How to run the crawler to crawl PPLive channels?

1. Clarification

This is a stable version of our crawler and it works from 2006 to early 2008. Because PPLive is a closed-source application and no technical details about this application is published, PPLive protocol may also change in the future, so our crawler may not work perfectly in the future. However, we believe the principle and approach of our crawler remain useful for research community.

Our crawler can crawl multiple channels simultaneously. The crawler has two operations: snapshot and partner discovery. Because a very large fraction of PPLive peers are behide NAT and firewall, thus they are unresponsive. The crawler therefore suffers from this unresponsive portion, especially the partner discovery operation because links between unresponsive nodes may not be obtained.

To prepare and run the crawler, we need 2 machines: one is Windows to install PPLive client, Wireshark to capture UDP packets; the other is Linux (with thread programming in C/C++) to run the crawler after you capture the UDP packets. The crawler actually uses the captured packets in text format and then translates the packets into hex format and sends to PPLive membership servers and other peers attending the same channel with our PPLive client. When receiving our crawler's packets, membership servers and peers will return UDP packets back to our crawler, which extracts and put crawled peers into a list and continues sending packets.

I recommend you to read this document entirely before you start. Please email me if you need more information about the crawler. Good luck!

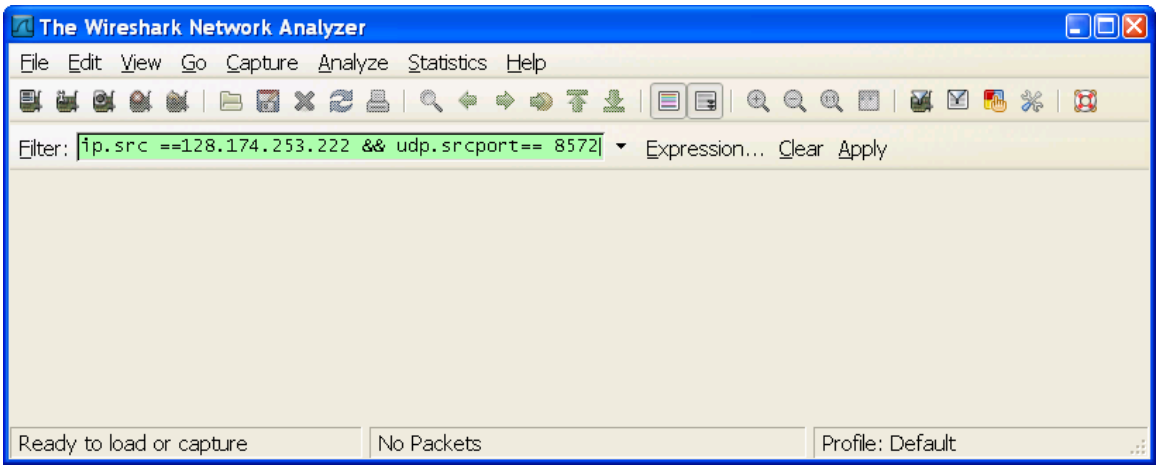
2. Software and Platform

The first step is installing Wireshark in Windows (http://www.wireshark.org/). This software is free and it is used to capture packets between our Windows client and peers attending the same channel. It also captures packets between our Windows client and PPLive membership servers and PPLive channel management servers. Simply download and run the Wireshark binary file and install.

3. Getting the channel ID

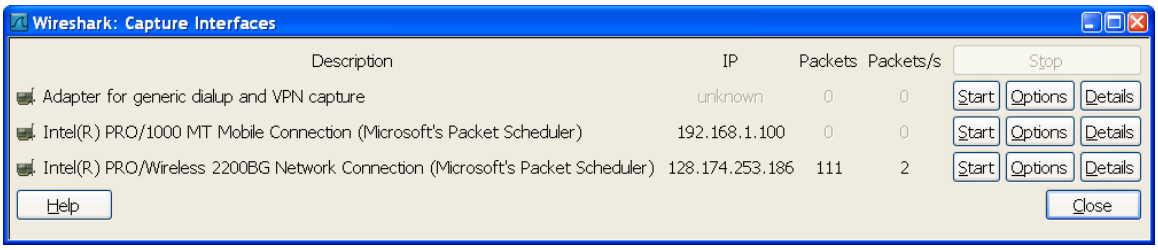
3.1. Configuring Wireshark

1. Run Wireshark
2. Go to the menu Capture/option and deselect the checkbox "Capture the packet in promiscuous mode"
3. The next step is to create a filter to filter UDP packets *sent* (thus udp.srcport) by PPLive client to PPLive servers and other peers as shown below. You can simply click on "Filter" and then edit the expression or you can just enter the expression directly to the textbox.



How to know the UDP port of PPLive? There are several ways. First, you can open PPLive client, right click on PPLive icon in the bottom right hand side of the Windows taskbar and select "Parameter Settings". Here, you see the UDP port.

Second, you can use a blank filter in Wireshark and click Apply, then you select menu Capture/Interfaces and click on Start button of the interface you know PPLive client travel through (remember to deselect the checkbox "Capture the packet in promiscuous mode" in Capture/option menu).



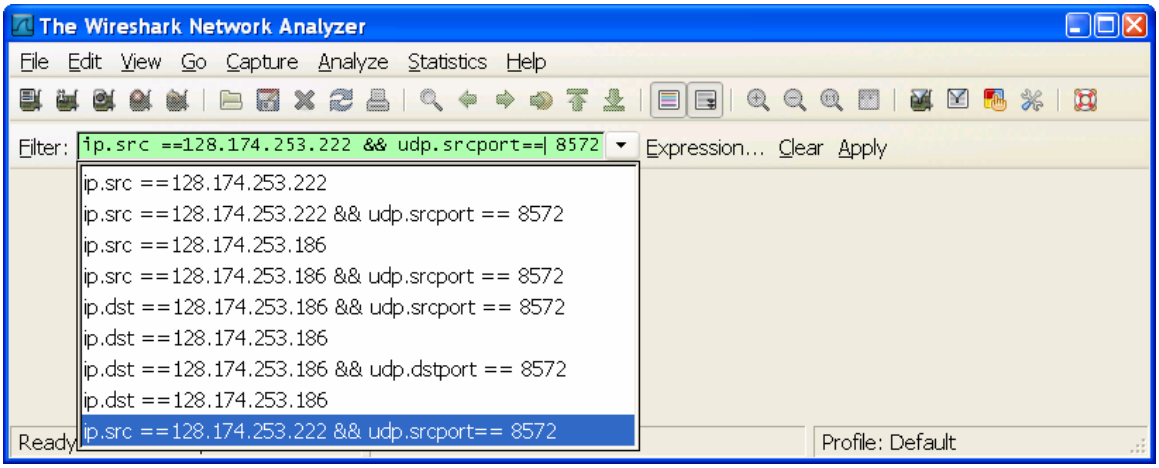
After that, open PPLive client, select a channel. You will see all packets captured by Wireshark. Click on a UDP packet sent by your PPLive client (you see the IP address of your client), you see the port of PPLive client.

So far, you have a filter to filter out all UDP packets sent by your PPLive client to its peers and PPLive servers when the PPLive client joins any channel. The next step is getting the channel ID to feed the crawler, which will crawl and give us population of the channel.

3.2. Getting the channel ID

Each PPLive channel has an ID. To get the channel ID, perform following steps:

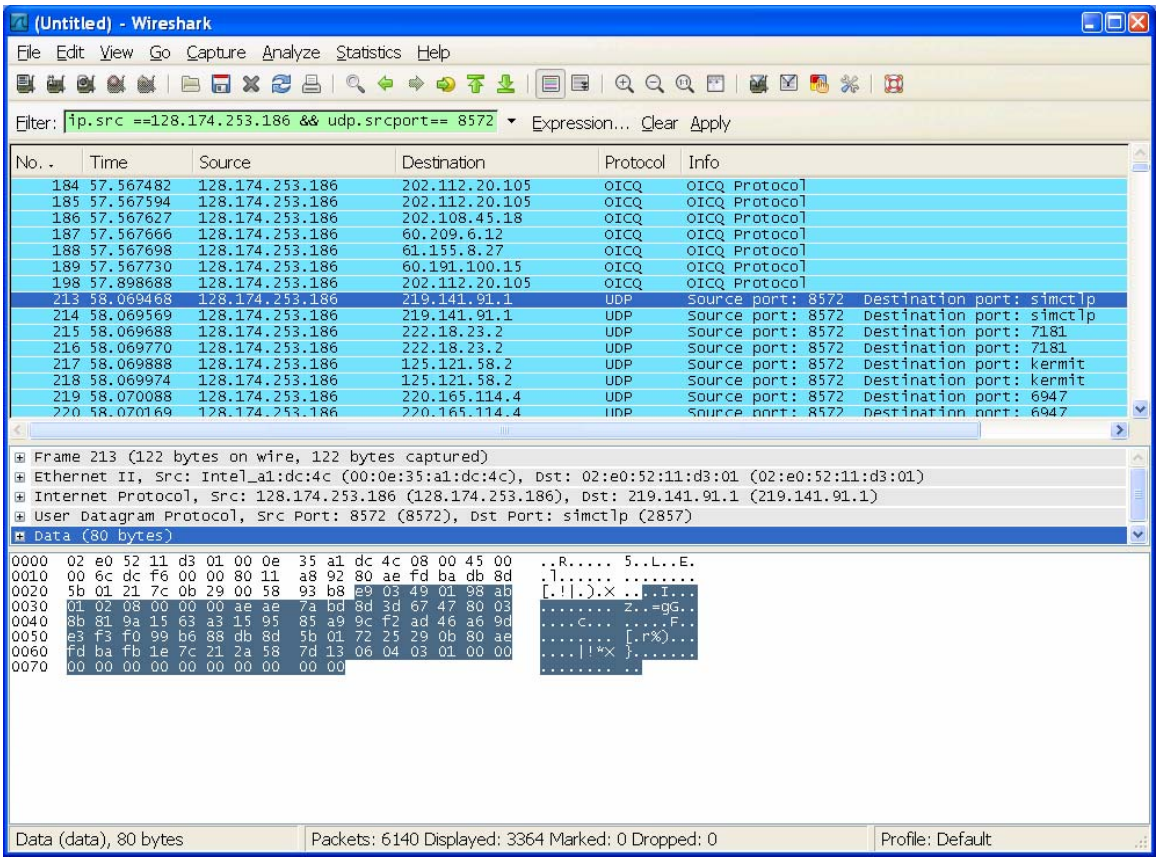
1. Run Wireshark and edit the filter as presented in previous section (if you already enter the filter, you can select it from the drop-down menu). Then, select menu Capture/Interfaces. Select Start button of the interface which PPLive client travels through.



At the moment, because you have not started PPLive client, there should be no packet captured.

2. Now, you start your PPLive client. After the PPLive client starts, you can select a channel.

3. At the moment, Wireshark starts capturing UDP packets sent by your client.



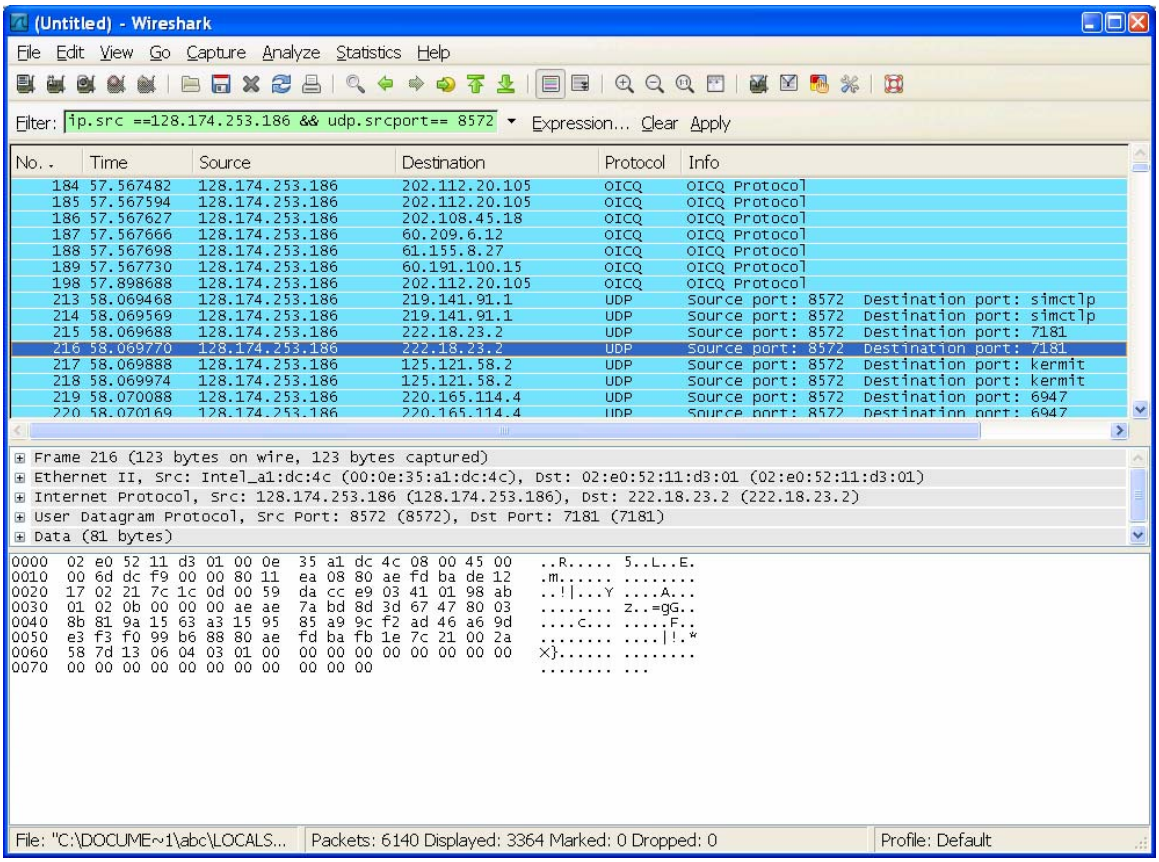
In this image, you see the upper part is OICQ Protocol (line 184-198), an application-oriented protocol of PPLive. From this image we know that the client first sends UDP packet to PPLive membership servers to get a list of peers attending the client's entering channel. After sending UDP packets to servers and gets back the peer list, client starts sending UDP packets to peers to ask for their neighbors to connect and get video content (starting from line 213). Notice that our PPLive client send different packet formats to the servers and clients. Our crawler needs to simulate this process by sending appropriate packets to servers to get peer list and then sending appropriate packets to clients to get their neighbors. *[One question may be: the crawler may just keep sending the UDP packets to servers to ask for peers. However, PPLive is closed-source, we do not know how the servers return the peer list to a client. So, we need to span the network to get peers ourselves.]*

To extract the correct packets to send to the servers and clients, we use a Perl script (below). So, we need to export the UDP packets captured by Wireshark to a text file. After that, the Perl script can extract patterns from this text file, and calls the crawler with these patterns as input parameters.

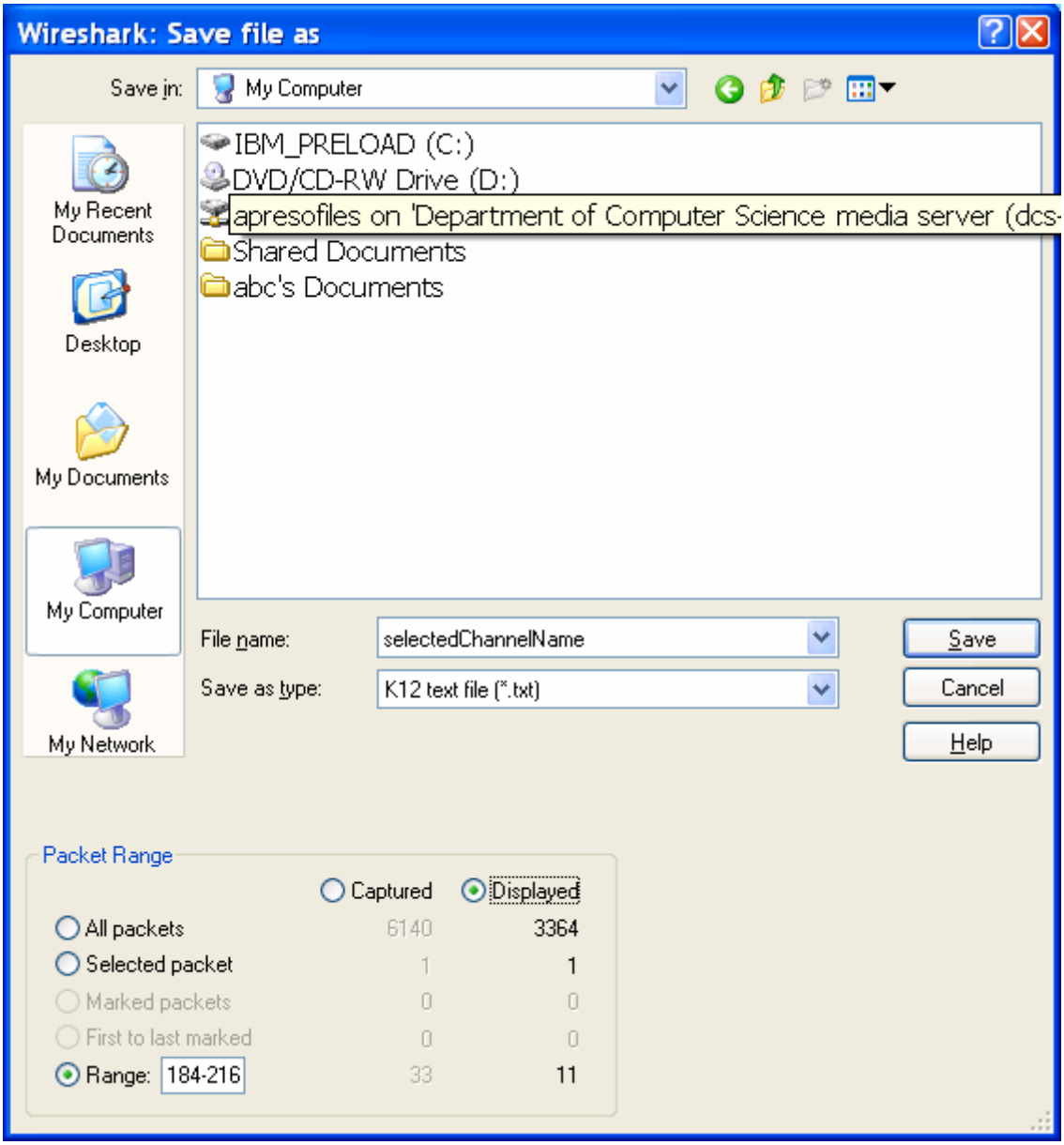
Perform following steps to export packets to a text file (you definitely have other ways to get the packet format and put into the crawler. This is just one way). This is the most tedious part ☺.

Because the PPLive client will send 2 packets to a peer A to ask for the A's neighbors, we need to select a line with the second packet. For example, here we choose 216 instead of 215 because line

216 is the second packet sent to 222.18.23.2. This selection is important for our Perl script to extract correctly later - please notice that. Remember that in this image we have lines from 184 to 216 (from the beginning to the line 216)



Then, select menu “File/Save As” and enter the corresponding information.



Notice that we only capture UDP packets in a range. The type of file is K12 text file. You can enter your channel name, for example you named your text file “channelA.txt”. Then click Save. You are done this part.

4. Compile the crawler

The crawler will run on a Linux box with thread programming and C/C++. In 2006, I use Fedora and in 2007 I use Ubuntu. I think our crawler will work on a Linux box in general.

Now, you need to change to the current directory of the crawler. Next, you copy file “channelA.txt” from Windows to the directory “channelID” inside our release directory.

Because our crawler uses thread programming, we use the following to compile:

g++ -lpthread crawler.cpp –o crawler

In our release tar.gz file, “data” is the directory to store crawled data. “channelID” is the directory to store text file we have in section 2 above. You need to copy the text file in section 2 to “channelID” directory (by SSH for example). If you need to crawl multiple channels, simply put multiple text files in this directory. The “server.lst” is a list of server IPs that I find from my experiments. This list will change over time. So, you can use Wireshark and get IPs of PPLive server (OICQ packets sent at the beginning by the PPLive Windows client) and update this list accordingly. For example, if your PPLive client connects (by OICQ protocol) to some server, which does not exist in the server.lst file, you can insert a new IP in this “server.lst” file.

Remember to set the write permission on “data” directory, read permission on “channelID” directory and “server.lst” file so the crawler can read and write.

5. Run crawler

After everything is set, at the directory of crawler, simply run:

perl runCrawler.pl crawler server.lst 50

Here, “50” means number of maximum number of responses you want the crawler to receive from remote peers. Because the crawler sends UDP requests in a round-robin fashion on the crawled list of peers, when a peer answers more than 50 times, crawler will not take the subsequent answers of this peer. Usually, the crawler runs from 1 to 4 minutes, depending on channel size, so 50 is a reasonable number. You can tune this value by go to the “peerElem.h” and find “addPartner” method.

6. Contact

If you have any questions, please feel free to email me.
Long Vu
longvu2@uiuc.edu
Homepage: http://cairo.cs.uiuc.edu/~longvu2