Implementation of TCP in ns-3

This contains notes from the talk delivered by Mohit P Tahiliani at WNS3 2017.
The video link can be found here.
Compiled by: Kaushik S Kalmady - Github, Blog

> The source code for TCP in ns-3 can be found in the `internet` module at `src/internet/model/`

Important Classes

TcpHeader

- Implements all the flags required for the TCP Header
- Setters and getters are provided in this class for each of the flags
- Contains methods for serialization and deserialization

TcpSocket

- Abstract base class for all TcpSockets
- It contains TCP socket attributes that can be reused across different implementations
- Examples of attributes: *SndBufSize, RcvBufSize, InitialCwnd, SegmentSize* etc.
- We set congestion window in terms of bytes

TcpSocketFactory

- Abstract base class which defines APIs for TCP sockets
- It contains global default variables to initialize sockets

TcpSocketFactoryImpl

- This is the implementation of socket factory for ns3 TCP
- It creates the sockets of the type TcpSocketBase
- Use this class to create new TCP sockets whenever you need to

> Here is a breif overview on Unix Sockets

TcpSocketBase

- Base class for implementing TCP stream sockets

- Contains all the essential components of TCP
- Provides a socket interface for upper layers to call
- This is used to extract necessary information that the upper layers require
- An intermediate `TcpSocketState` class records all the info
- Components include:
    - Sliding Window Mechanism
    - Fast Recovery
    - Fast Retransmit
    - Enable/Disable window scaling, timestamps
    - Congestion Control Interface
    - Congestion control state

TcpSocketState

- Used to record the congestion state of a connection
- It saves the information passed between the socket and the congestion control algorithm
- This includes
    - Last sequence number acknowledged
    - Next sequence number to be transmitted
    - What is the current state of my congestion control algorithm (CA_OPEN, CA_RECOVERY etc)
    - Congestion window state
    - Current ssthresh value
    - This is useful in test case scenrios when we need to monitor the valued of above said fields and draw inferences

TcpCongestionOps

- Interface between main socket code and congestion control algorithm
- It is the abstract class for congestion control
- Most TCP implementatons use `TcpNewReno` which inherits this class, you can also use `TcpCongestionOps`
- Variables are stored in TcpSocketState
- Design inspired by linux
- If you inherit from this class you will have to implement most functions that are defined here that pertian to the functioning of your TCP
- For e.g. How do you increase congestion window?
- If you are to implement a new TCP the functions in this abstract class are the one's where you will implement the important parts

Examples

TCP examples can be found in `examples/tcp/`

The demonstration of example programs begins at around 34:20 in the video.

Steps to add a new TCP extension to ns-3

1. Create the `tcp-new.h` and `tcp-new.cc` files in `src/internet/model/` .
2. Create a new class for your TCP extension that can be inherited from `TcpCOngestionOps` or `TCPNewReno` .
3. Implement specific methods like the following. These will be specific to your implementation
   - GetSsThresh
   - IncreaseWindow
   - PktsAcked
4. Make the necessary modifications in `src/internet/wscript/` . This is where you make ns-3 aware of the new files that you have added so that they can be compiled.
5. Configure and build, fix errors if any.
6. Write an example program for this extension, or you can use the one that already exists.
7. Write tests and add documentation at `src/internet/doc/tcp.rst`

References

- Links to all the classes described above can be found here : TCP : ns-3 documentation