Landlock LSM

Towards unprivileged sandboxing

michael@kinvolk.io

Proposed new LSM by Mickaël Salaün

First RFC March 2016, "seccomp-object: From attack surface reduction to sandboxing"

Today in iteration v7

Goal

"empower any process, including unprivileged ones, to securely restrict themselves"

Note: current version (Landlock patch v7) requires `CAP_SYS_ADMIN`

Patchset v7

- a minimum viable product
- a stackable LSM
- using eBPF

  (new pogram type `BPF_PROG_TYPE_LANDLOCK_RULE`)
- focused on filesystem access control

source: https://landlock.io/talks/2017-09-14_landlock-lss.pdf

Why eBPF

- very limited kernel attack surface
- strict rules for policies (enforced through eBPF verifier)

Demo

```
./landlock landlock1_kern.o /usr/bin/bash
```

Events

- Landlock groups 33 filesystem-related LSM hooks into `LANDLOCK_SUBTYPE_EVENT_FS`
- an event "describes the kind of kernel object for which a rule will be triggered to allow or deny an action"

Actions

- events further distinguished by action type, e.g. `LANDLOCK_ACTION_FS_WRITE`
- or subevent specific arg, e.g. ioctl request

How it works

- `linux:security_init`: Landlock LSM hooks are set up
- user application loads Landlock program(s) with `bpf(2)` and applies with `seccomp(2)`
- prog is triggered for events matching the program subtype
- prog allows (`ret == 0`) or denies access (`ret != 0`)

Applying a rule

```
prctl(PR_SET_NO_NEW_PRIVS, 1, 0, 0, 0);
seccomp(SECCOMP_PREPEND_LANDLOCK_RULE, 0, &prog_fd);
```

where `prog_fd` is the fd of the eBPF Landlock program

Writing a rule requires ...

- a subtype
- a handler program

The subtype

```
SEC("subtype")
static const union bpf_prog_subtype _subtype = {
        .landlock_rule = {
                .abi = 1,
                .event = LANDLOCK_SUBTYPE_EVENT_FS,
                .ability = LANDLOCK_SUBTYPE_ABILITY_DEBUG,
        }
};
```

The handler program

```
SEC("landlock1")
static int landlock_fs_prog1(struct landlock_context *ctx)
{
        char fmt_event_fs[] = "received event LANDLOCK_SUBTYPE_EVENT_FS\n";
        char fmt_event_unknown[] = "received unknown event type\n";

        if (ctx->event & LANDLOCK_SUBTYPE_EVENT_FS) {
                bpf_trace_printk(fmt_event_fs, sizeof(fmt_event_fs));
        } else {
                // should not happen
                bpf_trace_printk(fmt_event_unknown, sizeof(fmt_event_unknown));
        }
        return 0; // allow all
}
```

Development

- LKML
- Patchset is based on net-next
- https://github.com/landlock-lsm/linux

Roadmap

- cgroups handling
- new eBPF map type for filesystem-related checks (map fsview)
- unprivileged mode

source: https://landlock.io/talks/2017-09-14_landlock-lss.pdf

Thank you

Questions?

Slides can be found here soon: https://speakerdeck.com/schu

michael@kinvolk.io

Resources

- https://landlock.io/
- https://landlock.io/linux-doc/landlock-v7/security/landlock/index.html
- https://landlock.io/talks/2017-09-14_landlock-lss.pdf
- https://landlock.io/talks/2017-06-21_landlock-linuxkit-sig.pdf
- https://lkml.org/lkml/2017/8/20/192
- https://man.openbsd.org/pledge.2
- https://www.kernel.org/doc/Documentation/security/LSM.txt