Interested in learning more about security?

SANS Institute
InfoSec Reading Room

This paper is from the SANS Institute Reading Room site. Reposting is not permitted without express written permission.

How to Install IC Radius and Extend via Custom Perl Script

Efficient, scalable multi-platform authentication poses many problems to all systems administrators. RADIUS is a flexible authentication protocol that can be used to centralise authentication. In this HOW TO I will investigate how for a typical company you can install and extend a freely available radius server. In addition, detailed steps also show how the extended radius server can be configured to authenticate a selection of different network elements.

Copyright SANS Institute

Author Retains Full Rights

AD

Michael T Meacle
GSEC Version 1.3 (amended December 12,2001)
.

Abstract 1
Setting The Scene 2
What is Radius 3
Installing Linux RH 7.2 4
Installing IC Radius 5
PERL Installation: 6
Berkley Sockets Installation: 6
MySQL Installation: 7
Securing MySQL (root account): 7
Securing MySQL (IC Radius specific accounts): 9
Install Data-Dumper: 10
Install Data-ShowTable: 10
Installing DBI: 11
Installing DBD: 11
Installing SNMP: 11
IC-Radius Installation: 12
Create IC-Radius Database: 13
Create IC-Radius Database Standard Tables: 13
Load the Dictionary File(s): 14
Enabling the CGI scripts: 14
Enabling the Perl test script: 15
Extending IC Radius 16
Basics of Radius Authentication: 17
Basics of IC Radius Authentication: 19
IC Radius Interacting with External Programs: 20
Understanding how radiusAVs.pl works: 21
Extending IC Radius with radiusAVs.pl: 23
Adding IC Radius records for zzz.org: 23
Using Test Script: 27
Using with Squid: 27
Using with BAY routers: 29
Using with 3Com routers: 30
Additional Considerations 31
Summary 32
References 32
Appendix 36
Appendix 1: RadiusAVs.pl (IC Radius Extension Script) 36
Appendix 2: Additional Custom Tables 45
Appendix 3: Bay Dictionary 47

© SANS Institute 2002, Author retains full rights.

Abstract

Efficient, scalable multi-platform authentication poses many problems to all systems administrators. RADIUS is a flexible authentication protocol that can be used to centralise authentication. In this HOWTO I will investigate how for a typical company you can install and extend a freely available radius server. In addition, detailed steps also show how the extended radius server can be configured to authenticate a selection of different network elements.

© SANS Institute 2002, As part of the Information Security Reading Room. Author retains full rights.

Setting The Scene

The problem of multi platform scalable, flexible authentication with a audit trial confronts every system administrator. To illustrate I will use an example, a small enterprising service provider "ZZZ Gadgets". Recently you as the head systems administrator provided a detailed report to management highlighting your concern that all staff had access to all systems and network elements using a common shared password. Management after reviewing your report have given permission to design and implement a scalable, centralised authentication system.

After categorizing all the equipment (see Table 1) identified in the audit, by using the "Principle of Least Privilege"[1], you have established the following as the initial permissions based on your company's new security policy. There are 110 employees in ZZZ Gadgets, the following is a summary based on one individual from each workgroup identified.

- mick (initial password of mickPW)
- Typical experienced 3Com router specialist, providing Bay router assistance
  - access to WEB via Squid
  - access to 3Com routers as manager
  - access to Bay routers as user

- trish (initial password of trishPW)
- Typical experienced BAY router specialist
  - access to WEB via Squid
  - access to Bay routers as manager

- luke (initial password of lukePW)
- Typical level 1, Data Communications employee
  - access to 3Com routers as user
  - access to Bay routers as user

- jacqui (initial password of jacquiPW)
- Typical administration staff employee
  - access to web

Hostname	IP Address	Purpose
Squid.zzz.org	10.1.1.8	Squid server with internet access
Bay1.zzz.org	10.1.2.1	Bay router
Bay2.zzz.org	10.1.2.2	Bay router
Threecom1.zzz.org	10.1.3.1	3Com router
Threecom2.zzz.org	10.1.3.2	3Com router
WsX.zzz.org	10.1.4.x	Users workstation x

Table1: Summary of core equipment used with zzz.org

After careful consideration of the types of Network Elements (NE) in use along with various authentication protocols available it was decided to implement a RADIUS solution. It was decided to initially implement a single server "radius.zzz.org" with a ip address of 10.1.1.1.

© SANS Institute 2002, As part of the Information Security Reading Room. Author retains full rights.

Some additional accounts have also been identified in the planning stage that will be used to administrator the central authentication server. These are sample accounts only; all administrators will have there own account so that changes can be audited.

- mysql application root account "root"
  - demonstrate setting and using initial password to "rootPW"

- mysql client read write administrator account "radiusRW"
  - demonstrate setting and using initial password to "radRWpw"

- mysql client read only administrator account "radiusRO"
  - demonstrate setting and using initial password to "radROpw"

- radius web cgi administration tool, "radWWW"
  - demonstrate setting and using initial password to "WWWpw"

What is Radius

RADIUS is a acronym for Remote Authentication Dial In User Service (RADIUS) as currently defined in rfc2865 [2]. Radius is a lightweight client server protocol used to transport authentication, authorization and configuration information on one udp port while accounting records can be transported on a second udp port. Radius originally developed by Livingston Enterprises is now owned by Lucent Technologies [3].

Radius protocol bonds together three primary cornerstones of user access control:

- Authentication: identifying and verifying who the user really is,
- Authorization : determining what a particular user is allowed to do,
- Accounting: record that a user done something.

Both authentication and authorization are done at the same time, this contrast with TACACS+ [4] where all three facets are handled separately.

Some of the key features of radius, as outlined in rfc2865 [2], are what makes radius so powerful:

- Client/Server:
  - provides a scalable centralized security solution.
- Network Security:
  - A shared secret known only by the server and the client are used to encrypt passwords being sent between client and server.
  - In addition message digest algorithms are also used to prevent reply attacks
- Flexible Authentication Mechanisms:
  - This allows each installation to match the actual authentication scheme to the required level of security. For example, is username/password sufficient or is a One-Time-Password mechanism required.
- Extensible Protocol:
  - Each transaction is self-defining consisting of a variable number of Attribute/Length/Value tuples. This allows a vendor, hopefully in a controlled manner, to define their own extensions.

© SANS Institute 2002, As part of the Information Security Reading Room. Author retains full rights.

As a result of these features, historically the radius protocol has been supported by a wide range of equipment manufactures. In addition there are a large variety of both commercial and non-commercial radius servers that can be used to leverage off this protocol. Widespread support by vendors is a stark contrast to the propriety and in some ways superior Cisco TACACS+ [5].

The radius protocol as originally defined by Livingston in rfc2058 [6] used UDP ports 1645 for authentication and port 1646 for accounting information. Unfortunately these ports had already been officially assigned to other parties: 1645 to Datametrics, and 1646 to samsg-port. Shortly after in a revised rfc2138 [7] the ports were redefined to the now officially registered udp ports of 1812 and 1813. We will use the new official ports of 1812/1813 within this document. A simplified explanation of the operation of the radius protocol is documented later. See "Basics of Radius Authentication".

Installing Linux RH 7.2

As this server is going to have a pivotal role in all authentication's, I suggest you start with a fresh minimal install of Redhat Linux 7.2. This will maximize your chance of having a clean secure authentication server. I also further suggest that this server be dedicated to authentication. This step is optional if you already have a suitable server to install IC Radius.

DISCLAIMER: Some people have had more wives than I have built Linux servers, thus this is by no means the best way to build a server but merely a example. Feel free to use your own preferred and proven method.

Some of the major steps during the fresh install

- Select a graphical install
- Select a custom install
- Use GRUB with a password
- When selecting Firewall Configuration select "HIGH" and "Custom"
  - Only allow 'ssh', and other ports of '1812:udp, 1813:udp'
- Make sure on the "Authentication and Configuration" screen you select
  - Enable MD5 passwords
  - Enable shadow passwords
- On "Package Group Selection" check "select individual Packages"
  - Be as aggressive as you can and deselect as much as possible "remember the less you install the less hardening required and the fewer vulnerabilities you will have"
  - You can always add a packages as required latter "so if in doubt leave it out"
- If any conflicts arise with the selected packages allow the installation program to make your combination consistent
- After rebooting we will do the initial hardening.
  - This is absolute minimalist approach; for further detail please refer to any / all of the following
    - http://rr.sans.org/linux/sec_install.php [8]
    - http://www.enteract.com/~lspitz/linux.html [9]
    - http://www.bastille-linux.org/ [10]

© SANS Institute 2002, As part of the Information Security Reading Room. Author retains full rights.

- http://rr.sans.org/linux/bastille.php [11]
- Ensure that only essential ports are open
  - netstat –na
- Ensure that only the services that you want running at the various runlevels are enabled how you want them.
  - chkconfig –list
  - obviously I was not aggressive enough in de-selecting packages during the initial install as I had to explicitly disable some services using the following commands. Removal would be advised.
    - chkconfig –level 2345 sendmail off
    - chkconfig –level 2345 lpd off
    - chkconfig –level 2345 xfs off
    - chkconfig –level 2345 isdn off
    - chkconfig –level 345 netfs off
    - chkconfig –level 345 portmap off
    - chkconfig –level 345 nfslock off
  - you may wish to explicitly turn off sgi_fam
    - edit /etc/xinetd.d/sgi_fam
    - add 'disable = yes' just prior to the closing brace
- reboot, revalidate active ports
  - shutdown –r now
  - netstat –na
    - in my case only TCP 22 was open

Installing IC Radius

We will install the current version of IC Radius [12]. This is a free radius server based on Cistron Radius. It provides very similar functionally as the original Livingston implementation except that it uses a MySQL relational database to store configuration data. If you already have a fully functional copy of IC Radius feel free to skip to "Extending IC Radius".

WARNING: Recently CERT published two vulnerabilities [13]. Unfortunately IC Radius is vulnerable to both. As yet there appears no patch for IC Radius. Please re-check for a vendor patch before proceeding.

If you do skip forward please read, understand and allow for the material covered in

- Securing MySQL (root account)
  - only step 1 is covered to any extent in the README [15],
  - steps 2 and 3 are new and essential in my opinion
- Securing MySQL (IC Radius specific accounts)
  - also not covered at all in the README [15]

First obtain the latest from ftp://ftp.innercite.com/pub/icradius [14].
I used 'icradius-0.18.1.tar.gz' which is the current version.

© SANS Institute 2002, As part of the Information Security Reading Room. Author retains full rights.

I suggest you extract and read the very extensive and helpful README [15] file contained within the zipped tarball prior to starting. It is newer but very similar to http://radius.innercite.com/ICRADIUS.README.html [16] which is currently version 0.17b.

- tar –xzvOf icradius-0.18.1.tar.gz icradius-0.18.1/doc/README > /tmp/README

Unlike the README [15] file I am going to use RPM's wherever possible. I prefer to do this as it facilitates easier maintenance in my opinion. RPM's love them!

The README [15] is very well laid out thus I am going to use it as a basic structure of the major steps during the initial install of IC Radius. This will allow anyone who is not using Redhat to easily keep in sync. Essentially if a RPM is used very few of the steps contained with the README [15] are required. I suggest you fully read both this HOWTO and the README [15] prior to beginning.

WARNING: When I refer to package ABC-4.6.8-4 I'm referring to the exact major and minor version number as found on my installation. You should wherever possible match or better the version. And before going live check all products for any updates especially of a security / stability nature. http://www.redhat.com/apps/support/errata/ [17]

PERL Installation:

You should have installed it during the initial build. You can verify this by using either of the following methods.

- rpm –qa | grep –i perl
  - look for something similar to "perl-5.6.0-17"
- perl –v

If it is not installed you can install it from the second cdrom by doing the following

- Insert the 2nd cdrom into your cdrom drive
- Mount it by using
  - mount /dev/cdrom /mnt/cdrom
- Install software
  - rpm –ivv /mnt/cdrom/Redhat/RPMS/perl-5.6.0-17.src.rpm

Berkley Sockets Installation:

You should have installed them during the initial build, you can verify this by using the following command.

- rpm –qa db*
- look for the following
  - db1-1.85-7
  - db2-2.4.14-7
  - db3-3.2.9-4

If they are not installed you can install them from the 2nd cdrom by doing the following

- Insert the 2nd cdrom into your cdrom drive
- Mount it by using
  - mount /dev/cdrom /mnt/cdrom

© SANS Institute 2002, As part of the Information Security Reading Room. Author retains full rights.

- Install software
  - rpm –ivv /mnt/cdrom/Redhat/RPMS/db1-1.85-7.rpm
  - rpm –ivv /mnt/cdrom/Redhat/RPMS/db2-2.4.14-7rpm
  - rpm –ivv /mnt/cdrom/Redhat/RPMS/db3-3.2.9-4.rpm

MySQL Installation:

You should have installed it during the initial build, you can verify this by using the following command.

- rpm –qa | grep –i mysql
- look for the following
  - mysql-3.23.41-1
  - mysql-server-3.23.41-1
  - mysql-devel-3.23.41-1
  - mysqlclient9-3.23.41-1

If they are not installed you can install them from the 1st cdrom by doing the following

- Insert the 1st cdrom into your cdrom drive
- Mount it by using
  - mount /dev/cdrom /mnt/cdrom
- Install software (order is important)
  - rpm –ivv /mnt/cdrom/Redhat/RPMS/mysql-3.23.41-1.rpm
  - rpm –ivv /mnt/cdrom/Redhat/RPMS/mysql-server-3.23.41-1.rpm
  - rpm –ivv /mnt/cdrom/Redhat/RPMS/mysql-devel-3.23.41-1.rpm
  - rpm –ivv /mnt/cdrom/Redhat/RPMS/mysqlclient9-3.23.22-6.rpm
- Secure the server's configuration file as the RPM doesn't do it
  - chmod 600 /etc/my.cnf

Please note some differences between a rpm install and source install:

- step 7 in the notes would result in mysql being installed in /usr/local/mysql; if you use the rpm it is installed in /usr/bin.
- step 13 would also be
  - chmod 755 /etc/rc.d/init.d/mysqld
- step 14 should also be
  - chown –R root /usr/bin/mysql
- if using the source install please do not do step 21-24 as I have extended them. For more detail refer to two securing MySQL sections.

Securing MySQL (root account):

I will only cover some basic securing issues. More can be found on the Internet and a good starting point is the MySQL online documentation itself. In particular,

- http://www.mysql.com/doc/C/o/Connecting.html [18],
- http://www.mysql.com/doc/P/r/Privileges.html [19]

© SANS Institute 2002, As part of the Information Security Reading Room. Author retains full rights.

Firstly a default install has a 'root' account without a password. There are a couple of ways to change it, the following is but one extracted from README [15].

- Start the MySQL client in interactive mode pre-selecting the 'mysql' database
  - mysql mysql
  - update user set password = Password("rootPW") where user = "root";
  - flush privileges;
  - exit;
- Please note from now on when you start the mysql client you have to explicitly tell it that the password is not null, you have three basic choices
  - Start the client providing the password on the command line
    - mysql –prootPW other_options_go_here
      - note: there is no space after the –p
      - note: this is the least preferred method due to 'ps –waux' allowing anyone to learn your password, its use should be discouraged
  - Start the client and get it to prompt for the password
    - mysql –p other_options_go_here
      - note the whitespace after the –p option
  - Create a MySQL configuration file in the users home directory
    - ~/.my.cnf
      - [client]
      - user=root
      - password=rootPW
    - minimize access to configuration file
      - chmod 600 ~/.my.cnf
    - For more details see http://www.mysql.com/doc/O/p/Option_files.html [20]

Secondly a default install also allows for anonymous accounts. You can delete the anonymous accounts by doing the following.

- Start the MySQL client in interactive mode pre-selecting the 'mysql' database, and prompting for password.
  - mysql –p mysql
  - delete * from user where user = "";
  - flush privileges;
  - exit;

Third a default install has some default "test" user accounts without a password. You should delete them as follows.

- Start the MySQL client in interactive mode pre-selecting the 'mysql' database, and prompting for password.
  - mysql –p mysql
  - delete from db where Host = "%" and Db like "test%";
  - flush privileges;
  - exit;

© SANS Institute 2002, As part of the Information Security Reading Room. Author retains full rights.

Fourth optional step is to minimize DNS poising exposure. As this is a server and consequently its IP address should very rarely change you can hard code any host domain address to its IP addresses.

- Start the MySQL client in interactive mode pre-selecting the 'mysql' database
  - mysql –p mysql
  - update user set Host="10.1.1.1" where Host = "radius.zzz.org";
  - flush privileges;
  - exit;

Securing MySQL (IC Radius specific accounts):

The 'root' account has full access to the whole MySQL application and ALL the databases it contains. Thus its id/password is very powerful; access to it needs to be very limited. Using the "Principle of least Privilege" [1] I suggest we create at least two additional accounts which should be used at all times to administer the 'radius' database.

The first additional account "radiusRW" will have full "Read/Write" access to the 'radius' database however it will not be able to grant any of its rights to anyone else. In addition it can only be used from the localhost itself to access the database locally. I see no reason to allow network access to this account. Because mysql clients can access the local host using either 'localhost' or it local IP address (10.1.1.1) you will see two entries.

- Start the MySQL client in interactive mode pre-selecting the 'mysql' database
  - mysql mysql -p
  - grant select,insert,update,delete,create
    on radius.*
    to radiusRW@localhost
    identified by 'radRWpw';
  - grant select,insert,update,delete,create
    on radius.*
    to radiusRW@10.1.1.1
    identified by 'radRWpw';
  - exit;
- This account will typically be used to do interactive maintenance of the 'radius' database
- You should use this as a starting template of all RW 'administrative' accounts for the 'radius' database

The second additional account "radiusRO" will have "Read Only" access to the 'radius' database and similarly will not be able to grant any of its rights to anyone else. In addition it can only be used from the localhost itself to access the database locally. I see no reason to allow network access to this account. Because mysql clients can access the local host using either 'localhost' or it local IP address (10.1.1.1) you will see two entries.

- Start the MySQL client in interactive mode pre-selecting the 'mysql' database
  - mysql mysql -p
  - grant select,
    on radius.*

© SANS Institute 2002, As part of the Information Security Reading Room. Author retains full rights.

to radiusRO@localhost
identified by 'radROpw';
- grant select,
  on radius.*
  to radiusRO@10.1.1.1
  identified by 'radROpw';
- exit;

- This account will typically be used to do "Read Only" interactive maintenance of the 'radius' database
- You should use this as a starting template of all RO 'administrative' accounts for the 'radius' database

Install Data-Dumper:

You should have installed it during the initial build, you can verify this by using the following commands.

- /etc/cron.daily/slocate
- locate DataDumper.pm

If it was not installed you can install them from the 2nd cdrom by doing the following

- Insert the 2nd cdrom into your cdrom drive
- Mount it by using
  - mount /dev/cdrom /mnt/cdrom
- Install software
  - rpm –ivv /mnt/cdrom/Redhat/RPMS/perl-5.6.0-17.src.rpm

Install Data-ShowTable:

This is not part of Redhat and can optionally be installed as follows.

- Obtain the source zipped tarball from search.cpan.org.
  - http://www.cpan.org/authors/id/AKSTE/Data-ShowTable-3.3.tar.gz [21]
- copy zipped tarball into /usr/local/src
  - cp /tmp/Data-ShowTable-3.3.tar.gz /usr/local/src/
- Extract, Configure, Build, Install
  - cd /usr/local/src
  - tar –xzvf Data-ShowTable-3.3.tar.gz
  - cd Data-ShowTable-3.3
  - make Makefile.PL
  - Note: There is a bug in ShowTable.pm line 724, two missing ">"
    - Change form / to
      - [, I<\@title_formats [, I<\@data_formats [, I<$table_attrs>]]]]]]];
      - [, I<\@title_formats> [, I<\@data_formats> [, I<$table_attrs>]]]]]]];
    - the README [15] highlights this error unfortunately they indicate the error is in the Makefile when it is in fact in ShowTable.pm
  - make
  - make test
  - make install

© SANS Institute 2002, As part of the Information Security Reading Room. Author retains full rights.

Installing DBI:

You should have installed them during the initial build, you can verify this by using the following command.

- rpm –qa | grep –i DBI
- look for the following
  - perl-DBI-1.18-1

If it is not installed you can install them from the 2nd cdrom by doing the following

- Insert the 2nd cdrom into your cdrom drive
- Mount it by using
  - mount /dev/cdrom /mnt/cdrom
- Install software
  - rpm –ivv /mnt/cdrom/Redhat/RPMS/perl-DBI-1.18-1.rpm

Installing DBD:

You should have installed them during the initial build, you can verify this by using the following command.

- rpm –qa | grep –i DBD
- look for the following
  - perl-DBD-MySQL-1.2216-4

If it is not installed you can install them from the 1st cdrom by doing the following

- Insert the 1st cdrom into your cdrom drive
- Mount it by using
  - mount /dev/cdrom /mnt/cdrom
- Install software
  - rpm –ivv /mnt/cdrom/Redhat/RPMS/perl-DBD-MySQL-1.2216-4.rpm

Installing SNMP:

This is optional. It is typically used by IC Radius to verify if a user is still logged in if his/her account has a maximum connection count limit. With the recent vulnerabilities exposed within SNMP [22] you should seriously consider the benefits of such a feature. Certainly our mock company zzz.org doesn't use or need this feature.

That aside you could have installed them during the initial build, you can verify this by using the following command.

- rpm –qa | grep –i snmp
- look for the following
  - ucd-snmp-4.2.1-7.rpm
  - ucd-snmp-utils-4.2.1-7.rpm

If they are not installed you can install them from the 2nd cdrom by doing the following

- Insert the 2nd cdrom into your cdrom drive
- Mount it by using
  - mount /dev/cdrom /mnt/cdrom

© SANS Institute 2002, As part of the Information Security Reading Room. Author retains full rights.

- Install software
  - rpm –ivv /mnt/cdrom/Redhat/RPMS/ucd-snmp-4.2.1-7.rpm
  - rpm –ivv /mnt/cdrom/Redhat/RPMS/ucd-snmp-utils-4.2.1-7.rpm

IC-Radius Installation:

Ensure that you have the latest source zipped tarball from ftp://ftp.innercite.com/pub/icradius/ [14].

- Copy zipped tarball into /usr/local/src
  - cp icradius-0.18.1.tar.gz .tar.gz /usr/local/src/
- Extract, Configure, Build, Install
  - cd /usr/local/src
  - tar –xzvf icradius-0.18.1.tar.gz
  - cd /usr/local/src/icradius-0.18.1/src/
- copy the relevant makefile
  - cp Makefile.lnx Makefile
  - Since we used a RPM to install MySQL we need to update the common Makefile
    - vi /usr/local/src/icradius-0.18.1/Makefile
      near top change from / to
      - LDFLAGS = -L /usr/local/lib/mysql
      - LDFLAGS = -L/usr/lib/mysql
- If you are going to authenticate a user against the OS then I suggest you enable pam support. This gives you a lot of flexibility on actual authentication mechanism. For example you could configure PAM to use a OTP mechanisms such as cryptocard.
  - vi /usr/local/src/icradius-0.18.1/Makefile and enable the following 2 lines
    - PAM = -DPAM
    - PAMLIB = -lparm -ldl
  - cp /usr/local/src/icradius-01.8.1/redhat/radiusd-pam /etc/pam.d/radiusd
- make
- make install
- Copy startup script and enable automatic start on reboot
  - cp /usr/local/src/icradius-01.8.1/redhat/rc.radiusd-redhat /etc/rc.d/init.d/radiusd
  - chmod 500 /etc/rc.d/init.d/radiusd
  - chkconfig –add radiusd
- Secure radius server configuration file
  - chmod 600 /etc/raddb/radius.conf
- Copy and secure the WEB cgi files to your Apache WEB cgi directory
  - mkdir –p /var/www/cgi-bin/radius
  - cp /usr/local/src/icradius-0.18.1/scripts/*.cgi /var/www/cgi-bin/radius/
  - chown apache:apache /var/www/cgi-bin/radius/*.cgi
    - (where apache is the account your web server runs as)
  - mkdir /var/www/cgi-bin/radius/icons
  - cp /usr/local/src/icradius-0.18.1/scripts/icons/*.cgi /var/www/cgi-bin/radius/icons/
  - chown apache:apache /var/www/cgi-bin/radius/icons/*.gif
    - (where apache is the account your web server runs as)
- Create the ascii radius accounting directory

© SANS Institute 2002, As part of the Information Security Reading Room. Author retains full rights.

- mkdir /var/log/radacct
- Create a session file to be used with the web administration tool
  - mkdir /var/www/cgi-bin/radius/session
  - chown apache:apache /var/www/cgi-bin/radius/session
  - chmod 770 /var/www/cgi-bin/radius/session
  - touch /var/www/cgi-bin/radius/session/icradiusweb.session
  - touch /var/www/cgi-bin/radius/session/radsess
  - chown apache:apache /var/www/cgi-bin/radius/session/icradiusweb.session
  - chown apache:apache /var/www/cgi-bin/radius/session/radsess
- Install the automatic log rotation script
  - cp /usr/local/src/icradius-01.8.1/redhat/radiusd-logrotate /etc/logrotate.d/radiusd
- Install the relevant support scripts
  - mkdir /usr/local/sbin/radius
  - chmod 700 /usr/local/sbin/radius
  - cp /usr/local/src/icradius-01.8.1/scripts/acctexport.pl /usr/local/sbin/radius/
  - cp /usr/local/src/icradius-01.8.1/scripts/acctimport.pl /usr/local/sbin/radius/
  - cp /usr/local/src/icradius-01.8.1/scripts/acctsummarize.pl /usr/local/sbin/radius/
  - cp /usr/local/src/icradius-01.8.1/scripts/radius.db /usr/local/sbin/radius/
  - cp /usr/local/src/icradius-01.8.1/scripts/radiusfixup.pl /usr/local/sbin/radius/
  - cp /usr/local/src/icradius-01.8.1/scripts/radlast /usr/local/sbin/radius/
  - cp /usr/local/src/icradius-01.8.1/scripts/radwatch /usr/local/sbin/radius/
  - cp /usr/local/src/icradius-01.8.1/scripts/radwho /usr/local/sbin/radius/
  - cp /usr/local/src/icradius-01.8.1/scripts/synaccounting.pl /usr/local/sbin/radius/
  - cp /usr/local/src/icradius-01.8.1/scripts/testrad /usr/local/sbin/radius/
  - cp /usr/local/src/icradius-01.8.1/scripts/userexport.pl /usr/local/sbin/radius/
  - cp /usr/local/src/icradius-01.8.1/scripts/userimport.pl /usr/local/sbin/radius/
  - cp /usr/local/src/icradius-01.8.1/scripts/userlineconv.pl /usr/local/sbin/radius/
  - cp /usr/local/src/icradius-01.8.1/scripts/dictimport.pl /usr/local/sbin/radius/
  - chmod 700 /usr/local/sbin/radius/[a-z]*

Create IC-Radius Database:

A default install of IC Radius expects a database called ‘radius’. Using the mysql client create the database.

- Start client (assuming you done the steps to Secure MySQL)
  - mysql –u root -p
- Create the database
  - create database radius;
- Exit the client
  - exit;

Create IC-Radius Database Standard Tables:

A default install of IC Radius expects a number of tables to be contained within a database called ‘radius’. Using the mysql client create the tables from the predefined definition file.

© SANS Institute 2002, As part of the Information Security Reading Room. Author retains full rights.

- Build tables from command line piping in SQL commands
  - mysql –u root –p radius < /usr/local/sbin/radius/radius.db
- Test that it worked using MySQL client
  - echo 'show tables;' | mysql –u radiusRW –p radius

Load the Dictionary File(s):

Dictionary files are used by the server to decompose requests and construct reply packets in accordance with the radius rfc2865 [2]. Typically a vendor has their own attributes defined in a dictionary file. A selection of dictionaries came with the source zipped tarball and will have been extracted into /usr/local/src/icradius-0.18.1/raddb.

A particular vendor specific file is identified as "dictionary.VENDORNAME" for example "dictionary.cisco" .In addition to loading the base definition file "dictionary" you should also load any of the vendor specific file now by using the following steps.

- Change into the directory where the scripts are now stored
  - cd /usr/local/sbin/radius
- edit the script dictimport.pl
  - vi dictimport.pl
    - change $dbusername to "radiusRW" ie RW username as created in Securing MySQL.
    - change $dbpassword to "radRWpw" ie RW password as created in Securing MySQL.
- Load the base definition thus.
  - ./dictimport.pl /usr/local/src/icradius-0.18.1/raddb/dictionary
- Now load any vendor specific thus; zzz.org doesn't require any, you may
  - ./dictimport.pl /usr/local/src/icradius-0.18.1/raddb/<dictionary_name>

Enabling the CGI scripts:

Previously during IC Radius installation we installed two cgi scripts into /var/www/cgi-bin/radius. We will now configure these to allow easy administration of the radius server. This step is optional especially if you are going to extend the IC Radius installation as it only supports the default tables, no support is available for the new extended tables.

Install the RADIUS Perl module provided by Innercite.

- Download the latest from
  - ftp://ftp.innercite.com/pub/icradius/IC-Radius-0.4.tar.gz [23]
- copy it into local source
  - cp /tmp/IC-Radius-0.4.tar.gz /usr/local/source/
- extract, configure and install
  - cd /usr/local/src
  - tar –xzvf IC-Radius-0.4.tar.gz
  - cd /usr/local/src/IC-Radius-0.4
  - perl Makefile.PL
  - make
  - make install

© SANS Institute 2002, As part of the Information Security Reading Room. Author retains full rights.

Next we need to create a account, which will be used by the cgi scripts, within the radius database with the special attribute / value pair of “Radius-Operator” / “Yes”.

- If mysql server is not running start it by
  - /etc/rc.d/init.d/mysql start
- Start the mysql client , this time we will use the RW id/password created specifically for radius administration. This will verify that they are set up OK. The ‘-p’ will force the client to prompt you for the current RW password.
  - mysql –u radiusRW –p radius
  - Create a WEB administration account “radWWW” by entering the following SQL commands
    - insert into radcheck values (“”, “radWWW”, “Password”, password(“WWWpw”));
    - insert into radcheck values (“”, “radWWW”, “Radius-Operator”, “Yes”);
    - insert into radcheck values (“”, “radWWW”, “Auth-Type”, “Crypt-Local”);
    - exit;
- edit /var/www/cgi-bin/radius/radius.cgi updating the following
  - change ~ line 44 from / to
    - my $sessionsfile = "/usr/local/apache/cgi-bin/icradiusweb.sessions";
    - my $sessionsfile = "/var/www/cgi-bin/radius/sessions/icradiusweb.sessions";
  - change ~ line 62 updating radius id and radius pw, from / to
    - $radius->init('radius', 'radiuspass', 'localhost', $database);
    - $radius->init('radiusRW', 'radRWpw', 'localhost', $database);
- edit /var/www/cgi-bin/radius/usage.cgi updating the following
  - change $dbusername to "radiusRW" ie RW username as created in Securing MySQL.
  - set $dbpassword to "radRWpw" ie RW password as created in Securing MySQL.
  - change ~ line 34 from / to
    - my $session_file = "/var/lib/apache/cgi-bin/radsess";
    - my $session_file = "/var/www/cgi-bin/radius/sessions/radsess";
- Change $cookiedomain to a valid domain for where you are working
  - From experience I suggest you initially set it to “” e.g. null, and once it is working set it in accordance to your real domain. A additional suggestion in isolating a configuration error, especially if you are continually being re-prompted for id / password, is to enable prompting when a cookie is received within your browser.
- Manually run the both scripts to check for error/warnings
  - /var/www/cgi-bin/radius/radius.cgi
    - to abort press <ctrl>c
  - /var/www/cgi-bin/radius/usage.cgi
    - to abort press <ctrl>c

Enabling the Perl test script:

During the installation of IC Radius you installed a very powerful test tool testrad into /usr/local/sbin/radius/ directory. Essentially it is a perl script that simulates a NAS client. It allows thorough yet flexible testing of your IC Radius installation. It will become invaluable in

© SANS Institute 2002, As part of the Information Security Reading Room. Author retains full rights.

your toolbox in maintaining your radius server. It is optional, but I strongly suggest you install the additional components to enable it to work.

There are a number of steps required to get the script working including the installation of a couple of perl libraries from cpan.

Install a MD5 support library.

- Download the latest MD5 library from cpan
  - http://www.cpan.org/authors/id/GAAS/MD5-2.02.tar.gz [24]
- copy it into local source
  - cp /tmp/MD5-2.02.tar.gz /usr/local/source/
- extract, configure and install
  - cd /usr/local/src
  - tar –xzvf MD5-2.02.tar.gz
  - cd /usr/local/src/MD5-2.02
  - perl Makefile.PL
  - make
  - make install

Install a RadiusPerl support library.

- Download the latest RadiusPerl library from cpan
  - http://www.cpan.org/authors/id/CARL/RadiusPerl-0.05.tar.gz [25]
- copy it into local source
  - cp /tmp/RadiusPerl-0.05.tar.gz /usr/local/source/
- extract, configure and install
  - cd /usr/local/src
  - tar –xzvf RadiusPerl-0.05.tar.gz
  - cd /usr/local/src/RadiusPerl-0.05
  - perl Makefile.PL
  - make
  - make install

NOTE: for some reason the install doesn't actually install the Perl module nor does it error.

- You can however manually install it by doing the following.
  - mkdir /usr/lib/perl5/site_perl/5.6.0/Authen
  - cp /usr/local/src/RadiusPerl-0.05/Authen/Radius.pm /usr/lib/perl5/site_perl/5.6.0/Authen/
  - chmod 444 /usr/lib/perl5/site_perl/5.6.0/Authen/Radius.pm
  - chown root:root /usr/lib/perl5/site_perl/5.6.0/Authen/Radius.pm

If you wish the RadiusPerl module can be optionally replaced with the Radius module from Innercite. Instructions on how to install the suitable replacement is contained in the previous section, "Enabling the CGI scripts".

Extending IC Radius

This is the heart of this document, I hope you're still with me.

© SANS Institute 2002, Author retains full rights.

Thus far we have:

- installed a base installation of Linux,
- installed a working copy IC Radius,
- performed basic hardening of MySQL root account,
- performed basic hardening of MySQL IC Radius specific accounts.

Most of what has been covered thus far is covered in various forms including

- IC Radius README [15] from ftp://ftp.innercite.com/pub/icradius [14]
- Redhat installation notes http://www.redhat.com/support/resources/install_upgrade/installing_linux.html [26]
- MySQL helpfile http://www.mysql.com/documentation/mysql/full/ [27]

The installation at this stage would allow you to authenticate and authorize users via Radius however it would have some limitations. For example a user with access to a Bay router would not be able to be given access to 3Com devices. This is due to the fact both 3Com and Bay routers use individual and incompatible attributes during the authentication and authorization phase. That is, if you configure the Radius server to send both 3Com and Bay attributes for a particular user then their access is revoked by both routers. This is interesting as the rfc2865 [2] clearly states that either a client or server can ignore attributes of unknown type.

"A RADIUS client MAY ignore Attributes with a unknown Type" rfc2865 [2] section 5. Interestingly the Squid authenticator discussed shortly doesn't care how many additional attributes are sent to it, all it considers is the type of response, e.g. is it '2' Access-Accept .

The original Livingston radius implementation provides a possible solution to this problem using prefix and suffixes[28]. I have found this to be both confusing and un-scalable, not to mention confusing to the average end user.

Wouldn't it be nice to be able to extend and customize IC Radius to dynamically create the correct attributes based on intersection of user, technology type and access level allowed.

Basics of Radius Authentication:

IC Radius does provide some hooks, which allow us to do this, however lets first refresh our minds on the basic operation of Radius.

Figure 1 represents diagrammatically the simplified sequence of radius authentication as described in rfc2865[2]. The Radius Client is any Network element, which has been configured to authenticate its users by a central radius server. Examples include 3Com and Bay routers, Squid proxy with a radius authentication plugin.

A brief explanation of the phases depicted in figure 1 are as follows.

- user initially establishes a session with a radius client, for example telnets to a Bay router.
- Radius client (router) prompts the user for his/her id.
- user enters his/her id.
- Radius client (router) then prompts the user for his/her password.
- User enter his/her password

© SANS Institute 2002, As part of the Information Security Reading Room. Author retains full rights.

- Radius client constructs a "Access-Request" in accordance to rfc2865 [2] and sends it to the first registered Radius server
- Radius server depending on the username / password and where the request originated from will determine one of the following
  - Access-Deny : refuse connection to user
  - Access-Accept : to accept the user
  - Access-Challenge : if the Radius server needs more information from the user

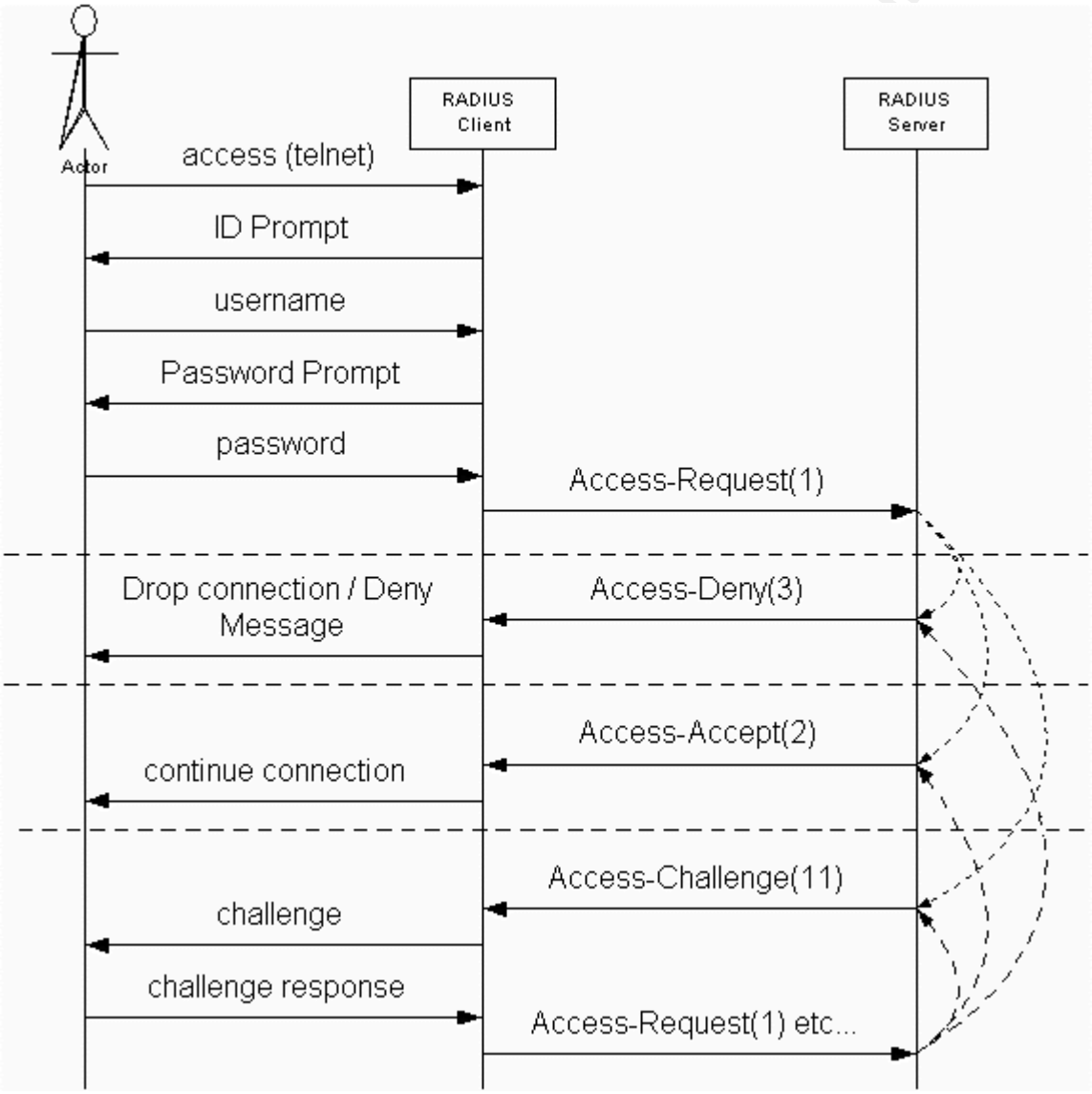

Figure 1: Simplified Sequence diagram of Radius authentication

© SANS Institute 2002, As part of the Information Security Reading Room. Author retains full rights.

Basics of IC Radius Authentication:

Once the radius server receives an "Access-Request" it has to be broken apart and checked against conditions defined within IC Radius's MySQL database. Figure 2 is a simplified representation of how IC Radius performs these tasks. It should be understood it is not intended to fully explain how IC Radius performs all its tasks but merely a simplified way of representing the minimum detail required by you to understand and customize the perl extension script.

The diagram has been built up from:

- What the rfc2865 [2] says it should do,
- Selected tests against a installation of IC Radius,
- Reading README [15] from Innercite,
- Reading searchable online FAQ http://radius.innercite.com/cgi-bin/faqdb.cgi [29],
- Reading some of the source code /usr/local/src/icradius-0.18.1/src/

A brief explanation of the main stages of IC Radius Authentication are depicted in Figure 2.

1) Radius client construct a "Access-Request" and sends it to the Radius Server
2) IC Radius verifies whether it has a "secret" associated with the source address of the radius packet.
3) No corresponding NAS record could be found
   - silently discards the request as per rfc2865 [2]
   - logs it in radius.log
4) extracts the password based on the "secret" associated with radius source address
5) process the "radgroupcheck" and "radcheck" conditions, in particular checking the previously decoded password against the associated username name.
6) If a challenge is required it will be constructed and returned to the user
7) Radius client, extracts the challenge and sends it to the user
8) Has any of the check values failed yet, for example "Simultaneous-Use" exceeded
9) Send "Access-Denied" if any conditions fail
10) All the initial conditions, having been meet, determine the reply attributes.
    - NOTE: this is the phase we plug our customization script into
    - Access can still be denied at this phase using the exit code of external program(s)
11) Has access been denied due to external programs
12) Send "Access-Denied" as some condition has failed
13) Send "Access-Accept" and any associated attributes/value pairs.

© SANS Institute 2002, As part of the Information Security Reading Room. Author retains full rights.

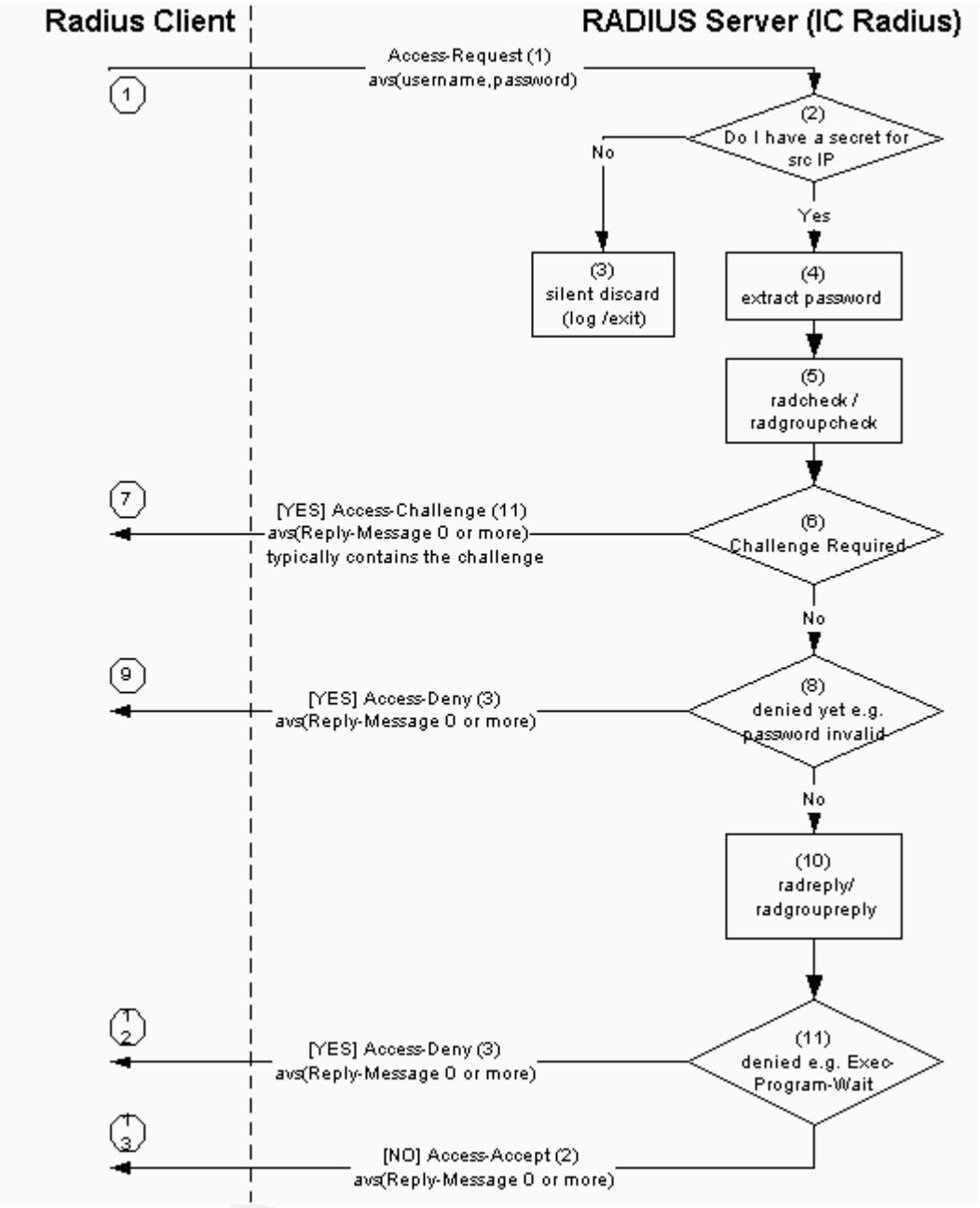

Figure 2: Simplified Stages of IC Radius Authentication.

IC Radius Interacting with External Programs:

As indicated in step 10 of the previous section you can customize the radius server to dynamically create reply attributes during the "radreply/radgroupreply" stage. This can be done by invoking one or more external programs written in whatever language you wish.

All that is required is the external program return the following

- Deny the user
  - exit code not 0

© SANS Institute 2002, As part of the Information Security Reading Room. Author retains full rights.

- Permit the user
  - exit code 0
  - List of attributes/values pairs out STDOUT

The README [15] documents the following attributes which can be passed to external programs. Please refer to README [15] for more detailed examples and explanations.

- Taken from the original request
  - %p port number
  - %n nas IP address
  - %u user name
  - %a Protocol (SLIP/PPP)
  - %s Speed (connect string – e.g. 28800/V42.BIS)
  - %i calling station ID
- Taken from the reply as defined thus far:
  - %f Framed IP address
  - %c Callback-Number
  - %t MTU

Please note "%n" is actually a attribute in the original request it is not necessarily the same as the radius request source address but in practice generally is. This subtlety allows thorough yet flexible testing of the radius server using the previously installed test program.

The external program "radiusAVs.pl" is only interested in two values "%n" and "%u". By using these two values and some additional tables, we conveniently store in the "RADIUS" database, we can determine dynamically attribute/value pairs to return on a user by user basis.

Understanding how radiusAVs.pl works:

The perl program 'radiusAVs.pl' is a custom program I have developed to allow the dynamic creation of custom attribute / value pairs based on,

- username,
- NAS type,
- Level the user is allowed.

A brief explanation of the script (contained in Appendix 1)

1) a radreply/radgroupreply is configured to spawn the external script for a particular user, it will pass two command line arguments the "NAS IP address" and the username.
2) Initially determine some details of the NAS client based on NAS IP address
3) Determine the maximum level for this user based on the intersection of his/her username and groups of elements they are a member of.
4) Is the maximum level attainable already been achieved
5) Do you wish to use a "slow" but sometimes more flexible Regular Expression match to determine a level for this user
   - NOTE: Regular Expression was how version 1 of the script worked. It is no longer required as version 2 prefers to use the faster and easier to understand group membership

© SANS Institute 2002, As part of the Information Security Reading Room. Author retains full rights.

approach. I have left the RE section in the script for completeness, it can be removed or disabled by the variable 'skipREmatch'.

6) Determine level using a RE intersection of username and full NAS name
7) Sanity check the maximum the user level determined during all previous phases
8) Err on the side of Deny if a illegal level was determined
9) Determine all dynamic Attribute value pairs based on maximum level and NAS type
10) Indicate success, and return Attribute / Value pair(s) via STDOUT

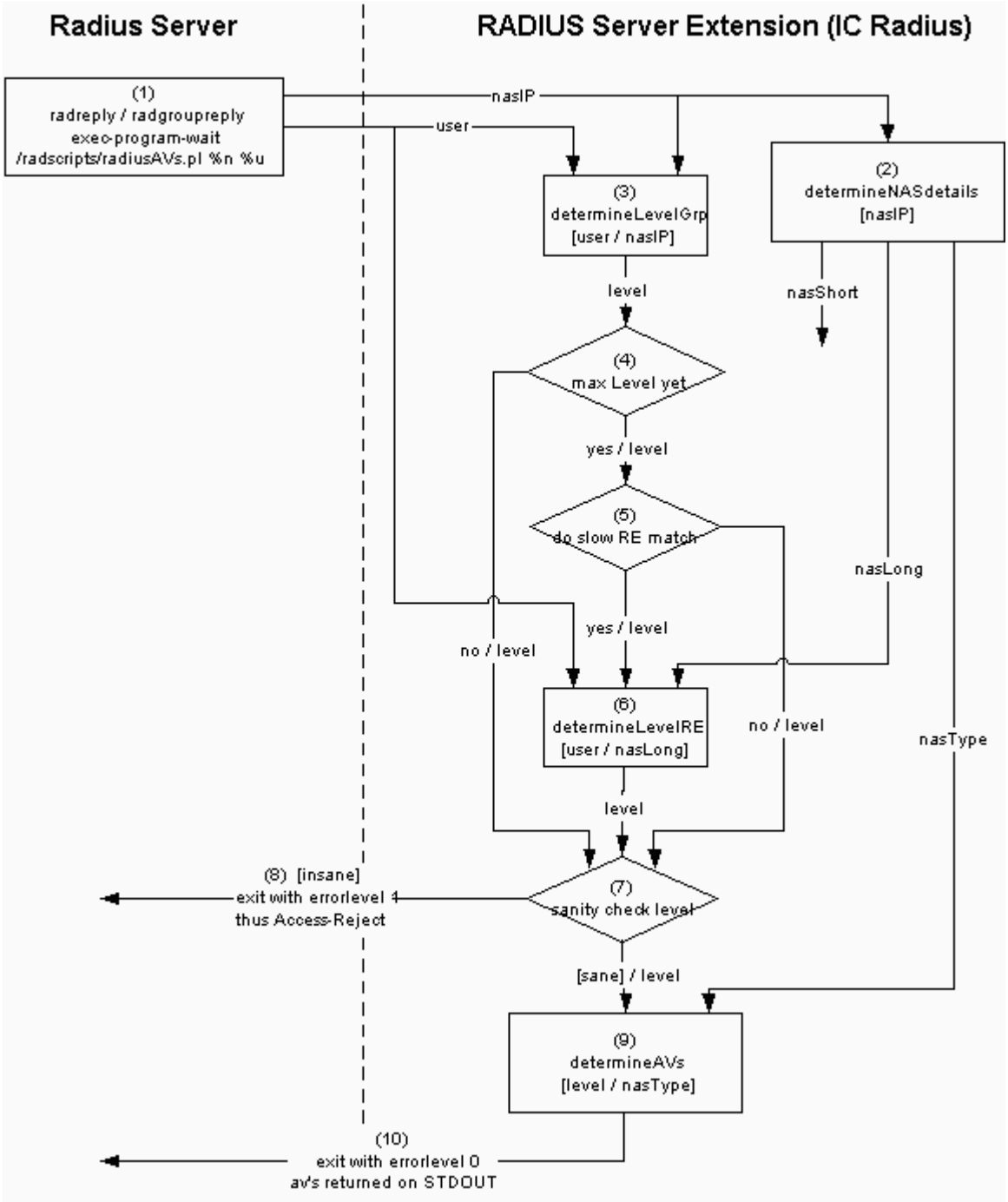

Figure 3: Basic Flowchart of radiusAVs.pl.

© SANS Institute 2002, As part of the Information Security Reading Room. Author retains full rights.

Extending IC Radius with radiusAVs.pl:

- Install the external script
  - mkdir /radscripts
  - chmod 700 /radscripts
  - chown radius:radius /radscripts
  - save a copy of it from appendix 1 to /radscripts/radiusAVs.pl
  - chmod 700 /radscripts/radiusAVs.pl
  - chown radius:radius /radiusAVs.pl
- Make the support tables
  - Save a copy of appendix 2 to /tmp/tables
  - cat /tmp/tables | mysql -p –u radiusRW radius
  - Check that the tables were created
    - echo 'show tables;' | mysql –u radiusRW –p radius
    - look for the following new tables
      - grp
      - nasXgrp
      - typeXlevel2avs
      - userXgrp2level
      - userXnas2level

A brief explanation of the primary purpose of these additional tables:

- grp
  - used to store comments about each group
  - optional table which can help in maintenance
- nasXgrp
  - used to associate a NAS ip's to any number of groups
  - a more meaningful name of the group can be found in the table "grp"
- typeXlevel2avs
  - the final stage, given the type of nas, level of a user, lookup required av's
- userXgrp2level
  - used to determine a user's level based on group membership and NAS groupings
- userXnas2level
  - used to determine a user's level of access based on their username and NAS long hostname,
  - used in the optional slow Regular expression level determination phase.

I will not discuss the script any further as I have heavily commented it.

Adding IC Radius records for zzz.org:

The README [15] has a detailed section with examples explaining how check and reply records can be assigned to a user. Please refer to it to obtain a greater understanding of how various records interact. I will however focus on adding records required for 'zzz.org' organization and

© SANS Institute 2002, As part of the Information Security Reading Room. Author retains full rights.

in particular calling the external program 'radiusAVs.pl'. Essentially it can be broken into two distinct phases: Adding a new NAS, Adding a User.

- Adding NAS's
  - For each NAS we need to create a record in the NAS table
  - Start mysql interactively and add the following; based on zzz.org
    - mysql –p radiusRW radius
      - insert into nas values ('', 'radius.zzz.org', 'radius', '10.1.1.1', 'livingston', '0', 'secret8', 'public', 'off');
      - insert into nas values ('', 'squid.zzz.org', 'squid', '10.1.1.8', 'squid', '0', 'secret8', 'public', 'off');
      - insert into nas values ('', 'bay1.zzz.org', 'bay1', '10.1.2.2', 'bay', '0', 'secret8', 'public', 'off');
      - insert into nas values ('', 'bay2.zzz.org', 'bay2', '10.1.2.2', 'bay', '0', 'secret8', 'public', 'off');
      - insert into nas values ('', 'threecom1.zzz.org', 'threecom1', '10.1.3.1', '3com', '0', 'secret8', 'public', 'off');
      - insert into nas values ('', 'threecom2.zzz.org', 'threecom2', '10.3.2', '3com', '0', 'secret8', 'public', 'off');
      - exit;
    - details extracted from table 1
    - NOTE: whenever you make changes to your NAS table you must restart the radius server for the changes to take effect
      - /etc/rc.d/init.d/radiusd restart

  - For each NAS ensure we have necessary Groups defined in 'grp' table
  - Start mysql interactively and add the following; based on zzz.org
    - mysql –p radiusRW radius
      - insert into grp values ('', '0', 'UNKNOWN', 'For any device unassigned to groups');
      - insert into grp values ('', '1', 'ALL', 'All devices are a member of this group');
      - insert into grp values ('', '2', 'BAY', 'All BAY router are a member of this group');
      - insert into grp values ('', '3', '3COM', 'All 3Com router are a member of this group');
      - insert into grp values ('', '4', 'WWW', 'All Squid Proxies are a member of this group');
      - exit;

  - For each NAS associate it with its appropriate groups as defined in the 'grp' table
  - Start mysql interactively and add the following; based on zzz.org
    - mysql –p radiusRW radius
      - insert into nasXgrp values ('', '10.1.1.8', '1');
      - insert into nasXgrp values ('', '10.1.1.8', '4');
      - insert into nasXgrp values ('', '10.1.2.1', '1');
      - insert into nasXgrp values ('', '10.1.2.1', '2');

© SANS Institute 2002, As part of the Information Security Reading Room. Author retains full rights.

- insert into nasXgrp values (‘’, ’10.1.2.2’, ‘1’);
- insert into nasXgrp values (‘’, ’10.1.2.2’, ‘2’);
- insert into nasXgrp values (‘’, ’10.1.3.1’, ‘1’);
- insert into nasXgrp values (‘’, ’10.1.3.1’, ‘3’);
- insert into nasXgrp values (‘’, ’10.1.3.2’, ‘1’);
- insert into nasXgrp values (‘’, ’10.1.3.2’, ‘3’);
- exit;

➢ For each type of NAS define the attribute / value pairs that you want returned based on level of access you wish to allow.
➢ Start mysql interactively and add the following; based on zzz.org
- mysql –p radiusRW radius
  - SQUID
    - This one is easy, it does not require any additional a/v’s
  - 3COM has two levels
    - User level (2)
      - insert into typeXlevel2avs values (‘’, ‘3com’, ‘Login-Service=Telnet‘ , ‘2’);
      - insert into typeXlevel2avs values (‘’, ‘3com’, ‘Service-Type=Login-User‘ , ‘2’);
    - Administrator level (15)
      - insert into typeXlevel2avs values (‘’, ‘3com’, ‘Login-Service=Telnet ‘ , ‘15’);
      - insert into typeXlevel2avs values (‘’, ‘3com’, ‘Administrative-User‘ , ‘15’);
  - BAY has two levels
    - User level (2)
      - insert into typeXlevel2avs values (‘’, ‘bay’, ‘Bay-User-Level=User‘ , ‘2’);
      - insert into typeXlevel2avs values (‘’, ‘bay’, ‘Bay-Audit-Level=User‘ , ‘2’);
    - Administrator level (15)
      - insert into typeXlevel2avs values (‘’, ‘bay’, ‘Bay-User-Level=Manager‘ , ‘15’);
      - insert into typeXlevel2avs values (‘’, ‘bay’, ‘Bay-Audit-Level=Manager‘ , ‘15’);

♦ Adding User’s
➢ For each User we need to create two check records in the ‘radcheck’ table
➢ Start mysql interactively and add the following; based on zzz.org
- mysql –p radiusRW radius
  - insert into radcheck values (‘’, ‘mick’, ‘Password’, password(‘mickPW’));
  - insert into radcheck values (‘’, ‘mick’, ‘Auth-Type’, ‘Crypt-Local’);
  - insert into radcheck values (‘’, ‘trish’, ‘Password’, password(‘trishPW’));
  - insert into radcheck values (‘’, ‘trish’, ‘Auth-Type’, ‘Crypt-Local’);
  - insert into radcheck values (‘’, ‘luke’, ‘Password’, password(‘lukePW’));
  - insert into radcheck values (‘’, ‘luke’, ‘Auth-Type’, ‘Crypt-Local’);

© SANS Institute 2002, As part of the Information Security Reading Room. Author retains full rights.

- insert into radcheck values (‘’, ‘jacqui’, ‘Password’, password(‘jacquiPW’));
- insert into radcheck values (‘’, ‘jacqui’, ‘Auth-Type’, ‘Crypt-Local’);
- exit;

- For each User we need to execute the external program ‘radiusAVs.pl’ passing it two values, ‘NAS ip address’, and ‘username’
- Start mysql interactively and add the following; based on zzz.org
  - mysql –p radiusRW radius
    - insert into radreply values (‘’, ‘mick’, ‘exec-program-wait’, ‘/radscripts/radiusAVs.pl %n %u’);
    - insert into radreply values (‘’, ‘trish’, ‘exec-program-wait’, ‘/radscripts/radiusAVs.pl %n %u’);
    - insert into radreply values (‘’, ‘luke’, ‘exec-program-wait’, ‘/radscripts/radiusAVs.pl %n %u’);
    - insert into radreply values (‘’, ‘jacqui’, ‘exec-program-wait’, ‘/radscripts/radiusAVs.pl %n %u’);

- For each User we now need to consider what devices they will have access to, we also need to consider the level allowed by each user. In our case refer to table 1.
- Start mysql interactively and add the following; based on zzz.org
  - mysql –p radiusRW radius
    - user ‘mick’ since he does not require full access to all devices we need to build up his group membership
      - access to web
        - insert into userXgrp2level values (‘’, ‘mick’, ‘4’, ‘2’);
      - access to 3Com routers as manager
        - insert into userXgrp2level values (‘’, ‘mick’, ‘3’, ‘15’);
      - access to Bay routers as user
        - insert into userXgrp2level values (‘’, ‘mick’, ‘2’, ‘2’);
    - user ‘trish’ since she does not require full access to all devices we need to build up her group membership
      - Access to web
        - insert into userXgrp2level values (‘’, ‘trish’, ‘4’, ‘2’);
      - access to Bay routers as manager
        - insert into userXgrp2level values (‘’, ‘trish’, ‘2’, ‘15’);
    - user ‘luke’ since he does not require full access to all devices we need to build up his group membership
      - Access to 3Com as user
        - insert into userXgrp2level values (‘’, ‘luke’, ‘3’, ‘2’);
      - Access to Bay as user
        - insert into userXgrp2level values (‘’, ‘luke’, ‘2’, ‘2’);
    - user ‘jacqui’ since she does not require full access to all devices we need to build up her group membership
      - Access to web
        - insert into userXgrp2level values (‘’, ‘jacqui’, ‘4’, ‘2’);
    - exit;

© SANS Institute 2002, As part of the Information Security Reading Room. Author retains full rights.

Using Test Script:

You previously installed a perl test script "/usr/local/sbin/radius/testrad".
Let's use it to test our installed radius server.

Run the script without any command line arguments. This will do two things. First it will ensure that all necessary support libraries are available. Second it will tell you the correct command line syntax.

- /usr/loca/sbin/radius/testrad
  - Usage: <radius server> <radius secret> <local IP> <username> <password>

Where:

- <radius server>
  - is the ip address of your radius server, in our case 10.1.1.1
- <radius secret>
  - is the secret for our source address as stored in the 'nas' table, in our case "secret8"
- <local ip>
  - this is sent as the attribute "NAS ip address"
  - this allows us to simulate very easily all the different types of NAS's defined. All you have to do is change this valid to a NAS of the type in question as defined in nas table.
- <username> / <password>
  - is a valid username and password.

For example, on the radius server lets simulate a valid 'squid' request.

- /usr/loca/sbin/radius/testrad 10.1.1.1 secret8 10.1.1.8 mick mickPW
- echo $?
  - 0

For example, on the radius server lets simulate a invalid 'squid' request.

- /usr/loca/sbin/radius/testrad 10.1.1.1 secret8 10.1.1.8 mick badpass
- echo $?
  - 1

For example, on the radius server lets simulate a valid '3com' request.

- /usr/loca/sbin/radius/testrad 10.1.1.1 secret8 10.1.3.1 mick mickPW
  - Login-Service=Telnet
  - Service-Type=Administrative-User
- echo $?
  - 0

Using with Squid:

You can easily enable your Squid Proxy Server to authenticate users over radius by installing a suitable radius agent.

- download the latest agent from

© SANS Institute 2002, Author retains full rights.

- http://selm.www.cistron.nl/~selm/authtools/squid_radius_auth-1.0.4.tar.gz [30]
- copy the downloaded file into local source directory
  - cp squid_radius_auth-1.0.4.tar.gz /usr/local/src/
- extract zipped tarball
  - cd /usr/local/src/
  - tar –xzvf squid_radius_auth-1.0.4.tar.gz
- cd /usr/local/src/squid_radius_auth-1.0.4
- cp Makefile.lnx Makefile
- manually apply a 'byte-order-patch' from
  - http://selm.www.cistron.nl/~selm/authtools/squid-rad-auth-byte-order-patch [31]
  - vi squid_radius_auth.c
    - change line ~505 form/to
      - auth->length = htonl(total_length);
      - auth->length = htons(total_length);
- Update where you wish to ultimately install the executable
  - if you do not want to install the executable into /opt/squid/auth/bin then you can modify /usr/local/src/squid_radius_auth-1.0.4/conf.h
    - change SQUID_RAD_CONF to something more suitable
- make clean
- make
- install the executable
  - cp /usr/local/src/squid_radius_auth-1.0.4/squid_rad_auth /opt/squid/auth/bin/
- then we need to make a configuration file
  - vi /opt/squid/auth/etc/squid_rad_auth.conf adding the following
    - # squid_rad_auth
    - server 10.1.1.1
    - secret secret8
- ensure that on this box /etc/services has radius on ports 1812 / 1813
  - grep –i rad /etc/services
    - radius 1812/tcp # Radius
    - radius 1812/udp # Radius
    - radius-acct 1813/tcp radacct # Radius Accounting
    - radius-acct 1813/udp radacct # Radius Accounting
- Ensure the radius server has a NAS record for this server and the secret is the same as in /opt/squid/auth/etc/squid_rad_auth
- test it manually by starting "/opt/squid/auth/bin/squid_rad_auth" then enter both a valid id and password one per line, <ctrl>c to abort
  - squiduser squid_password<enter>
    - OK or ERR
  - valid user with access to squid proxy
    - mick mickPW
    - OK
  - valid user without access to squid proxy
    - luke lukePW
    - ERR

© SANS Institute 2002, As part of the Information Security Reading Room. Author retains full rights.

- Invalid user, should also not have access
  - mick badpass
  - ERR
- modify squid.conf, Please note SQUID is extremely powerful and thus has many sometimes confusing configuration options the following is the simplest required to enable radius password authentication. More detailed configuration options are available from the developers site [32].
  - first tell squid what authentication program to use. Find the line starting with "authenticate_program" and change it to
    - authenticate_program /usr/local/squid/auth/bin/squid_rad_auth
  - Next we need to define a ACL of type proxy_auth, in your ACL section add
    - acl authRequired proxy_auth REQUIRED
    - acl sInternal src 10.1.4.0/255.255.255.0
  - Next combine the ACL with other ACL's in http_access, remember multiple acl's on one line are logically anded together
    - http_access allow sInternal authRequired
    - http_access deny all
  - Lets provide the user with a appropriate prompt for his/her id/password
    - proxy_auth_realm Company ZZZ Proxy Gateway and Virus Scanner
- Initiate squid to reread it configuration file
  - /etc/rc.d/init.d/squid reload
- restart your browser and test.

Using with BAY routers:

I have identified 2 suitable levels within a Bay router. I have decided to arbitrary set the two levels as 2 – User, 15 – Manager. We have successfully used these to manage:

- Bay Access Remote Node,
- Bay Access STAC Node,
- Nortel Passport 2430.

First we need to load into IC Radius Dictionary additional Attribute / Value pairs suitable for the Bay router. I do not claim that this list is complete, it was extracted from the following email http://lists.cistron.nl/pipermail/cistron-radius/2001-September/002139.html [33].

- Save a copy of Appendix 3 to /tmp/bay.dictionary
- Using mysql client add additional records into dictionary table of radius database
  - cat /tmp/bay.dictionary | mysql -p -u radiusRW radius
- Using mysql client check records were added into dictionary table of radius database
  - echo 'echo 'select * from dictionary where Attribute like "Bay%";' | mysql -u radiusRW –p radius

Second for each router, in our case bay1 and bay2, telnet into them and install the following additional configuration.

Relevant Bay router configuration from BCC for the CLIENT.

© SANS Institute 2002, As part of the Information Security Reading Room. Author retains full rights.

```
radius
   radius-client slot 1 address 10.1.2.1
      debug-message-level high
      authentication enabled
   back
   radius-server address 10.1.1.1
      authentication-udp-port 1812
      primary-server-secret secret8
      reset-timer 1
      automatic-reset enabled
   back
```

- Using a second telnet window check level 2 and level 15 access. When these are OK you should
  - change the manager password,
  - escrow the manager password,
  - delete all other accounts on the router,
  - save the configuration.

Three warnings based on the Bay routers I have tried this on:

- insert the whole configuration at once especially the section overriding the default port from 1645 to 1812. I found if you do not the router uses port 1645 until either a reboot or a complete removal and reinsertion of the radius configuration.
- The secret is limited to 8 characters. The router will accept longer shared secrets however I could not get them to work. This is a pity as the rfc2865 [2] recommends at least 16 characters.
- The manager password always works regardless of radius status, thus it should be suitably long and of cause escrowed.

Using with 3Com routers:

I have identified 2 suitable levels within a 3Com router. I have decided to arbitrary set the two levels as 2 – User, 15 – root. We have successfully used these to manage thousands of 432SI router.

This is based on trial and error. It appears to work as expected however no guarantees are given. Essentially I tried various combinations until I got it to work. I based what I was trying on details extracted from http://webserver3.isunet.net/elmerfud/tcm4/hiperarc5.0.9/dictnary .[34] The particular attribute/values needed were in fact loaded into IC radius when we loaded the default dictionary.

For each router, in our case threecom1 and threecom2, telnet into them and install the following additional configuration.

The relevant 3Com router configuration details which can be input via individual commands or the 'menu' command.

© SANS Institute 2002, As part of the Information Security Reading Room. Author retains full rights.

```
[1]threecom1#sh -ac conf
----------------------Access Control Global Configuration----------------------
ACcntUdpport = 1813
AUthUdpport = 1812
CurACcntSrvr = 10.1.1.1
CurAUthSrvr = 10.1.1.1
EXPirationTimer = 90 (days)
PrimACcntSrvr = 10.1.1.1
PrimAUthSrvr = 10.1.1.1
RESolutionOrder = Local
RetransTimer = 3
SecACcntSrvr = 0.0.0.0
SecAUthSrvr = 0.0.0.0
Secret = `*******'              ← "secret8"
```

- Using a second telnet window check level 2 and level 15 access. When these are OK you should:
  - change the root password,
  - escrow the root password,
  - delete all other accounts on the router.

Additional Considerations

This document is extremely long and hasn't covered everything you need to consider.
For example you should seriously consider:

- The fresh installation of Linux installed both IPChains and IPTables. Unfortunately only one can load, it is IPChains by default. IPTables is superior to IPChains and should be enabled. Additional information can be found at http://netfilter.samba.org/documentation/HOWTO/packet-filtering-HOWTO-7.html [35]
- The fresh installation of Linux also installed Tripwire, it needs to be configured and monitored through out the life of this server. Additional information can be found at http://www.linuxdoc.org/LDP/solrhe/Securing-Optimizing-Linux-RH-Edition-v1.3/tripwireASR.html [36]
- Linux, MySQL, IC Radius all create logs, they need to be monitored. One very suitable tool is swatch http://www.oit.ucsb.edu/~eta/swatch/ [37].
- All software has bugs please monitor Bugtrack and relevant vendor alerts. Additional information can be found at http://online.securityfocus.com/archive/1 [38]
- Backups are vital. In particular, your ever changing "radius" authentication database. You should as a minimum implement some sort of (tested) backup strategy using either of or a combination of:
  - mysqldump http://www.mysql.com/doc/m/y/mysqldump.html [39]
  - mysqlhotcopy http://www.mysql.com/doc/m/y/mysqlhotcopy.html [40]
- Depending on the nature of your installation you may also want to consider
  - Change log
  - Multiple systems

© SANS Institute 2002, Author retains full rights.

Summary
Authentication poses a number of challenges to system administrators. But by using a selection of free or very cheap off the shelf products (Linux, IC-radius, radius clients, MySQL) and a custom script access to a wide range of network elements can be controlled in a scalable way.

References

[1] "SANS Security Essentials V: Windows Basics", 2002(2002):2-3

[2] Rigney, C., Willens, S., Rubens, A., Simpson, W., "Remote Authentication Dial In User Service (RADIUS)", RFC2865, June 2000,
URL: http://www.doclib.org/rfc/rfc2865.html (1 March 2002)

[3] McCarthy, Ron., "RADIUS", Sys Admin February 1999 (1999):33-39

[4] Carrel, D., Grant, Lol., "The TACACS+ Protocol Version 1.78", January 1997,
URL: ftp://ftpeng.cisco.com/pub/tacacs/tac-rfc.1.78.txt (12 March 2002)

[5] Martin, Michael., "Router Expert: Building a secure TACACS+ environment", 21 January 2002,
URL: http://searchnetworking.techtarget.com/tip/1,289483,sid7_gci798349,00.html (5 February 2002)

[6] Rigney, C., Willens, S., Rubens, A., Simpson, W., "Remote Authentication Dial In User Service (RADIUS)", RFC2058, January 1997,
URL: http://www.doclib.org/rfc/rfc2058.html (1 March 2002)

[7] Rigney, C., Willens, S., Rubens, A., Simpson, W., "Remote Authentication Dial In User Service (RADIUS)", RFC2138, April 1997,
URL: http://www.doclib.org/rfc/rfc2138.html (1 March 2002)

[8] Retallack, Roger., "Securing Linux Installations", 15 June 2001,
URL: http://rr.sans.org/linux/sec_install.php (1 March 2002)

[9] Spitzner, Lance., "Preparing your Linux box for the Internet, Armoring Linux", 19 September 2001,
URL: http://www.enteract.com/~lspitz/linux.html (1 March 2002)

[10] "Bastille Linux", 14 June 2001,
URL: http://www.bastille-linux.org/ (12 March 2002)

[11] Grimaila, Michael Russell., "The Role of Bastille Linux in Information Security", 18 February 2002,
URL: http://rr.sans.org/linux/bastille.php (15 March 2002)

© SANS Institute 2002, As part of the Information Security Reading Room. Author retains full rights.

[12] "IC-RADIUS",
URL: http://radius.innercite.com/ (9 November 2001)

[13] "CERT® Advisory CA-2002-06 Vulnerabilities in Various Implementations of the RADIUS Protocol", 4 March 2002,
URL: http://www.cert.org/advisories/CA-2002-06.html (11 March 2002)

[14] "IC-RADIUS 0.18.1", 29 June 2001,
URL: ftp://ftp.innercite.com/pub/icradius/icradius-0.18.1.tar.gz (9 November 2001)

[15] Banks, James., Rathbun, Brad., "README", 30 June 2001,
URL: ftp://ftp.innercite.com/pub/icradius/icradius-0.18.1.tar.gz (9 November 2001)
(need to extract from tarball; contained within doc sub-directory)

[16] Rathbun, Brad., "IC-RADIUS README", 11 June 2001,
URL: http://radius.innercite.com/ICRADIUS.README.html, (11 November 2001)

[17] "Errata: Security Alerts, Bugfixes, and Enhancements",
URL: http://www.redhat.com/apps/support/errata/ (12 March 2002)

[18] "4.2.8 Connecting to the MySQL Server",
URL: http://www.mysql.com/doc/C/o/Connecting.html (March 12 2002)

[19] "4.2 General Security Issues and the MySQL Access Privilege System"
URL: http://www.mysql.com/doc/P/r/Privileges.html (12 March 2002)

[20] "4.1.2 my.cnf Option Files",
URL: http://www.mysql.com/doc/O/p/Option_files.html (12 March 2002)

[21] Stebbens, Alan K., "ShowTable routines to display tabular data in several formats.", 26 August 1997,
URL: http://www.cpan.org/authors/id/AKSTE/Data-ShowTable-3.3.tar.gz (12 November 2001)

[22] "CERT® Advisory CA-2002-03 Multiple Vulnerabilities in Many Implementations of the Simple Network Management Protocol (SNMP)", 7 March 2002,
URL: http://www.cert.org/advisories/CA-2002-03.html (12 March 2002)

[23] Wilder-Goodwin, Drew., "ICRADIUS Interface Module", 24 April 2001,
URL: ftp://ftp.innercite.com/pub/icradius/IC-Radius-0.4.tar.gz (11 November 2001)

[24] Aas, Gilse., "MD5 Perl interface to the MD5 Message-Digest Algorithm", 14 March 2001,
URL: http://www.cpan.org/authors/id/GAAS/MD5-2.02.tar.gz (12 November 2001)

[25] Declerck, Carl., "Authen::Radius provide simple Radius client facilities", 19 August 1997,
URL: http://www.cpan.org/authors/id/CARL/RadiusPerl-0.05.tar.gz (12 November 2001)

© SANS Institute 2002, As part of the Information Security Reading Room. Author retains full rights.

[26] "Red Hat Linux 7.2, The Official Red Hat Linux x86 Installation Guide",
URL: http://www.redhat.com/docs/manuals/linux/RHL-7.2-Manual/install-guide/ (12 March 2002)

[27] "MySQL Reference Manual for version 4.0.2-alpha."
URL: http://www.mysql.com/documentation/mysql/full/ (12 March 2002)

[28] "RADIUS for UNIX Administrator's Guide", Pleaston: Lucent Technologies, February 1999
URL: http://portmasters.com/tech/docs/pdf/radius.pdf (1 February 2002): Chapter 4/page 12.

[29] "IC-RADIUS On-line FAQ",
URL: http://radius.innercite.com/cgi-bin/faqdb.cgi (12 March 2002)

[30] van Selm, Marc ., "Squid RADIUS authenticator V1.04"
URL: http://selm.00www.cistron.nl/~selm/authtools/squid_radius_auth-1.0.4.tar.gz (6 November 2001)

[31] McCauley, Mike., "Onderwerp: Squid radius authenticator bug fix", 22 January 2001,
URL: http://selm.www.cistron.nl/~selm/authtools/squid-rad-auth-byte-order-patch (6 November 2001)

[32] "Squid 2.4 Stable1Configuration Manual",
URL: http://squid.visolve.com/squid24s1/contents.htm (12 March 2002)

[33] Abzug, T Mordechai., "Bay/Nortel 14.20 and Cistron radius", 20 September 2001,
URL: http://lists.cistron.nl/pipermail/cistron-radius/2001-September/002139.html (27 February 2002)

[34] "This file is a sample RADIUS dictionary for MERIT RADIUS server to interact with 3Com Hiper Access Router Card",
URL: http://webserver3.isunet.net/elmerfud/tcm4/hiperarc5.0.9/dictnary (27 February 2002)

[35] Russell, 'Rusty' Paul., "Using ip tables", 2000,
URL: http://netfilter.samba.org/documentation/HOWTO/packet-filtering-HOWTO-7.html (18 February 2002)

[36] Mourani, Gerhard.,"Chapter 18. Linux Tripwire ASR 1.3.1",
URL: http://www.linuxdoc.org/LDP/solrhe/Securing-Optimizing-Linux-RH-Edition-v1.3/tripwireASR.html (12 March 2002)

[37] Atkins, E. Todd., "SWATCH", Simple Watcher 3.0.4, 6 November 2001,
URL: http://www.oit.ucsb.edu/~eta/swatch/ (14 March 2002)

[38] "Bugtrack",
URL: http://online.securityfocus.com/archive/1 (12 March 2002)

© SANS Institute 2002, As part of the Information Security Reading Room. Author retains full rights.

[39] "4.8.5 mysqldump, Dumping Table Structure and Data",
URL: http://www.mysql.com/doc/m/y/mysqldump.html (12 March 2002)

[40] "4.8.6 mysqlhotcopy, Copying MySQL Databases and Tables",
URL: http://www.mysql.com/doc/m/y/mysqlhotcopy.html (12 March 2002)

© SANS Institute 2002, Author retains full rights.

Appendix

Appendix 1: RadiusAVs.pl (IC Radius Extension Script)

```
#!/usr/bin/perl -wT
######                                          ##########
#                                                  #
# WARNING : DO NOT MAKE THIS SCRIPT GLOBAL READABLE                    #
#         : IT CONTAINS A FULL READ ONLY ID/PW TO YOU RADIUS DATABASE  #
#                                                  #
######                                          ##########
######
#
# DISCLAIMER: Use at your own risk, no warranty or guarantee given
#
######
# Version: 1.10 11 February 2002
#
# Purpose:
# To automatically create Vendor Specific A/V pairs based on NAS source Address
# when used with ICRadius, it will also take into consideration user level allowed
#
# usage:  radiusAVs.pl <nas_ip_address> <user_id>
#
# that is configure your radius box as
#  insert into radreply values ('', 'user', 'Exec-Program-Wait', '/radscripts/radiusAVs.pl %n %u');
# WARNING: the last attribute has a total limit of 40 characters that is why I suggest radscripts
#          yep pollute you root directory - Sorry
#######

#
# You will also need to create additional tables within your 'radius' database
#

#######
# typeXlevel2avs - used to create a list of a/v's based on level of access and type of nas
# mysql> create table typeXlevel2avs (
#    -> id int(10) DEFAULT '0' NOT NULL auto_increment,
#    -> type varchar(30) NOT NULL,
#    -> response varchar(128) NOT NULL,
#    -> level int(10) DEFAULT '0' NOT NULL,
#    -> PRIMARY KEY (id)
#    -> );
# where
#    id - a counter id field
#    type - matched to nas.type
```

© SANS Institute 2002, As part of the Information Security Reading Room. Author retains full rights.

```
#    response - a a/v pair to be sent back
#    level - which level the user has to have to get this attribute
#

#######
# userXnas2level - used to determine a user's level of access based on his username and nas
longhostname
# mysql> create table userXnas2level (
#    -> id int(10) DEFAULT '0' NOT NULL auto_increment,
#    -> user varchar(30) NOT NULL,
#    -> nasRE varchar(128) NOT NULL,
#    -> level int(10) DEFAULT '0' NOT NULL,
#    -> PRIMARY KEY (id)
#    -> );
# where
#    id - a counter id field
#    user - user name
#    nasRE - regular expression to match against nas'es longname
#    level - the highest level available to this user for accessing from this nas
#

#######
# grp - a table in which store a comment about each group (not needed but can help in
administration)
# mysql> create table grp (
#    -> id int(10) DEFAULT '0' NOT NULL auto_increment,
#    -> grpID int(10) DEFAULT '0' NOT NULL,
#    -> grpName varchar(30) NOT NULL,
#    -> grpDescription text NOT NULL,
#    -> PRIMARY KEY (id)
#    -> );
# where
#    id - a counter id field
#    grpID - group ID number
#    grpName - a short name for the group
#    grpDescription - long description of this group to allow full detail of what it is used for
#

#######
# nasXgrp - used to associate a NAS's ipaddress to any number of groups
# mysql> create table nasXgrp (
#    -> id int(10) DEFAULT '0' NOT NULL auto_increment,
#    -> ipaddr varchar(15) NOT NULL,
#    -> grpID int(10) DEFAULT '0' NOT NULL,
#    -> PRIMARY KEY (id),
#    -> INDEX index_nas ( ipaddr )
```

© SANS Institute 2002, As part of the Information Security Reading Room. Author retains full rights.

```
#    -> );
# where
#    id - a counter id field
#    ipaddr - ip address of nas
#    grpID - group ID number
#

#######
# userXgrp2level - used to determine a user's level based on group membership and NAS
grouping
# mysql> create table userXgrp2level (
#    -> id int(10) DEFAULT '0' NOT NULL auto_increment,
#    -> user varchar(30) NOT NULL,
#    -> grpID int(10) DEFAULT '0' NOT NULL,
#    -> level int(10) DEFAULT '0' NOT NULL,
#    -> PRIMARY KEY (id),
#    -> INDEX index_user ( user )
#    -> );
# where
#    id - a counter id field
#    user - user name
#    grpID - group ID number
#    level - the highest level available to this user for accessing from this nas
#

# Revision History:
# 1.00 20/Nov/2001 mtm original
# 1.01 26/Nov/2001 mtm added timeout just in case
# 1.10 11/Feb/2002 mtm added support for group tables and conversion to level thus av's

use strict;
use DBI;

#
# Configuration Section Begin
#
my $database = 'radius';        # database in mysql which contains all the radius info
my $host = 'localhost';         # host which is running the mysql database
my $idAdmin = 'radiusRO';       # account with read permission to $database
my $pwAdmin = 'radROpw';        # valid password for $idAdmin to $database
my $timelimit = 10;             # total time to make a decision else fail
my $maxlevel = 15;              # set to the maximum level you have implemented suggest 15 e.g.
cisco
my $skipREmatch = 1;            # for speed set to 1 to disable RE matching, 0 to enable
#
# Configuration Section End
```

© SANS Institute 2002, As part of the Information Security Reading Room. Author retains full rights.

```
#

my @avs = ();
my $nasIP = "";
my $nasType = "";
my $nasNameLong = "";
my $nasNameShort = "";
my $level = 0;
my $user = "";

my $dbh = "";           # database handle

sub alarm_h {
#################
# name:         alarm_h
# purpose:      registered function to handle registered time expiry
# variablesIN:  nil
# expects:      to have been registered previously
#               to have timed out
# returns:      directly -  nil
#               indirectly - nil
# comments:     will be called by Perl handler if the previously
#               registered time limit has been exceeded
# last reviewed: mtm 08/Jul/2001
#################
  alarm(0);
  if ( $dbh ) {
    $dbh->disconnect;
  }
  exit 1;
}


sub determineNASdetails {
########################
# name:         determineNASdetails
# purpose:      to determine NAS details by using radius NASip source address and looking up
radius.nas table
# variablesIN:  $dbh - initialized database handle
#               $nasIP - radius packet NAS ip address,
#                     note this is from a attribute it may be different to source ip address
#               $pNasType - pointer to return NAS type as extracted from radius.nas.type
#               $pNasNameLong - pointer to return NAS type as extracted from radius.nas.type
#               $pNasNameShort - pointer to return NAS type as extracted from radius.nas.type
# expects:      $dbh handle to be valid
# returns:      directly - status
```

© SANS Institute 2002, As part of the Information Security Reading Room. Author retains full rights.

```
#                       TRUE - if all ok
#                       FALSE - if any failure
#               indirectly -
#                       $pNasType has been set to what is in radius.nas.type
#                       $pNasNameLong has been set to what is in radius.nas.nasname
#                       $pNasNameShort has been set to what is in radius.nas.shortname
# comments:     nil
# last reviewed: mtm 20/Nov/2001
#########################
  my @data = ();
  my $dbh = "";
  my $nasIP = "";
  my $pNasType = "";
  my $pNasNameLong  = "";
  my $pNasNameShort = "";
  my $query = "";
  my $status = "";
  my $sth = "";

  ($dbh, $nasIP, $pNasType, $pNasNameLong, $pNasNameShort) = @_;
  $query .= "select type, nasname, shortname ";
  $query .= "from nas ";
  $query .= "where ipaddr=?";

  $sth = $dbh->prepare($query);
  if(!$sth->execute($nasIP)) {
    $status = FALSE();
  } else {
    if ( @data = $sth->fetchrow_array ) {
      $$pNasType = $data[0];
      $$pNasNameLong = $data[1];
      $$pNasNameShort = $data[2];
      $status = TRUE();
    } else {
      $$pNasType = "";
      $status = FALSE();
    }
  }
  $sth->finish;
  return $status;
}


sub determineAVs {
#########################
# name:         determineAVs
```

© SANS Institute 2002, As part of the Information Security Reading Room. Author retains full rights.

```
# purpose:      to construct a array of av's from radius.typeXlevel2avs based on
#                 - user level
#                 - nas type
#             if the level requested is 0 then by default deny all access
# variablesIN:  $dbh - initialized database handle
#             $nasType - the type of NAS as extracted from radius.nas
#             $level - user level required
#             $pAVs - pointer to array to return each matching record in
# expects:      $dbh handle to be valid
# returns:      directly - status
#                 TRUE - if all ok
#                 FALSE - if any failure
#             indirectly - $pAVs array will have necessary records added
# comments:     nil
# last reviewed: mtm 20/Nov/2001
#########################
  my @data = ();
  my $dbh = "";
  my $level = 0;
  my $nasType = "";
  my $pAVs = "";
  my $query = "";
  my $status = "";
  my $sth = "";

  ($dbh, $nasType, $level, $pAVs) = @_;

  # deny all level 0 accesses by default
  if ( $level == 0 ) {
    return FALSE();
  }

  $query .= "select response ";
  $query .= "from typeXlevel2avs ";
  $query .= "where type=? and level=?";

  $sth = $dbh->prepare($query);
  if(!$sth->execute($nasType, $level)) {
    $status = FALSE();
  } else {
    $status = TRUE();
    while ( @data = $sth->fetchrow_array ) {
      push ( @$pAVs, $data[0] );
    }
  }
  $sth->finish;
```

© SANS Institute 2002, As part of the Information Security Reading Room. Author retains full rights.

```
  return $status;
}


sub determineLevelRE {
#########################
# name:        determineLevelRE
# purpose:      to determine the user level based on RE match of username X nasLongName
#                  this will typically be level 15 (e.g. read write only),
#                  this will typically be level 2  (e.g. read only),
#                  if no match default level is 0  (e.g. disable / reject)
# variablesIN:  $dbh - initialized database handle
#              $user - user extracted from radius packet
#              $nasNameLong - the fqdn of the nas source extracted from radius.nas.nasname
#              $pLevel - pointer to return calculated level
# expects:      $user to contain the username
# returns:      directly - status
#                  TRUE - if all ok
#                  FALSE - if any failure
#              indirectly - $pLevel has been updated to actual user level
# comments:
# last reviewed: mtm 12/02/2002
#########################
  my @data = ();
  my $dbh = "";
  my $user = "";
  my $pLevel = "";
  my $nasNameLong = "";
  my $query = "";
  my $status = "";
  my $sth = "";

  ($dbh, $user, $nasNameLong, $pLevel) = @_;
  #
  # prepare for RE lookup
  #
  $query .= "select level, nasRE ";
  $query .= "from userXnas2level ";
  $query .= "where user=? ";
  $query .= "order by level desc";

  #
  # assume bad news
  #
  $$pLevel = 0;
```

© SANS Institute 2002, As part of the Information Security Reading Room. Author retains full rights.

```
  $sth = $dbh->prepare($query);
  if(! $sth->execute($user)) {
    $status = FALSE();
  } else {
    $status = TRUE();
    while ( @data = $sth->fetchrow_array ) {
      if ( $nasNameLong =~ /$data[1]/ ) {
        $$pLevel = $data[0];
        last;
      }
    }
  }
  $sth->finish;
  return $status;
}


sub determineLevelGrp {
#########################
# name:         determineLevelGrp
# purpose:      to determine the max user level, based on
#               - username and groups that user has access to
#               - device groups
#             if no match default level is 0
# variablesIN:  $dbh - initialized database handle
#             $user - user extracted from radius packet
#             $nasIP - radius packet NAS ip address
#             $pLevel - pointer to return calculated level
# expects:      $user to contain the username
#             $dbh to be initialized
#             $nasIP to be valid
# returns:      directly - status
#                 TRUE - if all ok
#                 FALSE - if any failure
#             indirectly - $pLevel has been updated to actual user level
# comments:     nil
# last reviewed: mtm 11/02/2002
#########################
  my @data = ();
  my $dbh = "";
  my $nasIP = "";
  my $pLevel = "";
  my $query = "";
  my $status = "";
  my $sth = "";
  my $user = "";
```

© SANS Institute 2002, As part of the Information Security Reading Room. Author retains full rights.

```
  ($dbh, $user, $nasIP, $pLevel) = @_;
  # determine MAX level based on group crossed by user and nasIP
  # sub queries would be good - do not blame me
  $query .= "select userXgrp2level.level ";
  $query .= "from userXgrp2level,nasXgrp ";
  $query .= "where userXgrp2level.grpID = nasXgrp.grpID ";
  $query .= "and userXgrp2level.user=? ";
  $query .= "and nasXgrp.ipaddr=? ";
  $query .= "order by userXgrp2level.level DESC ";
  $query .= "limit 1 ";


  #
  # assume bad news
  #
  $$pLevel = 0;

  $sth = $dbh->prepare($query);
  if(! $sth->execute($user, $nasIP)) {
    $status = FALSE();
  } else {
    $status = TRUE();
    if ( @data = $sth->fetchrow_array ) {
      $$pLevel = $data[0];
    }
  }

  $sth->finish;
  return $status;
}


sub TRUE  { 1 };
sub FALSE { 0 };

#
# Check and assign command line options
#
if ($#ARGV == 1 ) {
  $nasIP = shift;
  ($nasIP)  = ($nasIP =~ /.*?(\d+\.\d+\.\d+\.\d+).*?/ );
  $user = shift;
  ($user) = ($user =~ /.*?(\w+).*?/ );
  if ( ! ( ( defined $nasIP ) && (  defined $user ) ) ) {
    exit 1;
```

© SANS Institute 2002, Author retains full rights.

```
  }
} else {
  exit 1;
}

#
# install a precautionary timeout timer in case
#
$SIG{'ALRM'} = \&alarm_h;
alarm($timelimit);

if (!($dbh = DBI->connect("DBI:mysql:${database}:${host}", "$idAdmin", "$pwAdmin") )) {
  exit 1;
}

($dbh->disconnect, alarm(0), exit 1 ) unless (determineNASdetails($dbh, $nasIP, \$nasType,
\$nasNameLong, \$nasNameShort));
($dbh->disconnect, alarm(0), exit 1 ) unless (determineLevelGrp($dbh, $user, $nasIP, \$level));
if ( ( $level != $maxlevel ) && ( $skipREmatch != 1 ) ) {
  #
  # note matching by determineLevelGrp is much preferred
  # in fact I suggest that you skip determineLevelRE if you can
  #
  my $tmpLevel = $level;
  ($dbh->disconnect, alarm(0), exit 1 ) unless (determineLevelRE($dbh, $user, $nasNameLong,
\$level));
  $level = ( $level > $tmpLevel ) ? $level : $tmpLevel;
}
#
# sanity check the level
#
($dbh->disconnect, alarm(0), exit 1 ) unless (( $level >= 0 ) && ( $level <= $maxlevel ) );
($dbh->disconnect, alarm(0), exit 1 ) unless (determineAVs($dbh, $nasType, $level, \@avs));

$dbh->disconnect;
alarm(0);

for (@avs) {
  print "$_\n";
}
exit 0;
```

Appendix 2: Additional Custom Tables

```
#
# Table structure for table 'grp'
```

© SANS Institute 2002, As part of the Information Security Reading Room. Author retains full rights.

```
#

CREATE TABLE grp (
  id int(10) NOT NULL auto_increment,
  grpID int(10) NOT NULL default '0',
  grpName varchar(30) NOT NULL default '',
  grpDescription text NOT NULL,
  PRIMARY KEY (id)
) TYPE=MyISAM;

#
# Table structure for table 'nasXgrp'
#

CREATE TABLE nasXgrp (
  id int(10) NOT NULL auto_increment,
  ipaddr varchar(15) NOT NULL default '',
  grpID int(10) NOT NULL default '0',
  PRIMARY KEY (id),
  KEY index_nas(ipaddr)
) TYPE=MyISAM;

#
# Table structure for table 'typeXlevel2avs'
#

CREATE TABLE typeXlevel2avs (
  id int(10) NOT NULL auto_increment,
  type varchar(30) NOT NULL default '',
  response varchar(128) NOT NULL default '',
  level int(10) NOT NULL default '0',
  PRIMARY KEY (id)
) TYPE=MyISAM;

#
# Table structure for table 'userXgrp2level'
#

CREATE TABLE userXgrp2level (
  id int(10) NOT NULL auto_increment,
  user varchar(30) NOT NULL default '',
  grpID int(10) NOT NULL default '0',
  level int(10) NOT NULL default '0',
  PRIMARY KEY (id),
  KEY index_user(user)
) TYPE=MyISAM;
```

© SANS Institute 2002, As part of the Information Security Reading Room. Author retains full rights.

```
#
# Table structure for table 'userXnas2level'
#

CREATE TABLE userXnas2level (
  id int(10) NOT NULL auto_increment,
  user varchar(30) NOT NULL default '',
  nasRE varchar(128) NOT NULL default '',
  level int(10) NOT NULL default '0',
  PRIMARY KEY (id)
) TYPE=MyISAM;
```

Appendix 3: Bay Dictionary

```
INSERT INTO dictionary VALUES ('','VENDOR','Bay','1584','');
INSERT INTO dictionary VALUES ('','ATTRIBUTE','Bay-Local-IP-
Address','35','ipaddr','Bay');
INSERT INTO dictionary VALUES ('','ATTRIBUTE','Bay-Primary-DNS-
Server','54','ipaddr','Bay');
INSERT INTO dictionary VALUES ('','ATTRIBUTE','Bay-Secondary-DNS-
Server','55','ipaddr','Bay');
INSERT INTO dictionary VALUES ('','ATTRIBUTE','Bay-Primary-NBNS-
Server','56','ipaddr','Bay');
INSERT INTO dictionary VALUES ('','ATTRIBUTE','Bay-Secondary-NBNS-
Server','57','ipaddr','Bay');
INSERT INTO dictionary VALUES ('','ATTRIBUTE','Bay-User-Level','100','integer','Bay');
INSERT INTO dictionary VALUES ('','VALUE','Bay-User-Level','Manager','2','');
INSERT INTO dictionary VALUES ('','VALUE','Bay-User-Level','User','4','');
INSERT INTO dictionary VALUES ('','VALUE','Bay-User-Level','Operator','8','');
INSERT INTO dictionary VALUES ('','ATTRIBUTE','Bay-Audit-Level','101','integer','Bay');
INSERT INTO dictionary VALUES ('','VALUE','Bay-Audit-Level','Manager','2','');
INSERT INTO dictionary VALUES ('','VALUE','Bay-Audit-Level','User','4','');
INSERT INTO dictionary VALUES ('','VALUE','Bay-Audit-Level','Operator','8','');
```

© SANS Institute 2002, As part of the Information Security Reading Room. Author retains full rights.

Last Updated: May 22nd, 2013

Upcoming SANS Training

Click Here for a full list of all Upcoming SANS Events by Location

SANS Malaysia @ MCMC 2013	**Cyberjaya, MY**	**Jun 03, 2013 - Jun 08, 2013**	**Live Event**
SANS Pen Test Berlin 2013	**Berlin, DE**	**Jun 03, 2013 - Jun 08, 2013**	**Live Event**
Industial Control Systems Security Training - Houston	**Houston, TXUS**	**Jun 10, 2013 - Jun 15, 2013**	**Live Event**
Security Impact of IPv6 Summit 2013	**Washington, DCUS**	**Jun 14, 2013 - Jun 16, 2013**	**Live Event**
SANSFIRE 2013	**Washington, DCUS**	**Jun 14, 2013 - Jun 22, 2013**	**Live Event**
SANS Canberra 2013	**Canberra, AU**	**Jul 01, 2013 - Jul 13, 2013**	**Live Event**
Digital Forensics & Incident Response Summit 2013	**Austin, TXUS**	**Jul 09, 2013 - Jul 16, 2013**	**Live Event**
SANS London Summer 2013	**London, GB**	**Jul 09, 2013 - Jul 16, 2013**	**Live Event**
SANS Rocky Mountain 2013	**Denver, COUS**	**Jul 14, 2013 - Jul 20, 2013**	**Live Event**
SANS Mumbai 2013	**Mumbai, IN**	**Jul 22, 2013 - Jul 27, 2013**	**Live Event**
SEC528 SANS Training Program for the CompTIA® New Advanced Security Practitioner (CASP) Certification	**Washington, DCUS**	**Jul 22, 2013 - Jul 26, 2013**	**Live Event**
SANS San Francisco 2013	**San Francisco, CAUS**	**Jul 29, 2013 - Aug 03, 2013**	**Live Event**
SANS SEC 560: Network Penetration Testing @ Bangalore 2013	**Bangalore, IN**	**Aug 05, 2013 - Aug 10, 2013**	**Live Event**
SANS Boston 2013	**Boston, MAUS**	**Aug 05, 2013 - Aug 10, 2013**	**Live Event**
Critical Security Controls Summit	**Washington, DCUS**	**Aug 12, 2013 - Aug 18, 2013**	**Live Event**
Industrial Control Systems Security Training - DC	**Washington, DCUS**	**Aug 12, 2013 - Aug 16, 2013**	**Live Event**
SANS Thailand 2013	**Bangkok, TH**	**Aug 19, 2013 - Aug 31, 2013**	**Live Event**
SANS Virginia Beach 2013	**Virginia Beach, VAUS**	**Aug 19, 2013 - Aug 30, 2013**	**Live Event**
Mobile Device Security Summit 2013	**OnlineCAUS**	**May 30, 2013 - Jun 06, 2013**	**Live Event**
SANS OnDemand	**Books & MP3s OnlyUS**	**Anytime**	**Self Paced**