Global Knowledge®

Expert Reference Series of White Papers

Building, Installing, and Configuring a RADIUS Server

Building, Installing, and Configuring a RADIUS Server

George Mays, CCNA, A+, Network+, Security+, CTT+, I-Net+

Introduction

I work often with a variety of networking devices from different manufacturers. In many cases the equipment is simply being evaluated, configured for demonstration purposes, or incorporated into a lab for classroom use. RADIUS is supported by a lot of these gadgets but I have never found a handy-dandy inexpensive and easy to use RADIUS server. Well, let's build one.

RADIUS is an acronym for Remote Authentication Dial-In User Services. It is an AAA tool intended to be useful in instances where the user would like to centralize management of authentication, authorization, and accounting functions (hence the AAA). Authentication, or proving that users are who they claim to be, is the primary reason that most people are driven to want a RADIUS server. But the authorization (limiting what a user is allowed to do once they have been authenticated) and accounting (logging and billing functions) will be of interest to some people, too.

Where I most often like to demonstrate the use of RADIUS is in the configuration of Ethernet switches and IEEE 802.11 access points. For switches, RADIUS is most often used in conjunction with IEEE 802.1x port-based network access controls, which can in turn be used to control the identity of users who are allowed access to specific ports. For access points the same mechanism is actually in play, but it is used to limit who can associate with the wireless network. You might have bumped into this feature if you ever clicked on the "enterprise" settings for WPA or WPA2 while configuring your AP.

Raspberry Pi is a handy platform to building a simple RADIUS server. I chose the Raspberry Pi 2, which has a multicore ARM processor and 1GB of RAM. For the OS I decided on Raspbian, which is basically Debian Linux for the Pi. A case and heatsinks creates a nice package about the size of a pack of cigarettes. To build and configure the server you will also require a display that supports HDMI and a USB keyboard and mouse initially. Once the appliance is built you will no longer need the latter items as you can manage it via the network.

The cost is minimal. The Raspberry Pi 2 costs $35 and the case and heatsinks add less than $15. You do need a class 10 micro SD card with at least 16GB, so add another $10. For less than $60 you can have a RADIUS server. My advice though would be to spend a little more and get a 32GB SD card—the 16GB option does not leave as much room for growth and experimentation. That bumps the total to about $75. By the way, if you have an older Raspberry Pi Model B or B+ laying around, it will work. The minimum SD card size is 8GB.

There are quite a few steps involved in getting things up and running as you will see in the details later in this paper. But the big picture is not all that tough to understand. Here are the major tasks:

1. Acquire and build your Raspberry Pi and assemble it with mouse, keyboard, display, and Internet connection. All of the connections are standard off-the-shelf cables.

2. Download Raspbian and install it on the SD card. Boot it up and go through the initial OS installation and customization.

3. Install and configure Apache, MySQL, and PHP, turning this into a LAMP server.

4. Install and configure FreeRADIUS and the daloRADIUS GUI.

5. Test, tune, experiment, and enjoy.

Let's get started.

Building the hardware platform

As a starting point I suggest that you visit raspberrypi.org. There are links there to various distributors. I bought my Raspberry Pi 2 Model B from MCM Electronics. I purchased the case and heatsinks from Amazon. Be careful. Make sure your case is for the Raspberry Pi 2—not its predecessors. The class 10 micro SD card came from Walmart. I already had the other things laying around my house. For the display I used a small 1080p TV. The mouse and keyboard have to be USB devices. Note that the Pi is powered by a USB connection; so that requires a separate power supply. In my case the TV had a USB port on the back so I simply took power for the Pi from that. Cables you will need include: 1) an Ethernet cable, 2) an HDMI cable, and 3) a USB cable for power with USB A connector on one end and a micro USB connector on the other.

Obtaining Raspbian and installing the OS

On the raspbian.org website navigate to the Downloads page and scroll down to the Raspbian section. There are two important links here. First, the image installation guides will describe the steps needed to download and install the Raspbian system image. Second, there is a link to actually download your Raspbian image. I used the ZIP download.

Image installation guides are available for Windows, Mac OS X, and Linux. I went by the Windows guide. Using Win32DiskImager I burned the extracted Raspbian ISO image to the SD card. My laptop has an SD slot so this was painless. If your computer lacks that convenience then you will need some sort of adapter that lets you read and write SD cards.

After the system image was transferred to the SD card I installed the card into the Pi and connected everything up. Do not forget the network connection. When you power it on for the first time you will see the familiar Linux-style boot sequence. The Raspberry Pi Software Configuration Tool starts automatically on this first boot. Here are a few of the things that I did at this juncture:

1. Expanded the file system.

2. Enable dBoot to Desktop/Scratch. Chose the graphical desktop.

3. Went through Internalisation Options. I used en-US, set my time zone, and set generic 104-key U.S. keyboard. Careful, default settings are for the U.K.

4. Overclock is up to you. I set it to a modest 800MHz.

5. In Advanced Settings I changed the hostname to "radiuspi" and enabled SSH. It is very important to change the hostname.

6. Rebooted.

When the system comes alive you will be at the Raspbian graphical desktop. You are automatically logged in with the default user id "pi" and the default password "raspberry". From here on out you have a choice: you can use the LX Terminal feature to access the bash command shell, or you can SSH into the system from your favorite PC. At a minimum you will want to fire off LX Terminal and enter the ifconfig command to discover the IP address that you obtained.

From the command prompt you should update and upgrade your software. The commands are:

"`sudo apt-get update`" and "`sudo apt-get upgrade`".

At this point you have a perfectly nice bare bones Raspberry Pi Raspbian system. Now, it is time to customize it to meet our needs.

Install the LAMP components

LAMP is an acronym for Linux, Apache, MySQL, and PHP. Apache is the web server software that we will use. MySQL is the database software and PHP is the web scripting facility. This software cocktail is a powerful one that serves as a basis for many web-based applications. While these tools are not strictly needed to build a RADIUS server using FreeRADIUS, they are required for managing it via the web interface, daloRADIUS. From my perspective, being able to interact with the server using my browser is extremely convenient because I use this only occasionally and don't care to re-educate myself every time I go to make a change to the user information used by RADIUS.

I have to give credit where credit is due. There is a RaspiPress tutorial on the LAMP installation for the Pi and I used that to guide me through the installation. I deviated a bit here and there. But it is very nice to have the shared insights of others who have been through the process—much appreciated.

The vast majority of the steps that follow have to be carried out in the context of the root user. To avoid having to constantly type in the `sudo` command just enter the command "`sudo su`". Notice that the prompt changes from a "$" to a "#" to indicate that you are operating as the root user. Now just take it one piece at a time.

Apache

- `apt-get install apache2 apache2-doc apache2-utils`

PHP

- `apt-get install libapache2-mod-php5 php5 php-pear php5-xcache`
- `apt-get install php5-mysql`

MySQL

- `apt-get install mysql-server mysql-client`

Note: Make sure that you remember the root password that you enter here, you'll need it. I used the password "raspberry".

Change Your IP Address
Using nano (or your favorite editor) you need to modify /etc/network/interfaces. Where you see the line “iface eth0 inet dhcp” replace it with the following:

- auto eth0
- iface eth0 inet static
- address x.x.x.x (Note: I used 192.168.0.199)
- netmask x.x.x.x (Note: I used 255.255.255.0)
- gateway x.x.x.x (Note: My router is 192.168.0.1)
- dns-nameserver x.x.x.x (Note: My DNS server is 192.168.0.101)

Congratulations. You now have a LAMP server. It is a good idea to reboot, then you can test it over the network from your browser. The default home page simply reports “It works!” to you.

Install and Configure FreeRADIUS and the daloRADIUS GUI

Now, it is time to install RADIUS itself and the GUI that will make it a bit friendlier. We will be using FreeRADIUS as our RADIUS server software and daloRADIUS, a PHP application, to provide the web interface. Handy tips came from the Binary Heartbeat blog. Again, let’s proceed a step at a time.

FreeRADIUS Installation

- sudo su
- apt-get install freeradius freeradius-mysql php5-gd php-db

daloRADIUS Installation

- cd /usr/src
- wget http://downloads.sourceforge.net/project/daloradius/daloradius/daloradius0.9-9/daloradius-0.9-9.tar.gz (Note: careful, dash before the zero in the file name)
- tar xvfz daloradius-0.9-9.tar.gz -C /var/www
- mv /var/www/daloradius-0.9-9/ /var/www/daloradius
- cd /var/www/daloradius

Initialize the MySQL Database for FreeRADIUS to Use

- mysql -u root -p
- CREATE DATABASE radiusdb ;
- exit
- mysql -u root -p radiusdb < /var/www/daloradius/contrib/db/fr2-mysql-daloradius-and-freeradius.sql
- mysql -u root -p
- CREATE USER ‘radiususer’@’localhost’ ;
- SET PASSWORD FOR ‘radiususer’@’localhost’ = PASSWORD(‘radiuspass’) ;
- GRANT ALL ON radiusdb.* to ‘radiususer’@’localhost’ ;
- exit

Point daloRADIUS at the MySQL Database

You need to edit /var/www/daloradius/library/daloradius.conf.php. I used nano to do this. Change the following configuration values:

- $configValues['CONFIG_DB_USER'] = 'radiususer' ;
- $configValues['CONFIG_DB_PASS'] = 'radiuspass' ;
- $configValues['CONFIG_DB_NAME'] = 'radiusdb' ;

Configure FreeRADIUS

You need to edit /etc/freeradius/users. Again, I used nano. =The following lines are commented out; you need to remove the leading "#."

- #"John Doe" Cleartext-Password := "hello"
- # Reply-Message = "Hello, %(User-Name)"

You may as well edit /etc/freeradius/clients.conf at this point. I added the following lines at the bottom of the file so that any device on my office network could access the server.

- # Allow any address that starts with 192.168
- client 192.168.0.0/16 {
- secret = testing123
- shortname = office-network
- }

Test FreeRADIUS

Now make sure that FreeRADIUS initializes successfully using the following commands. You should see "Info: Ready to process requests" at the end of the initialization process.

- `service freeradius stop`
- `freeradius -XXX`

- If FreeRADIUS starts okay then you can type Ctrl-C to exit the program and restart it with:
- `service freeradius start`

There is a command line tool called radtest that can be used to exercise the RADIUS server. Type:

- `radtest "John Doe" hello localhost 0 testing123`

You should receive a response that says "Access-Accept".

Point FreeRADIUS to the MySQL Database

You need to edit /etc/freeradius/radiusd.conf. In the "modules" section remove the leading "#" from the following lines in the file:

- $INCLUDE sql.conf
- $INCLUDE sql/mysql/counter.conf

Then, edit /etc/freeradius/sql.conf to reflect the following:

- server = "localhost"
- login = "radiususer"
- password = "radiuspass"
- radius_db = "radiusdb"

Finally, you should edit /etc/freeradius/sites-enabled/default. Remove the leading "#" from the lines that say just "sql" in the following sections of the configuration file:

- "authorize"
- "accounting"
- "session"
- "post-auth"

Then, reboot your system with:

- reboot

Conclusion

Congratulations, you have a RADIUS server! Go ahead and give it a try. From another PC on the network, fire up your browser and point it at your new server. For me the URL was http://192.168.0.199/daloradius. Naturally, you will have to specify your own server's IP address. The daloRADIUS username is "administrator" and the password is "radius."

The steps outlined above have created a basic RADIUS server with the daloRADIUS graphical (web) user interface. There are a few things that I had to tweak to fix a couple of "gotchas." Here are my last few tips:

- Make sure that the owner and group on /var/www/daloradius/library/daloradius.conf.php is "www-data". Commands "chown" and "chgrp" are used to fix things if they are wrong.

- Adjust the permissions on the dmesg and syslog files in /var/log so that users other than root can read them. I used "chmod 644 …" on both files.

- Using daloRADIUS, click Config, click Logging Settings, and then enable logging of Queries and Actions.

- Adjust the permissions for the /var/log/freedadius directory. I used "chmod 755 …" on that.

- I did not have great luck using the Epiphany web browser installed with Raspbian. So I installed Chromium, and that worked quite a bit better.

I used a 64GB class 10 micro SD card because I did not know what to expect when I began the project. It turns out that the software installation consumed less than 3GB. That leads me to believe that one could get away with just a 16GB card or even an 8GB card.

The Raspberry Pi 2 has 1GB of RAM on board. Previous Pis have only 512MB, or for the older ones, just 256MB. Therefore the memory footprint of all of this software warrants our attention. On my system there is only 224MB of RAM in use, which lead me to believe that one might well get away with an older Raspberry Pi.

As far as the processor is concerned, the quad-core ARM processor in the Pi 2 makes for a snappy and very responsive system. The older Pis are not so blessed. I ran an experiment in which I repeated these instructions using a Raspberry Pi Model B, which has a single core ARM processor and 512MB of RAM. The SD card was only 8GB. It worked. Not only that, it was surprisingly responsive. This would be a perfectly nice platform for testing, demonstrations, and experimentation.
I hope you enjoy your new RADIUS server.

Learn More

Learn more about how you can improve productivity, enhance efficiency, and sharpen your competitive edge through training.

Understanding Networking Fundamentals
TCP/IP Networking
Enterprise Wi-Fi Security (CWSP)

Visit **www.globalknowledge.com** or call **1-800-COURSES (1-800-268-7737)** to speak with a Global Knowledge training advisor.

About the Author

George Mays is the president of G.W. Mays & Associates, Inc. based in San Antonio, Texas. George co-authored Villanova University's Mastering IS Security+ course and their CISSP and CASP courses. Additionally, he wrote ANRC's Network Traffic Analysis and Advanced Network Traffic Analysis courses, which he taught on a regular basis to security agencies in the United States. George has over 45 years of experience in computing, data communications, and networking.