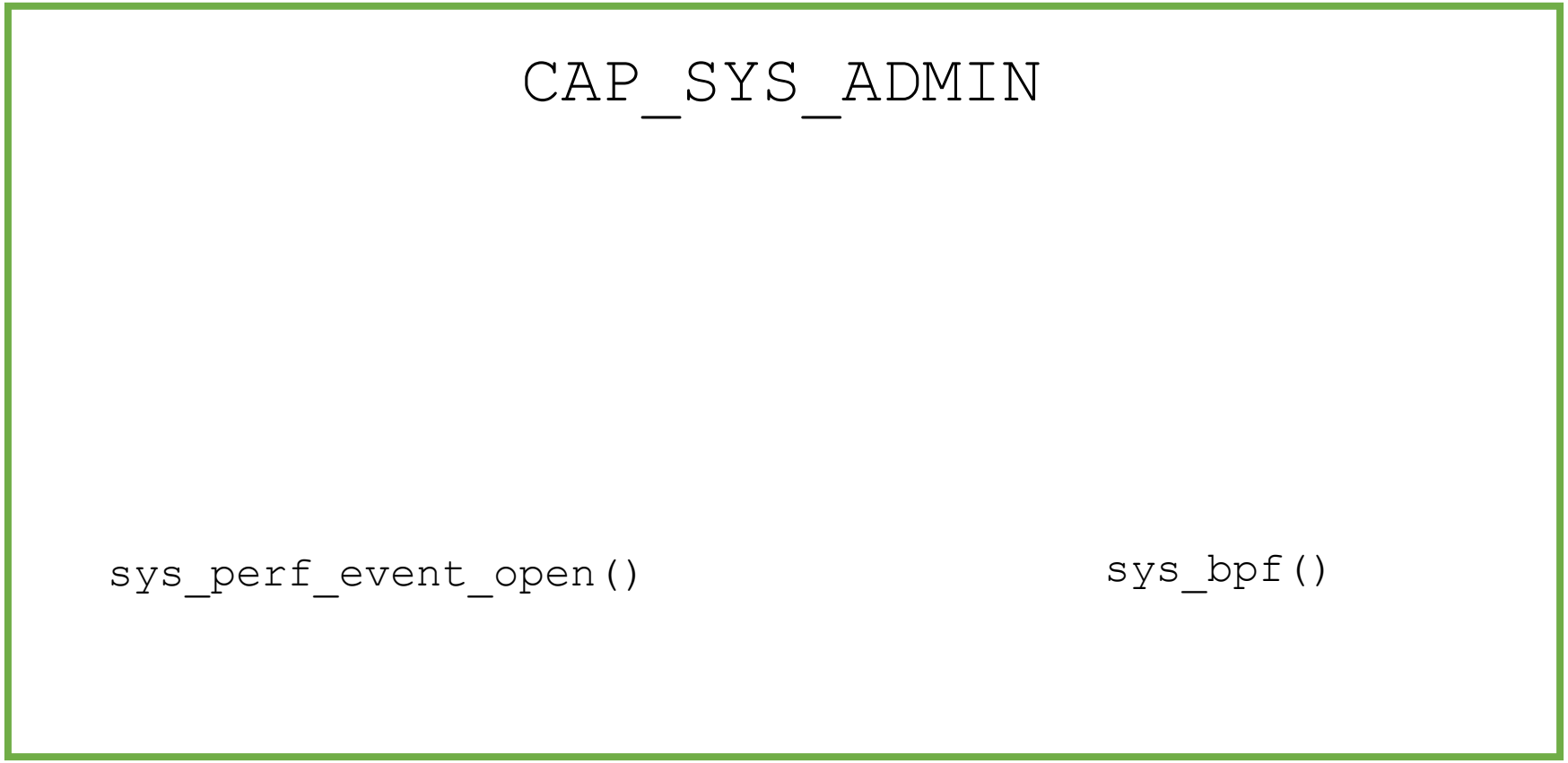
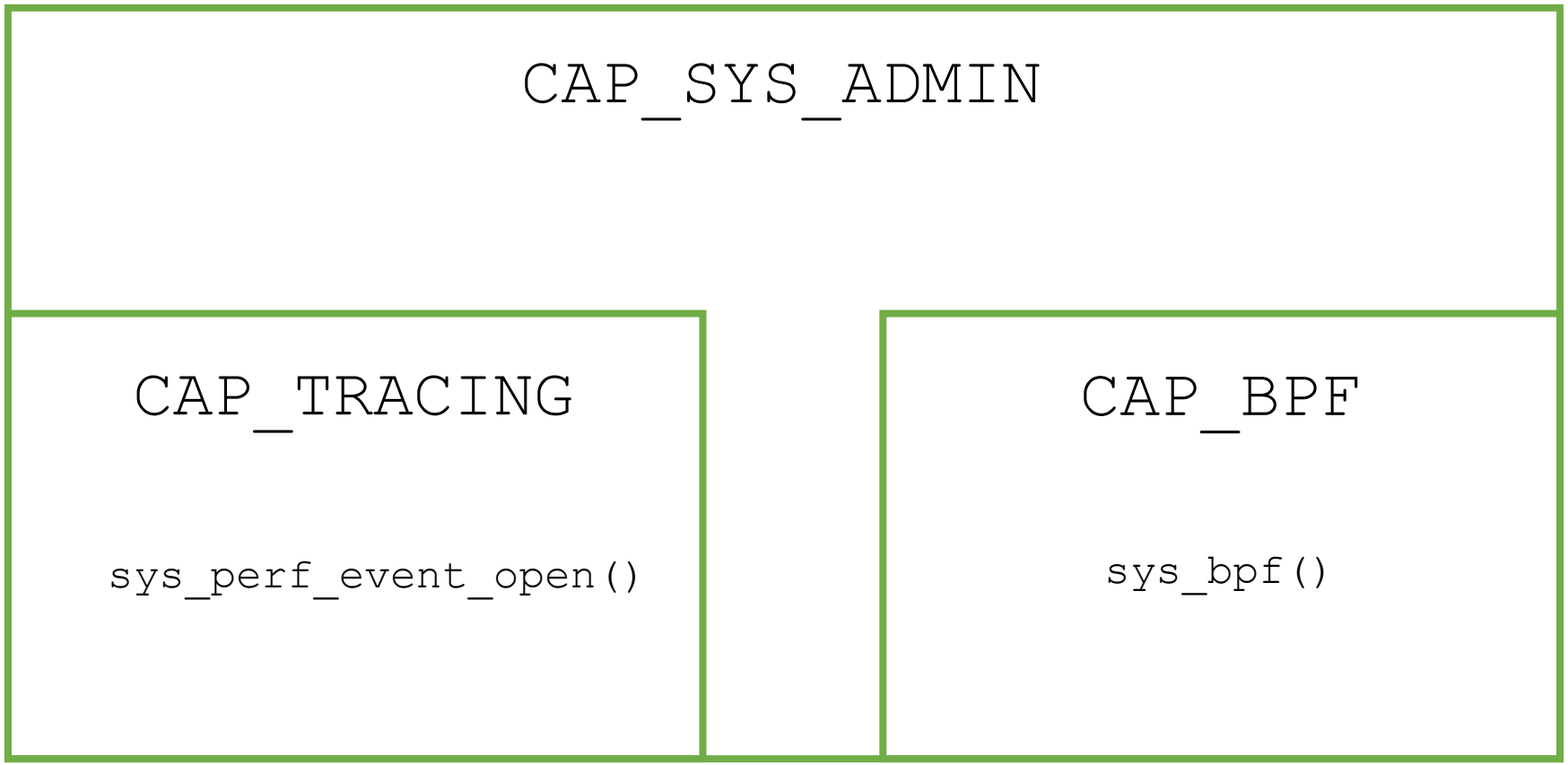
CAP_BPF and CAP_TRACING

Alexei Starovoitov, Song Liu

Overview



Overview



Use Cases

- BPF tracing users
  - `CAP_BPF` + `CAP_TRACING`
- BPF networking users
  - `CAP_BPF` + `CAP_NET_ADMIN`

sys_bpf() map

only CAP_BPF	+ CAP_TRACING	+ CAP_NET_ADMIN	+ CAP_SYS_ADMIN
BPF_MAP_CREATE except	create stackmap	create devmap	create cpumap
BPF_PROG_LOAD	BPF_RAW_TRACEPOINT_OPEN	BPF_PROG_ATTACH	hardware offload
BPF_BTF_LOAD	BPF_TASK_FD_QUERY	BPF_PROG_DETACH	
BPF_*_GET_NEXT_ID		BPF_PROG_QUERY	
BPF_*_GET_FD_BY_ID		BPF_PROG_TEST_RUN	
BPF_OBJ_GET_INFO_BY_FD			

with map FD
BPF_MAP_*

with access to bpffs
BPF_OBJ_PIN
BPF_OBJ_GET

Do **NOT** `setcap cap_bpf bpftool`

- **NOR**
  - `setcap cap_bpf bpftrace`
  - `setcap cap_tracing perf`

- Existing tools are too powerful for untrusted users

- `CAP_BPF` and `CAP_TRACING` make it possible to build bpf/perf/tracing tools for untrusted users
  - Modified `funccount.py`