# Lecture 19: Proxy-Server Based Firewalls

## Lecture Notes on "Computer and Network Security"

by Avi Kak (kak@purdue.edu)

March 21, 2017
3:29pm

## Goals:

- The SOCKS protocol for anonymizing proxy servers
- Socksifying application clients
- The Dante SOCKS server
- **Perl and Python scripts for accessing an internet server through a SOCKS proxy**
- Squid for controlling access to web resources (and for web caching)
- The Harvest system for information gathering, indexing, and searching
- How to construct an SSH tunnel through a web proxy

# CONTENTS

## 19.1: FIREWALLS IN GENERAL (AGAIN)

- To expand on what was mentioned at the beginning of Lecture 18, firewalls can be designed to operate at any of the following three layers in the TCP/IP protocol stack:

  – the Transport Layer (example: packet filtering with iptables)

  – the Application Layer (example: HTTP Proxy)

  – the layer between the Application Layer and the Transport Layer (example: SOCKS proxy)

- Firewalls at the Transport Layer examine every packet, check its IP headers and its higher-level protocol headers (in order to figure out, say, whether it is a TCP packet, a UDP packet, an ICMP packet, etc.) to decide whether or not to let the packet through and to determine whether or not to change any of the header fields. (**See Lecture 18 on how to design a packet filtering firewall.**)

- A firewall at the Application Layer examines the requested session for whether they should be allowed or disallowed based on where the session requests are coming from and the purpose of the requested sessions. Such firewalls are built with the help of what are known as **proxy servers**.

- For truly application layer firewalls, you'd need a separate firewall for each different type of service. For example, you'd need separate firewalls for HTTP, FTP, SMTP, etc. Such firewalls are basically access control declarations built into the applications themselves. As a network admin, you enter such declarations in the server config files of the applications.

- A more efficient alternative consists of using a protocol between the application layer and the transport layer – this is sometimes referred to as the **shim layer** – to trap the application-level calls from *intranet* clients for connection to the servers in the internet. [The shim layer corresponds to the Session Layer in the 7-layer OSI model of the TCP/IP protocol stack. See Lecture 16 for the OSI model.]

- Using a shim layer protocol, a proxy server can monitor all session requests that are routed through it in an *application-independent manner* to check the requested sessions for their legitimacy. *In this manner, only the proxy server, serving as a firewall, would require direct connectivity to the internet and the local* intranet *can "hide" behind the proxy server.* The computers in the internet at large would not even know about the existence of your machine in the local intranet behind the firewall.

- When a proxy is used in the manner described above, it may also be referred to as an **anonymizing proxy**.

- Some folks like to use anonymizing proxies for privacy reasons. Let's say you want to visit a web site but you do not wish for that site to know your IP address, you can route your access through a third-party anonymizing proxy.

- There are free publicly available proxy servers that you can use for such purpose. Check them out by entering a string like "public proxy server" in a search engine window. You can also use publicly available scanners to search for publicly available proxy servers within a specific IP range. The website `http://publicproxyservers.com` claims to offer a marketing-pitch-free listing of the public proxy servers.

- In addition to achieving firewall security, a proxy server operating at the application layer or the shim layer can carry out data caching (this is particularly true of HTTP proxy servers) that can significantly enhance the speed at which the clients download information from the servers. If the gateway machine contains a current copy of the resource requested, in general it would be faster for a client to download that copy instead of the version sitting at the remote host.