHUP])
read(3<TCP:[127.0.0.1:45678->127.0.0.1:22]>, "", 2048) = 0
shutdown(3<TCP:[127.0.0.1:45678->127.0.0.1:22]>, SHUT_RD) ↩
⟶ = -1 ENOTCONN (Transport endpoint is not connected)
close(3<TCP:[127.0.0.1:45678->127.0.0.1:22]>) = 0
+++ exited with 0 +++
```

Rozbor signálových masek deskriptorů signalfd: --decode-fds=signalfd

strace -q -yy –trace=/signalfd,/epoll_ctl –syscall-limit=3 \
/usr/lib/systemd/systemd-portabled

```
signalfd4(-1, [INT], 8, SFD_CLOEXEC|SFD_NONBLOCK) = 4<signalfd:[INT]>
epoll_ctl(3<anon_inode:[eventpoll]>, EPOLL_CTL_ADD, 4<signalfd:[INT]>, ↩
⟶ events=EPOLLIN, data=u32=112224560, u64=94214514829616) = 0
signalfd4(4<signalfd:[INT]>, [INT TERM], 8, SFD_CLOEXEC|SFD_NONBLOCK) ↩
⟶ = 4<signalfd:[INT TERM]>
```

Rozbor různé informace spojené s ID procesů

--decode-pids=*set*

comm : výpis jména příkazu (právě `comm`) spojeného s ID vlákna nebo procesu

pidns : výpis ID vlákna, procesu, procesové skupiny, nebo seance v jmenném prostoru strace, pokud sledovaný proces je v jiném jmenném prostoru PID

Implicitní hodnoty a pseudonymy

- Standardně je **--decode-pids=none**
- **-Y** je pseudonym pro **--decode-pids=comm**
- **--decode-pids=all** je pseudonym pro **--decode-pids=comm**,**pidns**

Výpis jmen příkazů u PID: --**decode-pids=comm**, -**Y**

strace -f -**Y** -e%process timeout 1 sleep 2

```
execve("/usr/bin/timeout", ["timeout", "1", "sleep", "2"], 0x7ffcfb9bb348 /* 17 vars */) = 0
clone(child_stack=NULL, flags=CLONE_CHILD_CLEARTID|CLONE_CHILD_SETTID|SIGCHLD, ↩
⟶ child_tidptr=0x7f116942da10) = 123452<timeout>
strace: Process 123452 attached
[pid 123451<timeout>] wait4(123452<timeout>, 0x7ffc132b1a28, WNOHANG, NULL) = 0
[pid 123452<timeout>] execve("/bin/sleep", ["sleep", "2"], 0x7ffc132b1ce0 /* 17 vars */) = 0
[pid 123451<timeout>] -- SIGALRM {si_signo=SIGALRM, si_code=SI_TIMER, ↩
⟶ si_timerid=0, si_overrun=0, si_int=0, si_ptr=NULL} --
[pid 123451<timeout>] kill(123452<sleep>, SIGTERM) = 0
[pid 123452<sleep>] -- SIGTERM {si_signo=SIGTERM, si_code=SI_USER, ↩
⟶ si_pid=123451<timeout>, si_uid=1000} --
[pid 123451<timeout>] kill(0, SIGTERM) = 0
[pid 123451<timeout>] -- SIGTERM {si_signo=SIGTERM, si_code=SI_USER, ↩
⟶ si_pid=123451<timeout>, si_uid=1000} --
[pid 123451<timeout>] kill(123452<sleep>, SIGCONT) = 0
[pid 123452<sleep>] +++ killed by SIGTERM +++
-- SIGCHLD {si_signo=SIGCHLD, si_code=CLD_KILLED, si_pid=123452<sleep>, ↩
⟶ si_uid=1000, si_status=SIGTERM, si_utime=0, si_stime=0} --
kill(0, SIGCONT)                        = 0
-- SIGCONT {si_signo=SIGCONT, si_code=SI_USER, si_pid=123451<timeout>, si_uid=1000} --
wait4(123452<sleep>, [WIFSIGNALED(s) && WTERMSIG(s) == SIGTERM], WNOHANG, NULL) = 123452
exit_group(124)                         = ?
+++ exited with 124 +++
```

Výpis jmen příkazů a překlad PID mezi jimennými prostory

strace --**decode-pids=all** -q -f --trace=clone,kill \
unshare -Urpf sh -c 'sleep 2 & sleep 1 && kill $!'

```
clone(child_stack=NULL, flags=CLONE_CHILD_CLEARTID|CLONE_CHILD_SETTID|SIGCHLD, ↩
⟶ child_tidptr=0x7ff30b283a10) = 123456<unshare>
[pid 123456<sh>] clone(child_stack=NULL, flags=CLONE_CHILD_CLEARTID|CLONE_CHILD_SETTID|SIGCHLD, ↩
⟶ child_tidptr=0x7f6dccd89a10) = 2<sh> /* 123457 in strace's PID NS */
[pid 123456<sh>] clone(child_stack=NULL, flags=CLONE_CHILD_CLEARTID|CLONE_CHILD_SETTID|SIGCHLD, ↩
⟶ child_tidptr=0x7f6dccd89a10) = 3<sh> /* 123458 in strace's PID NS */
[pid 123458<sleep>] +++ exited with 0 +++
[pid 123456<sh>] -- SIGCHLD {si_signo=SIGCHLD, si_code=CLD_EXITED, ↩
⟶ si_pid=3, si_uid=0, si_status=0, si_utime=0, si_stime=0} --
[pid 123456<sh>] kill(2<sleep> /* 123457 in strace's PID NS */, SIGTERM) = 0
[pid 123457<sleep>] -- SIGTERM {si_signo=SIGTERM, si_code=SI_USER, ↩
⟶ si_pid=1<sh> /* 123456 in strace's PID NS */, si_uid=0} --
[pid 123457<sleep>] +++ killed by SIGTERM +++
[pid 123456<sh>] +++ exited with 0 +++
-- SIGCHLD {si_signo=SIGCHLD, si_code=CLD_EXITED, si_pid=123456, ↩
⟶ si_uid=0, si_status=0, si_utime=0, si_stime=0} --
+++ exited with 0 +++
```

Výpis zásobníku volání funkci: -**k**

strace -qq -P /dev/full cat /dev/null > /dev/full

```
fstat(1, st_mode=S_IFCHR|0666, st_rdev=makedev(1, 7), ...) = 0
close(1)                                = 0
```

strace -**k** -qq -P /dev/full cat /dev/null > /dev/full

```
fstat(1, st_mode=S_IFCHR|0666, st_rdev=makedev(0x1, 0x7), ...) = 0
 > /usr/lib/x86_64-linux-gnu/libc.so.6(__fxstat+0x13) [0x107653]
 > /usr/bin/cat(main+0x1a8) [0x2548]
 > /usr/lib/x86_64-linux-gnu/libc.so.6(__libc_start_call_main+0x79) [0x29c89]
 > /usr/lib/x86_64-linux-gnu/libc.so.6(__libc_start_main@@GLIBC_2.34+0x84) [0x29d44]
 > /usr/bin/cat(_start+0x29) [0x2f09]
close(1)                                = 0
 > /usr/lib/x86_64-linux-gnu/libc.so.6(__close_nocancel+0x7) [0xfac57]
 > /usr/lib/x86_64-linux-gnu/libc.so.6(_IO_file_close_it@@GLIBC_2.2.5+0x62) [0x83492]
 > /usr/lib/x86_64-linux-gnu/libc.so.6(fclose@@GLIBC_2.2.5+0x102) [0x77932]
 > /usr/bin/cat(close_stream+0x1c) [0x631c]
 > /usr/bin/cat(close_stdout+0x12) [0x3492]
 > /usr/lib/x86_64-linux-gnu/libc.so.6(__run_exit_handlers+0x155) [0x41a35]
 > /usr/lib/x86_64-linux-gnu/libc.so.6(exit+0x19) [0x41b69]
 > /usr/lib/x86_64-linux-gnu/libc.so.6(__libc_start_call_main+0x80) [0x29c90]
 > /usr/lib/x86_64-linux-gnu/libc.so.6(__libc_start_main@@GLIBC_2.34+0x84) [0x29d44]
 > /usr/bin/cat(_start+0x29) [0x2f09]
```

Výpis místa volání v zdrojovém kódu: -**kk**

strace -qq -P /dev/full cat /dev/null > /dev/full

```
fstat(1, st_mode=S_IFCHR|0666, st_rdev=makedev(1, 7), ...) = 0
close(1)                                = 0
```

strace -**kk** -qq -P /dev/full cat /dev/null > /dev/full

```
fstat(1, st_mode=S_IFCHR|0666, st_rdev=makedev(0x1, 0x7), ...) = 0
 > /usr/lib/x86_64-linux-gnu/libc.so.6(__fxstat+0x13) [0x107653] ../sysdeps/unix/sysv/linux/fxstat64.c:51
 > /usr/bin/cat(main+0x1a8) [0x2548] /usr/include/x86_64-linux-gnu/sys/stat.h:469
 > /usr/lib/x86_64-linux-gnu/libc.so.6(__libc_start_call_main+0x79) [0x29c89] ../sysdeps/nptl/libc_start_call_main.h:58
 > /usr/lib/x86_64-linux-gnu/libc.so.6(__libc_start_main@@GLIBC_2.34+0x84) [0x29d44] ../csu/libc-start.c:360
 > /usr/bin/cat(_start+0x29) [0x2f09]
close(1)                                = 0
 > /usr/lib/x86_64-linux-gnu/libc.so.6(__close_nocancel+0x7) [0xfac57] ../sysdeps/unix/sysv/linux/close_nocancel.c:26
 > /usr/lib/x86_64-linux-gnu/libc.so.6(_IO_file_close_it@@GLIBC_2.2.5+0x62) [0x83492] ./libio/fileops.c:142
 > /usr/lib/x86_64-linux-gnu/libc.so.6(fclose@@GLIBC_2.2.5+0x102) [0x77932] ./libio/iofclose.c:53
 > /usr/bin/cat(close_stream+0x1c) [0x631c] lib/close-stream.c:60
 > /usr/bin/cat(close_stdout+0x12) [0x3492] lib/closeout.c:119
 > /usr/lib/x86_64-linux-gnu/libc.so.6(__run_exit_handlers+0x155) [0x41a35] ./stdlib/exit.c:108
 > /usr/lib/x86_64-linux-gnu/libc.so.6(exit+0x19) [0x41b69] ./stdlib/exit.c:138
 > /usr/lib/x86_64-linux-gnu/libc.so.6(__libc_start_call_main+0x80) [0x29c90] ../sysdeps/nptl/libc_start_call_main.h:74
 > /usr/lib/x86_64-linux-gnu/libc.so.6(__libc_start_main@@GLIBC_2.34+0x84) [0x29d44] ../csu/libc-start.c:360
 > /usr/bin/cat(_start+0x29) [0x2f09]
```

Výpis kontextů SELinux: --**secontext**

strace -P /dev/null cat /dev/null

```
openat(AT_FDCWD, "/dev/null", O_RDONLY) = 3
newfstatat(3, "", {st_mode=S_IFCHR|0666, st_rdev=makedev(0x1, 0x3), ...}, AT_EMPTY_PATH) = 0
fadvise64(3, 0, 0, POSIX_FADV_SEQUENTIAL) = 0
read(3, "", 131072)                       = 0
close(3)                                  = 0
+++ exited with 0 +++
```

strace --**secontext** -P /dev/null cat /dev/null

```
[unconfined_t] openat(AT_FDCWD, "/dev/null" [null_device_t], O_RDONLY) = 3 [null_device_t]
[unconfined_t] newfstatat(3 [null_device_t], "", {st_mode=S_IFCHR|0666, ↩
⟶ st_rdev=makedev(0x1, 0x3), ...}, AT_EMPTY_PATH) = 0
[unconfined_t] fadvise64(3 [null_device_t], 0, 0, POSIX_FADV_SEQUENTIAL) = 0
[unconfined_t] read(3 [null_device_t], "", 131072) = 0
[unconfined_t] close(3 [null_device_t]) = 0
+++ exited with 0 +++
```

Výpis kontextů SELinux: --secontext=mismatch

Příklad nesouladu kontextů SELinux

```
$ matchpathcon $PWD/config.h
/home/me/strace/src/config.h unconfined_u:object_r:user_home_t:s0
$ ls -Z $PWD/config.h
system_u:object_r:user_home_t:s0 /home/me/strace/src/config.h
```

strace --secontext=full,mismatch

```
$ strace -secontext=full,mismatch -trace=%file stat config.h
...
[unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023] statx(AT_FDCWD, "config.h" ↩
⟶ [system_u:object_r:user_home_t:s0!!unconfined_u:object_r:user_home_t:s0], ↩
⟶ AT_STATX_SYNC_AS_STAT|AT_SYMLINK_NOFOLLOW|AT_NO_AUTOMOUNT, ↩
⟶ STATX_ALL, stx_mask=STATX_ALL|STATX_MNT_ID, stx_attributes=0, stx_mode=S_IFREG|0644, ↩
⟶ stx_size=50008, ...) = 0
```

Formát jmenných konstant a příznaků: -X/--const-print-style

strace -e /open cat /dev/null

```
openat(AT_FDCWD, "/etc/ld.so.cache", O_RDONLY|O_CLOEXEC) = 3
openat(AT_FDCWD, "/lib64/libc.so.6", O_RDONLY|O_CLOEXEC) = 3
openat(AT_FDCWD, "/dev/null", O_RDONLY) = 3
+++ exited with 0 +++
```

strace -X verbose -e /open cat /dev/null

```
openat(-100 /* AT_FDCWD */, "/etc/ld.so.cache",
       0x80000 /* O_RDONLY|O_CLOEXEC */) = 3
openat(-100 /* AT_FDCWD */, "/lib64/libc.so.6",
       0x80000 /* O_RDONLY|O_CLOEXEC */) = 3
openat(-100 /* AT_FDCWD */, "/dev/null", 0 /* O_RDONLY */) = 3
+++ exited with 0 +++
```

strace -X raw -e /open cat /dev/null

```
openat(-100, "/etc/ld.so.cache", 0x80000) = 3
openat(-100, "/lib64/libc.so.6", 0x80000) = 3
openat(-100, "/dev/null", 0)              = 3
+++ exited with 0 +++
```

Filtrovaní podle návratného stavu systémového volání

-e status=*set*

successful	: systémové volání nevrátilo chybu, pseudonym pro **-z**
failed	: systémové volání vrátilo chybu, pseudonym pro **-Z**
unfinished	: systémové volání nevrátilo
detached	: proces byl odpojen do návratu systémového volání
unavailable	: systémové volání se vrátilo, ale se nepodařilo vyzvednout návratnou hodnotu

Standardně je **-e status=all**.

Filtrovaní podle návratného stavu systémového volání: -z, -Z

env -i LD_LIBRARY_PATH=/lib64 strace -z -e%file /bin/cat < /dev/null

```
execve("/bin/cat", ["/bin/cat"], 0x7ffc2b781ca0 /* 1 var */) = 0
openat(AT_FDCWD, "/lib64/libc.so.6", O_RDONLY|O_CLOEXEC) = 3
+++ exited with 0 +++
```

env -i LD_LIBRARY_PATH=/lib64 strace -Z -e%file /bin/cat </dev/null

```
access("/etc/ld.so.preload", R_OK) = -1 ENOENT (No such file or directory)
openat(AT_FDCWD, "/lib64/tls/x86_64/x86_64/libc.so.6", O_RDONLY|O_CLOEXEC) = -1 ENOENT (No suc
stat("/lib64/tls/x86_64/x86_64", 0x7ffdcc6a0c20) = -1 ENOENT (No such file or directory)
openat(AT_FDCWD, "/lib64/tls/x86_64/libc.so.6", O_RDONLY|O_CLOEXEC) = -1 ENOENT (No such file
stat("/lib64/tls/x86_64", 0x7ffdcc6a0c20) = -1 ENOENT (No such file or directory)
openat(AT_FDCWD, "/lib64/tls/x86_64/libc.so.6", O_RDONLY|O_CLOEXEC) = -1 ENOENT (No such file
stat("/lib64/tls/x86_64", 0x7ffdcc6a0c20) = -1 ENOENT (No such file or directory)
openat(AT_FDCWD, "/lib64/tls/libc.so.6", O_RDONLY|O_CLOEXEC) = -1 ENOENT (No such file or dire
stat("/lib64/tls", 0x7ffdcc6a0c20) = -1 ENOENT (No such file or directory)
openat(AT_FDCWD, "/lib64/x86_64/x86_64/libc.so.6", O_RDONLY|O_CLOEXEC) = -1 ENOENT (No such fi
stat("/lib64/x86_64/x86_64", 0x7ffdcc6a0c20) = -1 ENOENT (No such file or directory)
openat(AT_FDCWD, "/lib64/x86_64/libc.so.6", O_RDONLY|O_CLOEXEC) = -1 ENOENT (No such file or d
stat("/lib64/x86_64", 0x7ffdcc6a0c20) = -1 ENOENT (No such file or directory)
openat(AT_FDCWD, "/lib64/x86_64/libc.so.6", O_RDONLY|O_CLOEXEC) = -1 ENOENT (No such file or d
stat("/lib64/x86_64", 0x7ffdcc6a0c20) = -1 ENOENT (No such file or directory)
+++ exited with 0 +++
```

Filtrovaní podle návratného stavu systémového volání: agregace u -z [1/2]

strace -o log -f -e signal=none -e trace=execve,nanosleep \
sh -c 'sleep 0.1 & sleep 0.2 & sleep 0.3' && cat log

```
13475 execve("/bin/sh", ["sh", "-c", "sleep 0.1 & sleep 0.2 & sleep 0."...],
      0x5631be4f87a8 /* 42 vars */) = 0
13476 execve("/bin/sleep", ["sleep", "0.1"], 0xe4c4f0 /* 33 vars */ <unfinished ...>
13477 execve("/bin/sleep", ["sleep", "0.2"], 0xe4c4f0 /* 33 vars */ <unfinished ...>
13478 execve("/bin/sleep", ["sleep", "0.3"], 0xe4c4f0 /* 33 vars */ <unfinished ...>
13476 <... execve resumed>)              = 0
13477 <... execve resumed>)              = 0
13478 <... execve resumed>)              = 0
13476 nanosleep(tv_sec=0, tv_nsec=100000000,  <unfinished ...>
13477 nanosleep(tv_sec=0, tv_nsec=200000000,  <unfinished ...>
13478 nanosleep(tv_sec=0, tv_nsec=300000000,  <unfinished ...>
13476 <... nanosleep resumed>NULL)       = 0
13476 +++ exited with 0 +++
13477 <... nanosleep resumed>NULL)       = 0
13477 +++ exited with 0 +++
13478 <... nanosleep resumed>NULL)       = 0
13478 +++ exited with 0 +++
13475 +++ exited with 0 +++
```

Filtrovaní podle návratného stavu systémového volání: agregace u -z [2/2]

strace -o log -z -f -e signal=none -e trace=execve,nanosleep \
sh -c 'sleep 0.1 & sleep 0.2 & sleep 0.3' && cat log

```
13475 execve("/bin/sh", ["sh", "-c", "sleep 0.1 & sleep 0.2 & sleep 0."...],
      0x5631be4f87a8 /* 42 vars */) = 0
13476 execve("/bin/sleep", ["sleep", "0.1"], 0xe4c4f0 /* 33 vars */) = 0
13477 execve("/bin/sleep", ["sleep", "0.2"], 0xe4c4f0 /* 33 vars */) = 0
13478 execve("/bin/sleep", ["sleep", "0.3"], 0xe4c4f0 /* 33 vars */) = 0
13476 nanosleep(tv_sec=0, tv_nsec=100000000, NULL) = 0
13476 +++ exited with 0 +++
13477 nanosleep(tv_sec=0, tv_nsec=200000000, NULL) = 0
13477 +++ exited with 0 +++
13478 nanosleep(tv_sec=0, tv_nsec=300000000, NULL) = 0
13478 +++ exited with 0 +++
13475 +++ exited with 0 +++
```

Problémy s filtrací systémových volání

glibc: open nebo openat?

```
glibc-2.25$ strace -qq -e open 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)                  = 3
```

Problémy s filtrací systémových volání

glibc: open nebo openat?

```
glibc-2.25$ strace -qq -e open 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)                  = 3
glibc-2.26$ strace -qq -e open cat /dev/null
```

Problémy s filtrací systémových volání

glibc: open nebo openat?

```
glibc-2.25$ strace -qq -e open 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)                  = 3
glibc-2.26$ strace -qq -e open cat /dev/null
glibc-2.26$ strace -qq -e openat cat /dev/null
openat(AT_FDCWD, "/etc/ld.so.cache", O_RDONLY|O_CLOEXEC) = 3
openat(AT_FDCWD, "/lib64/libc.so.6", O_RDONLY|O_CLOEXEC) = 3
openat(AT_FDCWD, "/dev/null", O_RDONLY) = 3
```

Problémy s filtrací systémových volání

glibc: open nebo openat?

```
glibc-2.25$ strace -qq -e open 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)                  = 3
glibc-2.26$ strace -qq -e open cat /dev/null
glibc-2.26$ strace -qq -e openat cat /dev/null
openat(AT_FDCWD, "/etc/ld.so.cache", O_RDONLY|O_CLOEXEC) = 3
openat(AT_FDCWD, "/lib64/libc.so.6", O_RDONLY|O_CLOEXEC) = 3
openat(AT_FDCWD, "/dev/null", O_RDONLY) = 3
glibc-2.25$ strace -qq -e openat cat /dev/null
```

Problémy s filtrací systémových volání

glibc: open nebo openat?

```
glibc-2.25$ strace -qq -e open 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)                  = 3
glibc-2.26$ strace -qq -e open cat /dev/null
glibc-2.26$ strace -qq -e openat cat /dev/null
openat(AT_FDCWD, "/etc/ld.so.cache", O_RDONLY|O_CLOEXEC) = 3
openat(AT_FDCWD, "/lib64/libc.so.6", O_RDONLY|O_CLOEXEC) = 3
openat(AT_FDCWD, "/dev/null", O_RDONLY) = 3
glibc-2.25$ strace -qq -e openat cat /dev/null
```

Tradiční přístup nefunguje na všech architekturách

```
riscv$ strace -e open,openat
strace: invalid system call 'open'
```

Nové možností zadaní filtru systémových volání

Rychlé řešení pomocí regulárních výrazů

```
$ strace -e /open
```

Nové možnosti zadaní filtru systémových volání

Rychlé řešení pomocí regulárních výrazů

```
$ strace -e /open
```

Skutečný výsledek rychlého řešení

Regulárnímu výrazu `.*open.*` odpovídají taky následující systémové volání:

- `mq_open`
- `open_by_handle_at`
- `perf_event_open`

Nové možnosti zadaní filtru systémových volání

Rychlé řešení pomocí regulárních výrazů

```
$ strace -e /open
```

Skutečný výsledek rychlého řešení

Regulárnímu výrazu `.*open.*` odpovídají taky následující systémové volání:

- `mq_open`
- `open_by_handle_at`
- `perf_event_open`

Přesný regulární výraz

```
$ strace -e '/^open(at)?$'
```

Nové možnosti zadaní filtru systémových volání

Rychlé řešení pomocí regulárních výrazů

```
$ strace -e /open
```

Skutečný výsledek rychlého řešení

Regulárnímu výrazu `.*open.*` odpovídají taky následující systémové volání:

- `mq_open`
- `open_by_handle_at`
- `perf_event_open`

Přesný regulární výraz

```
$ strace -e '/^open(at)?$'
```

Řešení pomocí volitelné specifikaci systémových volání

```
$ strace -e '?open,?openat'
```

Zadání argv[0] prováděného příkazu: --**argv0**=***name***

strace --argv0=insmod --trace=execve kmod --help

```
 execve("/bin/kmod", ["insmod", "-help"], 0x7fffeb80fa58 /* 42 vars */) = 0
Usage:
 insmod [options] filename [args]
Options:
 -V, -version     show version
 -h, -help        show this help
+++ exited with 0 +++
```

Spuštění jako oddělené vnouče: -D/--daemonize

Regulární volání strace

```
$ echo $$ && strace -e none sh -c 'echo $PPID'
1234
23456
+++ exited with 0 +++
$ echo $$ && strace -e none sh -c 'echo $PPID'
1234
23459
+++ exited with 0 +++
```

Volání strace jako démona

```
$ echo $$ && strace -D -e none sh -c 'echo $PPID'
1234
1234
+++ exited with 0 +++
```

Další oddělení strace: -DD, -DDD

strace -D

```
$ timeout -s KILL 1 strace -D -e/nanosleep sleep 2
strace: Process 123457 attached
clock_nanosleep(CLOCK_REALTIME, 0, {tv_sec=2, tv_nsec=0}, Killed
```

strace -DD

```
$ timeout -s KILL 1 strace -DD -e/nanosleep sleep 2
strace: Process 123457 attached
clock_nanosleep(CLOCK_REALTIME, 0, {tv_sec=2, tv_nsec=0}, <unfinished ...>) = ?
Killed
+++ killed by SIGKILL +++
```

--seccomp-bpf

- Automaticky vytvaruje a připojuje BPF program na filtraci systémových volání
- Zrychluje zpracování vyfiltrovaných systémových volání o přibližně stokrát

Hanebný příklad s dd

```
$ dd if=/dev/zero of=/dev/null bs=1 count=1M 2>&1 | grep -v records
1048576 bytes (1.0 MB, 1.0 MiB) copied, 0.281794 s, 3.7 MB/s

$ strace -f -qq --signal=none --trace=fchdir \
  dd if=/dev/zero of=/dev/null bs=1 count=1M 2>&1 | grep -v records
1048576 bytes (1.0 MB, 1.0 MiB) copied, 13.9573 s, 75.1 kB/s

$ strace -f -qq --signal=none --trace=fchdir --seccomp-bpf \
  dd if=/dev/zero of=/dev/null bs=1 count=1M 2>&1 | grep -v records
1048576 bytes (1.0 MB, 1.0 MiB) copied, 0.307412 s, 3.4 MB/s
```

Porovnání zpomalení

$0.307412/0.281794 \approx 1.09$ $13.9573/0.307412 \approx 45.4$ $13.9573/0.281794 \approx 49.5$

Omezení

- Nemá efekt pokud není specifikováno --**follow-forks** (-**f**)
- Nekompatibilní s --**syscall-limit**=***number***
- Nekompatibilní s --**detach-on**=**execve** (-**b execve**)
- Neplatí pro přípojené pomoci --**attach**=***pid*** (-**p** *pid*) programy

Proč?

- Program seccomp BPF, jež nainstalován stracem, používá rozhraní `SECCOMP_RET_TRACE`
- Až nainstalován, seccomp BPF program nemůže být odstraněn
- Když není žádný tracer, `SECCOMP_RET_TRACE` se interpretuje jako `SECCOMP_RET_ERRNO`, takže všechny dříve sledované systémové volání selhávají s chybou `ENOSYS`

Vzhledem k tomu, že --**seccomp-bpf** je optimalizace, vypíná se v případech, když nemůže být použit.

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

inject=*set* – injekce chyb pro specifikovanou sadu systémových volání
error=*errno* – kód chyby pro navracení
when=*expr* – kdy dělat injekci, v podobě *first[..last][+[step]]*
syscall=*syscall* – injekce specifikovaného systémového volání (ze skupiny **%pure**)
namísto -1

strace -e /open -e **inject**=all:**error**=EACCES:**when**=3 \
cat /dev/full /dev/null

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

Chyba v python3.5: nesprávné zpracovaní chyb otevíraní /dev/urandom

```
$ strace -P /dev/urandom -e inject=%file:error=ENOENT python3
openat(AT_FDCWD, "/dev/urandom", O_RDONLY|O_CLOEXEC)
 = -1 ENOENT (No such file or directory) (INJECTED)
Fatal Python error: Failed to open /dev/urandom
-- SIGSEGV {si_signo=SIGSEGV, si_code=SEGV_MAPERR, si_addr=0x50} --
+++ killed by SIGSEGV +++
Segmentation fault
```

Chyba v python3.5: nesprávné zpracovaní chyb čtení z /dev/urandom

```
$ strace -a0 -e read -P /dev/urandom -e inject=all:error=EIO python3
read(3, 0x8db610, 24) = -1 EIO (Input/output error) (INJECTED)
Fatal Python error: Failed to read bytes from /dev/urandom
-- SIGSEGV {si_signo=SIGSEGV, si_code=SEGV_MAPERR, si_addr=0x50} --
+++ killed by SIGSEGV +++
Segmentation fault
```

glibc <= 2.25 chyba v dynamickém linkeru

```
$ strace -e mprotect -efault=all:error=EPERM:when=1 pwd
mprotect(0x7fabcd00f000, 2097152, PROT_NONE)
 = -1 EPERM (Operation not permitted) (INJECTED)
mprotect(0x7fabcd20f000, 16384, PROT_READ) = 0
mprotect(0x606000, 4096, PROT_READ)      = 0
mprotect(0x7fabcd441000, 4096, PROT_READ) = 0
/
+++ exited with 0 +++
```

glibc >= 2.26 za správnou kontrolou

```
$ strace -e mprotect -efault=all:error=EPERM:when=1 pwd
mprotect(0x7fabcd00f000, 2097152, PROT_NONE)
 = -1 EPERM (Operation not permitted) (INJECTED)
pwd: error while loading shared libraries: libc.so.6:
 cannot change memory protections
+++ exited with 127 +++
```

Manipulovaní systémovými voláními: injekce návratnou hodnoty

-e **inject**=*set*:**retval**=*value*[:**when**=*expr*][:**syscall**=*syscall*]

inject=*set* – injekce pro specifikovanou sadu systémových volání
retval=*value* – návratná hodnota pro navracení
when=*expr* – kdy dělat injekci, v podobě *first[..last][+[step]]*
syscall=*syscall* – injekce specifikovaného systémového volání (ze skupiny **%pure**)
namísto -1

příklad: uchovaní dočasných souborů

```
$ cat script.sh
t=‘mktemp‘; trap ’rm -f "$t"’ 0; echo secret $$ > $t
$ strace -qq -f -e signal=none -e /unlink
  -e inject=all:retval=0 sh script.sh
[pid 347] unlinkat(AT_FDCWD, "/tmp/tmp.l1AlwyCYH3", 0) = 0 (INJECTED)
$ cat /tmp/tmp.l1AlwyCYH3
secret 345
```

Manipulovaní systémovými voláními: injekce zdržení

injekce zdržení systémového volání

strace -e **inject**=*set*:**delay_enter**=*usecs*
strace -e **inject**=*set*:**delay_exit**=*usecs*

dd if=/dev/zero of=/dev/null bs=1M count=10

```
10+0 records in
10+0 records out
10485760 bytes (10 MB, 10 MiB) copied, 0.00211354 s, 5.0 GB/s
```

strace -einject=write:delay_exit=100000 -ewrite -o/dev/null \ dd if=/dev/zero of=/dev/null bs=1M count=10

```
10+0 records in
10+0 records out
10485760 bytes (10 MB, 10 MiB) copied, 1.10658 s, 9.5 MB/s
```

-e **inject**=*set*:**poke_enter**=*@argN=DATAN,@argM=DATAM...*[:**when**=*expr*]

inject=*set* – injekce pro specifikovanou sadu systémových volání
poke_enter=*@argN=DATAN,@argM=DATAM,...* – pozměnění pamětí po ukazateli `arg`*N* na `DATAN`, po ukazateli `arg`*M* na `DATAM`, a t.d.
when=*expr* – kdy dělat injekci, v podobě *first[..last][+[step]]*

strace -P /etc/shadow cat /etc/shadow

```
openat(AT_FDCWD, "/etc/shadow", O_RDONLY) = -1 EACCES (Permission denied)
cat: /etc/shadow: Permission denied
+++ exited with 1 +++
```

hex=2F6465762F6E756C6C00
strace -P /etc/shadow –inject=openat:poke_enter=@arg2=$hex cat /etc/shadow

```
openat(AT_FDCWD, "/etc/shadow", O_RDONLY) = 3 (INJECTED: args)
+++ exited with 0 +++
```

-e **inject**=*set*:**poke_exit**=*@argN=DATAN,@argM=DATAM...*[:**when**=*expr*]

inject=*set* – injekce pro specifikovanou sadu systémových volání
poke_exit=*@argN=DATAN,@argM=DATAM,...* – pozměnění pamětí po ukazateli arg*N* na DATAN, po ukazateli arg*M* na DATAM, a t.d.
when=*expr* – kdy dělat injekci, v podobě *first[..last][+[step]]*

strace -s256 -e readlink readlink /etc/localtime

```
readlink("/etc/localtime", "../usr/share/zoneinfo/Asia/Jerusalem", 64) = 36
../usr/share/zoneinfo/Asia/Jerusalem
+++ exited with 0 +++
```

hex=2E2E2F7573722F73686172652F7A6F6E65696E666F2F4574632F555443
strace -e readlink –inject=readlink:retval=29:poke_exit=@arg2=$hex \
readlink /etc/localtime

```
readlink("/etc/localtime", "../usr/share/zoneinfo/Etc/UTC", 64) = 29 (INJECTED: args, retval)
../usr/share/zoneinfo/Etc/UTC
+++ exited with 0 +++
```

Systémový čas, strávený v systémových voláních: -c

strace -c sleep 1

```
% time     seconds  usecs/call     calls    errors syscall
------ ----------- ----------- --------- --------- ----------------
 31.45    0.000078          19         4           close
 25.40    0.000063          15         4           mprotect
 12.90    0.000032          32         1           nanosleep
 11.29    0.000028           5         5           mmap
  8.47    0.000021           5         4           brk
  8.06    0.000020          20         1           munmap
  2.42    0.000006           6         1           arch_prctl
  0.00    0.000000           0         1           read
  0.00    0.000000           0         2           fstat
  0.00    0.000000           0         1         1 access
  0.00    0.000000           0         1           execve
  0.00    0.000000           0         2           openat
------ ----------- ----------- --------- --------- ----------------
100.00    0.000248                    27         1 total
```

Reální čas, strávený v systémových voláních: -c -w

strace -c -w sleep 1

```
% time     seconds  usecs/call     calls    errors syscall
------ ----------- ----------- --------- --------- ----------------
 99.91    1.000184     1000183         1           nanosleep
  0.04    0.000367         366         1           execve
  0.02    0.000216          53         4           close
  0.01    0.000087          17         5           mmap
  0.01    0.000075          18         4           mprotect
  0.01    0.000052          13         4           brk
  0.00    0.000037          18         2           openat
  0.00    0.000027          26         1           munmap
  0.00    0.000024          12         2           fstat
  0.00    0.000019          19         1         1 access
  0.00    0.000015          14         1           read
  0.00    0.000013          13         1           arch_prctl
------ ----------- ----------- --------- --------- ----------------
100.00    1.001116                    27         1 total
```

strace -c cat < /dev/null

```
% time     seconds  usecs/call     calls    errors syscall
------ ----------- ----------- --------- --------- ----------------
 31.54    0.000205           6        30        13 openat
 23.23    0.000151           6        22           mmap
 14.00    0.000091           4        19           newfstatat
 12.92    0.000084           4        20           close
  4.46    0.000029           9         3           mprotect
  4.15    0.000027          13         2           munmap
  2.46    0.000016           4         4           read
  1.54    0.000010           3         3           brk
  0.92    0.000006           6         1           getrandom
...
------ ----------- ----------- --------- --------- ----------------
100.00    0.000650           5       116        15 total
```

Nastavení souhrnu volání: --**summary-columns**=***set***, -**U** ***set*** [2/2]

strace -c -U min-time,max-time,avg-time,calls cat < /dev/null

```
shortest  longest  usecs/call     calls syscall
---- ---- ------ ----- --------
0.000000 0.000016           5        30 openat
0.000000 0.000010           3        22 mmap
0.000000 0.000019           4        19 newfstatat
0.000000 0.000005           2        20 close
0.000009 0.000018          12         3 mprotect
0.000005 0.000015          10         2 munmap
0.000000 0.000006           3         4 read
0.000000 0.000006           3         3 brk
0.000005 0.000005           5         1 getrandom
...
---- ---- ------ ----- --------
0.000000 0.000019           4       116 total
```

strace --**tips**[=[[**id**:]*id*],[[**format**:]*format*]]

- *id* může být jedním s následujícího:
  - **random** vypisuje náhodnou radu (implicitní hodnota)
  - *číslo* vypisuje určenou radu podle specifikovaného čísla
- *format* může být jedním s následujícího:
  - **compact** vypisuje radu jen dost velkou pro obdržení všeho textu
  - **full** vypisuje radu ve své plné kráse
  - **none** nevypisuje žádnou radu

strace --tips=id:31

```
 ________________________________________________        ____
/                                                \      /    \
| Medicinal effects of strace can be achieved    |     |-. .-.|
| by invoking it with the following options:     \     (_@)(_@)
|                                                 \    .---_  \
|     strace -DDDqqq -enone --signal=none          _\  /..   \_/
|                                                /    |__.-^ /
|                                                |       } |
\________________________________________________/       |  [
                                                         [ ]
```

strace --tips=id:31

```
 ________________________________________________        ____
/                                                \      /    \
| Medicinal effects of strace can be achieved    |     |-. .-.|
| by invoking it with the following options:     \     (_@)(_@)
|                                                 \    .---_  \
|     strace -DDDqqq -enone --signal=none          _\  /..    \_/
|                                                /    |__.-^ /
|                                                |       }  |
\________________________________________________/       |   [
                                                         [  ]
```

https://github.com/strace/strace/issues/14

Document how to just get the 'medicinal' effects of strace with no overhead.

Otázky?

webové stránky

https://strace.io/

strace.git

https://github.com/strace/strace.git
https://gitlab.com/strace/strace.git

mailing list

strace-devel@lists.strace.io

IRC kanál

#strace@OFTC

ptrace(2)

- strace utilizes ptrace infrastructure for tracing
- ptrace(2) is a generic debugging interface that provides a set of commands (*requests*) that enable various operations: reading and writing tracee's memory, obtaining tracee's registers, and so on
- Almost all ptrace operations are performed on a stopped process
- The ptrace API (ab)uses the standard UNIX parent/child signaling over waitpid(2) in order to deliver notifications about changes in tracees' state (including ptrace-induced stops)
- This mechanism is used to notify tracer about all kinds of events: syscall stops (after resume with PTRACE_SYSCALL request), signal deliveries, group stops, forks, execs, etc.

How strace traces processes

- strace is waiting for events in wait4(2)
- Upon receiving a ptrace event, strace tries to figure out what happened (syscall stop, signal received by tracee, etc.)
  - The only information it has at this point is the status returned by the wait4(2) syscall; as a result, additional PTRACE_GETEVENTMSG request is required to distinguish some of the events
- For syscall stops, the additional information (syscall number and arguments on entering, return code on exiting) is retrieved
- Syscall-specific decoder function is called, which, in turn, may perform additional reads from tracee's memory for elaborate argument printing (structures, arrays, linked lists...)
- When the decoding is finished, tracee is resumed
- For each syscall, syscall stop is happened twice: on syscall entering and exiting

seccomp-bpf usage

- `seccomp` (for Secure Computing) is a Linux mechanism that provides an ability (in its `SECCOMP_SET_MODE_FILTER` mode) to attach a BPF program to a process
- Since Linux 3.5, a `seccomp` program has an ability to return `SECCOMP_RET_TRACE` as a result of its execution, which, in turn, notifies `ptrace`-based tracer with `PTRACE_EVENT_SECCOMP`
- When `--seccomp-bpf` command-line option is passed to `strace`, BPF bytecode is generated and attached to tracees, if possible
- The feature has been implemented as part of GSoC 2018 and 2019 projects by Chen Jingpiao and Paul Chaignon, and included in the 5.3 release of `strace`