cljNetPcap Prototype

Ruediger Gad

January 26, 2012

Contents

List of Figures

1 Description

cljNetPcap is a wrapper/adapter/facade (Whatever you wanna call it.) for using jNetPcap [3] with Clojure [1].

2 Requirements

Clojure 1.3

Java 1.6

Operating Systems: Linux and Windows (both x86 and x86_64) See also below on which configurations cljNetPcap could be successfully run and on which not.

For Linux `libpcap`[4] version 1.0.0 or later must be installed. For Windows `winpcap`[2] version 4 or later must be installed.

For building the distributable Jar file etc. and running the unit tests Leiningen [5] is recommended.

cljNetPcap has been successfully tested on: Linux (Fedora 16) x86_64 libpcap 1.1.1

Note that Windows seems not to provide an `any` device. Also note that there seems to be some issue with respect to the libpcap based filtering mechanism on Windows. Here no filter should be set (via the `-f ""` option) for first tests.

3 Running

3.1 cljNetPcap

Generally, cljNetPcap must be run as a user with the permissions required to list all network interfaces, to set a network interface into promiscuous mode, and to inject packtes into a network interface. Ususally, this means cljNetPcap needs to be run as root or Administrator respectively. For a guide on how to capture packets as non-root user in Linux see: `http://packetlife.net/blog/2010/mar/19/sniffing-wireshark-non-root-user/`.

3.2 Unit Tests

Most of the unit tests need to have full access to the network interface in order to sniff data. Hence, the tests need to be run as a user with the according permissions, e. g., as root / Administrator. For running the unit tests simply execute `lein test` from projects base directory.

4 Architecture

The main work is done in two threads (see Figure 1 on the following page). The "Sniffer Thread" is responsible for sniffing data from the network interface. This sniffed data is stored in a queue.

From this queue the "Forwarder Thread" reads the data and processes/forwards it further. At this point essentially anything could be done with the data: e. g., print to stdout, or write to a file.

The functiones that are used as packet handler and forwarder can be set via command line options. For more information see how cljNetPcap is started (see 3.1 on the previous page).

4.1 Filtering the Input Data

The input data can be filtered at two points: first, native libpcap filters can be used to filter the packets prior to sniffing (default). In this case only matching packets will be sniffed. Second, the already sniffed packets could be filtered in the Clojure packet handler code.

4.2 Forwarding the Data

References

[1] Rich Hickey. Clojure - home. `http://clojure.org`.

[2] Riverbed Technology. WinPcap. `http://www.winpcap.org`.

[3] Sly Technologies, Inc. jNetPcap OpenSource — Protocol Analysis SDK. `http://jnetpcap.com`.

[4] Tcpdump/Libpcap. TCPDUMP/LIBPCAP public repository. `http://www.tcpdump.org`.

[5] technomancy (Phil Hagelberg). technomancy/leiningen - GitHub. `https://github.com/technomancy/leiningen`.

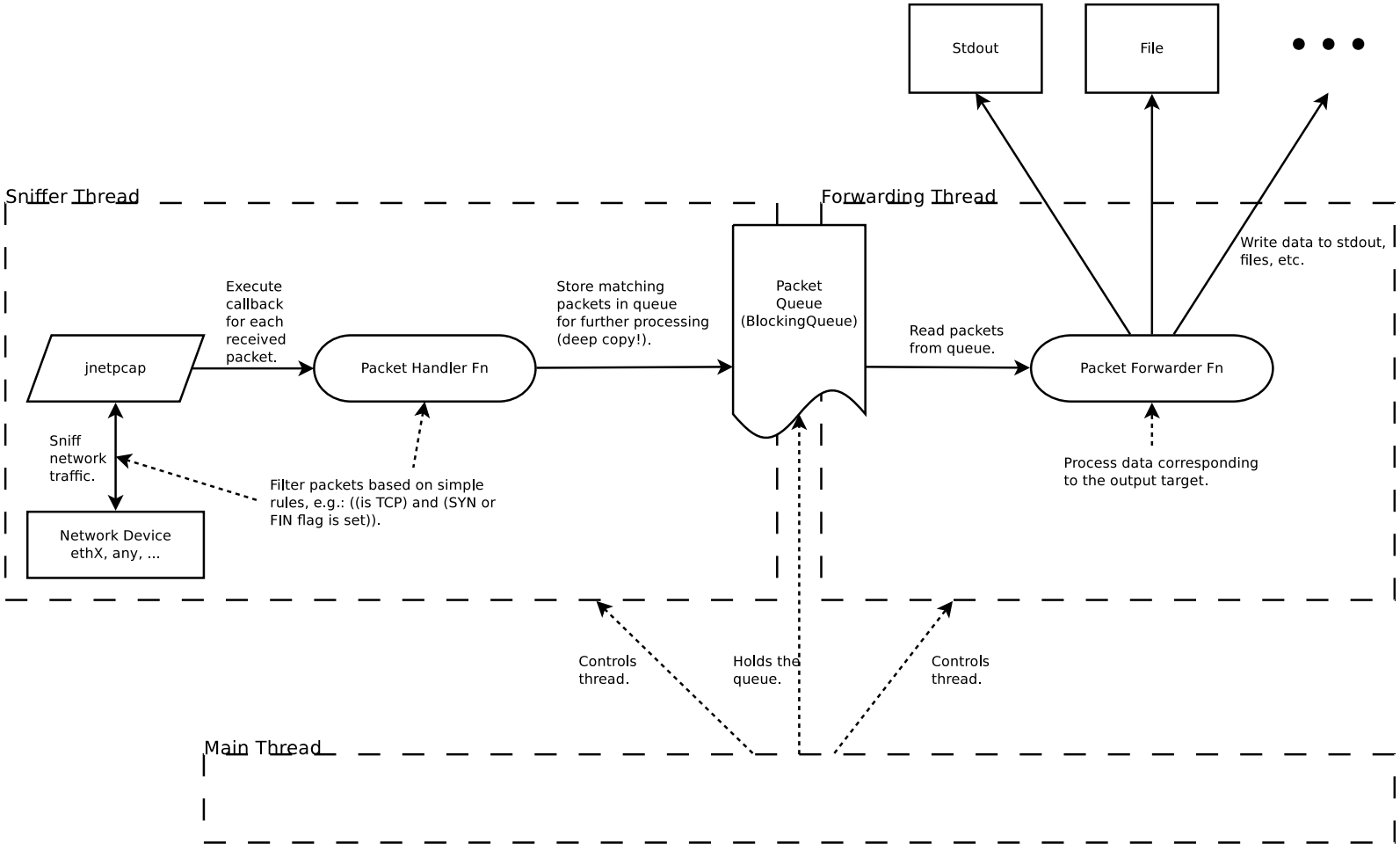

Figure 1: Simple Data Flow Diagram