Kernel Module Load, Data Component DC0031

Archived: 2026-04-05 13:34:38 UTC

The process of loading a kernel module into the operating system kernel. Kernel modules are object files that extend the kernel's functionality, such as adding support for device drivers, new filesystems, or additional system calls. This action can be legitimate (e.g., loading a driver) or malicious (e.g., adding a rootkit).

Data Collection Measures:

- Linux:
  - Auditd: Enable auditing of kernel module loading. Example rule: `-a always,exit -F arch=b64 -S init_module,delete_module`.
  - Syslog: Monitor `/var/log/syslog` or `/var/log/messages` for entries related to kernel module loads.
  - Systemd Journal: Use `journalctl` to query logs for module loading events: `journalctl -k | grep "Loading kernel module"`
- macOS:
  - Unified Logs: Use the `log` command to query kernel module events: `log show --predicate 'eventMessage contains "kextload"' --info`
  - Endpoint Security Framework (ESF): Monitor for `ES_EVENT_TYPE_AUTH_KEXTLOAD` (kernel extension loading events).
- Kernel-Specific Tools:
  - Lsmod: Use `lsmod` to list loaded kernel modules in real-time.
  - Kprobe/eBPF: Use extended Berkeley Packet Filter (eBPF) or Kernel Probes (kprobes) to monitor kernel events, including module loading. Example using eBPF tools like BCC: `sudo python /path/to/bcc/tools/kprobe -v do_init_module`
- Enable EDR Monitoring:
  - Configure alerts for: Suspicious kernel module loads from non-standard paths (e.g., /tmp). Unexpected or unsigned kernel modules.
  - Review detailed telemetry data provided by the EDR for insight into who initiated the module load, the file path, and whether the module was signed.
