In user space, there's `syslogd(8)` . This is a daemon that listens on a number of UNIX domain sockets (mainly `/dev/log` , but others can be configured too), and optionally to the UDP port 514 for messages. It also receives messages from `klogd(8)` (`syslogd(8)` doesn't care about `/proc/kmsg`). It then writes these messages to some files in `/log` , or to named pipes, or sends them to some remote hosts (via the `syslog` protocol, on UDP port 514), as configured in `/etc/syslog.conf` .

User-space applications normally use the `libc` function `syslog(3)` to log messages. `libc` sends these messages to the UNIX domain socket `/dev/log` (where they are read by `syslogd(8)`), but if an application is `chroot(2)` -ed the messages might end up being written to other sockets, f.i. to `/var/named/dev/log` . It is, of course, essential for the applications sending these logs and `syslogd(8)` to agree on the location of these sockets. For these reason `syslogd(8)` can be configured to listen to additional sockets aside from the standard `/dev/log` .

Finally, the `syslog` protocol is just a datagram protocol. Nothing stops an application from sending syslog datagrams to any UNIX domain socket (provided that its credentials allows it to open the socket), bypassing the `syslog(3)` function in `libc` completely. If the datagrams are correctly formatted `syslogd(8)` can use them as if the messages were sent through `syslog(3)` .