네트워크 모니터링 툴의 설계 및 구현

윤종철[0], 곽인섭, 강흥식
인제대학교 정보컴퓨터공학부
pcman95@hanmail.net kinsub98@cs.inje.ac.kr hskang@nice.inje.ac.kr

Design & Implementation of Network Monitering Tool

Jong-Chul Yun[0], In-Seub Gwag, Heung-Seek Kang
Dept of Information and Computer Engineering Inje University

요 약

인터넷과 네트워킹 기술의 비약적인 발전으로 인해 수많은 프로토콜들과 관련 기술, 그리고 서비스들이 새롭게 등장하였다. 하지만 설계상에서 보안에 대해 고려되지 않았던 많은 기술들은 이제 새로운 보안 위협을 발생시키는 등의 문제점을 드러내고 있다. 네트워크를 통한 크래킹 역시 이러한 문제점으로 지적되고 있는데, 이러한 위협으로부터 시스템을 보호하기 위해 방화벽, 침입탐지 시스템과 같은 정보보호 시스템들이 연구, 개발되었다. 본 논문에서 제안하는 네트워크 모니터링 도구는 스니핑이라는 해킹 기법으로 이용되기도 하는 다소 위험한 기술을 이용하여 네트워크상의 패킷을 실시간으로 수집, 분석함으로써 네트워크 관련 오류의 점검, 크래킹의 실시간 감시등에 이용할 수 있도록 해준다.

1. 서 론

인터넷의 확산과 분산 컴퓨팅 환경의 발전과 성장에 힘입어 원격 접속 컴퓨팅의 시대가 활짝 열렸다. 이로 인해 다양한 인터넷 프로토콜과 이를 기반으로 한 인터넷 서비스들이 적용되는 응용범위가 확장되고 있다. 하지만 프로토콜과 서비스들의 취약점을 이용한 외부에서의 불법침입과 중요 정보 유출 및 변경 등 역기능들이 날로 증가되고 있으며 그 피해 규모 또한 심각한 수준에 이르고 있다. 이러한 역기능들을 막기 위해 많은 연구가 진행되고 있으며 정보보호 시스템 또한 수없이 개발되고 있다. 그중에서도 가장 기본이 될 수 있는 네트워크 모니터링 도구는 네트워크 상의 떠도는 여러 패킷 중에서 관리자가 원하는 여러 패킷 중에서 관리자가 원하는 패킷만을 수집하여, 정보를 제공함으로써 네트워크 관련 오류 점검, 크래킹의 실시간 감시등의 이상 징후들을 미리 발견할 수 있도록 해준다.

본 논문에서 제시하는 네트워크 모니터링 도구 역시 기존에 나와 있는 분석도구와 크게 다르지 않으나, 국내에서 사용되는 네트워크 모니터링 도구의 대부분이 외국산이라는 점과 모니터링 도구의 실행 원리와 제작 기술을 터득함으로써 보다 향상된 정보 보호 시스템 개발의 초석으로 삼기위해 본 연구를 진행하였다.

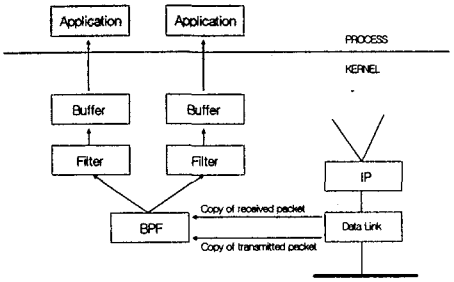
2. 관련연구

네트워크 모니터링 도구는 네트웍 상의 트래픽을 수집하기 위해서 스니핑이(sniffing)라는 기술을 사용하게 된다. 물리적인 네트워크 트래픽을 사용자 수준에서 접근하기 위해서는 모니터링을 수행하고자 하는 호스트의 디바이스 드라이버와 상호 동작할 수 있는 기술이 필요한데 대부분의 운영체제에서는 사용자 수준에서 패킷수집 기능을 두어서 네트워크 감시를 할 수 있도록 하였다.

2.1 패킷수집기술

먼저 패킷 수집기술에 대해서 이해를 하기 위해서는 네트워크에 관한 기본 개념이 있어야 한다. 네트워크 상의 각각의 노드들은 OSI 7계층을 가지는데 통신의 당사자에 해당하는 노드들은 패킷이 7계층을 거치는데 반해 중간 노드들은 패킷을 가져와서 자기에게 온 것이 아니라고 판단되면 패킷을 버리게 되는데 네트워크 모니터링 도구에서는 버리지 않고 가져오게 함으로써 자기가 목적지가 아닌 패킷들을 볼 수 있도록 제공한다.

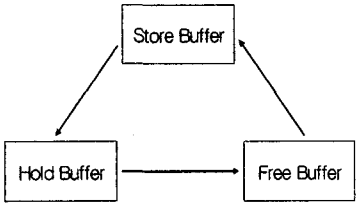
이러한 기술을 패킷 필터라고 하는데 네트워크 탭(tap)을 promiscuous 모드로 설정함으로써 가능하다. 현재 대부분의 유닉스 계열의 운영체제에서 나름대로의 패킷 필터를 제공하고 있다. 1980년 Carnegie-Mellon 대학에서 개발한 패킷 필터가 최초이며 현재의 패킷필터들은 이 패킷 필터에 기반을 두고 있다. 그 중 BPF는 현재까지 알려진 가장 강력한 패킷 필터로서 buffered read, promiscuous 네트워크 접근 및 waiting test를 제공한다. BPF는 non-shared buffer모델과 레지스터 구조라는 특징으로 인해 같은 하드웨어에서 다른 패킷 수집기술들 보다 훨씬 우수한 성능을 보인다. BPF는 수집된 패킷을 처리하기 위하여 3가지 버퍼를 관리하고 있는데 그 기능은 다음과 같다



[그림 1] BPF의 구조

데이터링크 드라이버는 데이터링크 계층을 통해서 들어오고 나가는 패킷들이 생길때 마다 BPF를 호출한다. [그림 1]에서 보듯이 BPF는 필터링 기술을 이용하여 필요한 패킷만을 버퍼로 보내게 된다.

●Store 버퍼 : 네트워크 인터페이스로부터 수신되는 패킷을 저장하는 역할
●Hold 버퍼 : 프로세스에 의해 읽혀질 패킷을 저장한다.
●Free 버퍼 : 비어있는 버퍼를 나타낸다.



[그림 2] 회전 버퍼들

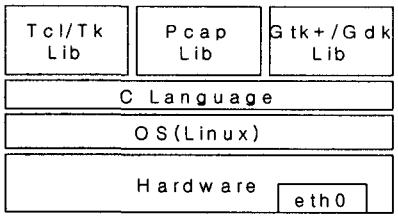
일단 패킷이 필터링을 거쳐 버퍼 쪽으로 들어오게 되면 Store Buffer가 가득 찬 상태라면 패킷을 Free Buffer에 저장한 후 나중에 Store Buffer에 복사하게 된다. 그렇지 않으면 바로 Store Buffer에 저장하게 된다. 이렇게 Store Buffer에 저장된 패킷은 한꺼번에 Hold Buffer로 옮겨져 프로세스에 의해 읽히게 되고 Store Buffer는 빈 상태가 된다.

3. 설계 및 구현

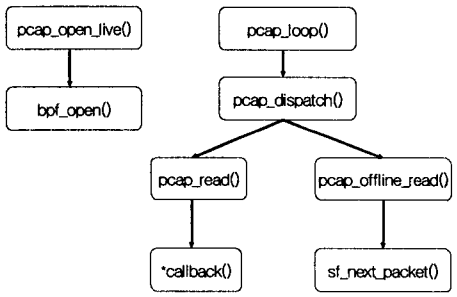
본 논문에서 제안하는 네트워크 모니터링 도구에서는 패킷 수집을 위해 BPF(BSD Packet Filter)를 이용한 Libpcap 라이브러리를 사용하고 있다. 이 PCAP 라이브러리는 사용자 수준에서 시스템에 상관없이 패킷 수집을 용이하도록 도와주는 라이브러리이다. Tcpdump를 만든 Lawrence Berkeley 연구소에서 제작되었으며 네트워크 통계치 수집, 보안 모니터링 및 네트워크 디버깅 등의 다양한 응용 프로그램에 효과적으로 이용될 수 있으며 가장 큰 특징은 시스템과 운영체제에 따라서 패킷 수집을 가능케 하는 각기 다른 인터페이스를 제공하므로 환경에 크게 구애받지 않는다는 점이다.

3.1 설계

네트워크 모니터링 툴의 구조는 [그림 3,4]와 같으며 Tcl/Tk는 결과물을 GUI환경으로 꾸미기위한 스크립트 언어이며, Gtk+/Gdk라이브러니는 전체적인 인터페이스를 구성하는데 사용되는 라이브러리이다.



[그림 3] 네트워크 모니터링 도구의 구조



[그림 4] pcap 라이브러리를 이용한 패킷 캡처

(가) pcap_open_live()는 일정한 크기의 메모리를 할당하고 bpf_open(p, ebuf)을 호출하여 가상 디바이스를 open한다. 그런 다음 ioctl함수를 이용하여 데이터링크 계층의 형태를 알아내고 timeout을 설정한다. 그런 후, NIC카드의 모드를 promiscouse모드로 전환하고 네트워크 디바이스의 버퍼크기를 알아낸다.

(나) pcap_loop()는 pcap_dispatch()를 호출하여 cnt라는 변수만큼 읽을 때까지 pcap_dispatch()를 계속 수행한다. pcap_dispatch()는 저장된 파일이 있으면 pcap_offline_read()를 호출하고 그렇지 않으면 pcap_read()를 호출한다. pcap_offline_read()는 sf_next_packet()의 리턴 값이 0이 아닐때 까지 계속 sf_next_packet()을 호출하고 나중에 bpf_filter()을 호출한다. sf_next_packet()는 sf_reaf파일에서 데이터를 읽고 hdr에 header를 buf에 content를 저장하여 리턴한다. pcap_read()함수는 p->fd에서 p->bufsize까지 읽어서 p->buffer에 저장하고 저장된 패킷에 callback 함수를 적용시킨다.

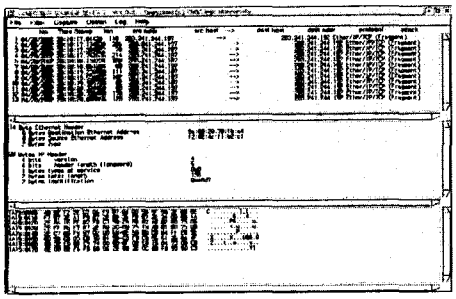
3.2 구현

아래의 소스코드는 pcap 라이브러리 일부 함수를 사용

하여 네트워크 상의 패킷 데이터를 수집하기 위해 응용 프로그램에서의 패킷 수집 모듈 세팅 방법을 의사코드(pseudo code)로 나타낸것이다.

```
static pcap_t  *pd;     /*라이브러리에서 사용되는 구조*/
char           *device_name;
char           filter_rule[]= "some filter rules ...";
int            needed_length = 100;
u_char         *packet_data;

/*가상 패킷 디바이스 이름을 알아낸다.*/
device_name = find_pseudo_packet_device_name();

/*패킷 디바이스를 연다. 패킷을 열 때 로컬 네트워크
내의 모든 패킷을 읽어 들이기 위해서 promiscuous 모드
로 디바이스를 설정한다.*/
pd = open_pseudo_device_for_monitoring(device_name);

/*필요한 정보만을 패킷에서 읽어 들이기 위해 해당 길이
를 설정한다.*/
set_needed_length_ofpacket(pd, needed_length);

/*필요한 패킷만을 읽어 들이기 위해 필터를 설정한다.*/
set_filter_to_pseudo_device(pd, filter_rule);

/*패킷을 읽어 들여 원하는 작업을 한다.*/
while(1) {
    packet_data = get_packet(pd);
    /*do something*/
}
```

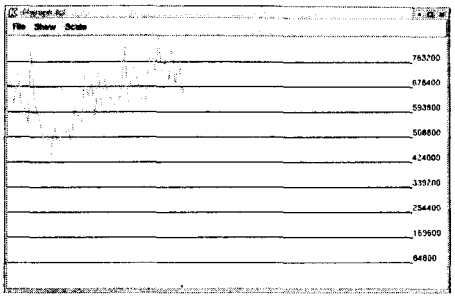
3.3 결과화면



[그림 5] 네트워크 패킷 탐지 도구 결과 화면1

4. 결론 및 향후 연구 방향

본 논문에서는 이더넷 카드의 promiscuous모드와 libpcap 라이브러리를 이용하여 패킷을 수집하는 기술들을 간단히 소개 하였고 네트워크 상의 패킷을 수집할 수 있는 툴을 설계 및 구현하였다. 그렇게 함으로써 네트워크 상에서 어떤 정보들이 오고 가는지에 대해서 조금이



[그림 6] 네트워크 패킷 탐지 도구 결과 화면2

나마 이해하고, 네트워크 오류와 크래킹에 대한 조기 탐지가 가능하도록 하였다. 그러나 아직은 네트워크 모니터링 도구가 구현 초기단계에 있으므로 개선의 여지가 많다고 하겠다. 설계상에서는 언급하지 않았지만 무분별하게 수집된 패킷들 중에서 꼭 필요한 데이터만 가려낼 수 있는 축약 기술에 관한 연구도 필요한 실정이다. 나아가 이러한 툴에서 특정 패킷의 공격 시그니처를 읽어내고 이를 탐지 해내는 기술이 더해진다면 침입탐지 시스템으로서의 활용 가능성 또한 높다고 하겠다.

참 고 문 헌

[1] 리눅스 통합 보안 시스템, 2000.02, 비트프로젝트50호
[2] Security Plus For UNIX, The Pohang University of Science and Technology, February. 2001
[3] UNIX network programming Networking APIs: Socket and XTI, W.RICHARD STEVENS 1998
[4] "Effective TCP/IP Programming"/44 Tips to Improve your Network Programs, Jon C.Sander
[5] 네트워크 기반 프로토콜 공격에 대한 침입탐지 시스템의 구성방안, 2001년 한국정보처리학회 추계 학술발표논문집 제8권 2호