Name

bpftool-cgroup − tool for inspection and simple manipulation of eBPF progs

SYNOPSIS

bpftool [*OPTIONS*] **cgroup** *COMMAND*

OPTIONS := { { **−j** | **−−json** } [{ **−p** | **−−pretty** }] | { **−d** | **−−debug** } | { **−f** | **−−bpffs** } }

COMMANDS := { **show** | **list** | **tree** | **attach** | **detach** | **help** }

CGROUP COMMANDS

bpftool cgroup { **show** | **list** } *CGROUP* [**effective**]
bpftool cgroup tree [*CGROUP_ROOT*] [**effective**]
bpftool cgroup attach *CGROUP ATTACH_TYPE PROG* [*ATTACH_FLAGS*]
bpftool cgroup detach *CGROUP ATTACH_TYPE PROG*
bpftool cgroup help

PROG := { **id** *PROG_ID* | **pinned** *FILE* | **tag** *PROG_TAG* | **name** *PROG_NAME* }
ATTACH_TYPE := { **cgroup_inet_ingress** | **cgroup_inet_egress** |
cgroup_inet_sock_create | **cgroup_sock_ops** |
cgroup_device | **cgroup_inet4_bind** | **cgroup_inet6_bind** |
cgroup_inet4_post_bind | **cgroup_inet6_post_bind** |
cgroup_inet4_connect | **cgroup_inet6_connect** |
cgroup_unix_connect | **cgroup_inet4_getpeername** |
cgroup_inet6_getpeername | **cgroup_unix_getpeername** |
cgroup_inet4_getsockname | **cgroup_inet6_getsockname** |
cgroup_unix_getsockname | **cgroup_udp4_sendmsg** |
cgroup_udp6_sendmsg | **cgroup_unix_sendmsg** |
cgroup_udp4_recvmsg | **cgroup_udp6_recvmsg** |
cgroup_unix_recvmsg | **cgroup_sysctl** |
cgroup_getsockopt | **cgroup_setsockopt** |
cgroup_inet_sock_release }
ATTACH_FLAGS := { **multi** | **override** }

DESCRIPTION

bpftool cgroup { show | list } *CGROUP* **[effective]**

List all programs attached to the cgroup *CGROUP*.

Output will start with program ID followed by attach type, attach flags and program name.

If **effective** is specified retrieve effective programs that will execute for events within a cgroup. This includes inherited along with attached ones.

bpftool cgroup tree [*CGROUP_ROOT***] [effective]**

Iterate over all cgroups in *CGROUP_ROOT* and list all attached programs. If *CGROUP_ROOT* is not specified, bpftool uses cgroup v2 mountpoint.

The output is similar to the output of cgroup show/list commands: it starts with absolute cgroup path, followed by program ID, attach type, attach flags and program name.

If **effective** is specified retrieve effective programs that will execute for events within a cgroup. This includes inherited along with attached ones.

bpftool cgroup attach *CGROUP ATTACH_TYPE PROG* **[***ATTACH_FLAGS***]**

Attach program *PROG* to the cgroup *CGROUP* with attach type *ATTACH_TYPE* and optional *ATTACH_FLAGS*.

ATTACH_FLAGS can be one of: **override** if a sub−cgroup installs some bpf program, the program in this cgroup yields to sub−cgroup program; **multi** if a sub−cgroup installs some bpf program, that cgroup program gets run in addition to the program in this cgroup.

Only one program is allowed to be attached to a cgroup with no attach flags or the **override** flag. Attaching another program will release old program and attach the new one.

Multiple programs are allowed to be attached to a cgroup with **multi**. They are executed in FIFO order (those that were attached first, run first).

Non−default *ATTACH_FLAGS* are supported by kernel version 4.14 and later.

ATTACH_TYPE can be one of:

- **ingress** ingress path of the inet socket (since 4.10)
- **egress** egress path of the inet socket (since 4.10)
- **sock_create** opening of an inet socket (since 4.10)
- **sock_ops** various socket operations (since 4.12)
- **device** device access (since 4.15)
- **bind4** call to bind(2) for an inet4 socket (since 4.17)
- **bind6** call to bind(2) for an inet6 socket (since 4.17)
- **post_bind4** return from bind(2) for an inet4 socket (since 4.17)
- **post_bind6** return from bind(2) for an inet6 socket (since 4.17)
- **connect4** call to connect(2) for an inet4 socket (since 4.17)
- **connect6** call to connect(2) for an inet6 socket (since 4.17)
- **connect_unix** call to connect(2) for a unix socket (since 6.7)
- **sendmsg4** call to sendto(2), sendmsg(2), sendmmsg(2) for an unconnected udp4 socket (since 4.18)
- **sendmsg6** call to sendto(2), sendmsg(2), sendmmsg(2) for an unconnected udp6 socket (since 4.18)
- **sendmsg_unix** call to sendto(2), sendmsg(2), sendmmsg(2) for an unconnected unix socket (since 6.7)
- **recvmsg4** call to recvfrom(2), recvmsg(2), recvmmsg(2) for an unconnected udp4 socket (since 5.2)
- **recvmsg6** call to recvfrom(2), recvmsg(2), recvmmsg(2) for an unconnected udp6 socket (since 5.2)
- **recvmsg_unix** call to recvfrom(2), recvmsg(2), recvmmsg(2) for an unconnected unix socket (since 6.7)
- **sysctl** sysctl access (since 5.2)
- **getsockopt** call to getsockopt (since 5.3)
- **setsockopt** call to setsockopt (since 5.3)
- **getpeername4** call to getpeername(2) for an inet4 socket (since 5.8)
- **getpeername6** call to getpeername(2) for an inet6 socket (since 5.8)
- **getpeername_unix** call to getpeername(2) for a unix socket (since 6.7)

- **getsockname4** call to getsockname(2) for an inet4 socket (since 5.8)
- **getsockname6** call to getsockname(2) for an inet6 socket (since 5.8)
- **getsockname_unix** call to getsockname(2) for a unix socket (since 6.7)
- **sock_release** closing a userspace inet socket (since 5.9)

bpftool cgroup detach *CGROUP ATTACH_TYPE PROG*
Detach *PROG* from the cgroup *CGROUP* and attach type *ATTACH_TYPE*.

bpftool prog help
Print short help message.

OPTIONS

−h, **−−help**
Print short help message (similar to **bpftool help**).

−V, **−−version**
Print bpftool's version number (similar to **bpftool version**), the number of the libbpf version in use, and optional features that were included when bpftool was compiled. Optional features include linking against LLVM or libbfd to provide the disassembler for JIT−ted programs (**bpftool prog dump jited**) and usage of BPF skeletons (some features like **bpftool prog profile** or showing pids associated to BPF objects may rely on it).

−j, **−−json**
Generate JSON output. For commands that cannot produce JSON, this option has no effect.

−p, **−−pretty**
Generate human−readable JSON output. Implies **−j**.

−d, **−−debug**
Print all logs available, even debug−level information. This includes logs from libbpf as well as from the verifier, when attempting to load programs.

−f, **−−bpffs**
Show file names of pinned programs.

EXAMPLES

 # mount −t bpf none /sys/fs/bpf/
 # mkdir /sys/fs/cgroup/test.slice
 # bpftool prog load ./device_cgroup.o /sys/fs/bpf/prog
 # bpftool cgroup attach /sys/fs/cgroup/test.slice/ device id 1 allow_multi

 # bpftool cgroup list /sys/fs/cgroup/test.slice/

```
ID       AttachType      AttachFlags     Name
1        device          allow_multi     bpf_prog1
```

 # bpftool cgroup detach /sys/fs/cgroup/test.slice/ device id 1
 # bpftool cgroup list /sys/fs/cgroup/test.slice/

```
ID       AttachType      AttachFlags     Name
```

SEE ALSO

bpf(2), **bpf−helpers**(7), **bpftool**(8), **bpftool−btf**(8), **bpftool−feature**(8), **bpftool−gen**(8), **bpftool−iter**(8), **bpftool−link**(8), **bpftool−map**(8), **bpftool−net**(8), **bpftool−perf**(8), **bpftool−prog**(8), **bpftool−struct_ops**(8), **bpftool−token**(8)