

**Name**

bpftool-feature – tool for inspection of eBPF-related parameters for Linux kernel or net device

**SYNOPSIS**

**bpftool** [*OPTIONS*] **feature** *COMMAND*

*OPTIONS* := { { **-j** | **--json** } [{ **-p** | **--pretty** }] | { **-d** | **--debug** } }

*COMMANDS* := { **probe** | **help** }

**FEATURE COMMANDS**

**bpftool feature probe** [*COMPONENT*] [**full**] [**unprivileged**] [**macros** [**prefix** *PREFIX*]]

**bpftool feature list\_builtins** *GROUP*

**bpftool feature help**

*COMPONENT* := { **kernel** | **dev** *NAME* }

*GROUP* := { **prog\_types** | **map\_types** | **attach\_types** | **link\_types** | **helpers** }

**DESCRIPTION**

**bpftool feature probe** [**kernel**] [**full**] [**macros** [**prefix** *PREFIX*]]

Probe the running kernel and dump a number of eBPF-related parameters, such as availability of the **bpf()** system call, JIT status, eBPF program types availability, eBPF helper functions availability, and more.

By default, bpftool **does not run probes** for **bpf\_probe\_write\_user()** and **bpf\_trace\_printk()** helpers which print warnings to kernel logs. To enable them and run all probes, the **full** keyword should be used.

If the **macros** keyword (but not the **-j** option) is passed, a subset of the output is dumped as a list of **#define** macros that are ready to be included in a C header file, for example. If, additionally, **prefix** is used to define a *PREFIX*, the provided string will be used as a prefix to the names of the macros: this can be used to avoid conflicts on macro names when including the output of this command as a header file.

Keyword **kernel** can be omitted. If no probe target is specified, probing the kernel is the default behaviour.

When the **unprivileged** keyword is used, bpftool will dump only the features available to a user who does not have the **CAP\_SYS\_ADMIN** capability set. The features available in that case usually represent a small subset of the parameters supported by the system. Unprivileged users **MUST** use the **unprivileged** keyword: This is to avoid misdetection if bpftool is inadvertently run as non-root, for example. This keyword is unavailable if bpftool was compiled without libcap.

**bpftool feature probe dev** *NAME* [**full**] [**macros** [**prefix** *PREFIX*]]

Probe network device for supported eBPF features and dump results to the console.

The keywords **full**, **macros** and **prefix** have the same role as when probing the kernel.

**bpftool feature list\_builtins** *GROUP*

List items known to bpftool. These can be BPF program types (**prog\_types**), BPF map types (**map\_types**), attach types (**attach\_types**), link types (**link\_types**), or BPF helper functions (**helpers**). The command does not probe the system, but simply lists the elements that bpftool knows from compilation time, as provided from libbpf (for all object types) or from the BPF UAPI header (list of helpers). This can be used in scripts to iterate over BPF types or helpers.

**bpftool feature 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.

**SEE ALSO**

**bpf(2)**, **bpftool(8)**, **bpftool-btf(8)**, **bpftool-cgroup(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)**