Journal of Information Security, 2011, 2, 151-157
doi:10.4236/jis.2011.24015 Published Online October 2011 (http://www.SciRP.org/journal/jis)

Vulnerabilities of LDAP as an Authentication Service

Charlie Obimbo*, Benjamin Ferriman
School of Computer Science, *University of Guelph*, *Guelph*, *Canada*
E-mail: {*cobimbo, bferrima}@uoguelph.ca
Received June 24, 2011; *revised July* 12, 2011; *accepted July* 18, 2011

Abstract

Lightweight Directory Access Protocol (LDAP) servers are widely used to authenticate users in enterprise level networks. Organizations such as universities and small to medium-sized businesses use LDAP for a variety of applications including E-mail clients, SSH, and workstation authentication. Since many organizations build dependencies on the LDAP service, a Denial-of-Service (DoS) attack to the service can cause a greater number of services disrupted. This paper examines the danger in the use of LDAP for user authentication by executing a DoS attack exploiting the TCP three-way handshake required when initializing a connection to an LDAP server.

Keywords: LDAP, SYN Flooding, Denial-of-Service, Authentication Service

1. Introduction

In computing today organizations including universities and small to medium-sized businesses need to provide a wide range of services to a vast number of users. Many of these services require a form of authentication and/or authorization to securely verify the identity of their respective subscribers. Services that may require such authentication include email clients like Zimbra and remote terminal clients such as SSH. A denial-of-service attack on a Lightweight Directory Access Protocol Server (LDAP server) left vulnerable could effectively disrupt productivity and/or economic gains of an organization.

Since LDAP servers are critical [1] in business environments, they are typically hidden behind firewalls and IDS software (see **Figure 1**). One major flaw that usually causes security policies to be degraded, is the fact that LDAP is also an active directory meaning that IT departments will usually make these servers open to the Internet. Despite the efforts of firewalls, well-crafted TCP SYN packets can often cause SYN flooding symptoms.

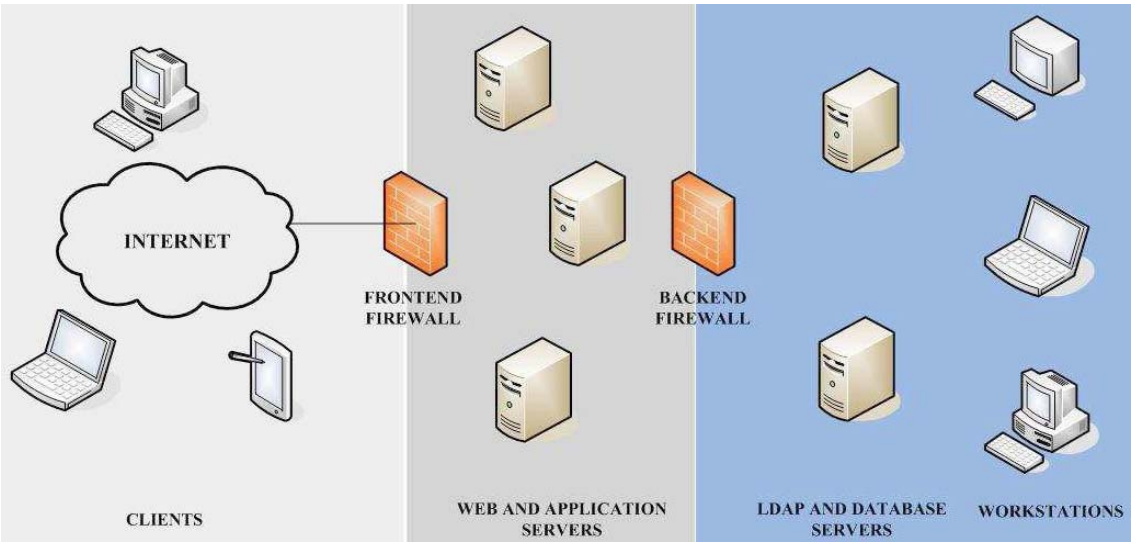

Figure 1. Standard enterprise network configuration [2].

This paper intends to assert the argument that active directory systems like LDAP in their current states are poor choices as authentication services through the design and implementation of a SYN flooding denial-of-service attack. The attack is intended as a simple denial-of-service scenario to bring forth issues that may arise when a LDAP server is used as an authentication service.

1.1. LDAP Overview

LDAP directories are hierarchical databases [1] that hold information about people and entities [3] (such as workstation PAM or SAM files). Inside each directory, data is stored in a tree structure with every level of the tree being a different domain. This structure resembles that of DNS servers; the top-level domain (TLD) is .com or .ca and the fully qualified domain name (FQDN) is ldap. example.com. All sub-directories also follow this structure (see **Figure 2**).

LDAP is designed for providing directory services with other open systems [3]. This means that by design LDAP is an open system for accepting and returning queries. The difference between directories and regular databases is that a directory typically has its data organized to allow quick search results for rapid querying [4].

1.2. Security in LDAP

Originally passwords were sent over networks in plaintext. Since LDAP was designed to facilitate communication among directories for organizations, LDAP's design assumed it would be implemented inside existing (secure) network infrastructures. To combat this shortcoming, LDAP had to incorporate the use of SSL to provide encryption of traffic containing plain-text passwords. The result was that a listener had to be opened on port 636 to support SSL. The solution provided the intended confidentiality but still was an ad-hoc solution. A better solution proposed in LDAP v3 was the incorporation of a Transport Layer Security (TLS) session [6] when initializing a connection with a LDAP server. Though LDAPs protocol security has been implemented there still exists many LDAP servers that allow less secure binding methods. This is usually due to lack of server configuration and/or interaction with legacy systems.

1.3. LDAP Authentication Model

LDAP as an authentication service follows the client/server model. The LDAP model has two main steps when a user requests non-TLS bind authentication. These are (in order):

1) TCP three-way handshake (SYN, SYN/ACK, ACK)

2) LDAP bind() function (performed synchronous or asynchronous)

All TCP traffic to a LDAP server is typically sent to port 389 [7], although v2 of the protocol allows communication with port 636 over the Secure Socket Layer (SSL). Since v3, the protocol has introduced the Simple Authentication and Security Layer (SASL) [8,9] using port 389, port 636 has become obsolete but still remains in use due to legacy directories still using v2 and client applications seeking confidentiality through SSL.

1.4. LDAP Authentication Protocol

As seen in the section above LDAP has two actions when initializing an authentication request. The three-

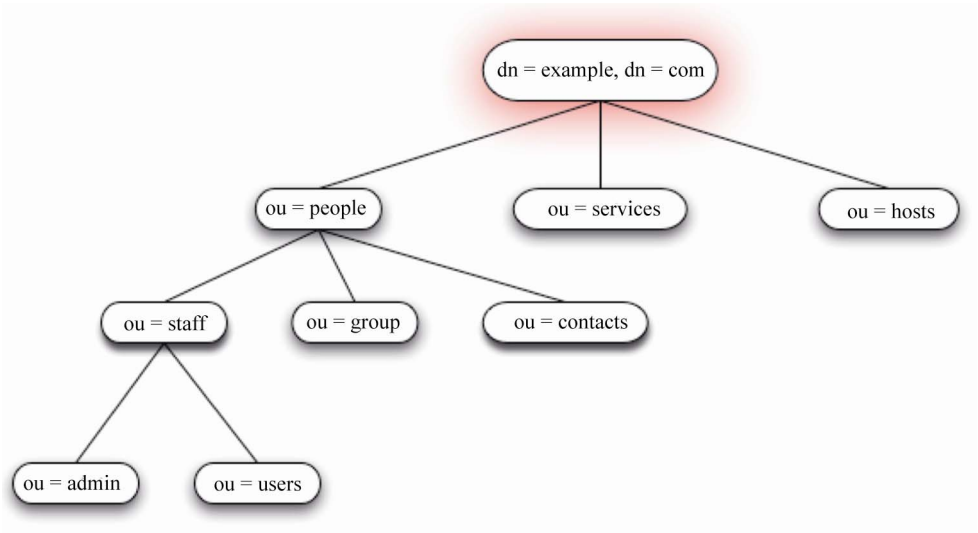

Figure 2. Basic LDAP heirachy [5].

way handshake required of a TCP connection forms the first step of authenticating. The second step requires the use of the LDAP protocol. The LDAP protocol is encapsulated in the TCP layer of a packet band has three standard fields. They are the messageID, protocolOp, and controls. Since authentication with LDAP only adds data in with the messageID and the protocolOp, the controls field will not be addressed.

The messageID field holds a unique value to the session from 0 to 231-1. Message IDs cannot be reassigned until a client has received a response corresponding to that message.

The protocol Op field holds three choices that are important functions of authenticating with LDAP. They include:

- bindRequest
- bindResponse
- unbindRequest

A bindRequest follows the following syntax: version, name, and authentication. Version is used to specify v2 or v3. The name field follows the LDAP standard for querying the directory (see **Figure 3**), while the authentication field specifies the encryption used.

In **Figure 3** the term simple refers to a password that has no encryption (*i.e.* plain-text). This practice is still common among many organizations.

1.5. Related Work

A lot of research has been done on LDAP injections [1,4] while far less is known about proper protection and implementation of LDAP servers. Denial-of-service of LDAP usually targets one of two OSI network layers. Attacks discovered on the **application layer** [10] of LDAP communication include *null byte injection*, where a carefully crafted *POST request* with a null byte inside can cause *unauthenticated authentication* to a system [11]. On the transport layer [12] denial-of-service attacks including *SYN flooding* have also been used to disrupt services. In another paper, security policy was adjusted from semantic threat graphs [13] that were generated by conducting *SYN flooding* on vulnerable high usage systems including LDAP. Though threat analysis is not the intention of this paper, the findings did show how network systems such as routers and servers reacted to heavy attacks. Also illustrated are many default configurations to prevent such an attack.

```
version: 3
name: uid = username, ou = people, o = uoguelph.ca
authentication: simple
simple: 55555555
```

Figure 3. Common format of bind request.

2. Proposed Attack

An attack was chosen to demonstrate the vulnerability of a denial-of-service attack and reinforce the idea that LDAP directory servers are not good candidates for authentication services. The proposed attack is a **SYN flood** attack on the three-way handshake of TCP protocol. Since we are attacking the way which TCP initializes a connection, the attack is to the transport layer. *SYN* **flood** was chosen as the best-suited attack for its simplicity and to emphasize that the problem is the usage of LDAP as an authentication system. In the TCP three-way handshake, a client and server send three packets between each other to initiate a synchronous connection. The three packets consist of:

1) a **SYN** (synchronize) client request

2) a **SYN/ACK** (synchronize/acknowledgement) server reply

3) a **ACK** (acknowledgement) client reply

An example of the handshake can be seen in **Figure 4**.

Since every TCP connection commences with a **SYN request**, attacks can be constructed with raw sockets [15] to spoof sender IP addresses causing server side SYN/

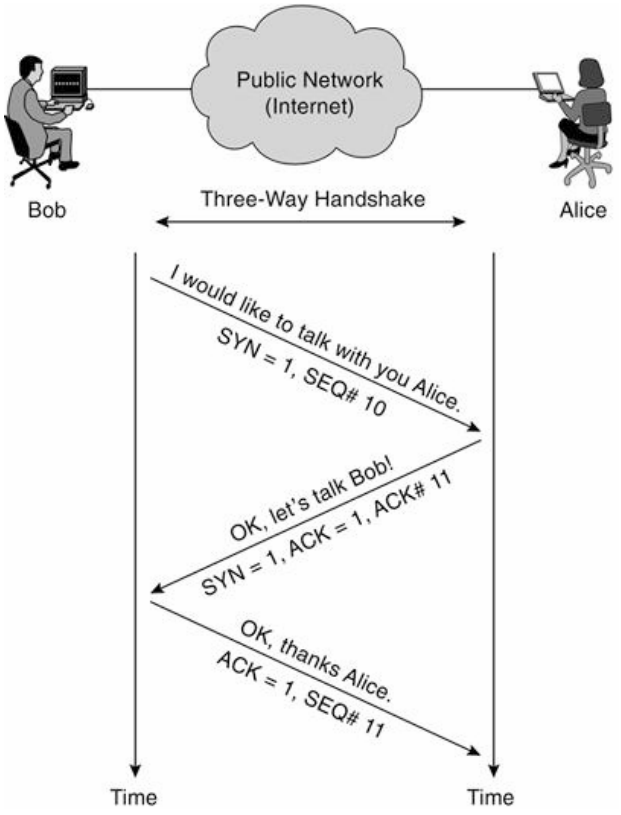

Figure 4. Basic TCP three-way handshake [14].

ACK packets to be redirected to the spoofed address. SYN flooding can potentially cause denial-of-service to two victims. One victim is the destination address if the service cannot properly handle many half-open connections. The other victim can be the spoofed address if it's service cannot handle random traffic well. An example of SYN flooding can be seen in **Figure 5**.

2.1. Packet Design

The packets were carefully constructed to impersonate the genuine *SYN* requests to a LDAP server running on port 38.

A standard IP header was created with the *spoofed* source IP address and the LDAP server IP as the destination address. The IP header takes up 20 bytes of the packet size.

The TCP header consisted of a *spoofed* source port and the LDAP destination port (389) as well as having the *SYN flag* bit flipped on. The TCP header also initializes its offset value to 6 (for six 32 bit words; 5 for the TCP header and 1 for TCP options) and sets the window size to 5840. The TCP header size is also 20 bytes. As an option the maximum segment size (MSS) is set to 1460. The option adds an additional 4 bytes to the TCP header size. In total the packet size is 44 bytes (IP and TCP headers). The TCP header used can be seen in **Figure 6**.

2.2. Attack Implementation

The TCP SYN packets were generated using raw sockets in C. The software sends any number (n) of SYN packets from a source address to a destination address and port (see **Figure 7**).

The attack was tailored for a LDAP server (tested against OpenLDAP [17]) but also has a testing suite made up of a client and server that constantly send and reply to messages. A live LDAP server was also used up until the implementation of the attack but was not able to be tested against since many systems were reliant of it. An implementation of OpenLDAP was used on a closed network.

To test whether the LDAP server was still reachable, a Python program was used to attempt authentication with the LDAP server. Every time a bind request with the correct credentials returned a connection error an alert (chime) would sound. To communicate with the LDAP server the program utilized the Python LDAP library.

3. Analysis of Attack

The number one adversary of this attack is the use of firewalls. That said the use of static firewall policies are highly ineffective to planned attacks and dynamic policy changes are needed. Even with dynamically written poli-

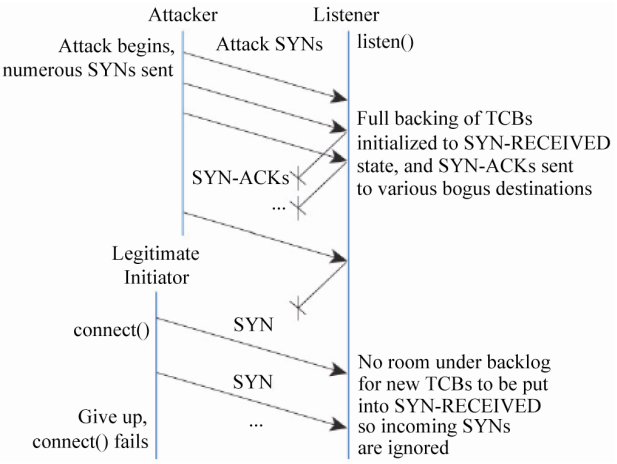

Figure 5. Basic SYN flooding attack [16].

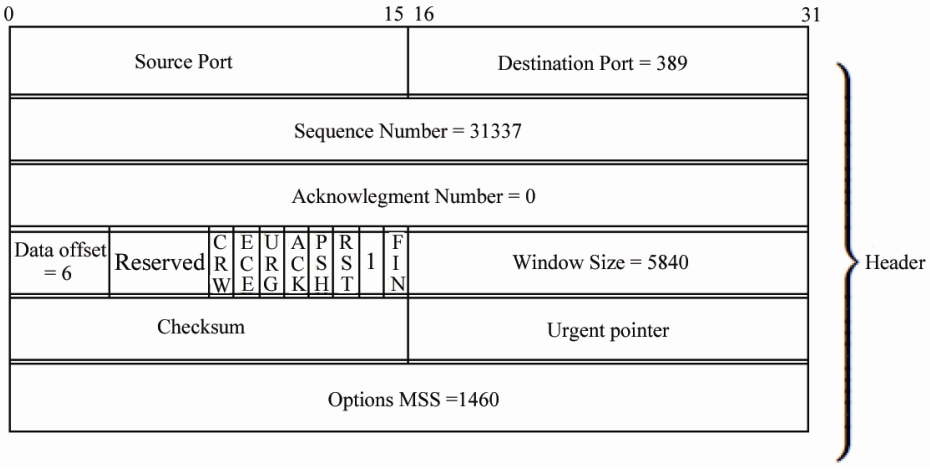

Figure 6. TCP Header—modified for connections with LDAP servers.


```
SYN flooding procedure:

begin
iphdr: = set_iphdr(src_ip, dst_ip);
tcphdr: = set_tcphdr(dst_port);
packet: = iphdr + tcphdr + payload;
for i: = 1 to n step 1 do
sendto(dst_ip, packet); od
where
proc set_iphdr(srcaddr, dstaddr) ≡
∗ip: = header;
ip− > ip_src: = srcaddr;
ip− > ip_dst: = dstaddr;
return(header);
  .
proc set_tcphdr(dstport) ≡
∗tcp: = header;
tcp− > th_sport: = rand();
tcp− > th_dport: = dstport;
tcp− > th_seq: = 31337;
tcp− > th_off: = 6;
tcp− > th_flags: = TH_SY N;
tcp− > th_win: = 5840;
comment: TCP Options (MSS = 1460);
∗tcpOp: = header + sizeof(tcp);
tcpOp[0]: = 2;
tcpOp[1]: = 4;
tcpOp[2]: = 5;
tcpOp[3]: = 180;
return(header);
  .
end
```

Figure 7. SYN flooding and basic header construction and execution algorithms; used in flood.c.

cies a lot of administrative effort is diverted to victimized firewall.

3.1. Packet Generation

In order to create obfuscation of the source of the attack, every packet sent changes its IP address from the previously sent address. This method makes it harder to implement an ad-hoc firewall policy that might disrupt an attack to regain control. It is for this same reason that the port number of the sender is randomized. The actual execution of the flood program (written in C) is done inside a shell script (see **Figure 8**) that changes the senders IP as well as adding the ability to set a *delay* between each packet sent.

3.2. Effectiveness of Attack

In the test environment the attack *successfully* denied service to all applications relying on LDAP for authentication. It must be noted that the LDAP server in the test environment handled many less queries than real-life implementations. Since the attack was devastatingly successful in the test environment, it is predicted to have the same effect in a real-life exercise. Just the increased amount of SYN requests is enough to require rapid modification and/or constant monitoring of firewall policies.

One major factor in the *effectiveness* of an attack is if the LDAP server has an IP that is resolvable to the Internet. This practice is still common since first and foremost LDAP is *a directory access protocol*. The random appearance of the source IP and port also prolonged the attacks effectiveness at defeating firewall policies. In terms of effectiveness, since LDAP is a critical authentication system and can effectively be denial-of-service, the attack (SYN flooding) is seen as highly effective when orchestrated properly.

4. Effectiveness as an Authentication Service

As of 2011 (when this paper was written) there are many choices of authentication services. One such example developed by the Massachusetts Institute of Technology is Kerberos [18]. Though Kerberos is also vulnerable to DoS attacks due to the fact that it a *centralized authentication server*, it addresses two extremely important flaws

```
#!/bin/bash
# 131.104.0.0 is part of the uoguelph.ca namespace
# -n 1 -w 2 -x 4 -y 5 -z 180: one packet sent, with MSS=1460 in TCP options

while true; do
  for j in `seq 254`
  do
    for i in `seq 256`
    do
      ./flood -s "131.104.$j.$i" -d 131.104.93.16 -p 389 -n 1 -w 2 -x 4 -y 5 -z 180
      sleep 0
    done
  done
done
```

Figure 8. Synflood.sh—script to control TCP SYN packets sent.

of LDAP. The two flaws presented are not design flaws of the protocol, rather implementation flaws due to increases in functionality.

4.1. Issues

The first problem with LDAP is the fact that it is an *active directory*. This means that it (the LDAP server) is constantly being inundated with new queries. An authentication service should never have **more traffic than necessary**. Since LDAP services provide more than just authentication, LDAP is a poor candidate as an authenticator. There are three measures that can be taken to better protect an organizations LDAP server(s).

1) Only bind to connections for authentication that are inside an organization's **IP range** and on **a known hosts list**

2) Bind all **blind authentication** connections to a **second physical LDAP server** that is a **clone** of the directory tree for the scope of a blind authentication

3) If allowing connections from the Internet, only allow **blind authentication**

The *first* measure ensures that only known clients inside the network have access to the directory for authentication and *privileged querying*. The *second* measure ensures that all non-critical traffic hitting the LDAP server is directed at a clone server instead ensuring **data integrity**. The *final* measure is ensured by the second measure that all Internet traffic is by policy sent to the clone server. With proper security policies set up **internal attacks** can also be traced easier and shut down faster since the abuse can be logged through internal networks.

The second flaw of LDAP is that since it was designed first for **directory access**, security was appended to the design, and not initially supported. As a result passwords can be sent over networks in *plain-text*. Any authentication service that allows *transmission* plain-text passwords of or *stores* plain-text passwords is not suited for use given computing in the 21st Century. Although v3 of the protocol allows TLS sessions [6], the use of such security has not fully carried over due to historic security policies using the obsolete SSL-session method, which can be easily compromised by *SSL certificate spoofing* [19]. There are also three precautions that can be taken for the second flaw in LDAP.

1) Not allowing **plain-text** passwords to be used for authentication; hash them with at least SHA-256

2) Using the **TLS service** LDAP supports

3) Having all authentication connections connect to server through a **virtual private network (VPN)**

Of course one could try and implement all of the above safe-gaurds but it would be much easier to use software designed for authentication. Due to the required extra policies needed to combat denial-of-service attacks, LDAP does not make a good authentication provider.

4.2. Alternative Authentication Services

As discussed previously in section 4.1, LDAP is a poor choice for authenticating users and entities. One service already described above is **Kerberos**. It is worth mentioning due to the fact that it is present in several systems including the BSD operating system and the X Window System [20]. Many other operating systems use a variant of Kerberos.

Kerberos incorporates the use of strong cryptography in order to ensure the **confidentiality** of authentication credentials. Kerberos is often used in conjunction with a LDAP server that only allows access from connections where an authentication ticket has been granted. Tickets are authentication tokens that verify a users identity to the requested service and tells the user where to create a connection with the service requested.

5. Conclusions

We have shown that the use of LDAP software in its current state is not suitable as an authentication service. In Section 3 the attack proposed was successful at causing denial-of-service due to SYN flooding and was thus able to render the LDAP service disrupted. In Section 3.2 it was argued that due to the fact authentication is a critical service a successful DoS attack is highly effective.

Section 4.1 brought forth two fundamental flaws of LDAP. They included protecting LDAP servers from DoS attacks and protecting user passwords from being discovered over a network. Finally section 4.2 suggested the use of Kerberos as an alternative authentication service to LDAP.

Attack Definition

The characteristics of the attack prompt the use of a better-suited definition: *denial-of-dependent-services* or DoDS.

Denial-of-dependent-services is a planned denial-of-service attack on a service with the intension to disrupt dependent services. This type of attack attempts to optimize the services denied while minimizing its (the attackers) targets. An example of an infrastructure that would be susceptible to this attack is *central authentication services*.

6. References

[1] J. M. Alonso, R. Bordon, M. Beltran and A. Guzman,

"LDAP Injection Techniques," 11*th IEEE Singapore International Conference on Communication Systems*, Guangzhou, 19-21 November 2008, pp. 980-986. doi:10.1109/ICCS.2008.4737330

[2] J. M. Alonso, R. Bordon, M. Beltran and A. Guzman, "LDAP Injection & Blind LDAP Injection," Figure 1 in URJC, 2008, ICCS 2008, p. 4.

[3] "RFC 4512: Light Directory Access Protocol (LDAP): Directory Information Models," 2006. http://tools.ietf. org/html/rfc4512

[4] J. M. Alonso, R. Bordon, M. Beltran and A. Guzman, "LDAP Injection & Blind LDAP Injection," URJC, 2008, ICCS 2008.

[5] "OpenLDAP—Secure Computing Wiki," 2010. http://www.secure-computing.net/wiki/index.php/OpenLDAP

[6] "RFC: 2830: Lightweight Directory Access Protocol (v3): Extension for Transport Layer Security," 2000, http://www.rfceditor.org/rfc/rfc2830.txt

[7] "RFC 1487: X.500 Lightweight Directory Access Protocol," 1993. http://www.faqs.org/rfcs/rfc1487.html

[8] "RFC 2251: Lightweight Directory Access Protocol (v3)," 1997. http://www.faqs.org/rfcs/rfc2251.html

[9] "RFC 4422: Simple Authentication and Security Layer (SASL)," 2006. http://tools.ietf.org/html/rfc4422

[10] "Application Layer-Wikipedia, the Free Encyclopedia," 2011. http://en.wikipedia.org/wiki/Application_Layer.

[11] A. Everett, "Unauthenticated Authentication: Null Bytes and the Affect on Web-Based Applications which Use LDAP," IT Information Security Office, Oklahoma State University, Stillwater, December 2006.

[12] "Transport Layer-Wikipedia, the Free Encyclopedia," 2011. http://en.wikipedia.org/wiki/Trans-port_Layer

[13] S. Foley and W. Fitzgerald, "An Approach to Security Policy Configuration Using Semantic Threat Graphs," Data and Applications Security XXIII, 2009. University College Cork Cork Constraint Computation Centre, Computer Science Department Ireland, Vol. 5645, pp. 33-48, 2009

[14] "TCP 3 WAY HANDSHAKE: Educational Resources, Tips, Tricks, and More," 2010. http:// www.3wayhandshake.com/

[15] "Raw Socket-Wikipedia, the Free Encyclopedia," 2011. http://en.wikipedia.org/wiki/Raw_so-cket

[16] W. Eddy, "Cisco—Defenses against TCP SYN Flooding Attacks," 2006. http://www.cisco.com/web/about/ac123/ac147/images/ipj/ipj_9-4/94_syn_fig2_lg.jpg

[17] "OpenLDAP, Download," 2011. http:// www.openldap.org/software/download/

[18] "MIT Kerberos Distribution Page," 2010. http://web.mit.edu/kerberos/dist/index.html

[19] "SSLSTRIP," 2009. http://tools.ietf. org/html/rfc4422

[20] "Kerberos: The Network Authentication Protocol," 2010. http://web.mit.edu/kerberos/what_is.

