나의 컴퓨터는 IP의 checksum은 0이고, TCP의 checksum는 계산해보면 이상하게도 pseudo header 와 tcp header의 값들을 동시에 계산해서 합쳐준 것과 같은게 아니라 pseudo header의 값들만을 더해준 것과 같게 나왔다. IP의 checksum과 TCP의 checksum을 동시에 바꿔줬더니 서버가 응답을 하긴 하지만 한번만 하지 않고 총 6번을 응답하게 됐다.

(이거는 오피시스템 노트필기에 존재)

1) Send a SYN packet to the server.

1-1)

- Modify your sniffer such that it can send a SYN packet instead of sniffing.

-- break out of the while loop after capturing the first SYN packet.

-- display the packet in raw bytes.

-- kill the server and the client (manually)

-- run the original sniffer

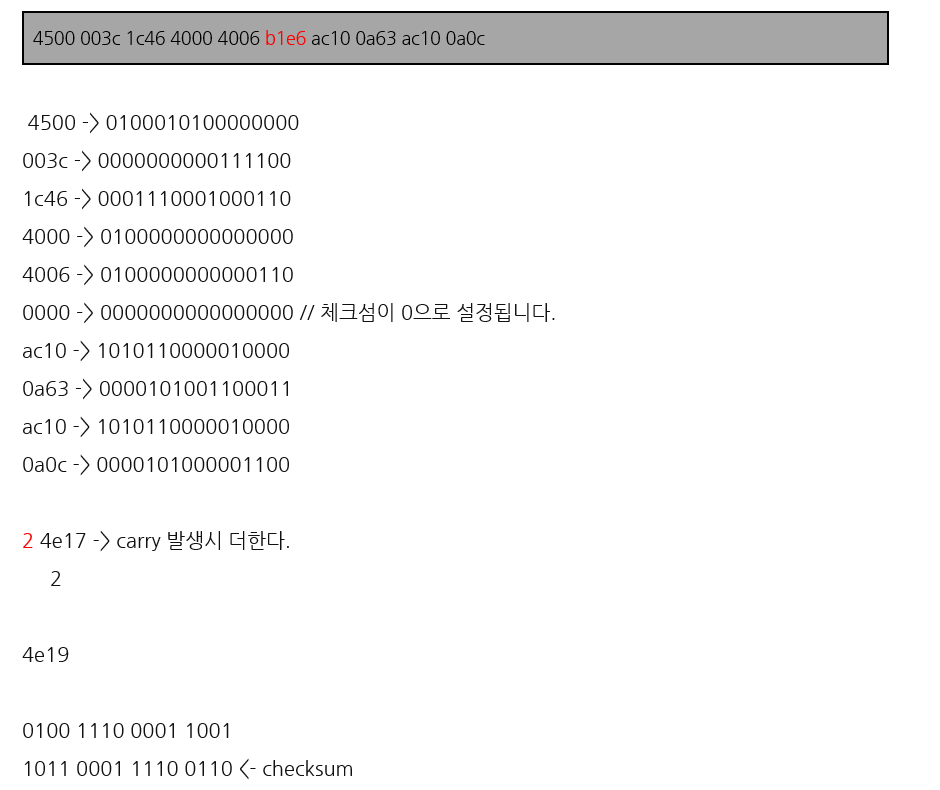
-- rerun the server

-- send the captured SYN packet to the server

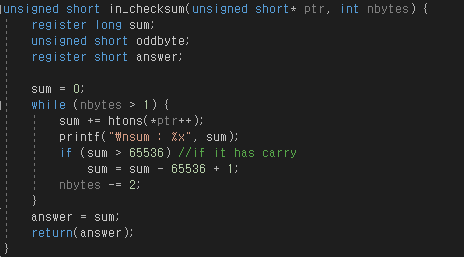
-- check if you can see this packet in the original sniffer

-- check if you can see the ACK packet from the server.

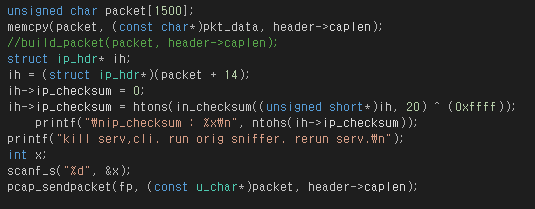
\*If the captured SYN packet shows checksum=0, do not send as it is: you have to compute the checksum by yourself (checksum offload case).



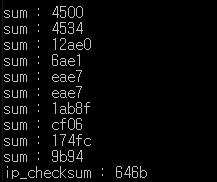
출처  
<https://ryufree.tistory.com/78>



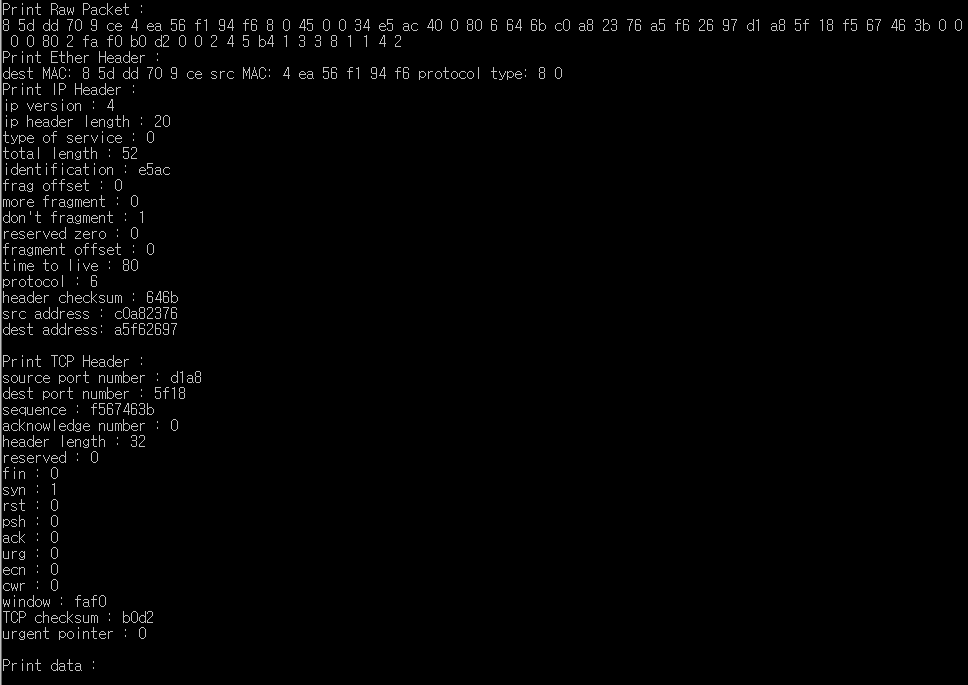
인터넷을 참고해서 in\_checksum 함수를 다시 작성했다. 65536은 carry의 값과 같아서 만약 캐리가 존재할 경우 그 캐리를 제거하고 계산된 값에 1을 더해주도록 했다. 값들이 정상적으로 더해지는지 확인하기 위해 중간에 printf를 넣어주었다.



마지막으로 계산된 값의 보수를 취해주기 위해 ^(0xffff)를 사용했다.



인터넷에 나온 방법으로 계산기로 처음부터 다 계산해봐도 9b94가 나오고, 9b94의 보수를 취해주면 정확히 646b가 나온다.



출력을 해봤을 때애도 IP의 checksum이 0에서 646b로 정확히 바꼈지만 서버는 끝내 응답하지 않았다.

1-2)

- Modify the sniffer further such that it re-computes ip and tcp checksum.

-- break out of the while loop after capturing the first SYN packet.

-- copy them into another buffer: pkt\_data=>packet

-- set ip\_check\_sum and tcp\_check\_sum to zero

-- recompute ip\_check\_sum

-- recompute tcp\_check\_sum

-- display the packet in raw bytes. this should be same as pkt\_data

-- kill the server and the client (manually)

-- run the original sniffer

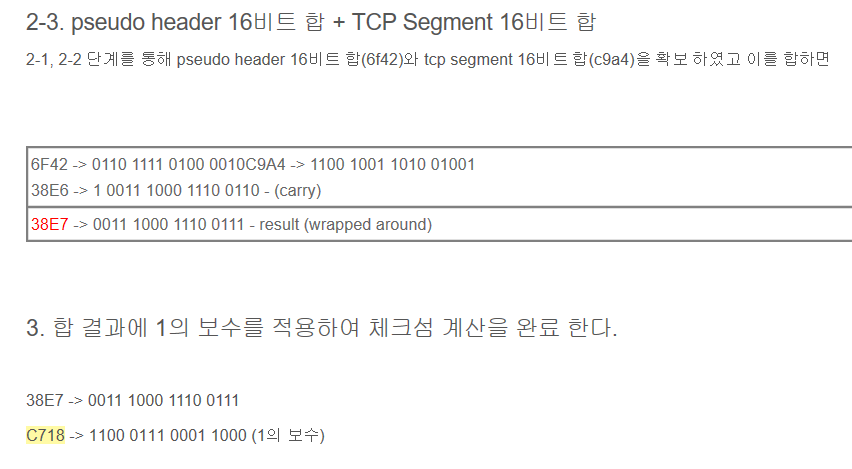
-- rerun the server

-- send packet to the server

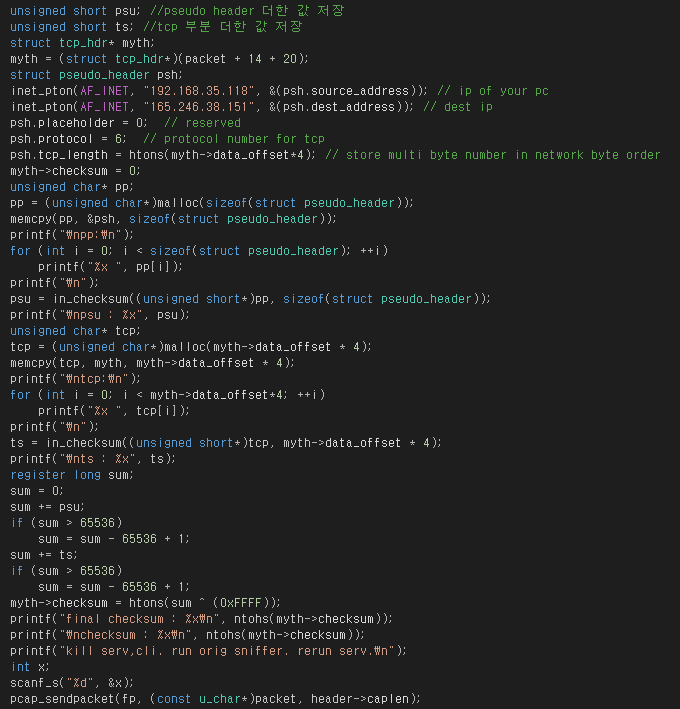
-- check if you can see this packet in the original sniffer

-- check if you can see the ACK packet from the server.



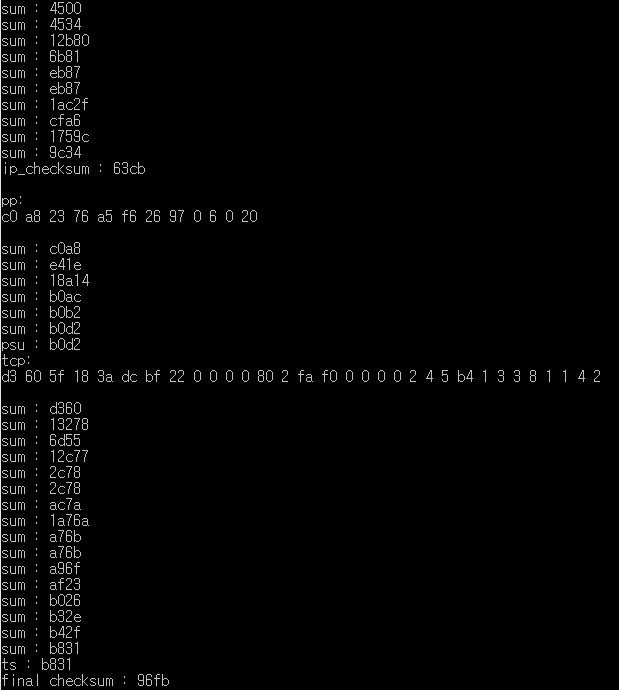


출처 : <https://securitynewsteam.tistory.com/entry/TCP%EC%B2%B4%ED%81%AC%EC%84%AC-%EA%B3%84%EC%82%B0%EB%B0%A9%EB%B2%95>

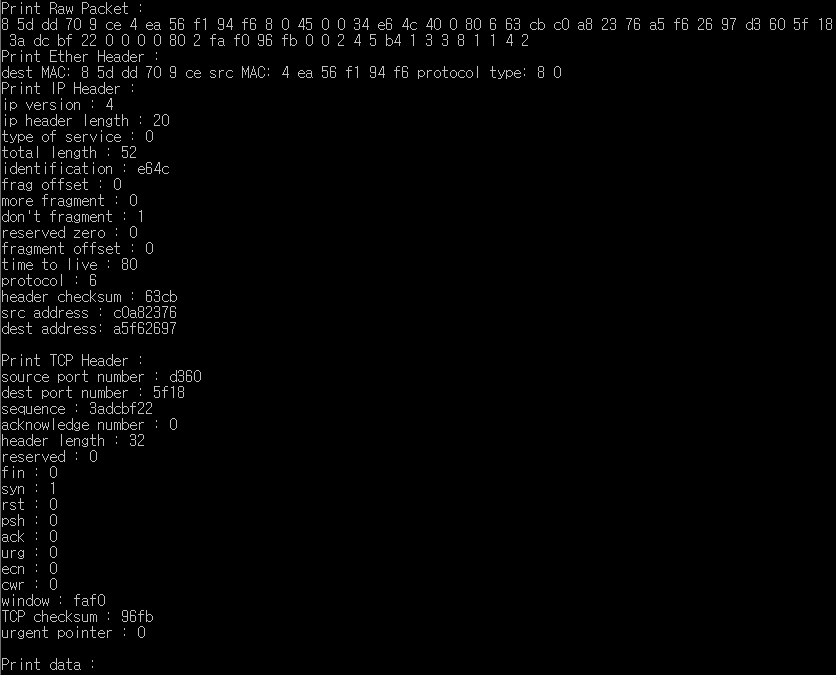


이번엔 tcp의 checksum을 다시 계산해주기 위해서 pseudo header인 psh를 만들어주고, source address에는 내 컴퓨터의 ip주소를 입력해주고 dest\_address에는 서버의 주소를 입력해주고 placeholder와 protocol에는 각각 0과 6을, tcp의 길이에는 data\_offset\*4를 입력해주었다. 인터넷을 참고하면 pseudo header의 값들과 tcp header의 값들을 따로 더해서 합친 후 1의 보수를 취해주면 원하는 checksum이 나온단걸 알 수 있다. 그래서 변수 psu와 ts를 만든 뒤 각각에 pseudo header의 값들을 더한 결과와 tcp header의 값들을 더한 결과를 넣어준 뒤 psu와 ts를 다시 더해서 1의 보수를 취해준 값을 최종 checksum값으로 정하였다.

(체크섬은 반대로 해서 집어넣어줘야 제대로 들어가게됨.)



여기서도 역시 인터넷에 나온 방법을 참고하여 raw packet을 보면서 전부다 계산기로 계산해봐도 위 사진에 나온 값들과 똑같이 나왔지만, 원래 패킷의 TCP checksum의 값은 final checksum 값인96fb가 아니라 pseudo header만을 계산한 값인 b0d2였다.





그 결과, SYN 패킷에는 IP checksum과 TCP checksum이 다 정상적으로 recompute 됐고, 서버는 ACK 패킷을 보내왔지만 한번이 아닌 6번을 보내왔다.