← ALL ADVISORIES

MEDIUM Disclosed 2026-06-12

OpenBSD - Remote Kernel MPLS Stack Disclosure

mpls_do_error copies (nstk+1) label-stack entries from a fixed 16-entry array when no BoS label is present, leaking 4 bytes of adjacent kernel stack memory in the ICMP/MPLS error response.

CVE	CVE-2026-56099
VENDOR	OpenBSD
PRODUCT	OpenBSD
AFFECTED	OpenBSD-current prior to 2026-06-18
FIXED IN	2026-06-18

Executive Summary

The `mpls_do_error` function in `sys/netmpls/mpls_input.c` parses an incoming MPLS label stack into a fixed-size local array `struct shim_hdr stack[MPLS_INKERNEL_LOOP_MAX]` (16 entries). When the parse loop completes without encountering a Bottom-of-Stack (BoS) label, `nstk` reaches `MPLS_INKERNEL_LOOP_MAX` (16). Several subsequent code paths then compute a copy length of `(nstk + 1) * sizeof(*shim)` — 17 entries — and use it with `icmp_do_exthdr`, `M_PREPEND`, and `m_copyback` against the 16-entry `stack`

object. This reads one `struct shim_hdr` (4 bytes) past the end of the array, and that data is reflected back to the sender inside the generated ICMP/MPLS error.

Details

The label-stack parse loop (line ~353) fills `stack[0..15]` and breaks only when the BoS bit is set:

```
struct shim_hdr stack[MPLS_INKERNEL_LOOP_MAX];   /* 16 entries */
...
for (nstk = 0; nstk < MPLS_INKERNEL_LOOP_MAX; nstk++) {
    ...
    stack[nstk] = *mtod(m, struct shim_hdr *);
    m_adj(m, sizeof(*shim));
    if (MPLS_BOS_ISSET(stack[nstk].shim_label))
        break;
}
```

With no BoS label anywhere in the stack, the loop runs to completion and `nstk == 16`. Later, the IP-version branch appends the stack as an ICMP extension object using a length derived from `(nstk + 1)`:

```
if (icmp_do_exthdr(m, ICMP_EXT_MPLS, 1, stack,
    (nstk + 1) * sizeof(*shim)))
    return (NULL);
```

The same `(nstk + 1)` length is used to prepend and `m_copyback` the stack onto the reflected packet:

```
M_PREPEND(m, (nstk + 1) * sizeof(*shim), M_NOWAIT);
...
m_copyback(m, 0, (nstk + 1) * sizeof(*shim), stack, M_NOWAIT);
```

With `nstk == 16`, each of these reads 17 entries from a 16-entry array, reading `stack[16]` — one `struct shim_hdr` (4 bytes) of adjacent kernel stack — and

includes it in the response sent back to the attacker.

Reachability

The path is reachable remotely via `mpls_input` (line ~90) → `mpls_do_error`, on systems that have MPLS enabled on an interface. The trigger is a crafted MPLS frame carrying 16 labels with no BoS bit set and an outermost TTL of 1, so that the error path is taken. See the proof of concept.

Impact

Each crafted packet leaks 4 bytes of kernel stack memory adjacent to the `stack` array. The leak is reflected in the ICMP/MPLS extension object of the error response, so an on-path or reachable attacker can harvest the leaked bytes. The primitive is repeatable: sending successive triggers leaks different adjacent stack bytes, enabling gradual kernel memory disclosure. No crash or panic results; the leak is silent.

Timeline

- 2026-06-12 — Reported to OpenBSD with proof of concept
- 2026-06-18 — Fix committed openbsd/src@6a23123

`#kernel` `#mpls` `#info-leak`