Global Information Assurance Certification Paper

Luciano R. Santos Jr.
Version 1.2e
Back to the Basics with TACACS (Terminal Access Controller Access Control System)

1. Introduction

In today's industry, it is common to have multiple external accesses to and from a network. It is important to find a centralized method of auditing and regulating these external accesses. Terminal Access Controller Access Control System (TACACS) is a method that can do the job. It is known to be easy to implement, adaptable and economical. The purpose of this document is to provide you some basic information on this centralized authentication server and to show you how this can improve remote access security.

2. History on TACACS

The Internet Engineering Task Force (IETF) classifies TACACS as an authentication, authorization and accounting (AAA) server. The IETF created a group that developed the AAA requirements for network access. They accomplished their goal by producing the TACACS protocol that supports a wide array of access models.

The TACACS protocol has been transformed twice in its lifecycle. The first version of TACACS is an older protocol that has been around for years. It was first developed during the early APRANET days. Originally TACACS utilized the UDP (User Datagram Protocol) transport. Within the UDP transport one TACACS packet was encapsulated in the UDP Data field. A client request (Network Access Servers - Figure 1) and a server response (Authentication Server – Figure 2, exchange the TACACS packets). In the eye's of today's industry this is considered unreliable and has very limited functionality.

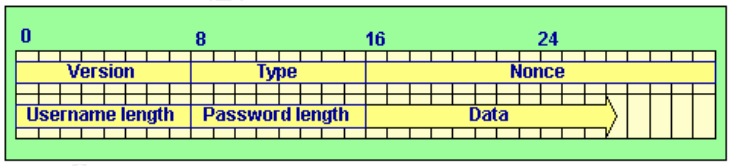

Figure 1: TACACS Request Packet Format[1]

Every packet that is sent to a server has all of the following information (Figure 1):

- Version (1 octet), simple form (0)
- Type (1octet), encoded request or response type
- Nonce (2 octets), arbitrary value to math a response with a request
- Username Length (1 octet), Length of username
- Password Length (1 octet), Length of password
- Data (n octets), Username and Password.

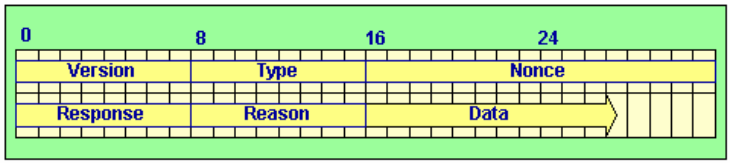

Figure 2: TACACS Request Packet Format[2]

Every response sent from a client contains the following information (Figure 2):

- Version (1 octet), simple form (0)
- Type (1octet), encoded request or response type
- Nonce (2 octets), arbitrary value to math a response with a request
- Response (1octet), Response codes (rejected or accepted).
- Reason (1 octet); Reason when Response code is rejected.

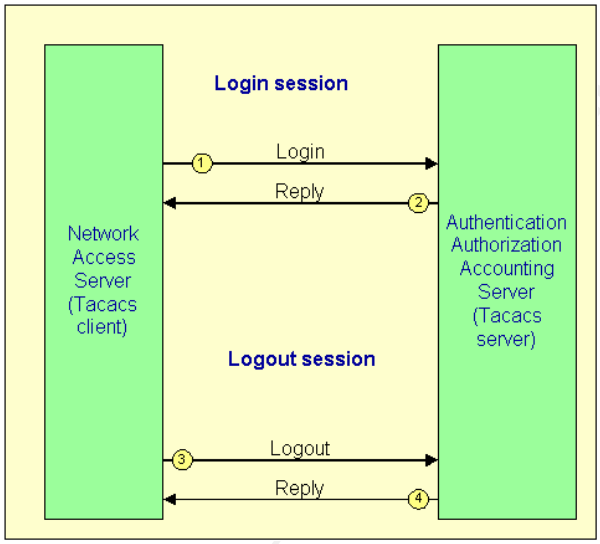

Figure 3: Sample TACACS Message Flow[3]

Adding additional functionality was key, so in the early 1990s it became a reality. The protocol that once utilized the UDP transport changed to the more reliable TCP (Transmission Control Protocol) transport. This was an extended version of the original protocol that had additional functionality and as a subset included the original functionality. The original functionality was included to maintain backward compatibility. They named the new protocol XTACACS (Extended TACACS). Most of the current TACACS Daemons are based on this new protocol. It has more variations of authentication requests and an expanded list of response codes (Per Request for Comments (RFC) 1492). Like the original TACACS, XTACACS packets are exchanged between the clients (Network Access Servers - Exhibit 4) that send an authentication/authorization request to the responding server (Authentication Server – Exhibit 5).

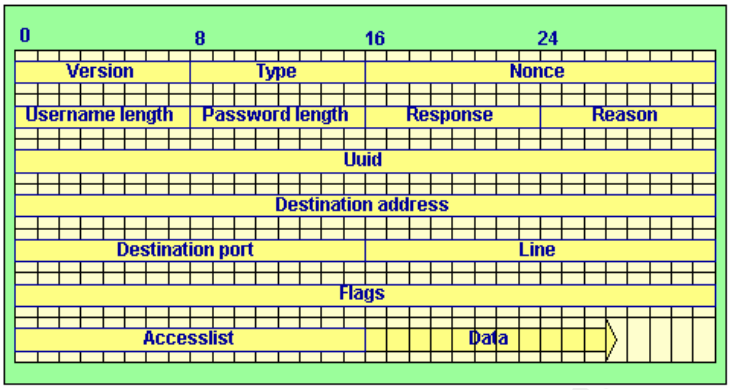

Figure 4: XTACACS Request Packet Format[4]

Every packet that is sent to a server has all of the following information (Figure 4):

- Version (1 octet), XTACACS (128)
- Type (1octet), encoded request or response type
- Nonce (2 octets), arbitrary value to math a response with a request
- Username Length (1 octet), Length of username
- Password Length (1 octet), Length of password
- Response, 0
- Reason, 0 (except for LOGOUT, SLIPOFF)
- Uuid, 0
- Destination Address, Destination address when using CONNECT, SLIPON or SLIPOFF
- Line, the line number
- Flags, 0
- Access list, 0
- Data, (n octets), Username and Password.

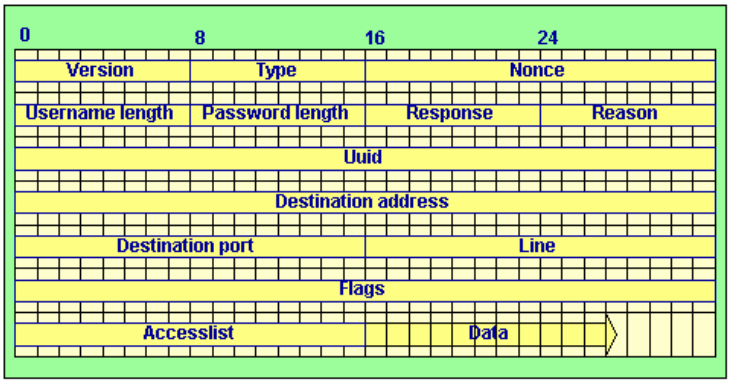

Figure 5: XTACACS Reply/Response Packet Format[5]

Every packet that is sent to a server has all of the following information:

- Version (1 octet), XTACACS (128)
- Type (1octet), encoded request or response type
- Nonce (2 octets), arbitrary value to math a response with a request
- Username Length (1 octet), Length of username
- Password Length (1 octet), Length of password
- Response, return accepted or rejected
- Reason, return the reason when rejected
- Uuid, Userid code assigned
- Destination Address, Destination address when using CONNECT, SLIPON or SLIPOFF
- Destination Port, Destination port when using the command CONNECT
- Line, the line number
- Flags, miscellaneous flags
- Access list, Access list for users

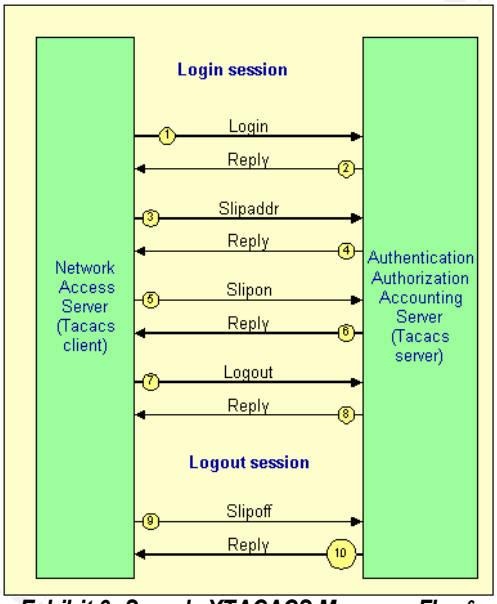

Exhibit 6: Sample XTACACS Message Flow[6]

Cisco Systems adopted TACACS and created a proprietary security implementation called TACACS+ for its AAA architecture. Enhancements were made to the product by "separating the authentication, authorization and accounting functions." Resulting in a more secure product that would allow encryption to all NAS (Network Access Servers) -server transmissions. Cisco's TACACS+ also permits arbitrary length and content parameters for authentication exchanges. Bearing no resemblance to the original TACACS, the TACACS+ packet formats are not backward compatible. The current IETF standard for TACACS is TACACS+. The remaining sections of this document will be based of the current IETF standard.

To get access to a router or network access server, TACACS+ provides a centralized validation of users. The TACACS+ daemon that runs on a Unix or Windows NT workstation maintains TACACS+ services on a

database. In order for the TACACS+ features on your access server to be available, you must have a TACACS+ server configured and accessed.

As stated earlier, TACACS+ provides for separate authentication, authorization, and accounting functions. Each function works independently because TACACS+ allows for a single TACACS+ daemon to provide for them. Depending on the capabilities of the daemon, each function can be tied into its own database to take advantage of other functions available on that server or on entire network. TACACS+'s ultimate goal is to construct a methodology that would manage multiple network access points from a single management service.

Communication protocols such as Point-to-Point Protocol (PPP), Compressed SLIP (CSLIP), Apple Talk Remote Access (ARA) and Serial Line Internet Protocol (SLIP) can be utilized through network access points that enable dumb terminals, workstations, home PCs, routers and more in conjunction with a modem, ISDN or other forms of adapters. Whether it is a single user to a network, sub-network or interconnected network; a network access server provides the connections. Network access clients are connected to the network through a network access server. With TACACS+ being administered through the AAA security services, it is able to provide some very secure functions. One of the functions is having the ability to ensure the confidentiality and integrity of the protocol exchanges. Each protocol exchange between a TACACS+ daemon and a network access server are encrypted.

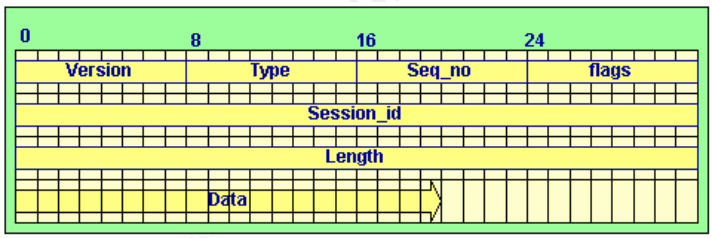

Figure 7: TACACS+ Packet Format[7]

Every authentication packet contains the following information when send to a server:

- Version number (1 octet), Version number consisting of a major and minor version number.
- Type (1 octet), Message type which equals (TAG_PLUS_AUTHEN: =1) authentication.
- Sequence number (1 octet), Sequence number of packets for a current session.
- Flags (1 octet), This octet contains various bitmap flags such as a flag indicating whether the data after the 'length' is encrypted or not.
- Session id (4 octets), A random number generated by the client for this session.
- Length (4 octets), The length of the remainder of the packet.
- Data depending on the authentication action and type.

Every authentication packet contains the following information when sent as a response to a client:

- Version number (1 octet), Version number consisting of a major and minor version number.

- Type (1 octet), Message type which equals (TAG_PLUS_AUTHEN: =1) authentication.
- Sequence number (1 octet), Sequence number of packets for a current session.
- Flags (1 octet), This octet contains various bitmap flags such as a flag indicating whether the data after the 'length' is encrypted or not.
- Session id (4 octets), The session id number belonging to this session.
- Length (4 octets), The length of the remainder of the packet.
- Data, (n octets) containing the reply of the authentication request.

When the server expects a challenge response, the client sends an authentication continue packet containing the following:

- Version number (1 octet), Version number consisting of a major and minor version number.
- Type (1 octet), Message type which equals (TAG_PLUS_AUTHEN: =1) authentication.
- Sequence number (1 octet), Sequence number of packets for a current session
- Flags (1 octet), This octet contains various bitmap flags such as a flag indicating whether the data after the 'length' is encrypted or not.
- Session id (4 octets), The session id number belonging to this session
- Length (4 octets), The length of the remainder of the packet.
- Data, (n octets) containing the CONTINUE packet body.

Every authorization packet contains the following information when sent to a server:

- Version number (1 octet), Version number consisting of a major and minor version number.
- Type (1 octet), Message type which equals (TAG_PLUS_AUTHOR: =2) authorization.
- Sequence number (1 octet), Sequence number of packets for a current session.
- Flags (1 octet), This octet contains various bitmap flags such as a flag indicating whether the data after the 'length' is encrypted or not.
- Session id (4 octets), A random number generated by the client for this session.
- Length (4 octets), The length of the remainder of the packet.
- Data depending on the authentication method.

Every authorization packet contains the following information when sent as a response to a client:

- Version number (1 octet), Version number consisting of a major and minor version number.
- Type (1 octet), Message type which equals (TAG_PLUS_AUTHOR: =2) authorization.
- Sequence number (1 octet), Sequence number of packets for a current session
- Flags (1 octet), This octet contains various bitmap flags such as a flag indicating whether the data after the 'length' is encrypted or not.
- Session id (4 octets), The session id number belonging to this session.
- Length (4 octets), The length of the remainder of the packet.
- Data, (n octets) containing the reply of the authorization request.

Every accounting packet contains the following information when sent to a server:

- Version number (1 octet), Version number consisting of a major and minor version number.
- Type (1 octet), Message type which equals (TAG_PLUS_ACCT: =3) accounting.
- Sequence number (1 octet), Sequence number of packets for a current session
- Flags (1 octet), This octet contains various bitmap flags such as a flag indicating whether the data after the 'length' is encrypted or not.
- Session id (4 octets), A random number generated by the client for this session.
- Length (4 octets), The length of the remainder of the packet.
- Data depending on the authentication action and type

Every accounting packet contains the following information when sent as response to a client:

- Version number (1 octet), Version number consisting of a major and minor version number.
- Type (1 octet), Message type which equals (TAG_PLUS_ACCT: =3) accounting.
- Sequence number (1 octet), Sequence number of packets for a current session.
- Flags (1 octet), This octet contains various bitmap flags such as a flag indicating whether the data after the 'length' is encrypted or not.
- Session id (4 octets), The session id number belonging to this session.
- Length (4 octets), The length of the remainder of the packet.
- Data, (n octets) containing the reply of the accounting request

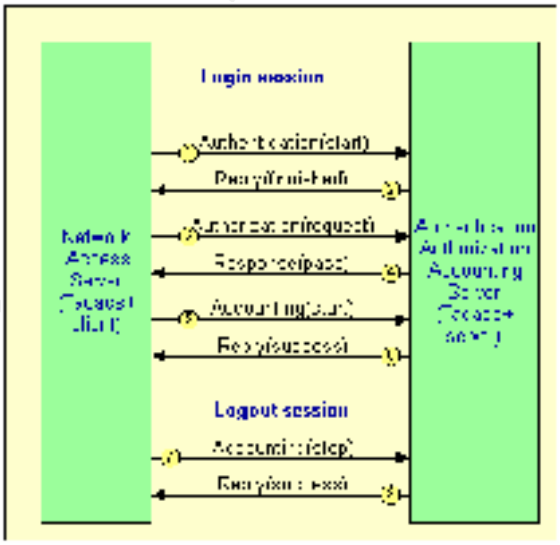

Figure 8: TACACS+ Message Flow[8]

3. How it Works

The protocol that is used to exchange Network Access Server information is TACACS+. TACACS+ exchanges this information between a centralized database and a network device. This protocol allows a separate TACACS+ server (the access server) to provide authentication, authorization and accounting

(AAA). While all these services are part of the TACACS+ protocol, they are independent of one another. In any given time a TACACS+ configuration can use any or all of the three services.

The TACACS+ servers can control server options, attribute/value pairs, define users, and control authentication services. You can specify through the options section operation parameters of the service, the accounting file name and shared secret key. A series of user and group definitions that are used to control authorization and authentication of the file is the remainder of the file. The format that is followed is user = username or group = user name, followed by single or multiple attribute/value pairs inside curly brackets.

The process is initiated when client conducts a TCP session. This TCP session passes attribute/value pairs to a server utilizing a standard header format. A variable length parameter field follows the standard header format. The service request type (AAA) is contained in the header and is sent in a clear form. The rest of the parameter field is encrypted to maintain its integrity and confidentiality. Extensibility and site-specific customization can be provided from TACACS variable parameter field, while reliability of delivery is ensured by the TCP protocol. One of the things that could impact a server's performance is the format and protocol. It increases the communication overhead.

4. Authentication

There are three types of TACACS authentication: Start, Reply and Continue. The description of the type of authentication comes from the start packet, which begin when the client starts the authentication. With simple authentication the packet will also include the users ID and password. The next step is the reply stage. The server responds to the client with a reply. If the client is requiring additional information it is passed with a client continue and server reply packets. By using various authentication protocols the transactions can include login and password change. When attribute/value pairs for connection configuration return, the TACACS authentication is classified as successful.

The identity of a user or any known entities is determined by how the authentication process controls access to the network devices. Challenge-response authentication, fixed passwords and one-time passwords are types of authentication that are compatible with the TACACS+ protocol. Authentication with TACACS+ can take place when an initial logon takes place on a machine or when service requests are sent that requires privileged access.

Logon begins when TACACS+ obtains the user's id and password information. The information is then encrypted by using the MD5 encryption algorithm and a TACACS+ packet header is added. Ways to identify attributes of packets such as: type, sequence number, encryption utilized and length are through the information in the packet header. That packet is transferred via the TACACS+ protocol to the TACACS+ server. After the server receives the packet, the authentication of the user information takes place. Then, the server notifies the client whether the authentication has been accepted or denied. Another scenario would include the server notifying the client requesting more information and that authentication will continue. Unless the authentication is accepted or denied, the challenge-response will continue. When a request for privileged or restricted services is sent, more information is asked for from the TACACS+ protocol to access these requests. When local password authentication is enabled the local password authentication is invoked and the TACACS+ password authentication fail. Local authentication is enabled by default. To re-enable local authentication, TACACS+ authentication must be disabled.

5. Authorization and Accounting

Request and Response attribute/value (AV) pairs make up the Authorization functions within TACACS. The attribute/value pairs are used to permit or deny addresses, commands, services or protocols. Request and Response attribute values are what make up the Authorization functions. These functions are used to: Set the level of user privileges, invokes callback actions, assigns networks addresses that are specific and permits or denies a variety of command, addresses, services or protocols. All of these functions can be incorporated within an authentication transaction or an authorization-specific request.

The Accounting functions within TACACS are very similar to the format of TACACS Authorization functions. Start, Stop, More, and Watchdog are part of what is included within the Accounting functions. The ability to validate TCP session, when data is not sent for extended periods of time is provided by the Watchdog function. Along with standard accounting data supported by RADIUS, the event logging capability that records system level changes in access rights or privileges is included within TACACS Accounting.

6. Capabilities

Recursive lookup and callout capabilities considerably enhance the TACACS authentication and authorization processes. Connection, Authentication and Authorization information can be spread across multiple entries by utilizing the recursive lookup capability. The user entry is where the AV pairs are first looked up. The group entry is where the unresolved pairs are looked up if the user is a member of a group. If a default value is specified in the group entry it is assigned that default. Recursive lookups can have any number of connection requirements, because TACACS+ allows groups to be apart of other groups. The callout capability that permits the execution of user-supplied programs is also supported by TACACS+. Any number of requirements can be accommodated, when callout is used to dynamically alter the authentication and authorization processes. The RADIUS static configuration is known to be less a versatile approach. Third-party authentication mechanisms such as Secure ID or Kerberos can be use to interface with callout. Callout can also be used for writing accounting and audit records and pulling parameters from a directory or database.

Like RADIUS, TACACS can be configured to use redundant servers. Since TACACS uses a reliable transport it has the ability to detect failed nodes like RADIUS. What is different about RADIUS and TACACS, is that TACACS cannot be configured to proxy NAS request. Without that capability the usefulness in cross-domain applications and large-scale applications is very limited.

7. Implementation

When it comes to TACACS server implementations, there are quite a few available including some freeware versions. On the client side, implementations are available through Cisco. Since Cisco distributes the TACACS+ and TACACS source code, the functionality and features can vary from implementation to implementation. The implementation known as the most robust of the commercial implementations is Cisco Secure. A compelling feature of Cisco Secure is that it can also support RADIUS functions. Before selecting server software and NAS components, make sure that you define functional requirements. TACACS+ will work well in a Cisco-centric environment. If you have a hybrid environment, consider server products that have capabilities for both TACACS+ and RADIUS.

8. Conclusion

In conclusion, the primary limitation is the lack of use of TACACS+. Currently, there are fewer server implementations and even less NAS implementation. There are no known custom extensions to the protocol or vendor-specific AV pairs outside of Cisco. There are some known scalability and performance issues with TACACS. TACACS utilizes TCP over multiple queries to establish connection, instead of the single-packet UDP that RADIUS would use. The ability to proxy requests through TACACS + servers are non-existent. Without that ability, TACACS could not support authentication across multiple domains.

In any case TACACS+ is still known to be easy to implement, adaptable and economical, with the ability to support an implementation of Cisco's NAS-based VPNs. TACACS+ adds support for third-party products by customizing the AAA functions through the "outcalls" capability. Being that TACACS+ is the IETF standard, NAS manufactures need would increase. Until that happen, TACACS+ will remain the Cisco-centric environments AAA solution.

9. References

Anderson, Brian. "TACACS User Identification Telnet Option – RFC927." 1 December 1984. URL: http://www.faqs.org/rfcs/rfc927.html (5 November 2001)

Ceti Solutions. "TACACS+ simple client library and login utility." 2001. URL: http://www.ceti.com.pl/~kravietz/progs/tacacs.html (5 November 2001)

Cisco Systems." PIX 5.1.x: TACACS+ and RADIUS." 2001. URL: http://www.cisco.com/warp/public/110/pix51.shtml (25 September 2001)

Cisco Systems." Configuring TACACS+." 9 August 2001. URL: http://www.cisco.com/univercd/cc/td/doc/product/software/ios121/121cgcr/secur_c/scprt2/scdtplus.htm#xtocid183612 (25 September 2001)

Cisco Systems. "Configuring Network Security." 1997.URL: http://www.cisco.com/univercd/cc/td/doc/product/lan/cat5000/rel_2_2/c5kcg2_2/10secure.htm (5 November 2001)

Finseth, C. "An Access Control Protocol, Sometimes Called TACACS – RFC1492." 1 July 1993. URL: http://search.ietf.org/rfc/rfc1492.txt (5 November 2001)

Internet Next Generation. "The Internet NG Project – TACACS." 1 November 1999. URL: http://ing.ctit.utwente.nl/WU5/D5.1/Technology/tacacs/ (5 November 2001)

Internet Next Generation. "The Internet NG Project – XTACACS." 1 November 1999. URL: http://ing.ctit.utwente.nl/WU5/D5.1/Technology/xtacacs/ (5 November 2001)

Internet Next Generation. "The Internet NG Project – TACACS+." 1 November 1999. URL: http://ing.ctit.utwente.nl/WU5/D5.1/Technology/tacacs+/ (5 November 2001)

Kiessling, Robert. "TACACS FAQ." 4 April 1998. URL: http://www.de.easynet.net/tacacs-faq/tacacs-faq.html#toc3 (5 November 2001)

Network Ice Corporation. "Port 49 TACACS, Login Host Protocol" 1998. URL: http://www.netice.com/Advice/Exploits/Ports/49/ (5 November 2001)

Schirmer, Neil. "TACACS." 24 July 2001. URL: http://searchsecurity.techtarget.com/sDefinition/0,,sid14_gci213464,00.html (5 November 2001)

Security Focus. "TACACS+ Denial of Service Vulnerability." 30 May 2000. URL: http://www.securityfocus.com/bid/1293 (5 November 2001)

Security Focus. "TACACS+ Protocol Flaws Vulnerabilities." 30 May 2000. URL: http://www.securityfocus.com/bid/1294 (5 November 2001)

Security Focus. "Cisco Secure ACS for Windows NT Oversized TACACS+ Packet DoS Vulnerability." 21 September 2000. URL: http://www.securityfocus.com/bid/1706 (5 November 2001)

Security Focus. "Cisco PIX TACACS+ Denial of Service Vulnerability." 3 October 2001. URL: http://www.securityfocus.com/bid/2551 (5 November 2001)

Seral, Devrim. "Tacacs+ RPM Distribution Home Page." 2001. URL: http://www.gazi.edu.tr/tacacs/index.php?page=what (5 November 2001)

Tipton, F.Harold. Information Security Management Handbook, 4th Edition. Boca Raton: Auerbach, 2001. P.33-50

Welcher, J. Peter. "Mentor Technologies – TACACS+." 1 December 1995. URL: http://www.ccci.com/welcher/papers/tacacs.htm (5 November 2001)

[1] Internet Next Generation, http://ing.ctit.utwente.nl/WU5/D5.1/Technology/tacacs/
[2] Internet Next Generation, http://ing.ctit.utwente.nl/WU5/D5.1/Technology/tacacs/
[3] Internet Next Generation, http://ing.ctit.utwente.nl/WU5/D5.1/Technology/tacacs/
[4] Internet Next Generation, http://ing.ctit.utwente.nl/WU5/D5.1/Technology/xtacacs/
[5] Internet Next Generation, http://ing.ctit.utwente.nl/WU5/D5.1/Technology/xtacacs/
[6] Internet Next Generation, http://ing.ctit.utwente.nl/WU5/D5.1/Technology/xtacacs/
[7] Internet Next Generation, http://ing.ctit.utwente.nl/WU5/D5.1/Technology/tacacs+/
[8] Internet Next Generation, http://ing.ctit.utwente.nl/WU5/D5.1/Technology/tacacs+/