1.6 공격받는 네트워크

인터넷의 모든 유용성과 역동성 뒤에,
'나쁜 친구들'이 인터넷에 연결된 컴퓨터에 해를 끼리고, 사생활을 침해하고, 우리가 의존하는 인터넷 서비스를 동작하지 못하게 함으로써
일상생활을 망가트리려고 하는 어두운 면이 있다.

네트워크 보안 분야는 나쁜 친구들이 어떻게 컴퓨터 네트워크를 공격할 수 있는가와
그러한 공격으로부터 네트워크를 방어할 수 있는가, 혹은 아예 그러한 공격에 영향을 받지 않는 새로운 구조의 설계 등을 다루는 분야다.

나쁜 친구들은 인터넷을 통해 여러분의 호스트에 멀웨어(악성코드)를 침투시킬 수 있다

우리는 인터넷에서 데이터를 수신/송신하기를 원하기 때문에 장치를 인터넷에 연결한다.

불행하게도 우리에게 전달되는 데이터들 중 해로운 것들도 포함되는데, 이들을 멀웨어(malware)라고 한다.
멀웨어는 우리들의 장치에 들어가서 나쁜 영향을 미친다.

e.g., 파일 삭제, 주민번호, 비밀번호, 키스트로크(keystroke, 키보드를 누르는 것) 등의 사적인 정보를 모으는 스파이웨어를 설치
→ 이러한 정보를 모아 나쁜 친구들에게 인터넷을 통해 다시 보낸다.

면역되지 않은 호스트는 수천의 비슷한 면역되지 않은 장치들로 구성된 네트워크, 즉 봇넷(botnet)에 등록될 수 있다.
나쁜 친구들은 목표로 하는 호스트에 대해 스팸 전자메일 분해 혹은 분산 DoS(Denial of Service) 공격을 위해 이 봇넷을 제어하고 이용한다.

오늘날 널리 퍼져 있는 많은 멀웨어는 자기복제(self-replicating)를 한다.
자기복제 멀웨어는 아래의 방법을 통해 기하급수적으로 퍼질 수 있다.

1. 한 호스트에 영향을 미치면, **그 호스트에서 인터넷을 통해 다른 호스트로의 엔트리를 찾는다.**
2. 새롭게 영향을 받은 호스트로부터 또 다른 많은 호스트로의 엔트리를 찾는다.

나쁜 친구들은 서버와 네트워크 인프라스트럭처를 공격할 수 있다

DoS(Denial-of-Service) 공격

> 네트워크, 호스트 혹은 다른 인프라스트럭처의요소들을 **정상적인 사용자가 사용할 수 없게 하는 것**

웹 서버, 전자메일 서버, DNS 서버, 기관 네트워크는 DoS 공격을 받을 가능성이 있다.

DoS 공격의 세 가지 범주

취약성 공격(vulnerability attack)

만약 올바른 순서의 패킷을 공격받기 쉬운 애플리케이션 혹은 운영체제에 보내면(교묘한 메시지를 보내는 것을 포함)
그 서비스는 중단되거나, 호스트가 동작을 멈출 수도 있다.

대역폭 플러딩(bandwidth flooding)

목표 호스트의 접속 링크가 동작하지 못하도록 **많은 패킷을 보내서** 정당한 패킷들이 그 서버에 도달하지 못하게 한다.

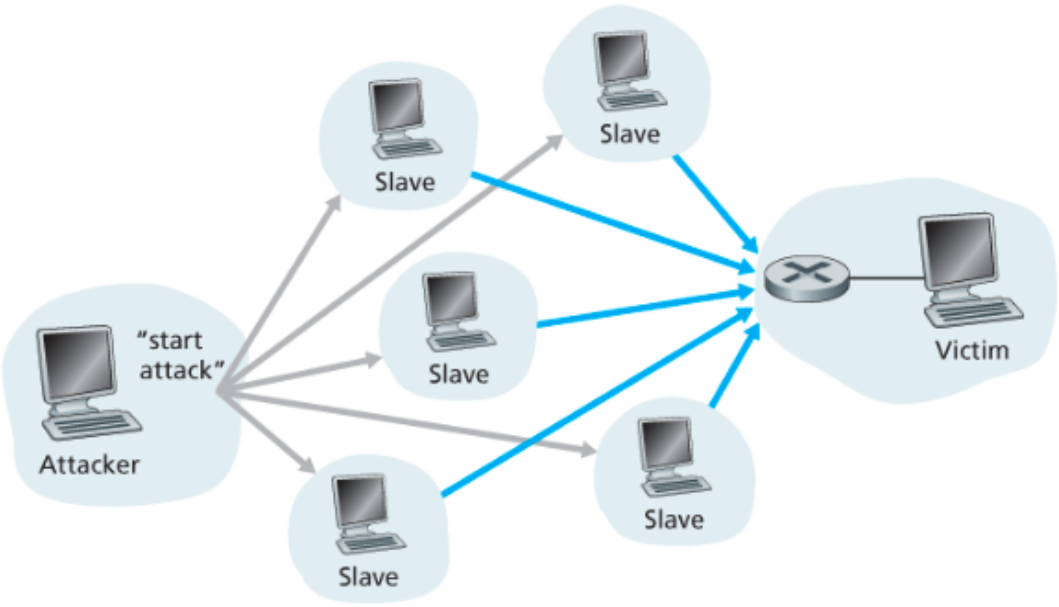
1.4.2절의 지연과 손실 분석 논의를 기억해보자.

만약 서버가 `R bps의 접속 속도`를 갖고 있다면, 공격자는 피해를 주기 위해 대략적으로 `R bps`의 속도로 트래픽을 전송하면 된다.

만약 R가 매우 크다면 단일 공격 소스는 서버에 나쁜 영향을 줄 수 있는 충분한 트래픽을 발생시킬 수 없다.

더 나아가, 모든 트래픽이 하나의 소스에서 방사된다면 업스트림 라우터는 그 공격을 발견할 수 있고
트래픽이 서버에 가까이 가기 전에 그 소스로부터 모든 트래픽을 차단(block)할 수 있다.

아래 그림처럼 `분산 DoS(DDoS) 공격`에서 공격자는 다중의 소스를 제어하고 각 소스는 목표에 트래픽을 보낸다.



이런 방법으로 모든 제어 소스에 걸친 통합 트래픽 속도가 서비스를 무능력하게 하기 위해서는 전송률이 약 R 이어야 한다.

수천개의 호스트로 구성된 봇넷을 이용하는 DDoS 공격은 오늘날 매우 흔하다.

연결 플러딩(connection flooding)

목표 호스트에 **반열림(half-open) 혹은 전열림(fully open)된 TCP 연결**을 설정한다.

호스트는 가짜 연결을 처리하느라 바빠서 정상적인 연결을 받아들이는 것을 중단하게 된다.

> 컴퓨터 네트워크 설계자는 DoS 공격을 방어하기 위해 무엇을 할 수 있는가?

→ 세 가지 유형의 DoS 공격에는 각기 다른 방어가 필요하다.

나쁜 친구들은 패킷을 탐지할 수 있다

유비쿼터스(ubiquitous) 인터넷 접속은 매우 편리하고 이동 사용자를 위한 애플리케이션이 가능하지만, 인터넷 접속은 주요한 보안 취약성을 창출했다.

무선 전송장치의 근처에 수동적인 수신자 즉, `패킷 스니퍼(packet sniffer)`를 위치시킴으로써 그 수신자는 전송되고 있는 **모든 패킷의 사본을 얻을 수 있다.**

스니퍼는 무선 뿐만 아니라, 유선 환경에서도 배치될 수 있다.
(이더넷 LAN, 케이블 접속 기술, 인터넷에 연결되는 기관의 접속 라우터 혹은 접속 링크 등)

나쁜 친구들은 이렇게 가로챈 패킷을 오프라인으로 분석하여 비밀번호, 주민등록번호, 영업 비밀 등 **모든 종류의 민감한 정보를 얻을 수 있다.**

패킷 스니퍼는 수동적이기 때문에, 즉 **스니퍼는 채널에 패킷을 삽입하지 않기 때문에 이를 탐지하기가 어렵다.**

그래서 무선 채널로 패킷을 보낼 때 어떤 나쁜 친구가 우리 패킷의 사본을 기록하고 있을 수 있다는 가능성을 받아들여야 한다.

패킷 스니핑을 방지하기 위한 가장 좋은 방어는 암호화를 포함하는 것이다. *(8장에서 다룸)*

나쁜 친구들은 여러분이 신뢰하는 사람인 것처럼 위장할 수 있다

임의의 출발지 주소, 패킷 내용, 목적지 주소를 갖는 패킷을 생성하고 이 패킷을 인터넷으로 보내는 것은 매우 쉽다.

가짜 출발지 주소를 가진 패킷을 인터넷으로 보내는 능력을 `IP 스푸핑(spoofing)`이라고 하며,
한 사용자가 다른 사용자인 것처럼 행동하는 여러 가지 방법 중 하나다.

이 문제를 해결하기 위해서는 `종단 인증(end-point authentication)`,
즉 **메시지가 실제로 와야 할 곳으로부터 온 것인지 확신할 수 있는 방법이 필요하다.**

네트워크 애플리케이션과 프로토콜의 경우 어떻게 이것을 할 수 있을까? (8장에서 다룸)

인터넷은 처음에 어떻게 보안 문제에 직면하게 되었을까?

> 인터넷은 원래 **'투명한 네트워크에 연결된 상호 신뢰할 수 있는 사용자 그룹'** 모델,
> 즉 **보안이 필요 없는 모델**에 기반을 둔 방식으로 설계되었다. [Blumenthal 2001]

따라서 원래 인터넷 구조의 많은 특성은 이러한 상호 신뢰를 반영하고 있다.
e.g., 한 사용자가 패킷을 다른 사용자에게 보내는 능력은 default, 사용자 식별은 default로 인증해야 하는 것보다는 선언된 액면 그대로 믿는 것이다.

그러나 오늘날 인터넷은 '상호 신뢰하는 사용자'를 분명히 포함하지 않는다.

그럼에도 불구하고 오늘날의 사용자들은 서로를 꼭 신뢰하지는 않을 때, 익명으로 통신하기를 원할 때,
제3자를 통해 간접적으로 통신할 때(이동 지원 에이전트, mobility-assisting agent) 등 이러한 상황에서도 여전

히 통신이 필요하다.

아직 우리 앞에는 많은 보안 관련 해결 과제가 존재한다.

우리는 스니핑, 종단 위장(end-point masquerading), 중간자 공격, DDoS 공격, 멀웨어 등에 방어하는 방법을 찾아야 한다.

> 💡 **상호 신뢰하는 사용자 간의 통신은 일반적인 것이 아니라 예외적인 것임을 명심해야 한다.**