사전조사 보고서

2014152019 심지섭

**WireShark 란?**

와이어샤크(Wireshark)는 자유 및 오픈 소스 패킷 분석 프로그램이다. 네트워크의 문제, 분석, 소프트웨어 및 통신 프로토콜 개발, 교육에 쓰인다. 원래 이름은 Ethereal이었으나 2006년 5월에 상표 문제로 말미암아 와이어샤크로 이름을 바꾸었다.

와이어샤크는 크로스 플랫폼으로, Qt 위젯 툴킷을 이용하여 사용자 인터페이스를 제공하며, pcap을 이용하여 패킷을 포획한다. 리눅스, 맥 OS X, BSD, 솔라리스를 포함한 다양한 유닉스 계열 운영 체제와 마이크로소프트 윈도에서 동작한다. GUI가 없는 터미널 기반 버전인 티샤크(TShark)도 제공한다. 와이어샤크, 또 TShark와 같은 프로그램과 함께 배포되는 여러 프로그램들은 자유 소프트웨어로, GNU 일반 공중 사용 허가서의 조건으로 공개된다

**WireShark에서 사용한다는 pcap이란?**

pcap(packet capture, 패킷 캡처)은 컴퓨터 네트워크 관리 분야에서 네트워크 트래픽 포착용 API를 구성하고 있다. 유닉스 계열 운영 체제들은 libpcap 라이브러리에 pcap을 포함하고 있다. 윈도는 WinPcap이라는 libpcap 포팅을 이용한다.

감시 소프트웨어는 libpcap이나 WinPcap을 이용하여 네트워크 상에 떠돌아다니는 패킷을 포착할 수 있으며 새로운 버전에서는 링크 레이어 상의 네트워크 패킷을 전송할 수 있을뿐 아니라 libpcap이나 WinPcap과 함께 이용할 수 있는 네트워크 인터페이스 목록을 가져올 수 있다.

pcap API는 C로 작성되었으며 자바, 닷넷 언어, 스크립트 언어와 같은 다른 언어들은 일반적으로 래퍼를 이용한다. 이러한 래퍼들은 libpcap이나 WinPcap 자체적으로 제공되지는 않는다. C++ 프로그램들은 C API에 직접 연결할 수도 있다. 오직 하나의 부분적인 객체 지향 C++ 래퍼는 현재 외부 소스로부터 이용할 수 있다.

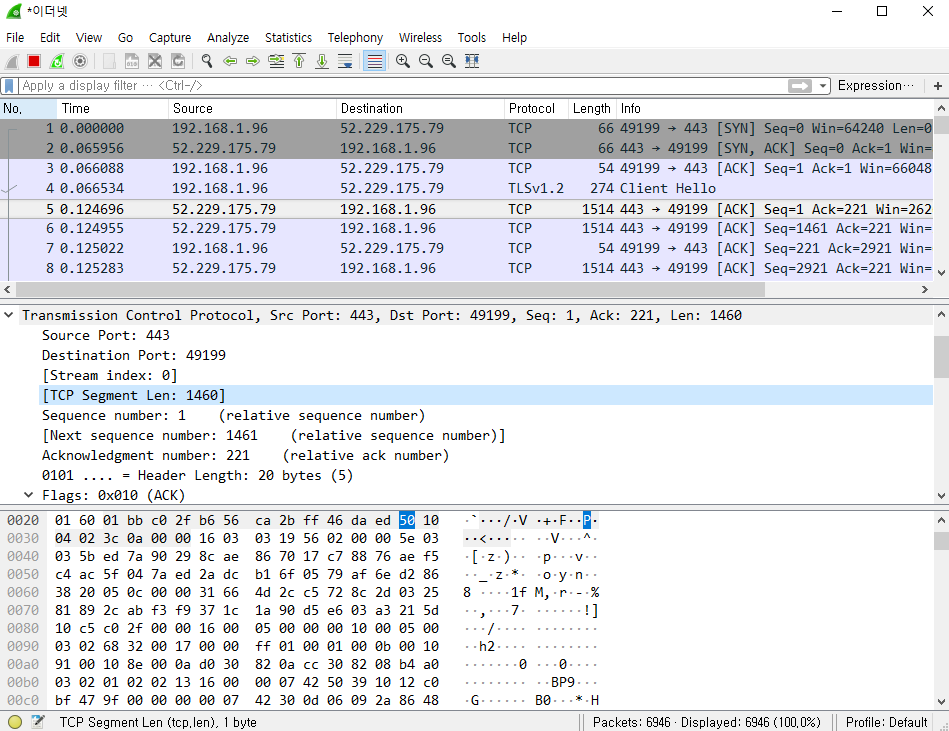
**WireShark 사용법**

상단의 패킷캡처의 분류기준을 보면 No, Time, Source, Destination, Protocol, Length, Info 가 있는것을 확인할 수 있다.

No = 패킷을 수집한 순서, Time = 패킷이 수집된 시간, Source = 패킷을 보낸 주소

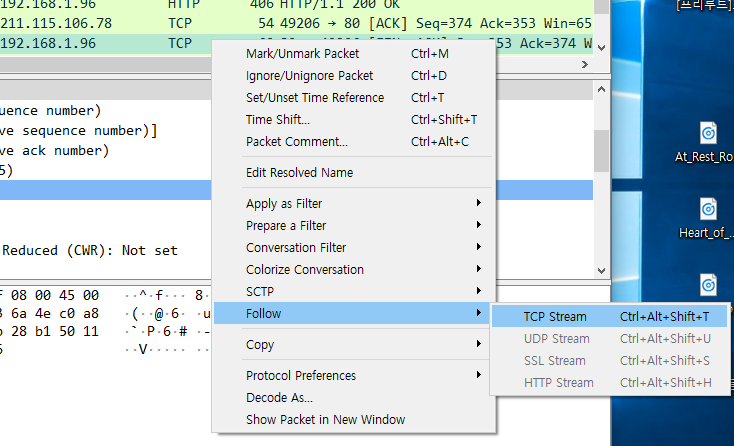
Destination = 패킷 도착주소, Protocol = 프로토콜 정보, Length = 패킷의 길이

Info = 패킷정보 이다.



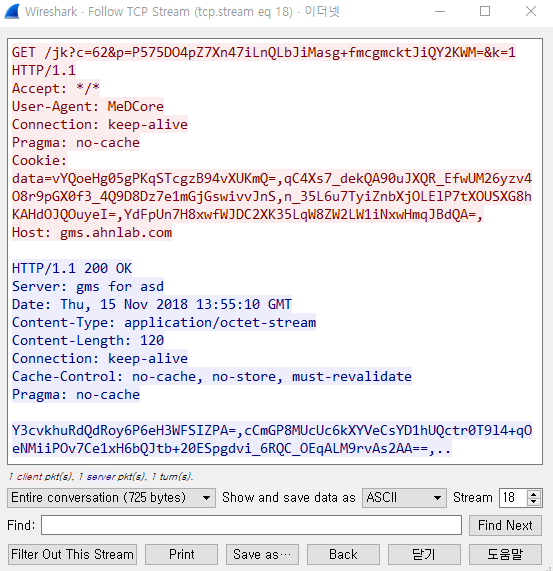
위 사진은 한 TCP관련 라인을 클릭한 상태이다.

SourcePort, DestinationPort, Sequence number, ack number 그리고 CheckSum 까지 볼 수 있다.



세션별로 조립한 것을 Stream이라고 하는데 위의 사진은 TCP로 통신되고 있는 패킷들을

조립해서 내용을 확인하는 메뉴이다.



18번째 스트림을 본 화면이다. HTTP프로토콜의 헤더가 출력된 것을 볼 수 있다.

GET명령어, HTTP버전, Connection을 Keep-alive 하는 것 까지 볼 수 있다.

Wire Shark를 이용하면 설계과제의 요구사항인 다양한 프로토콜의 헤더를 캡처링 할 수

있을것으로 보인다.

**Raw socket이란?**

RAW소켓은 헤더 정보들에 대해 프로그래머가 직접 제어할 수 있게 해준다.

즉, RAW소켓을 사용하면 IP헤더와 TCP헤더를 직접 제어할 수 있는데, 이를 모두 사용자 데이터로 취급하므로 네트워크 계층에 어떤 종류의 헤더가 와도 상관이 없게된다. RAW소켓을 사용하게 되면 IP주소나 포트주소가 다르더라도 물리계층인 인터넷 디바이스 드라이버에서 오는 모든 패킷을 전부 받아 확인이 가능하다. RAW소켓을 사용하는 대표적인 예로는 패킷들을 감지하는 WireShark(와이어샤크)가 있다. RAW소켓을 사용하지 않는 일반적인 애플리케이션들은 IP주소가 다르거나 자신의 포트 주소가 아닌 패킷은 받아볼 수 없다.

왜냐하면 운영체제의 프로토콜 스택에서 그러한 패킷은 이미 걸러져 버리고 해당 애플리케이션이 사용하는 ip 주소와 포트 주소에게 가는 패킷만 전달되기 때문이다.

그러나 RAW소켓을 사용하면 IP주소나 포트주소가 다르더라고 물리계층인 인터넷 디바이스 드라이버에서 오는 모든 패킷을 전부 받아볼 수 있다.

**RAW소켓을 사용하는 프로그램**

1. ping 프로그램(ICMP 프로토콜 또는 IGMP 프로토콜)

RAW소켓의 대표적인 프로그램이 바로 ping프로그램이다

ping프로그램은 특정 호스트에 패킷이 정상적으로 도달하는지 확인하기 위한 프로그램이다

ping프로그램은 내부적으로 ICMP 패킷을 보내는데 ICMP패킷을 받은 운영체제는 다시 ICMP패킷을 보내 자신이 살아 있다는 것을 확인 시켜준다

1. 패킷 캡쳐 프로그램

자신의 호스트로 오는 모든 패킷을 캡처해서 화면에 출력하는 역할을 한다

이런 패킷 캡쳐 프로그램도 역시 일반적인 소켓으로는 절대로 작성할 수 없다

그 이유는

첫째, 패킷의 헤더 구조를 알 수 없다

둘째, 애플리케이션에 전달되는 패킷은 자신의 IP주소와 포트 주소로 오는 패킷으로 한정되기 때문이다

이렇게 시스템에 오는 모든 패킷을 모니터링 하고자 한다면 RAW 소켓을 이용해야만 한다

대표적인 패킷 캡처 프로그램은

1. 위에서 설명한 WireShark
2. Fiddler(HTTP를 분석할 수 있다. 주요기능은 http 모니터링 및 웹사이트 성능 보안시험 이다.)
3. Angry ip Scanner(한번에 대규모의 라이브 호스트를 찾고 확인하는데 유용하다.)
4. Network Miner(네트워크 패킷 포렌식으로 많이 사용한다. 프로세스별로 통신하는 것을 쉽게 볼 수 있다.)

Raw socket을 이용한 프로그래밍은 간단하게 구현할 수 있다.

socket(PF\_INET, SOCK\_RAW, 145); 바로 소켓을 할당할 때 SOCK\_RAW 옵션을 주면 되는 것.

이후 sendto() 와 recv()함수로 통신하고, 받은 패킷을 (struct iphdr\*) 구조체에 넣으면 쉽게

패킷의 데이터를 분류할 수 있다.