USB Sniffing with tcpdump

From OMAPpedia

"usbmon (http://lxr.linux.no/#linux+v2.6.28.8/Documentation/usb/usbmon.txt) " is a facility in kernel which is used to collect traces of I/O on the USB bus. usbmon collects raw text/binary which are not easily human-readable. Here, the idea is to use Wireshark as frontend to produces a human-readable representation of these data. However Wireshark does not support usbmon raw data as is, so we have to parse these data in the pcap format. tcpdump is a good candidate to capture USB data from usbmon and generate pcap traces.

Contents

Building tcpdump

To capture USB, it is necessary to have a recent version of tcpdump/libpcap. You can use the scripted or manual method to build tcpdump or directly get a pre-compiled binary. Android NDK is required for both scripted and manual building (available from http://developer.android.com/tools/sdk/ndk).

Pre-compiled binaries

- File:Tcpdump-4.3.0-arm.tar.gz.
- File:Tcpdump-4.2.1-arm.tar.gz.

Using the build-tcpdump script

The build-tcpdump is a download & build script for Linux systems. Download File:Build-tcpdump.tar.gz

```
$ export NDK=/path/to/ndk
$ sh build-tcpdump
```

If script succeeded, tcpdump binary is built and ready to use on Android ARM platform.

Building manually from source

This method is adapted for tcpdump 4.3.0 with android-ndk-r8 and may require some adaptations in other cases.

Prepare toolchain

```
$ mkdir tcpdump
$ mkdir tcpdump/toolchain
$ android-ndk-r8/build/tools/make-standalone-toolchain.sh --platform=android-8 --install-dir=tcpdump/toolchain
$ export PATH=`pwd`/tcpdump/toolchain/bin:$PATH
$ export CC=arm-linux-androideabi-gcc
$ export RANLIB=arm-linux-androideabi-ranlib
$ export AR=arm-linux-androideabi-ar
$ export LD=arm-linux-androideabi-ld
```

Get source

Get the latest source for libpcap and tcpdump from http://www.tcpdump.org.

```
$ wget http://www.tcpdump.org/release/tcpdump-4.3.0.tar.gz
$ wget http://www.tcpdump.org/release/libpcap-1.3.0.tar.gz
$ tar -zxvf tcpdump-4.3.0.tar.gz
$ tar -zxvf libpcap-1.3.0.tar.gz
```

Build libpcap

```
$ cd libpcap-1.3.0
$ chmod +x configure runlex.sh
$ ./configure --host=arm-linux --with-pcap=linux ac_cv_linux_vers=2
$ make
$ cd ..
```

Build tcpdump

```
$ cd tcpdump-4.3.0
$ chmod +x configure
$ ./configure --host=arm-linux --with-pcap=linux --with-crypto=no ac_cv_linux_vers=2
```

Before compiling, you have to patch print-isakmp.c (setprotoent() and endprotoent() not "supported" on android).

```
$ sed -i".bak" "s/setprotoent/\/\/setprotoent/g" print-isakmp.c
$ sed -i".bak" "s/endprotoent/\/\/endprotoent/g" print-isakmp.c
$ make CFLAGS=-DNBBY=8
```

tcpdump binary is built and ready to use on Android ARM platform.

USB sniffing

Upload tcpdump on the Android target (adb push).

List Interfaces

```
$ tcpdump -D
1.eth0
2.usbmon1 (USB bus number 1)
3.any (Pseudo-device that captures on all interfaces)
4.lo
```

USB capturing

Choose usbmonX to listen USB bus X and parse its USB traffic in a pcap file.

```
$ tcpdump -i usbmon1 -w /data/usblog.pcap &
```

To stop sniffing, kill tcpdump.

```
$ killall tcpdump
```

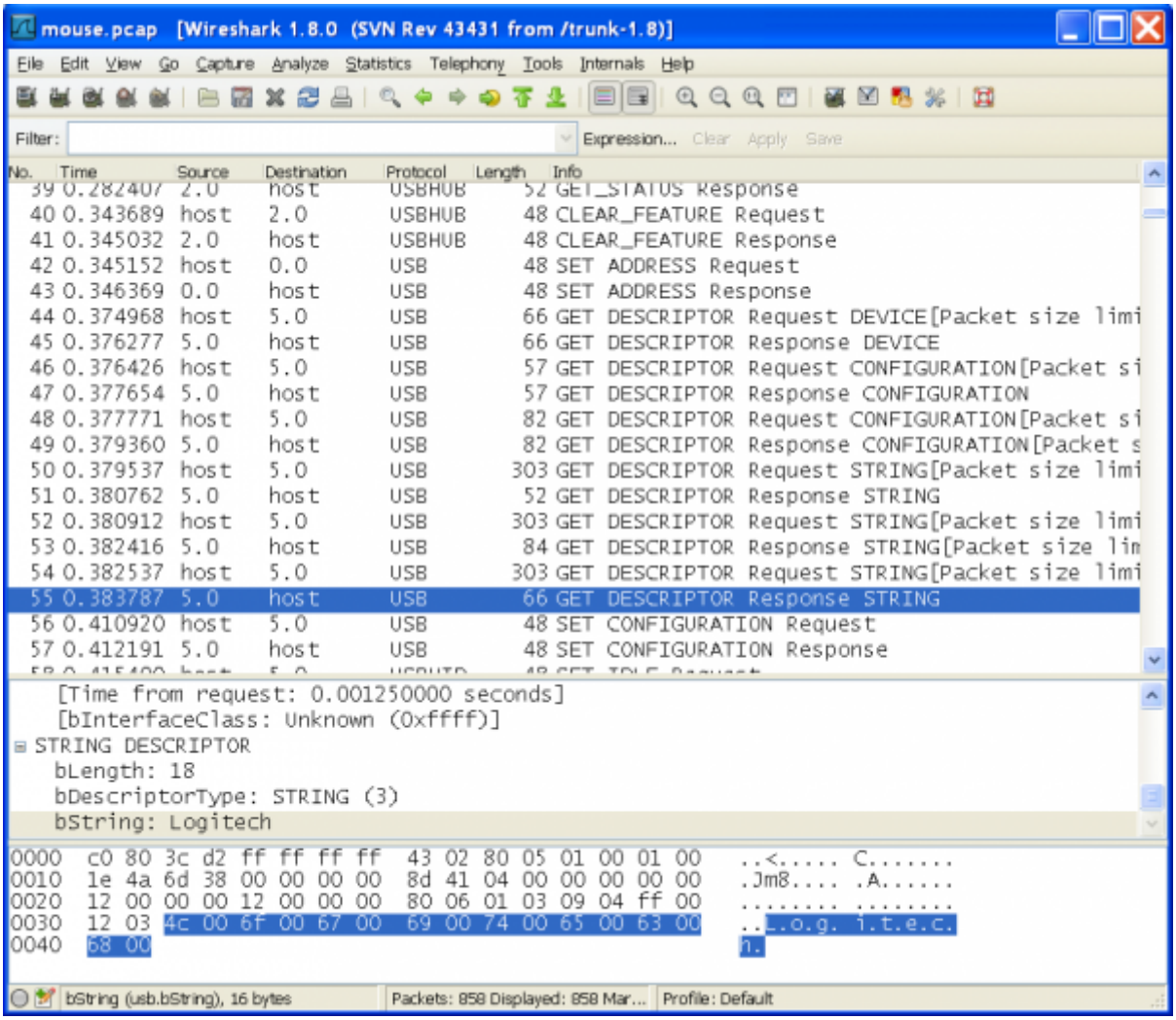
Wireshark

Retrieve pcap file from the target (adb pull) and open this file with Wireshark.

```
$ wireshark usblog.pcap
```

Example

USB traffic below was captured with tcpdump on OMAP5 platform on wich a USB mouse was plugged.



Appendices

Files

- Binary File:Tcpdump-4.3.0-arm.tar.gz
- Build script File:Build-tcpdump.tar.gz
- USB mouse pcap example File:Mouse.pcap.tar.gz

External Links

- http://wiki.wireshark.org/CaptureSetup/USB
- http://www.tcpdump.org/
- http://developer.android.com/tools/sdk/ndk/index.html
- http://www.kernel.org/doc/Documentation/usb/usbmon.txt

Loic-Poulain 14:16, 25 September 2012 (UTC) loic.poulain@gmail.com

- Disclaimers