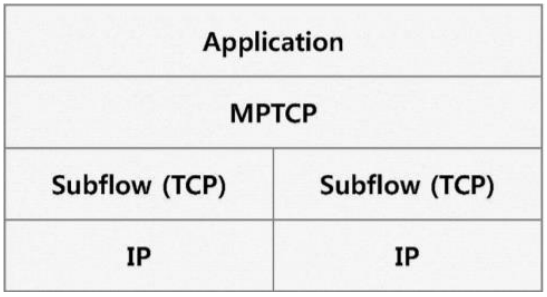
MPTCP Stack Packet Capture

개요

- MPTCP는 TCP의 확장버전으로 기존의 TCP를 사용하는 Application의 변경을 하지 않고 MPTCP를 사용할 수 있도록 한다.
- 아래의 그림에서 확인할 수 있듯이 MPTCP는 각 각의 개별 TCP 연결을 Subflow로 사용한다.
  또한 각 Subflow에서 송수신되는 Data 들은 MPTCP Stack에서 분배 및 재결합된다. (분배는 송신 시, 재결합은 수신 시)
- 각 Subflow는 각 각의 IP Stack을 타게 됨으로 서로 다른 경로(WIFI, Cellular 등)를 통해 전송되는 것을 알 수 있다.

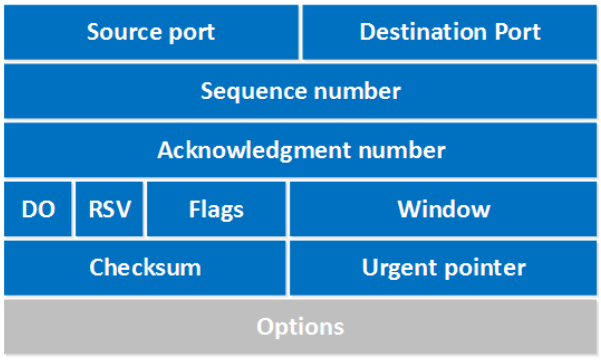


<그림> MPTCP Layer

- 아래의 모든 내용은 MPTCP 표준화 문서[RFC6824] 내용을 토대로 작성하였습니다.

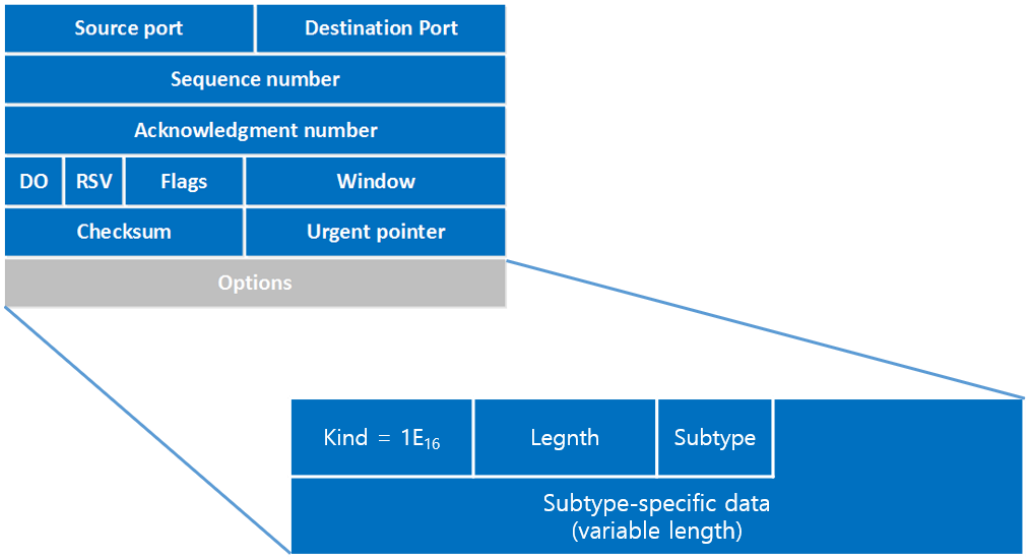
MPTCP Packet

- MPTCP Packet은 TCP Option 필드에 부착된다.



<그림> TCP Packet Header

- TCP Option필드의 하위에는 Kind, Length, Subtype이라는 필드가 존재한다. 세 필드의 역할은 다음과 같다.
  - Kind : TCP Option 종류 (Ex. MSS, Timestamps, SACK, MPTCP 등)
  - Length : TCP Option 패킷의 길이
  - Subtype : TCP Option 에서 사용하는 메시지 종류 (Ex. MPTCP의 경우 MP_CAPABLE, MP_JOIN, ADD_ADDR 등)
- MPTCP는 기본적으로 10진수 30의 값을 TCP Option의 Kind 필드에 고정적으로 담으며 Length와 Subtype은 MPTCP의 Signal마다 가변적으로 적용된다.



<그림> MPTCP Packet Header

- 앞서 말하였듯이 MPTCP Packet은 TCP Option 필드에 부착된다.
  또한, MPTCP Packet은 TCP Option의 하위 필드인 Kind 필드를 10진수 30값(16진수 1E)으로 고정한다.
  해당 Kind 값을 통해 Source와 Destination은 MPTCP 패킷임을 알 수 있다.

MPTCP Subtype Table

- 이번에는 MPTCP에서 사용하는 여러 Subtype들을 살펴보자.

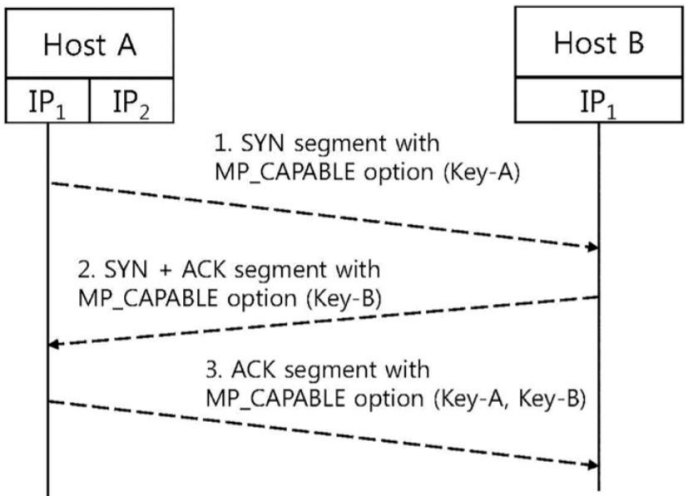
Symbol	Name	Value
MP_CAPABLE	Multipath capable	0x0
MP_JOIN	Join connection	0x1
DSS	Data sequence signal	0x2
ADD_ADDR	Add address	0x3
REMOVE_ADDR	Remove address	0x4
MP_PRIO	Change sub-flow priority	0x5
MP_FAIL	Fallback	0x6

<표> MPTCP Subtype Table

- Subtype
  - MP_CAPABLE : 가장 초기 MPTCP Connection을 연결할 때 사용되는 Subtype
  - MP_JOIN : 새로운 MPTCP의 Subflow를 연결할 때 사용하는 Subtype
  - DSS : 여러 Subflow에서 수신된 순서가 정렬되지 않은 데이터들을 MPTCP Layer에서 재결합하기 위해 사용하는 Subtype
  - ADD_ADDR : 연결을 위한 것이 아닌 잠재적인 NIC의 Address를 알려주기 위해 사용하는 Subtype
  - REMOVE_ADDR : 잠재적인 NIC의 Address를 삭제요청하기 위해 사용하는 Subtype
  - MP_PRIO : 현재 연결된 Subflow의 우선순위를 변경하기 위해 사용하는 Subtype (Backup, Primary)
  - MP_FAIL : MPTCP Connection에 문제가 발생하여 TCP Connection으로 Fall-back 하기 위해 사용하는 Subtype

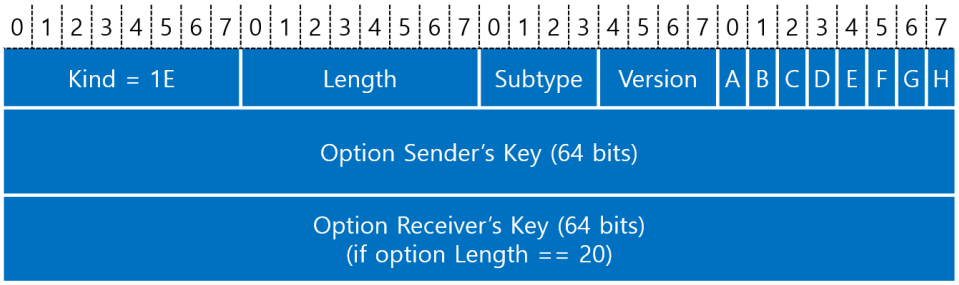
MP_CAPABLE - MPTCP 초기 연결

- 초기 MPTCP 연결은 하나의 Subflow의 SYN, SYN/ACK, ACK 패킷을 통해 진행된다.
  그리고 3-way handshake와 더불어 MPTCP의 MP_CAPABLE 옵션이 3개의 패킷(SYN, SYN/ACK, ACK)에 함께 부착된다.



<그림> MPTCP 초기 연결 시퀀스 다이어그램

- MPTCP 초기 연결 시에 두 호스트 간에는 암호화 알고리즘을 활용해 Key를 나눠 갖는다.
  현재 [RFC6824] 규격에 따르면 해당 암호화 알고리즘은 SHA1을 사용한다.
  이 Key의 목적은 MPTCP Connection에 새로운 Subflow를 추가할 때에 Connection에 등록될 새로운 Subflow를 인증할 때 사용한다.



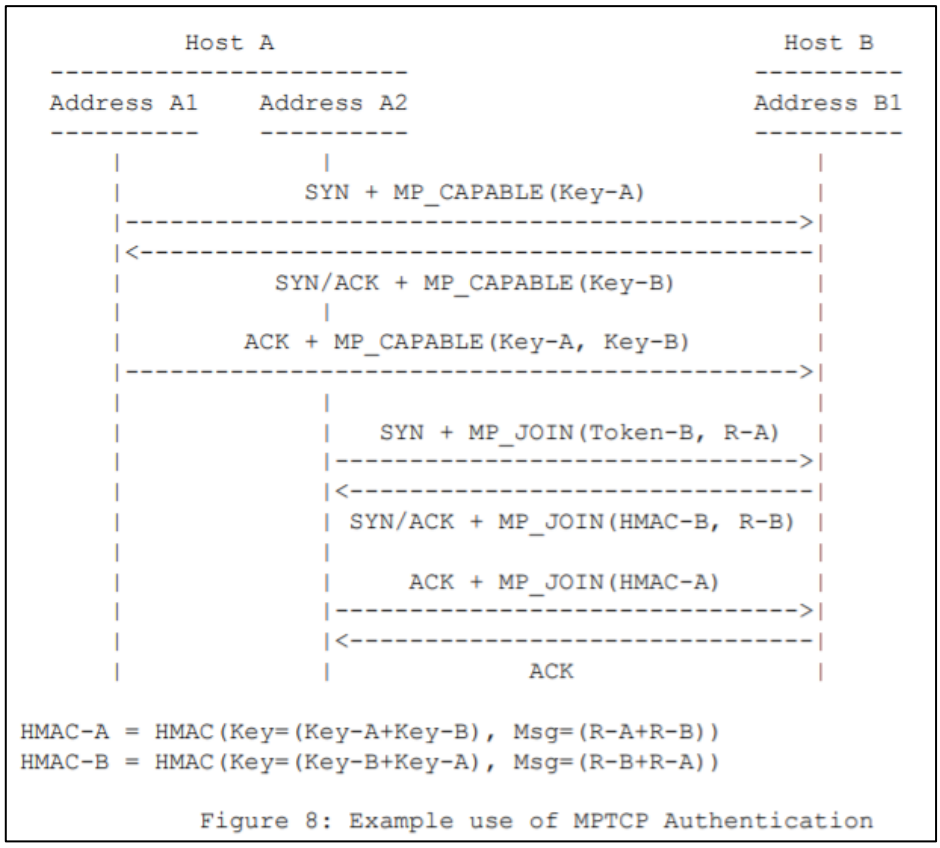
<그림> MP_CAPABLE Packet Format

- 위 그림은 초기 연결 시 사용되는 패킷의 포맷이다. 이 패킷의 필드들은 다음과 같은 의미를 가진다.
  - Kind
    : MPTCP를 의미하는 $1E_{16}$ 값으로 고정된다.
  - Length
    : 초기 연결 시에는 $12_{10}(0C_{16})$나 $20_{10}(14_{16})$값으로 고정된다. SYN, SYN/ACK일 때 12_{10}로 고정되며, ACK일 때엔 20_{10}으로 고정된다.
  - Subtype
    : MP_CAPABLE을 나타내는 0_{16}으로 고정된다.
  - Version
    : 현재 MPTCP Stack의 버전을 나타낸다.
  - A Flag
    : Checksum 사용 여부 (1-사용 / 0-미사용)를 나타낸다. 다시 말해, SYN에서 A를 1로 설정한 뒤 보내게 되면 Sender 측이 Checksum 사용을 원한다고 표현한 것이다. 만약, SYN/ACK에서 A가 0으로 설정되었다면 Receiver 측은 Checksum을 사용하지 않는다는 의미이다. 결론적으로 마지막 ACK 패킷의 A Flag 값을 통해서 이 Connection에서의 Checksum 사용 여부를 알 수 있게 된다.
  - B Flag
    : 확장성 Flag를 나타내며, MPTCP의 확장된 버전이 존재할 시 사용되는 Flag이다. 즉, 확장된 MPTCP 버전이라면 해당 Flag가 1로 설정된다.
  - C~H Flag
    : 암호화 알고리즘 협상 Flag이다. 현재는 H만 1로 설정하여 알고리즘을 협상하며, H Flag는 HMAC-SHA1 암호화 알고리즘을 나타낸다.
    이것을 통해서 MPTCP Connection을 인증하고 새로운 Subflow를 열 수 있기 때문에 반드시 암호화 알고리즘은 협상되어야 한다.
  - Sender Key
    : HMAC-SHA1 암호화 알고리즘을 통해 생성된 Sender측 Key 값
  - Receiver Key
    : HMAC-SHA1 암호화 알고리즘을 통해 생성된 Receiver Key 값

❖ SYN에서는 Sender Key만 확인되며, SYN/ACK에서는 Receiver Key만 확인된다. 마지막 ACK에서 Sender와 Receiver측의 두 Key값을 모두 확인할 수 있다. 또한, 이 세개의 Key값은 SHA-1에서 생성된 가장 중요하지 않은 64bit로 구성된다. (세 패킷의 MP_CAPABLE Length 필드 값이 다른 이유)

MP_JOIN - 새로운 Subflow 연결

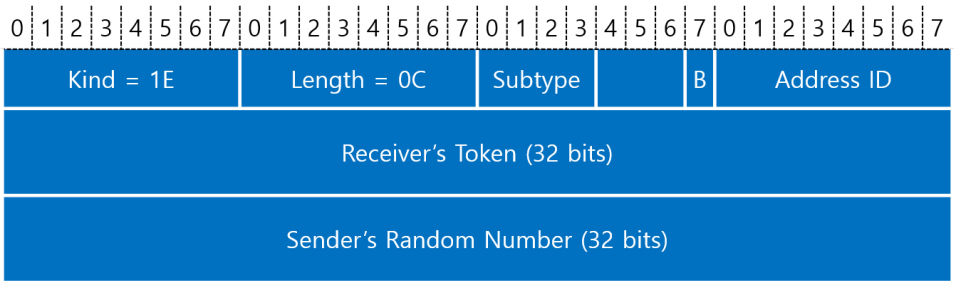
- MPTCP Connection이 MP_CAPABLE 교환으로 초기 연결이 완료되었으면 추가 Subflow를 Connection에 추가할 수 있다.
  또한, ADD_ADDR Signal을 통해 사용 가능한 주소를 미리 수신 측과 교환한 이후에 새 Subflow 연결을 수행할 수 있다. (Address ID 사전교환)



<그림> 새로운 Subflow 연결 시퀀스 다이어그램

- 새 Subflow는 일반적인 TCP 3-Way Handshake로 시작된다.
  또한, 3-Way Handshake 시에 사용되는 패킷과 함께 "MP_JOIN"를 함께 사용하여서 새로운 Subflow를 Connection에 인식시킨다.

- 새로운 연결하는 Subflow를 인증하기 위해 MP_CAPABLE 때에 사용되었던 Key 정보를 사용한다.

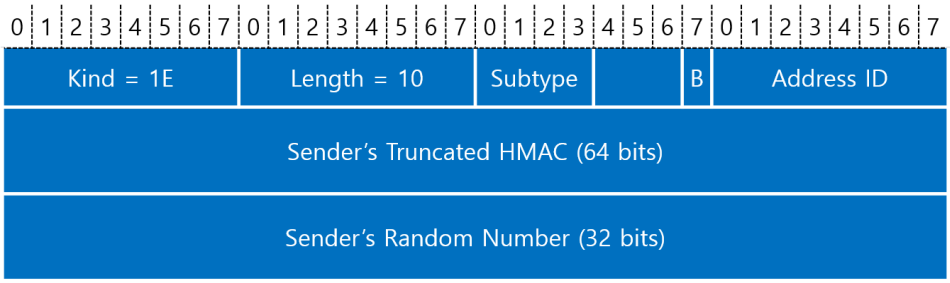


<그림> MP_JOIN Packet Format (SYN + MP_JOIN)

- 위 그림은 MPTCP의 새 Subflow 연결 요청(SYN)할 때 사용되는 패킷의 포맷이다. 이 패킷의 필드들은 다음과 같은 의미를 가진다.
  - Kind
    : MPTCP를 의미하는 $1E_{16}$ 값으로 고정된다.
  - Length

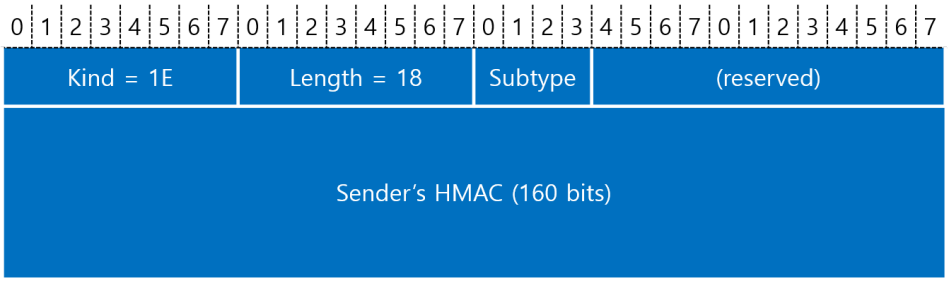
: 새 Subflow의 연결 요청(SYN + MP_JOIN) 패킷의 길이를 나타낸다. (16진수)

- Subtype
  : MP_JOIN을 나타내는 1_{16} 값으로 고정된다.
- B Flag
  : 새로 연결할 Subflow를 Backup으로 할지 Primary로 할지 설정하는 Flag이다. (Backup = 1, Primary = 0)
  : B Flag의 왼쪽 3bit는 0으로 고정된다.
- Address ID
  : Subflow의 Address ID를 나타내며 이 값은 ADD_ADDR을 통해 사전에 교환된 상태이다.
  : 양측의 Host는 Address ID와 IP Address를 미리 매핑 시켜 놓아야 한다.
  : 초기 MP_CAPABLE에서 사용된 Subflow의 Address ID 값은 default로 0이다.
  : 이 필드의 목적은 IP 헤더가 Middleboxes(NAT 등)에 의해 변경되더라도 패킷의 Source 주소 식별을 가능케 해준다.
- Receiver's Token
  : MPTCP Connection 식별, 초기 연결(MP_CAPABLE)할 때 생성된 32Bit Token(Host-A인 경우 Host-B에서 생성된 Token)
- Sender's Random Number
  : Replay Attack 방지를 위한 임의 번호(Nonce)



<그림> MP_JOIN Packet Format (SYN/ACK + MP_JOIN)

- 위 그림은 MPTCP의 새 Subflow 연결 요청 응답(SYN/ACK)할 때 사용되는 패킷의 포맷이다. 이 패킷의 필드들은 다음과 같은 의미를 가진다.
  - Length
    : 새 Subflow의 연결 요청 응답(SYN/ACK + MP_JOIN) 패킷의 길이를 나타낸다. (16진수)
  - Sender's Truncated HMAC
    : SYN + MP_JOIN을 받은 수신자는 송신 측의 HMAC(가장 왼쪽 64bit)로 응답한다.
  - Sender's Random Number
    : SYN + MP_JOIN을 받은 수신자는 Sender의 HMAC과 함께 임의 번호(Nonce)로 응답한다.

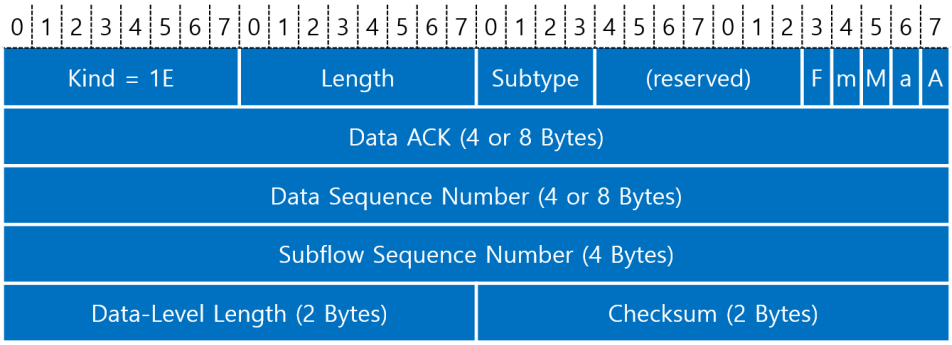


<그림> MP_JOIN Packet Format (ACK + MP_JOIN)

- 위 그림은 MPTCP의 새 Subflow 연결이 완료(ACK) 되었을 때 사용되는 패킷의 포맷이다. 이 패킷의 필드들은 다음과 같은 의미를 가진다.
  - Length
    : 새 Subflow의 연결 완료(ACK + MP_JOIN) 패킷의 길이를 나타낸다. (16진수)
  - Sender's HMAC
    : 연결 요청 측(이니시에이터)의 인증 정보를 담는 필드이다.
    : ACK + MP_JOIN 패킷을 전송하면 이니시에이터의 Subflow가 PRE_ESTABLISHED 상태가 되고, 수신 측의 ACK를 수신한 경우 ESTABLISHED 상태로 전환된다.

DSS - 데이터 송수신

- Sender Application에서 하나의 입력 Data Stream을 가져와 이것을 하나 이상의 Subflow로 분할하며,
  그것을 다시 조립하고 순서대로 Receiver Application에 전달될 수 있도록 MPTCP는 Data Sequence Signal(DSS) 옵션을 사용한다.



<그림> DSS Packet Format

- 위 그림은 MPTCP의 데이터 송수신시 사용되는 패킷의 포맷이다. 아 패킷의 필드들은 다음과 같은 의미를 가진다.
  - Kind
    : MPTCP를 의미하는 $1E_{16}$ 값으로 고정된다.
  - Length
    : DSS 패킷의 길이를 나타낸다.
  - Subtype
    : DSS를 나타내는 2_{16}로 고정된다.
  - (reserved)
    : 총 6bit로 padding 역할을 한다. 즉, 0으로 해당 6bit를 모두 채운다.
  - F Flag
    : DATA_FIN을 나타내는 Flag, MPTCP Connection에서 더 이상 보낼 Data가 없음을 의미한다. (Subflow-Level이 아닌 Connection-Level로 이해)
    : DATA_FIN은 Data의 최종 한 Byte를 차지한다. 즉, 보낼 Data가 10Bytes라면 DATA_FIN을 포함해서 총 Data 크기는 11Bytes가 되는 것이다.
  - m Flag
    : 해당 Flag가 1로 설정되었다면, Data Sequence Number 필드가 8 Bytes 길이임을 의미한다. (M flag가 1로 설정되었을 경우에만 의미를 가짐)
  - M Flag
    : 해당 Flag가 1로 설정되었다면, 해당 패킷에 Data Sequence Number, Subflow Sequence Number, Data-Level Length, Checksum이 존재함을 의미한다.
  - a Flag
    : 해당 Flag가 1로 설정되었다면, Data ACK 필드가 8 Bytes 길이임을 의미한다. (A flag가 1로 설정되었을 경우에만 의미를 가짐)
  - A Flag
    : 해당 Flag가 1로 설정되었다면, 해당 패킷에 Data ACK가 존재함을 의미한다.
  - Data ACK
    : MPTCP Connection-Level의 ACK이다. 이 ACK는 표준 TCP의 누적 ACK와 동일한 역할을 수행한다. 또한, 다음 수신할 DSN을 지정한다.
    : Data ACK는 Data와 모든 MPTCP 신호(MP_CAPABLE, MP_JOIN 등)가 원격 단에서 수신되고 승인되었음을 증명한다.
  - Data Sequence Number (DSN)
    : DSN은 0으로 시작할 수 없으며, 블라인드 세션에서 가로채기 더 어렵게 하기 위해 SHA-1 해시의 최소 64bit로 설정된다. (초기 DSN)
    : 송신단에서 전송하는 Application Data의 시작 데이터를 나타낸다.
  - Subflow Sequence Number
    : 각 Subflow에서 관리하는 필드이다. 각 Subflow에서 보내는 Data의 시작 위치를 나타내며, 해당 필드는 DSN에 의해 매핑 되어 재조립된다.
  - Data-level Length
    : MPTCP Conneciton에 의해서 전송되는 Data의 길이를 나타낸다.
    : 예를 들어, 송신할 때 DSN이 80, Data-level Length가 10 이라면 수신단에서는 Data ACK를 90으로 다음 Data를 요청하는 것이다.

: 만약 위와 같은 송신 패킷에 DATA_FIN이 설정되었던 경우라면 Data-level Length가 11이 되고 Data ACK는 91로 응답할 것이다.

- Checksum
  : MPTCP의 Data를 감지할 때 사용하는 것이 아닌, Middleboxes(NAT 등)에 의해 Payload가 조정되었는지 감지하는데 사용된다.
  : 만약 조정되었다면 MPTCP를 유지할 수 없음으로 일반 TCP로 Fall-back 된다.

MPTCP 연결 종료

- Application이 소켓으로 close()를 호출할 때, 이것은 더 이상 전송할 데이터가 없음을 의미한다.
  MPTCP는 이 때 DATA_FIN을 발생시킨다. (DSS의 'F' Flag)

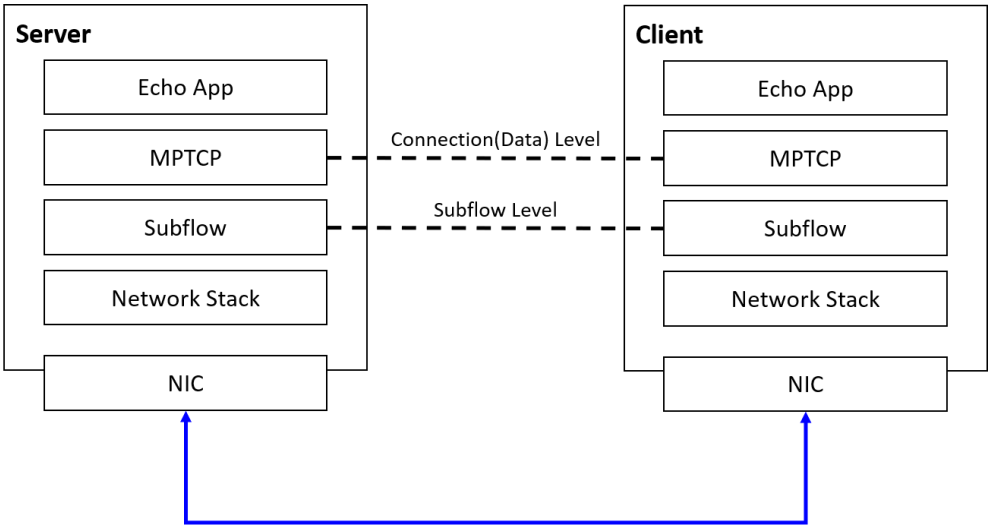
- 기본적으로 Host는 작동 중인 모든 Subflow를 닫아서는 안 된다.
  즉, 모든 Outstanding Data(보냈지만 ACK를 받지 않은 상태인 Data)가 DATA_ACKed가 되어야 지만 모든 Subflow를 닫을 수 있다.

- DATA_FIN을 수신하였다면 모든 Subflow는 표준 TCP FIN 4-way Handshake로 닫힌다.

- 위에서 설명한 대로 개별 Subflow는 표준 TCP FIN으로 종료됩니다.
  모든 Subflow가 FIN 교환으로 닫혔지만 DATA_FIN이 수신 및 승인되지 않은 경우 MPTCP Connection은 Timeout 후에만 닫힐 수 있습니다.
  이는 MPTCP Stack이 Subflow와 Connection Level에서 모두 TIME_WAIT 상태를 갖는다는 것을 의미합니다.
  따라서, 새로운 Subflow를 다시 설정하기 전에 모든 Subflow에서 연결이 끊기는 "Break-Before-Make" 시나리오가 허용됩니다.

MPTCP Physical Packet 캡처

- 다음은 Echo Server – Client 간에 송수신 되는 실제 물리적인 패킷을 캡처하여 나타낸 모습이다. Echo Server – Client의 구조는 다음과 같다.



- 아래의 패킷들은 캡처 된 시간 순서대로 정리해 놓은 것이다.
  - 순서는 MP_CAPABLE -> ADD_ADDR -> MP_JOIN -> DSS -> DATA_FIN이다.

- MP_CAPABLE
  - 초기 연결 시 발생되는 물리적 패킷은 다음과 같다.

```
[MP_CAPABLE] ------------------------------------------------------------------------------------------------------------------------------------------------

203.250.35.157.37746 > 203.250.33.51.9999: Flags [S], cksum 0x9d0e (correct), seq 3198590955, win 29200,
options [mss 1460,sackOK,TS val 2264349 ecr 0,nop,wscale 6,mptcp capable csum {0x877cfcac977e4b88}], length 0
3L >> 4500 0048 12f6 4000 3f06 4bf5 cbfa 239d cbfa 2133
4L >> 9372 270f bea6 9feb 0000 0000 d002 7210 9d0e 0000 0204 05b4 0402 080a 0022 8d1d 0000 0000 0103 0306
MPTCP >> 1e0c 0081 877c fcac 977e 4b88

203.250.33.51.9999 > 203.250.35.157.37746: Flags [S.], cksum 0xae4a (correct), seq 313062176, ack 3198590956, win 64260,
options [mss 1460,sackOK,TS val 1345427636 ecr 2264349,nop,wscale 7,mptcp capable csum {0x813f679475cb7f9f}], length 0
3L >> 4500 0048 0000 4000 4006 5deb cbfa 2133 cbfa 239d
4L >> 270f 9372 12a8 f320 bea6 9fec d012 fb04 ae4a 0000 0204 05b4 0402 080a 5031 98b4 0022 8d1d 0103 0307
MPTCP >> 1e0c 0081 813f 6794 75cb 7f9f

203.250.35.157.37746 > 203.250.33.51.9999: Flags [.], cksum 0xc5dd (correct), seq 1, ack 1, win 457,
options [nop,nop,TS val 2264350 ecr 1345427636,mptcp capable csum {0x877cfcac977e4b88,0x813f679475cb7f9f},mptcp dss ack 4007607362], length 0
3L >> 4500 0050 12f7 4000 3f06 4bec cbfa 239d cbfa 2133
4L >> 9372 270f bea6 9fec 12a8 f321 f010 01c9 c5dd 0000 0101 080a 0022 8d1e 5031 98b4
MPTCP >> 1e14 0081 877c fcac 977e 4b88 813f 6794 75cb 7f9f
MPTCP >> 1e08 2001 eedf 3c42

[MP_CAPABLE] ------------------------------------------------------------------------------------------------------------------------------------------------
```

- ADD_ADDR
  - 초기 연결 후 사용 가능한 NIC의 물리적 주소를 알려주기 위해 발생시키는 물리적 패킷은 다음과 같다.

```
[ADD_ADDR] --------------------------------------------------------------------------------------------------------------------------------------------

203.250.35.157.37746 > 203.250.33.51.9999: Flags [.], cksum 0x471f (correct), seq 1, ack 1, win 457,
options [nop,nop,TS val 2264350 ecr 1345427636,mptcp add-addr id 2 192.168.0.28,mptcp dss ack 4007607362], length 0
3L >> 4500 0044 12f8 4000 3f06 4bf7 cbfa 239d cbfa 2133
4L >> 9372 270f bea6 9fec 12a8 f321 c010 01c9 471f 0000 0101 080a 0022 8d1e 5031 98b4
MPTCP >> 1e08 3402 c0a8 001c 1e08 2001 eedf 3c42

203.250.35.157.37746 > 203.250.33.51.9999: Flags [.], cksum 0x4721 (correct), seq 1, ack 1, win 457,
options [nop,nop,TS val 2264350 ecr 1345427636,mptcp add-addr id 3 192.168.0.25,mptcp dss ack 4007607362], length 0
3L >> 4500 0044 12f9 4000 3f06 4bf6 cbfa 239d cbfa 2133
4L >> 9372 270f bea6 9fec 12a8 f321 c010 01c9 4721 0000 0101 080a 0022 8d1e 5031 98b4
MPTCP >> 1e08 3403 c0a8 0019 1e08 2001 eedf 3c42

[ADD_ADDR] --------------------------------------------------------------------------------------------------------------------------------------------
```

- DSS
  - 데이터 송수신시 발생되는 물리적 패킷은 다음과 같다.

```
[DSS] -------------------------------------------------------------------------------------------------------------------------------------------------

203.250.35.157.37746 > 203.250.33.51.9999: Flags [P.], cksum 0x5a55 (correct), seq 1:6, ack 1, win 457,
options [nop,nop,TS val 2264350 ecr 1345427636,mptcp dss ack 4007607362 seq 2208729441 subseq 1 len 5 csum 0x9e97], length 5
3L >> 4500 004d 12fa 4000 3f06 4bec cbfa 239d cbfa 2133
4L >> 9372 270f bea6 9fec 12a8 f321 d018 01c9 5a55 0000 0101 080a 0022 8d1e 5031 98b4
MPTCP >> 1e14 2005 eedf 3c42 83a6 8961 0000 0001 0005 9e97 6865 6c6c 6f

203.250.33.51.9999 > 203.250.35.157.37746: Flags [.], cksum 0x97d7 (correct), seq 1, ack 6, win 502,
options [nop,nop,TS val 1345427638 ecr 2264350,mptcp dss ack 2208729446], length 0
3L >> 4500 003c e05b 4000 4006 7d9b cbfa 2133 cbfa 239d
4L >> 270f 9372 12a8 f321 bea6 9ff1 a010 01f6 97d7 0000 0101 080a 5031 98b6 0022 8d1e
MPTCP >> 1e08 2001 83a6 8966

203.250.33.51.9999 > 203.250.35.157.37746: Flags [P.], cksum 0x603b (correct), seq 1:6, ack 6, win 502,
options [nop,nop,TS val 1345427638 ecr 2264350,mptcp dss ack 2208729446 seq 4007607362 subseq 1 len 5 csum 0x9878], length 5
3L >> 4500 004d e05c 4000 4006 7d89 cbfa 2133 cbfa 239d
4L >> 270f 9372 12a8 f321 bea6 9ff1 d018 01f6 603b 0000 0101 080a 5031 98b6 0022 8d1e
MPTCP >> 1e14 2005 83a6 8966 eedf 3c42 0000 0001 0005 9878 6865 6c6c 6f

203.250.33.51.9999 > 203.250.35.157.37746: Flags [.], cksum 0x6033 (correct), seq 6, ack 6, win 502,
options [nop,nop,TS val 1345427638 ecr 2264350,mptcp dss fin ack 2208729446 seq 4007607367 subseq 0 len 1 csum 0xdc4a], length 0
3L >> 4500 0048 e05d 4000 4006 7d8d cbfa 2133 cbfa 239d
4L >> 270f 9372 12a8 f326 bea6 9ff1 d010 01f6 6033 0000 0101 080a 5031 98b6 0022 8d1e 1e14 2015 83a6 8966 eedf 3c47 0000 0000 0001 dc4a

[DSS] -------------------------------------------------------------------------------------------------------------------------------------------------
```

- MP_JOIN
  - 새로운 Subflow 연결 시 발생되는 물리적 패킷은 다음과 같다.

```
[MP_JOIN] ------------------------------------------------------------------------------------------------------------------------------------------------------

203.250.33.175.52141 > 203.250.33.51.9999: Flags [S], cksum 0x876f (correct), seq 476878471, win 29200,
options [mss 1460,sackOK,TS val 2264350 ecr 0,nop,wscale 6,mptcp join id 2 token 0xa9a0dd64 nonce 0x7305469], length 0
3L >> 4500 0048 1b99 4000 3f06 4540 cbfa 21af cbfa 2133
4L >> cbad 270f 1c6c 9687 0000 0000 d002 7210 876f 0000 0204 05b4 0402 080a 0022 8d1e 0000 0000 0103 0306
MPTCP >> 1e0c 1002 a9a0 dd64 0730 5469

203.250.33.51.9999 > 203.250.33.175.52141: Flags [R.], cksum 0x2f48 (correct), seq 0, ack 476878472, win 0, length 0
3L >> 4500 0028 0000 4000 4006 5ff9 cbfa 2133 cbfa 21af
4L >> 270f cbad 0000 0000 1c6c 9688 5014 0000 2f48 0000

203.250.35.157.37746 > 203.250.33.51.9999: Flags [.], cksum 0x79e5 (correct), seq 6, ack 6, win 457, [!! 해당 패킷 의문 !!]
options [nop,nop,TS val 2264350 ecr 1345427638,mptcp dss ack 4007607367], length 0
3L >> 4500 003c 12fb 4000 3f06 4bfc cbfa 239d cbfa 2133
4L >> 9372 270f bea6 9ff1 12a8 f326 a010 01c9 79e5 0000 0101 080a 0022 8d1e 5031 98b6
MPTCP >> 1e08 2001 eedf 3c47

203.250.33.175.35332 > 203.250.33.51.9999: Flags [S], cksum 0x9f13 (correct), seq 3731747328, win 29200,
options [mss 1460,sackOK,TS val 2264350 ecr 0,nop,wscale 6,mptcp join id 3 token 0xa9a0dd64 nonce 0x74c5f35c], length 0
3L >> 4500 0048 7f1e 4000 3f06 e1ba cbfa 21af cbfa 2133
4L >> 8a04 270f de6d f200 0000 0000 d002 7210 9f13 0000 0204 05b4 0402 080a 0022 8d1e 0000 0000 0103 0306
MPTCP >> 1e0c 1003 a9a0 dd64 74c5 f35c

203.250.33.51.9999 > 203.250.33.175.35332: Flags [R.], cksum 0x5376 (correct), seq 0, ack 3731747329, win 0, length 0
3L >> 4500 0028 0000 4000 4006 5ff9 cbfa 2133 cbfa 21af
4L >> 270f 8a04 0000 0000 de6d f201 5014 0000 5376 0000

[MP_JOIN] ------------------------------------------------------------------------------------------------------------------------------------------------------
```

- DATA_FIN
  - 모든 데이터를 전송하였을 시 발생되는 물리적 패킷은 다음과 같다.

```
[DATA_FIN] -----------------------------------------------------------------------------------------------------------------------------------------------------

203.250.35.157.37746 > 203.250.33.51.9999: Flags [.], cksum 0x5a40 (correct), seq 6, ack 6, win 457,
options [nop,nop,TS val 2264350 ecr 1345427638,mptcp dss fin ack 4007607368 seq 2208729446 subseq 0 len 1 csum 0xe269], length 0
3L >> 4500 0048 12fc 4000 3f06 4bef cbfa 239d cbfa 2133
4L >> 9372 270f bea6 9ff1 12a8 f326 d010 01c9 5a40 0000 0101 080a 0022 8d1e 5031 98b6
MPTCP >> 1e14 2015 eedf 3c48 83a6 8966 0000 0000 0001 e269

203.250.33.51.9999 > 203.250.35.157.37746: Flags [F.], cksum 0x97ca (correct), seq 6, ack 6, win 502,
options [nop,nop,TS val 1345427645 ecr 2264350,mptcp dss ack 2208729446], length 0
3L >> 4500 003c e05e 4000 4006 7d98 cbfa 2133 cbfa 239d
4L >> 270f 9372 12a8 f326 bea6 9ff1 a011 01f6 97ca 0000 0101 080a 5031 98bd 0022 8d1e
MPTCP >> 1e08 2001 83a6 8966

203.250.33.51.9999 > 203.250.35.157.37746: Flags [.], cksum 0x97c9 (correct), seq 7, ack 6, win 502, [!! 해당 패킷 의문 !!]
options [nop,nop,TS val 1345427645 ecr 2264350,mptcp dss ack 2208729447], length 0
3L >> 4500 003c e05f 4000 4006 7d97 cbfa 2133 cbfa 239d
4L >> 270f 9372 12a8 f327 bea6 9ff1 a010 01f6 97c9 0000 0101 080a 5031 98bd 0022 8d1e
MPTCP >> 1e08 2001 83a6 8967

203.250.35.157.37746 > 203.250.33.51.9999: Flags [F.], cksum 0x79db (correct), seq 6, ack 7, win 457,
options [nop,nop,TS val 2264350 ecr 1345427645,mptcp dss ack 4007607368], length 0
3L >> 4500 003c 12fd 4000 3f06 4bfa cbfa 239d cbfa 2133
4L >> 9372 270f bea6 9ff1 12a8 f327 a011 01c9 79db 0000 0101 080a 0022 8d1e 5031 98bd
MPTCP >> 1e08 2001 eedf 3c48

203.250.33.51.9999 > 203.250.35.157.37746: Flags [.], cksum 0x97c8 (correct), seq 7, ack 7, win 502,
options [nop,nop,TS val 1345427645 ecr 2264350,mptcp dss ack 2208729447], length 0
3L >> 4500 003c e060 4000 4006 7d96 cbfa 2133 cbfa 239d
4L >> 270f 9372 12a8 f327 bea6 9ff2 a010 01f6 97c8 0000 0101 080a 5031 98bd 0022 8d1e
MPTCP >> 1e08 2001 83a6 8967

[DATA_FIN] -----------------------------------------------------------------------------------------------------------------------------------------------------
```

참 고 문 헌

- A. Ford, C. Raiciu, M. Handley, and O. Bonaventure, “TCP Extensions for Multipath Operation with Multiple Addresses,” IETF RFC 6824, April 2013.