Multi-addressed Multipath TCP

Compiled by :- Asst. Prof. Rashmi Pote

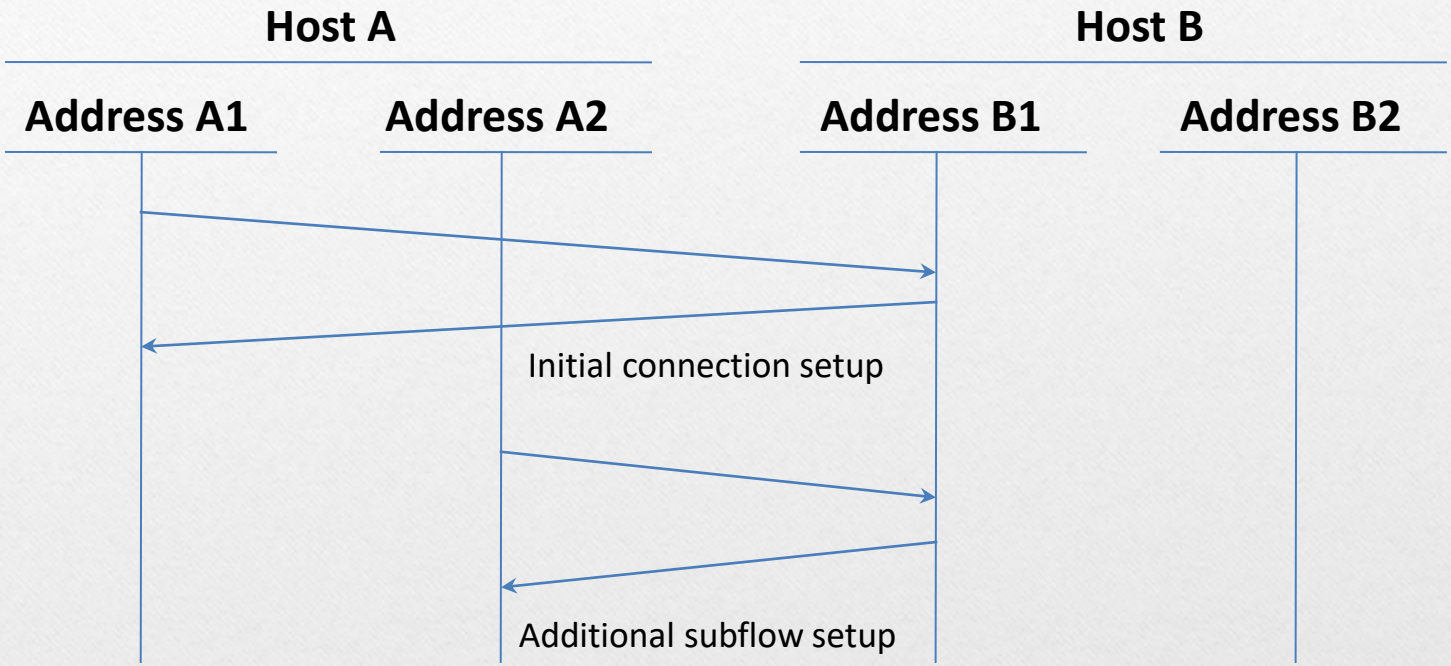
Objectives

- An easy-to-deploy Multipath TCP solution
- Our constraints:
  - Assume that one or both endpoints are multihomed and multi-addressed
  - Backwards-compatible with current, regular TCP
    - External: function through middleboxes
    - Applications: provide same service to regular TCP
    - Fall-back: to regular TCP in case of failure

MPTCP Operation

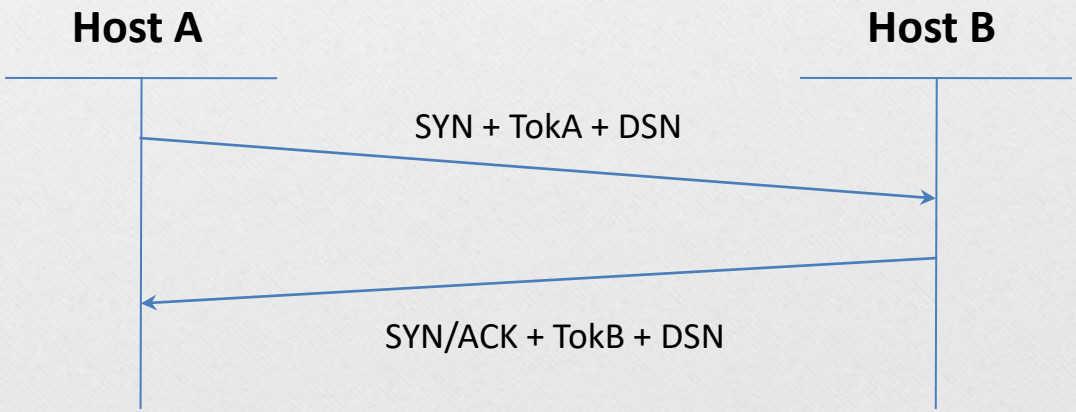
- An MPTCP *Connection* between endpoints is formed of one or more MPTCP *Subflows*
- Subflows will be created between different IP address pairs
- TCP Options used in subflows:
  - For a new subflow to join an existing connection
  - For data-level sequence numbering
  - For closing connections

MPTCP Example



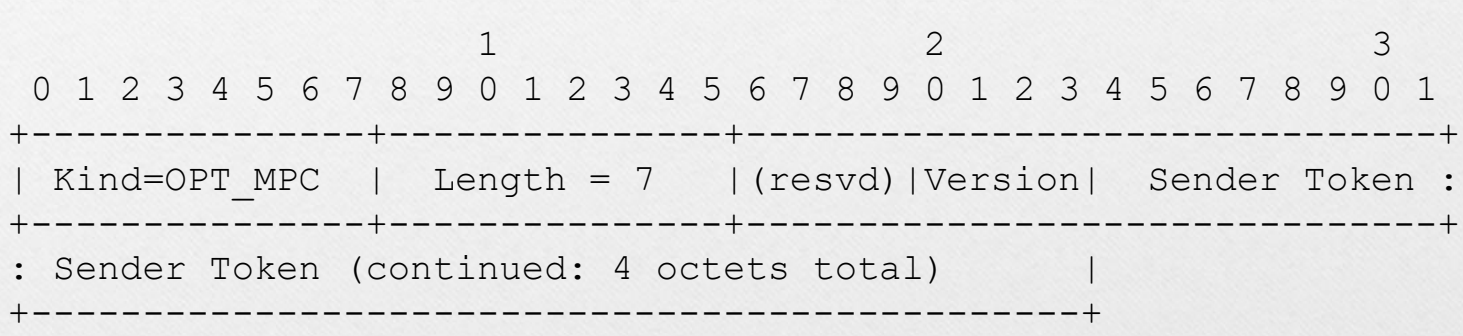
Session Initialisation

- Standard TCP SYN/ACK exchange with additional MPTCP option:
  - Sender's token to identify this connection
  - Initial data-level sequence number



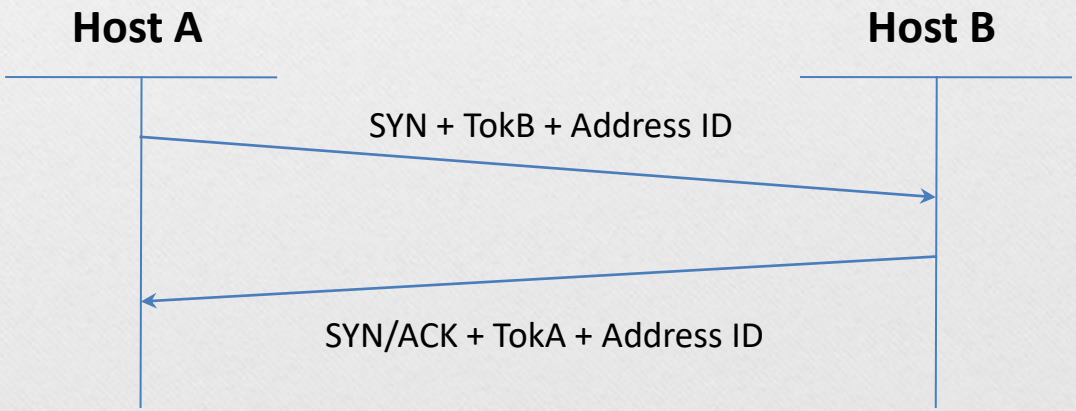
“Multipath Capable” Option

- To declare a sender’s token for this connection in the first SYN/ACK exchange



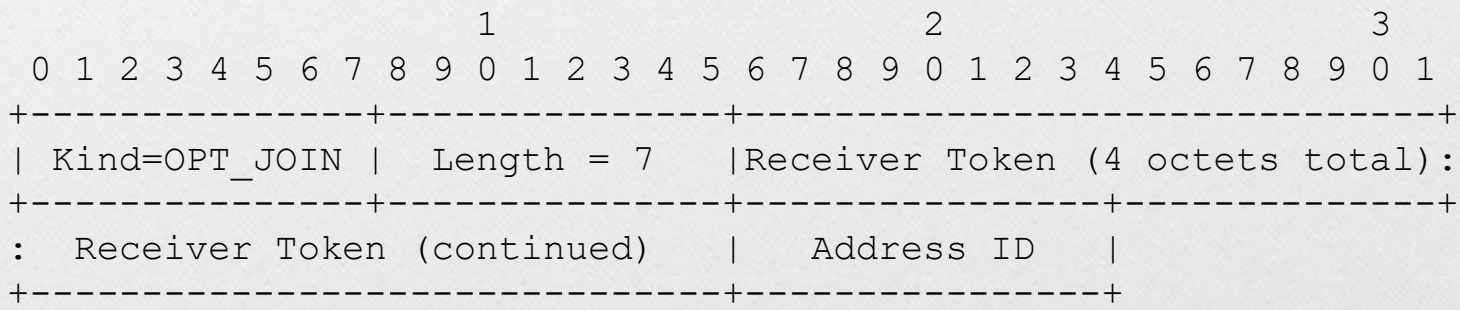
Starting a New Subflow

- One or both of local and remote IP addresses will be new
- TCP option contains:
  - Receiver's token to identify this connection
  - Sender's identifier for source address



“Join Connection” Option

- Added to SYN containing the receiver’s identifying token for the connection the sender wishes to join
- Also contains sender’s identifier for their source address

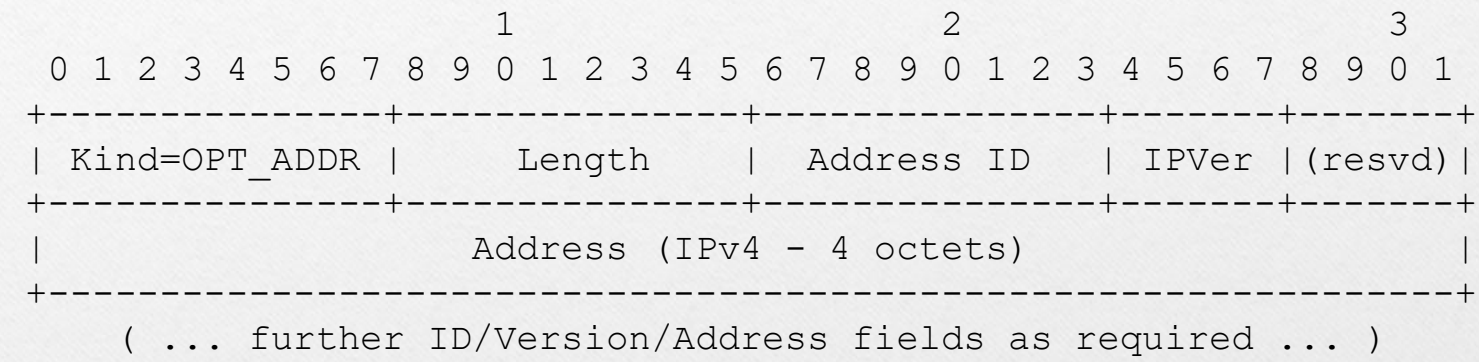


Address Knowledge Exchange

- But how to know a remote end's addresses?
- Add Address is signalled in TCP options on existing subflow, containing:
  - Address (v4 or v6 depending on IPVer)
  - Sender's unique Address Identifier
    - Can be correlated with that seen in SYN
- Remove address also possible when interface is unexpectedly lost
  - Removed by Address ID, to permit NATs

Adding Addresses

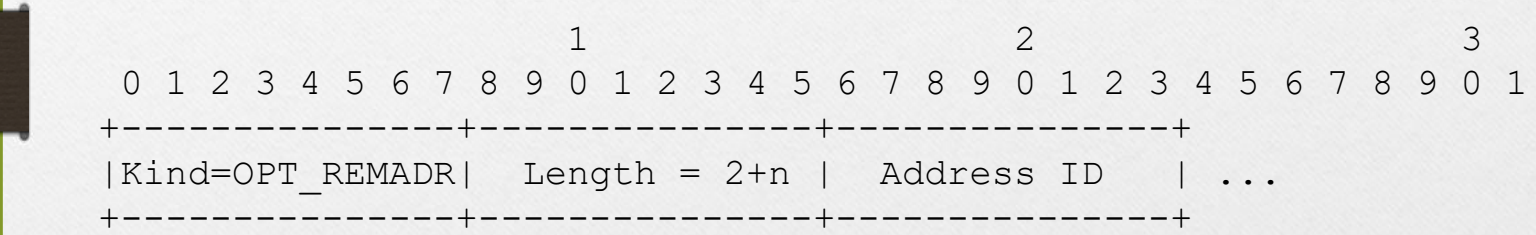
- Add Address: declares a local address to a remote endpoint, with local identifier



- Remote endpoint can attempt to set up new subflow(s) to this address

Removing Addresses

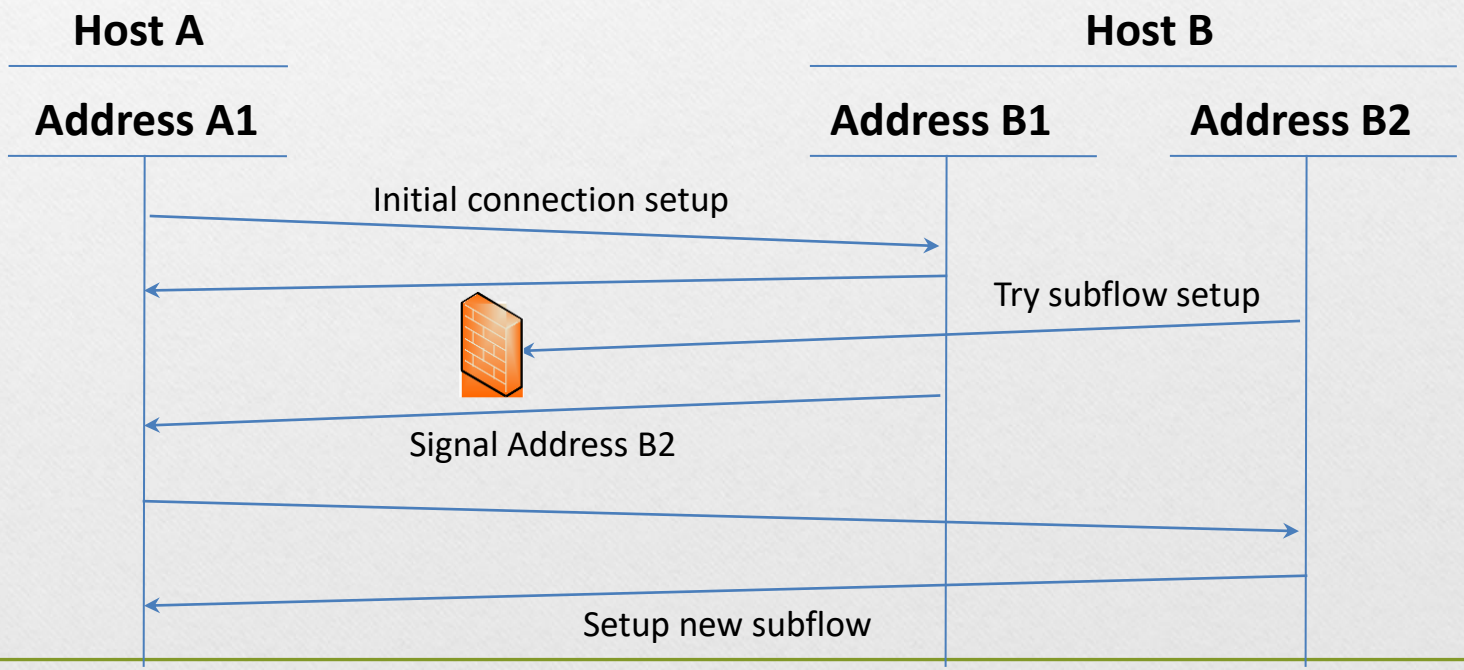
- Remove Address: declares a previously announced address as no longer valid and to be removed from connection



- Will trigger receiving endpoint to close all subflows using that address

Need for Address Signalling

- Many hosts will have firewalls/NATs that permit connections only in one direction
- Therefore, signalling addresses allows a receiver to open a connection in the opposite direction, e.g.

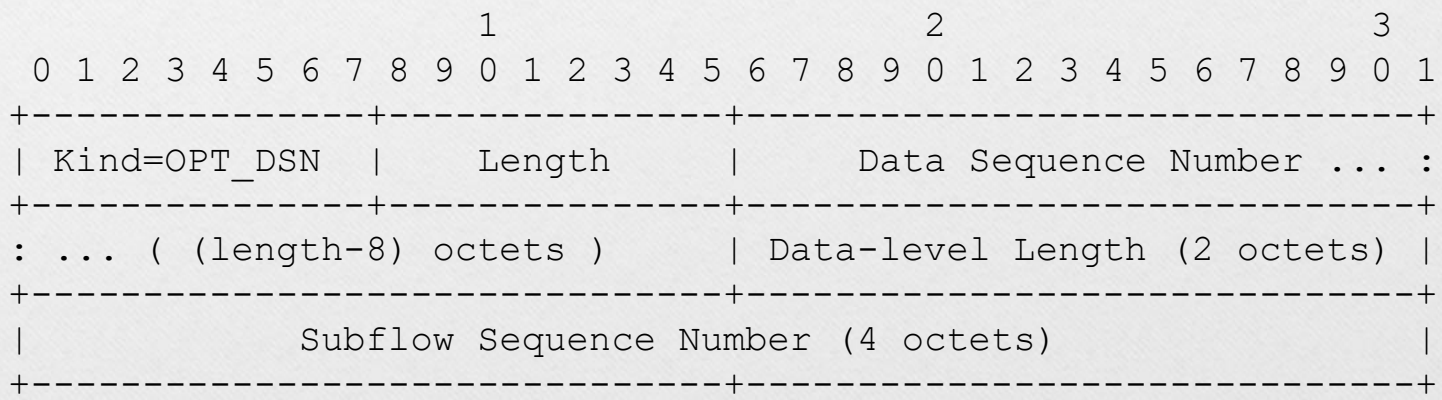


General Operation

- MPTCP connection is formed of multiple TCP subflows, with coupled congestion control and shared receive buffer
- Subflows have contiguous sequence numbering
- Data is reassembled by connection-level sequence numbering
- MPTCP can retransmit failures on original or new path (but if using a new path, must also retransmit on original for continuity)

Data Sequence Numbering

- Data Sequence Mapping option declares:
  - Subflow sequence number mapped to data-level sequence number
  - Number of bytes for which this mapping is valid
  - Data sequence number can be 64-bits



More on Data Sequence Mapping

- The Data Sequence Mapping need not be sent in every packet, and can be omitted when the sender is sure the receiver will understand (e.g. multiple packets covered by same length; or lone subflow)
- An initial data sequence number should be set at the first handshake

Closing Connection

- Subflows operate independently so can be closed with a FIN as normal
- If an interface is unexpectedly lost, the Remove Address option can initiate a rapid cleanup from both endpoints
- The DATA FIN option (on top of a TCP-level FIN) indicates the end of the connection
- RST has subflow-only relevance

Security and Threats

- Aim for baseline: no worse than current TCP
- Some security considerations so far:
  - Use of existing TCP for sub flows minimizes new threats from targeting third party addresses
  - Hijacking could be possible with token guessing
    - Additional security with e.g. ensuring correlation with signals on existing subflows
    - This would also reduce issues with time-shifted attacks

Open Issues

- Is proposed handshake mechanism sufficient?
- Receiver policy control – how?
- Do we want Subflow IDs or Address IDs?
- Do we need any data-level ACKs?
- Do we want Connection IDs given in every packet? (Possibility of aiding IDS etc?)