Introduction

One of the most difficult jobs a network administrator faces in today's network security environment is the limiting access to network services the authorized user encounters. Equally as challenging is closely monitoring what and when services are used, and their frequency of use. The need for network security puts an even greater constraint on network administrators, as networks continue to grow and become more decentralized. Routers and firewalls usually use filters (access-list) that are based on source and or destination IP addresses, along with ports to control access to services. This means that restrictions are attached to a device and not an individual. For example if I allow traffic from 10.1.0.5 to access a particular web server, then anyone who is sitting at the machine with the address of 10.1.0.5 will automatically have access to this web server. A more secure and flexible, filtering method to prevent this would be to attach the access to a specific individual. In other words only someone with the right username and password would be able to access the service. One solution would be to simply create a user database on every device that we want to restrict access to. This however would be a tremendous administrative nightmare, since it would entail attempting to update these individual databases every time a user was added or deleted. What is ideally needed is a central point of control - this is called an Access Control Server (ACS). This server would house the central user database, and obviously there could be more than one server depending on the size and need of the organization. There are different types of protocols that can be used to carry out this task (KERBEROS, RADIUS, and TACACS+).

This paper will focus on understanding and implementing TACACS+, however the same methodology can be applied to other protocols that handle access control. What is TACACS+? TACACS+ stands for Terminal Access Control Access Control Server. It is a derivative of the TACACS application used by Defense data network (DDN). Cisco made some enhancement to the TACACS application and thus TACACS+ came into existence as a Cisco Proprietary Protocol. The main goal of TACACS+ is to provide a centralized database against which to perform authentication. In actuality TACACS+ provides Authentication, Authorization, and Accounting (AAA).

Authentication - Refers to who is allowed to gain access to the network. Users are required to prove that they are really who they say they are. Traditionally authorized users were forced to use a password to verify their identity, however this has numerous security limitations. While TACACS+ can use usernames and passwords it can also use other mechanisms such as "one time" passwords. If standard passwords are used for authentication then adequate password aging should be in place to prevent hackers from accessing the system. For example: If a packet was intercepted and contain a users password the intercepted packet would have aged before the culprits are able to decode the encryption facilitating entry into the system.

Authorization - Refers to what the user is allowed to do, or what services the user has access to. For example: If a users dials into the network remotely and passes authentication, authorization could dictate what IP addresses the user has access to and what applications on those devices as well. Accounting -
Accounting - Refers to keeping track of what the user did, and when the services were used. This is extremely useful for a security auditing purposes. Accounting uses start and stop

messages to keep track of when a service was started and when it was terminated. Accounting records can either be stored locally or sent to another device such as a syslog server. TACACS+ uses a client-server model the server (running on either UNIX or NT) is queried by the client and the server in turn returns a reply stating weather the user passed or failed the authentication. It is important to note that the client is not the user or the user's machine but rather the device that is trying to determine if the user should be let through the network (typically a router or a firewall).

TACACS+ uses a client server model approach. The server (running on UNIX or NT) is questioned by the client and the server in turn reply by stating whether the user passed or failed the authentication. It is important to note that the client is not the user or the user's machine, but rather the device that is trying to determine if the user should be allowed entry into the network (typically a router or a firewall). TACACS+ uses TCP as the transport protocol – the default port is 49. If required, the server can be configured to listen on other ports. TACACS+ is similar to RADIUS (remote Access Dial In User Server) with a few key differences. RADIUS uses UDP for communication between the client and the server were as TACACS+ used TCP. With TCP being connection
oriented protocol and more reliable it makes for a more robust transport protocol of choice. Both TACACS+ and RADIUS use a shared secret key to provide encryption for communication between the client and the server. RADIUS encrypts the user's password when the client made a request to the server. This encryption prevents someone from sniffing the user's password using a packet analyzer. However other information such as username and services that is being performed can be analyzed. TACACS+ encrypts not
just only the entire payload when communicating, but it also encrypts the user's password between the client and the server. This makes it more difficult to decipher information about the communication between the client and the server. TACACS+ uses MD5 hash
function in its encryption and decryption algorithm. ftp://ftpeng.cisco.com/pub/tacacs/tac-rfc.1.78.txt)

Lastly in RADIUS the Authentication and Authorization checking are bundled together. When the client request authentication from the server; the server replies with the authentication attributes, as well as the authorization attributes. These functions can
not be performed separately. In TACACS+ all three AAA functions (authentication, authorization, and accounting) can be performed separately. This definitely gives the administrator much more flexibility when designing his AAA policy. For instance one method such as kerberos can be used for authentication, and a separate method such as TACACS+ can be used for authorization. Configuring TACACS+ Configuring TACACS+ consists of two parts 1) creating user profiles in the server's database, 2) setting up the clients to communicate with the server. In an attempt to provide extra security protection, we will focus on different examples of using TACACS+
effectively in a Cisco network environment. I will make the assumption that the reader has the basic knowledge of how to configure both a Cisco Routers as well as Cisco PIX firewalls. Example 1: Securing the routers/firewalls Routers and firewalls are a critical component of any network, and as such it is wise to closely limit who has access to these devices. We will focus on the Cisco router, however the same steps are needed for
implementation on the Cisco PIX firewalls. When accessing a Cisco router the Command line

interface (CLI), there are two main modes, the user mode and the enable mode. The user mode allows you limited access, to look at the routers configuration and statistic. Enable mode entitles you to "superuser" privileges, allowing you total control of the router. Both modes are typically protected by a separate password. Applying a filter to restrict from which IP addresses the router will accept telnet connections can further restrict access. While this is a good start there are several problems with these methods of access control. Firstly the use of a general password allowing access to the router, increases the likely hood that password may be leaked out. (I have worked in many organizations were I have heard the passwords being yelled across the room). The added restriction of using an access-list to filter the source address for telnet access means that if I can find out the password I can go to a machine that is allowed connection with the router and telnet in. In the event that something goes wrong, and we are forced to examine the syslogs, to pinpoint who was at the "controls" at that time, the only thing that the syslog will be able to indicate is the IP address from where the telnet originated, but not who was logged into the router. That would then mean it could be any one that had access to the room where the originating machine was located. Think of this scenario: Some employee does a change on the router without going through the proper change management channels (he was trying to save time, it was just a simple change), the change does not go as planned and a portion of the network is down. A look at the syslogs reveals that someone was logged into the router from an authorized IP address, however several departments have access to this particular machine - Engineering Implementation, and Operations. There is no way in definitively knowing, if this was merely a simple mistake or was the originating machine manipulated in an attempt to gain access to the router. Now the finger pointing begins. By using AAA we can move the access from the network level to the User level. In order to gain access to the router, the user logging in not only has to be coming from the right IP address, but also requires authentication by the TACACS+ server. If a device were intentionally compromised, there would be an audit trail showing not only the originating IP address, but also the user account the logged into the device. Another step is adding authorization. This would allow certain users more privileges than others. Level one engineers might only get authorization to execute a couple of "show " commands, while a select few Senior engineers are allowed "superuser" rights. Figure 1 shows a regular user configuration on a Unix Cisco Secure TACACS+ server, and Figure 2 demonstrates a user who has full enable rights to the router Note that for user look only the privilege level is set to 1, that is the number that denotes regular user mode, while user full right the privilege level is set to 15 that denotes enable mode.

```
user = look_only{
    password = clear "********"
    service=shell {
    set priv-lvl=1
       }
```

Figure 1

```
user = full_right{
    password = clear "********"
    service=shell {
    set priv-lvl=15
       }
```

Figure 2

The Next step is to configure the router or the pix to use the TACACS+ server to perform authentication and authorization

From the Command line the following commands are entered on the router:

tacacs-server host 192.168.2.1 key secretkey
This specified the host that is configured as the AAA server and the encryption key, the key must be defined on both the router and the server, and if the keys don't match they won't be able to decrypt each others packets
aaa new-model
Turns on AAA authentication
aaa authentication login default group tacacs+
Specifies that tacacs+ will be the method used for login authentication
aaa authorization exec default group tacacs+
Specified that tacacs+ will be the method used to determine what exec level or commands the user can execute
aaa accounting start-stop group tacacs+
enables accounting and specifies to keep track of the beginning and ending of commands this lets you track what commands were used and when

For PIX configuration the commands are as follows:

aaa-server sans host 192.168.2.1 secretkey
aaa-server sans protocol tacacs+
Specifies the address of the AAA server and the protocol used for AAA
aaa authentication serial console sans
aaa authentication telnet console sans
specifies that the previously defined authentication server "sans" should be used to authenticate both users connecting to the console directly or though telnet

The previous example showed how to use TACACS+ to restrict access to a router or PIX firewall, the next example will show how to control remote dial-in users' access to the network

With Cisco routers access to the network is typically restricted through the use of access-list The access-lists are applied to an interface and all traffic that passes through that interface is checked against the access-list (see figure 3).

```
Access-list 102 permit ospf any any
Access-list 102 permit tcp any host 192.168.2.10 eq 80

Interface e0
Ip access-group 102 in
```

Figure 3

Access-lists filter based on source and destinations IP addresses along with source and destination ports. The problem is that remote users usually get an address assigned to them dynamically when they dial into the network, so it is not known ahead of time what address a user will be using when he is trying to access a particular device on the network. How can I apply an access list if I don’t know what source address is going to be assigned to a particular user. A mechanism is needed whereby access lists can be tailored to specific users, and then applied to dynamically when the user connects to the network

Cisco refers to this mechanism as “Virtual Profile” or per-user profile. (http://www.cisco.com/univercd/cc/td/doc/product/software/ios120/12cgcr/dial_c/dcperusr.html)

The way that virtual profiles work is that when the user dials into the network a virtual interface is created (by virtual we mean logically, in other words it is not a physical interface) At the NAS (network access server, this is another name for the router that the user dials into it is called this because it serves as the entry point for the remote user to gain access to the local network). On the TACACS+ server a policy is created for that user, when the user dials in his policy is applied to his virtual interface. When the user disconnects form the network the virtual interface dynamically gets removed from the NAS. This method allows different policies (access-lists) to be applied to different users without affecting everyone else.
Figure 4 and 5 shows two different user profiles, Figure 6 shows the corresponding configurations needed on the access router(not the full router config)

```
user= randy
  Password= cleartext secret
  Service= PPP protocol= ip {

  ip:inacl#5=permit ip any
197.80.1.0 0.0.0.255 }
```

Figure 4

```
user= maria
  Password= cleartext secret
  Service= PPP protocol= ip {

  ip:inacl#5=permit tcp any
197.80.1.4 0.0.0.2 eq 80}
```

Figure 5

hostname Access
!
aaa new-model
aaa authentication ppp default group tacacs+
aaa authorization network default group tacacs+
aaa accounting network default start-stop group tacacs+

virtual-profile virtual-template 1
When a users dials in his traffic is process through a virtual interface
This command tells the router what interface to use as a “clone” when it makes a virtual interface for the user.

virtual-profile aaa
Uses the method defined in the “aaa authorization” statement to determine what polices need to be applied to a user’s virtual interface.

interface Virtual-Template1
ip unnumbered Ethernet0
no ip directed-broadcast
ppp authentication chap
Creates a template that can be copied or cloned when a virtural interfaces is needed. By cloned we mean that the properties of this interface will be inherited by the virtual interface.

tacacs-server host 1.1.1.9 key ciscokey
defines the address of the tacacs+ server and the encryption key to use when communicating with the server(server has to have same key)

Figure 7 shows some of the output from the router as the users dial into the network:

```
Access#sh acccess-list
Access#
(nothing shows up, indicating that currently there are no access lists on the router)

19:01:14: AAA/AUTHEN/START (2068838748): using "default" list
19:01:14: AAA/AUTHEN/START (2068838748): Method=tacacs+ (tacacs+)
AAA/AUTHEN (2068838748): status = PASS

(authentication has succeeded )
(now authorization is beginning to see what type of services the user can use )

19:01:17: Se0:22 AAA/AUTHOR/LCP (127545066): send AV protocol=lcp
19:01:17: Se0:22 AAA/AUTHOR/LCP (127545066): found list "default"
19:01:17: Se0:22 AAA/AUTHOR/LCP (127545066): Method=tacacs+ (tacacs+)
19:01:17: AAA/AUTHOR/TAC+: (127545066): user=randy
19:01:17: AAA/AUTHOR/TAC+: (127545066): send AV service=ppp
19:01:17: AAA/AUTHOR/TAC+: (127545066): send AV protocol=lcp

19:01:17: TAC+: (127545066): received author response status = PASS_ADD
(authorization succeeded)

19:01:19: %LINK-3-UPDOWN: Interface Virtual-Access2, changed state to up
( a virtual interface has been created to process the user's traffic)

19:01:19: Vi2 AAA/AUTHOR/IPCP: Processing AV addr=1.1.1.150
19:01:19: Vi2 AAA/AUTHOR/IPCP: Processing AV inacl#5=permit ip any  197.80.1.1 0.0.0.255
19:01:19: Vi2 AAA/AUTHOR/IPCP: Authorization succeeded
(The user's profile has been attached to the virtual interface ,this  includes assigning a user an IP address and
applying an access list to the virtual profile)


Access#sh access-list
Extended IP access list Virtual-Access2#0 (per-user)
    permit ip any 197.80.1.0 0.0.0.255

(now we see the access list on the router but only applied to the Virtual-Access2 interface)

Access#sh access-list
Extended IP access list Virtual-Access1#0 (per-user)
    permit tcp any 197.80.1.4 0.0.0.2 eq www
Extended IP access list Virtual-Access2#0 (per-user)
    permit ip any 197.80.1.0 0.0.0.255

(when maria dials in also we now see both access-lists, however they are applied to different virtual interfaces
```

Figure 7

© SANS Institute 2002, As part of the Information Security Reading Room.

Conclusion

Using TACACS+ to provide authentication and authorization and accounting gives network engineers an added layer of protection in securing networks. It allows access to network services to be regulated on a more granular basis. Remote users connecting to the network can be screened against the user database and a custom policy that controls not only what devices a user can access but also what services on a particular device that a user can access. If a users account is compromised that account can be disabled. Accounting provides an audit trail that can be used to track what services the user had access. As more workers begin to work remotely from home, TACACS+ can definitely make the task of administering user accounts and network services more efficient and secure. When Implementing TACACS+ remember that if the servers that are running the TACACS+ applications are compromised an attacker could have access to your organizations entire user/password database. The TACACS+ servers should not be running any other application; this minimizes the chance of the server being compromised through vulnerabilities in one of other applications. Communication with the TACACS+ servers should be limited to the devices (clients) that need to communicate with the server to perform authentication. In other words if a hacker were to gain access to an internal device, he should not be able to connect to the TACACS+ server from such device. This should be enforced by having appropriate filters (access control lists) on the routers and also by applying additional security measures to the server such as TCP wrappers.

Reference:

Cisco press, “Cisco IOS 12.0 Network Security”

“TACACS+ and RADIUS Comparison”
http://www.cisco.com/warp/public/480/10.html

request for comment 2138, “Remote Authentication Dial In User Service (RADIUS)”
URL: http://www.ietf.org/rfc/rfc2138.txt

“TACACS+ Authentication for HTTP Server Users”
http://www.cisco.com/warp/public/480/http-2.html

“Per-User Configuration”
http://www.cisco.com/univercd/cc/td/doc/product/software/ios120/12cgcr/dial_c/dcperusr.htm