DS Lab Assignment:

BitTorrent

J. van der Ham

fjeroen@os3.nlg

October 31, 2014

Abstract

Distributed systems are everywhere around us, and form the basis of the

Internet. A good example of distributed applications are peer-to-peer systems,

especially BitTorrent. In this lab exercise we will look at the behavior

of a BitTorrent client, and try to understand how the distribution of files is

implemented in this distributed system.

1 BitTorrent

The BitTorrent protocol was designed in 2001 by Bram Cohen, and was made

publicly available. BitTorrent is a peer-to-peer content distribution system where

peers cooperate to make content available to all peers. The goal of this system is

to work in such a way that no central coordination or repository is necessary.

Over the years BitTorrent has been very popular for sharing of copyrighted

media, but it is used for many legitimate purposes as well. Many Linux and Unix

distributions offer BitTorrent downloads for their disc-images, and it is used in

companies such as Blizzard, Twitter and Facebook as well.

2 Network Monitoring

In the following exercises we will be looking at the network traffic from the BitTorrent

client. The easiest method to examine network traffic is to use the Wireshark

utility, as it provides a graphical overview of the network traffic, and many different

analyses options. The exercises below are meant to help you understand how

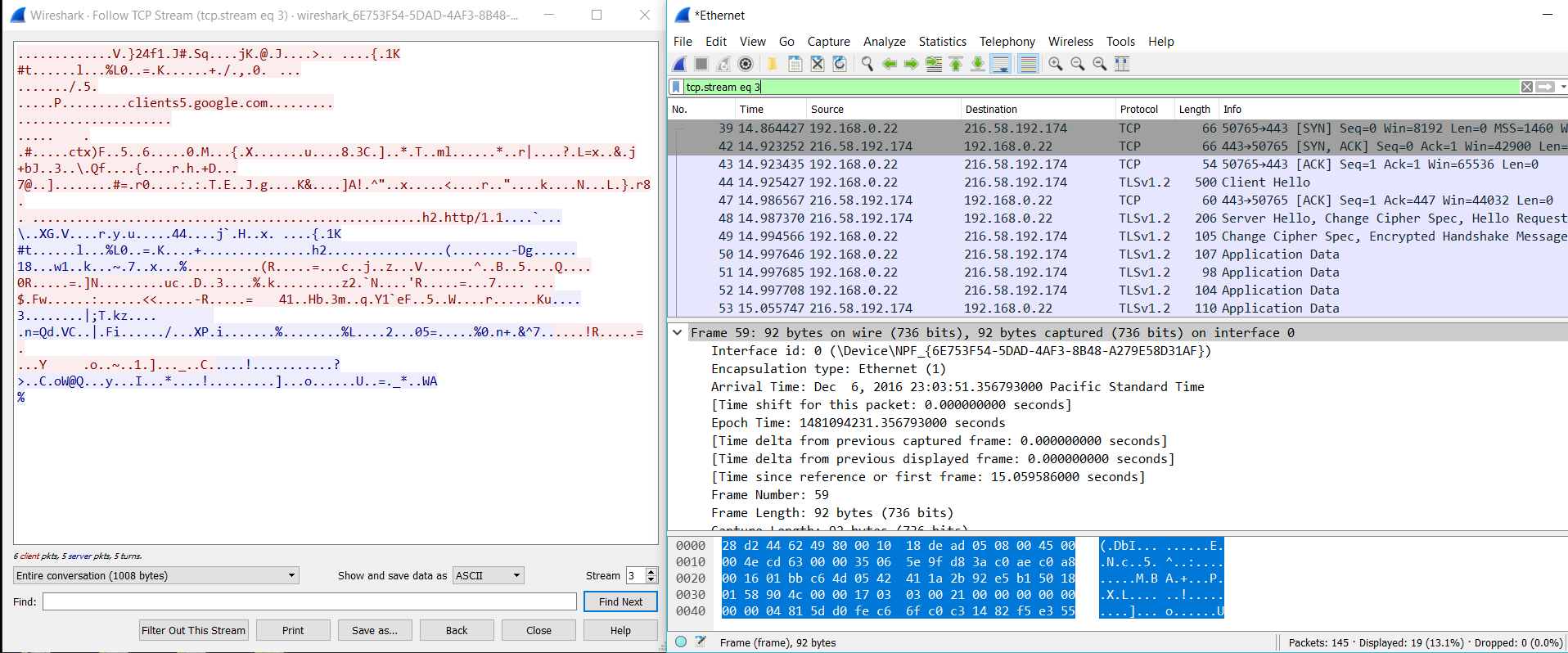
Wireshark works and how to analyze some simple network traffic.

Make sure that Wireshark is installed and launch it.

🡪 Start a new capture session with the ‘Capture Filter’ set to ‘port 25 or

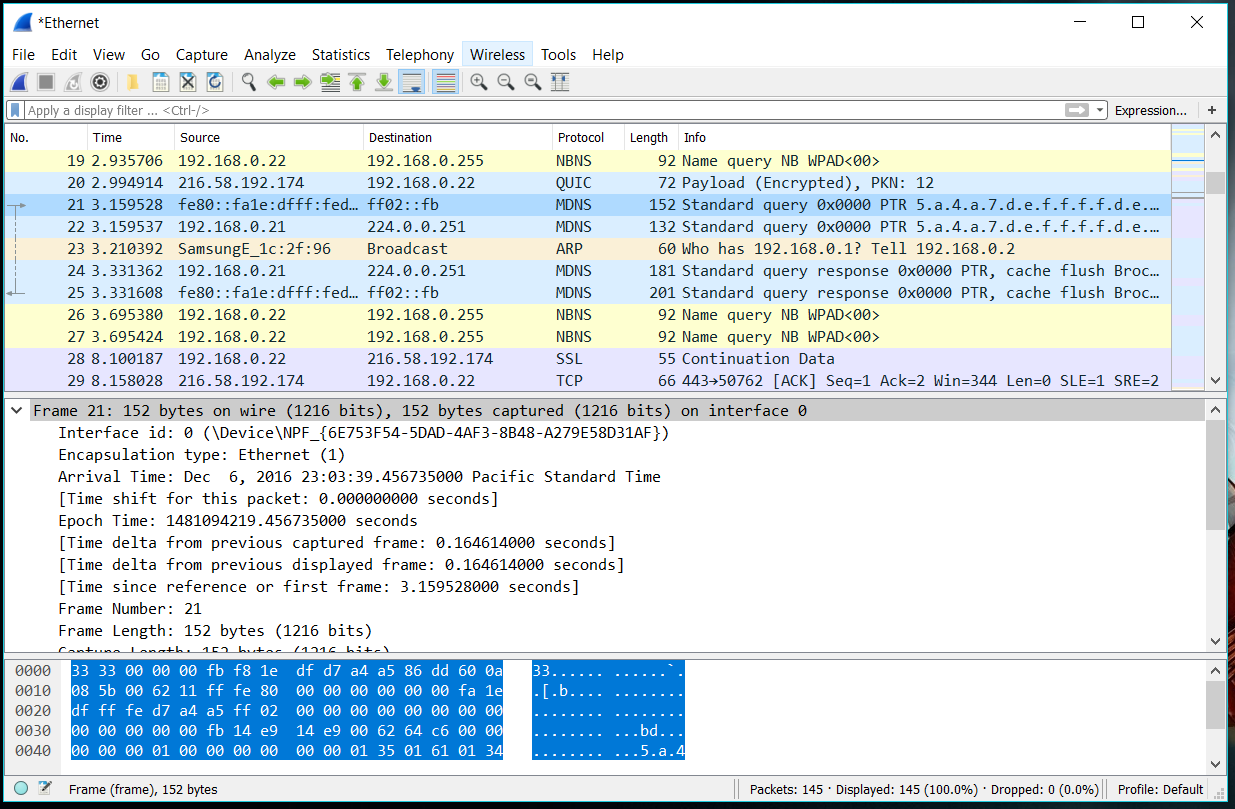
port 587’ and send an email using your favorite MUA / MSA. Look at

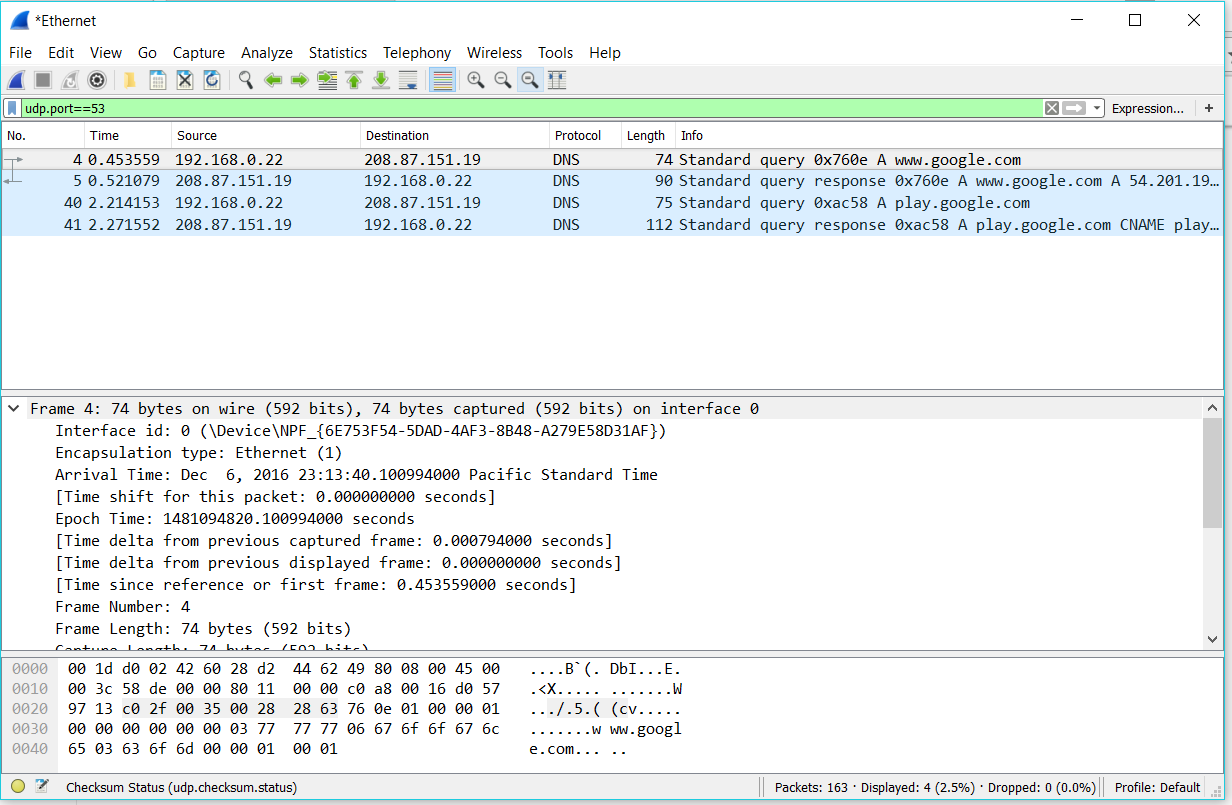
what is displayed, also try the ‘Analyze’🡪 ‘Follow TCP Stream’ feature.



🡪 Use a ‘Capture Filter’ set to port 53 and try some different DNS queries.

Try to generate both a UDP and a TCP stream.





3 BitTorrent Client

There are many different BitTorrent clients, but for this class we will be using the

Transmission client, as it is a stable client, with most features of the BitTorrent

protocol, and is available cross-platform as well.

Start Transmission without adding a torrent file yet. To ease the analysis, we are

limiting the bandwidth for downloading and uploading to maximally 100 kb/sec,

perhaps even less. Also make sure that you write down the listening port (we will

refer to it as $LPORT) in the Network tab of the Preferences, and deselect the ‘Pick

a random port’ option and the ‘micro Transport Protocol (uTP)’ option. Make sure

that you are generating as little network traffic as possible and quit email, IM and

other traffic generating applications.

Currently Wireshark seems to have a slight problem in recognizing BitTorrent

DHT traffic. In order to work around this, open up Wireshark preferences, open

the Protocols subsection, and select ‘BT-DHT’. Check the ‘Enable BT-DHT heuristic

dissection’ option. Once you have done that, start a new live capture. After you

have captured for a couple of seconds, you can stop it again.

🡺 Use the expression udp.port==$LPORT (where $LPORT is the value you

wrote down earlier) to filter out the BitTorrent DHT traffic. Use http://

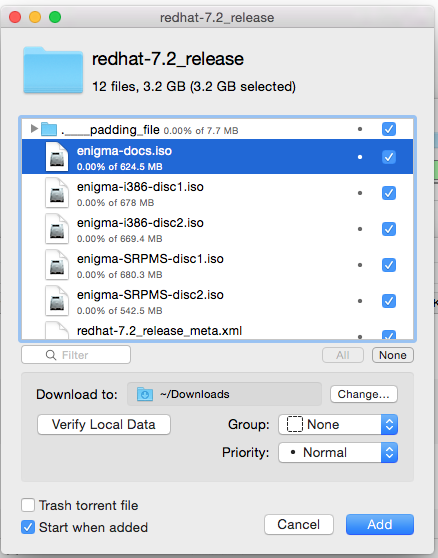
www.bittorrent.org/beps/bep\_0005.html to figure out what is going

on.

🡺 Pick a popular Linux or Unix distribution file image, and find the torrent

download option. Look at the downloaded file and try to figure what is in the

file.



Then start capturing traffic again in Wireshark. Make sure that you are capturing

traffic before starting the torrent download, or you will miss some initial

exchanges. Open the torrent file in Transmission, and start the downloading process.

Make sure that the client shows some progress in downloading data, and then

stop the capture in Wireshark after a few minutes. We will first look at the information

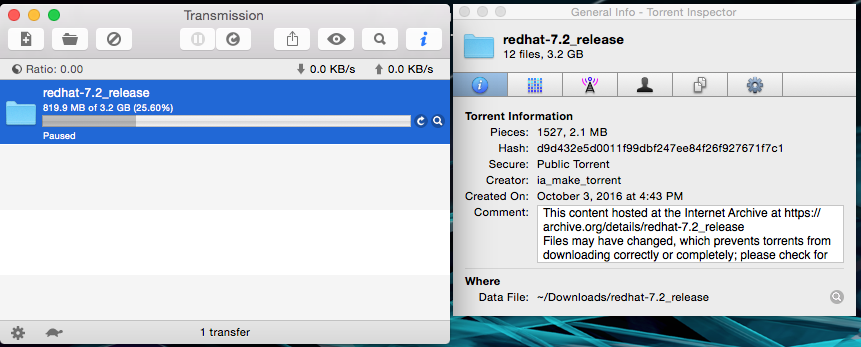
from the Transmission client itself before we look at the traffic. Leave the

downloading enabled.

🡺 Examine the behavior of the Transmission client, open the Inspection window

to see the properties of the torrent you are downloading. Try to understand

all the information presented to you.



Once you understand some of the basic operations in the Transmission client,

we are going to look at what happens on the wire.

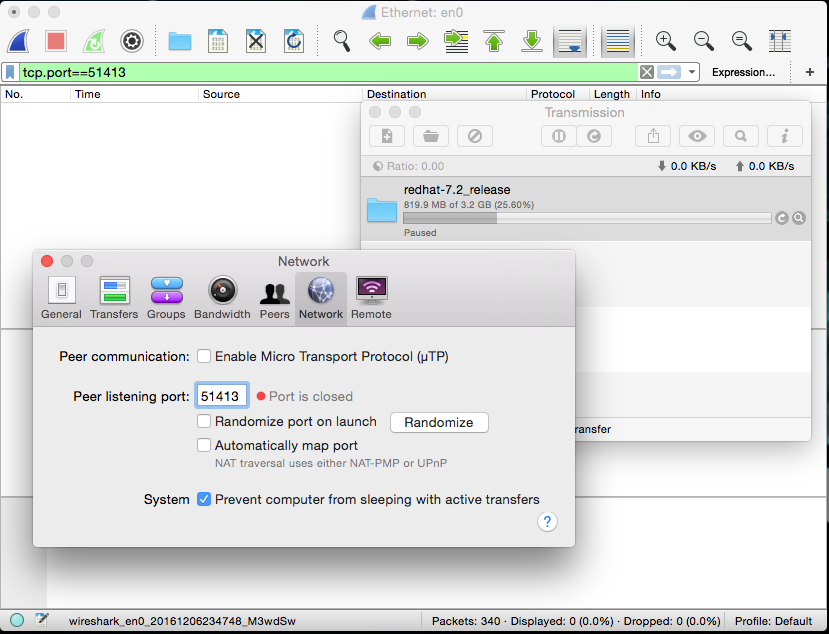
🡺 Examine the capture that you performed with Wireshark, start with the expression

tcp.port==$LPORT (where $LPORT is the value you wrote down

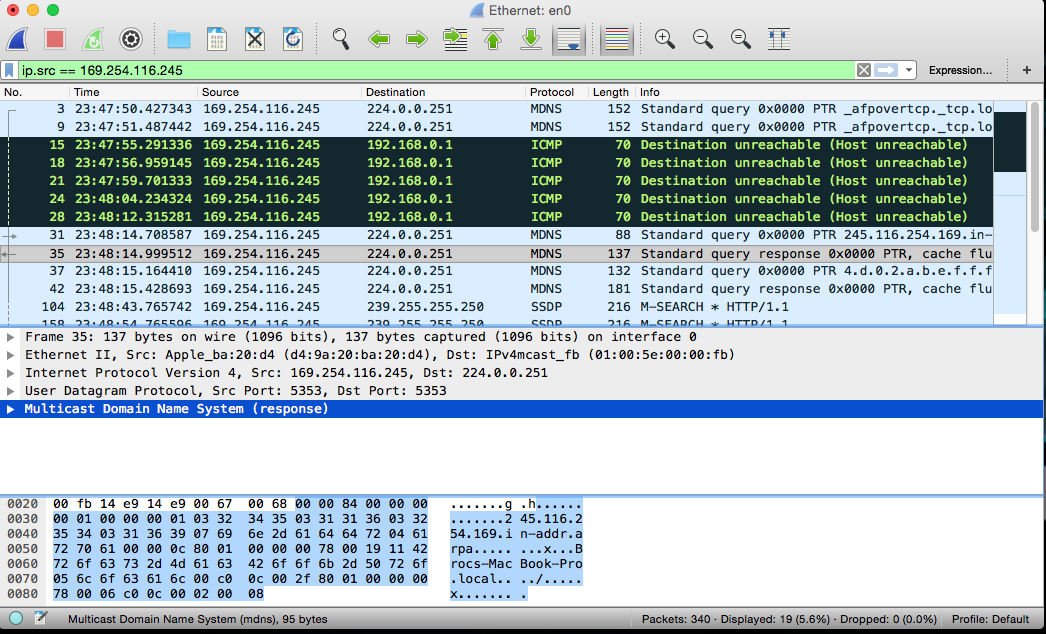
earlier). Select a packet with a ‘Handshake’, right-click on the IP-address

and select ‘Apply as Filter ! Selected’ to show the messages exchanged

with that particular IP.



searching for the LPORT returned nothing



using Apply as Filter ! Selected

🡺 Try to identify the steps involved in the communication between peers.

1st someone creates a torrent that contains meta data with trackers. When a node wants to find a peer, it uses the distance metric to compare the info has and then contacts the nodes and asks for the contact information of peers currently downloading the torrent. The returned value for a query includes a token that changes and is a hash of the ip address concatenated into a secret to prevent malicious hosts.

Now we have examined a traditional torrent using a torrent file. Next we start

with analyzing the behavior of a BitTorrent transfer using a magnet link. Make

sure that your Transmission still has the speed limits in place, and that you start a

Wireshark capture before you add the new transfer.

🡺 Add the magnet link:

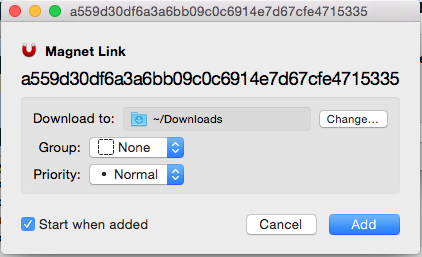
magnet:?xt=urn:btih:a559d30df6a3a6bb09c0c6914e7d67cfe4715335.

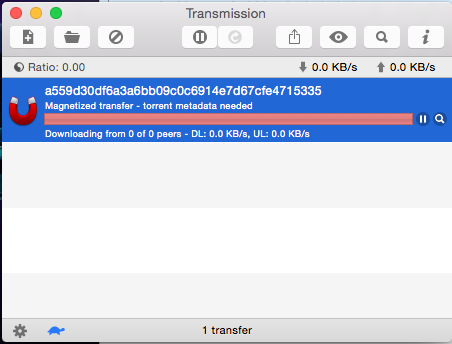
Analyze the traffic from Transmission regarding the magnet link and the final

transfer. Some questions to help your search: Can you see where the title

comes from? Where are you downloading the information from? And the

pieces?





Can you see where the title comes from?

No, the Transmission says the torrent metadata is needed.

Where are you downloading the information from? And the pieces?

The information and the pieces should be downloading from other peers.