Trapeze Networks Integration Guide

Revision	0.9
Date	27 May 2009
	Copyright © 2007 amigopod Pty Ltd
amigopod Head Office	amigopod Pty Ltd Suite 101 349 Pacific Hwy North Sydney, NSW 2060 Australia
	ABN 74 124 753 420
Web	www.amigopod.com
Phone	+61 2 8669 1140
Fax	+61 7 3009 0329

Table of Contents

Introduction

This document outlines the configuration process on both the Trapeze Networks Mobility Exchanges (MX) and the amigopod appliance to create a fully integrated Visitor Management solution. The solution leverages the captive portal functionality built into the Trapeze Mobility System Software (MSS). Trapeze uses the terminology of Web-Portal to refer to their internal captive portal functionality and it can be generally defined as follows:

Captive portal allows a wireless client to authenticate using a web-based portal. Captive portals are typically used in public access wireless hotspots or for hotel in-room Internet access. After a client associates to the wireless network, their device is assigned an IP address. The client must start a web browser and pass an authentication check before access to the network is granted. Captive portal authentication is the simplest form of authentication to use and requires no software installation or configuration on the client. The username/password exchange is encrypted using standard SSL encryption.

However, Captive Portal authentication does not provide any form of encryption beyond the authentication process; to ensure privacy of client data, some form of link-layer encryption (such as WEP or WPA-PSK) should be used when sensitive data will be sent over the wireless network.

Amigopod extends the standard Trapeze Web-Portal functionality by providing many advanced features such as a fully branded user interface, SMS integration for delivery of receipts, bulk upload of visitors for conference management, self provisioning of users for public space environments to name a few.

Trapeze Networks have reseller and OEM relationships with several other vendors and therefore the steps outlined in this document can equally be deployed on these 3rd party platforms.

Vendor	Products	amigopod verified	Partner Verified
TRAPEZE NETWORKS A BELDEN BRAND	MXR-2, MX-8, MX-216, MX-200, MX-2800	Yes – 7.0.9.6	
3com	WXR100, WX-1200, WX-2200		Yes -
NORTEL	2350, 2360, 2361, 2382, MX-2800		
enterasys	RBT-8100, RBT-8110, RBT-8400, RBT-8500		

Test Environment

The test environment referenced throughout this integration guide is based on a Trapeze MXR-2 Mobility Exchange. Although this low end hardware platform has been used, the testing and therefore this procedure is valid for all hardware variants from Trapeze and their OEM partners as it is the MSS software that is providing the integration points with amigopod.

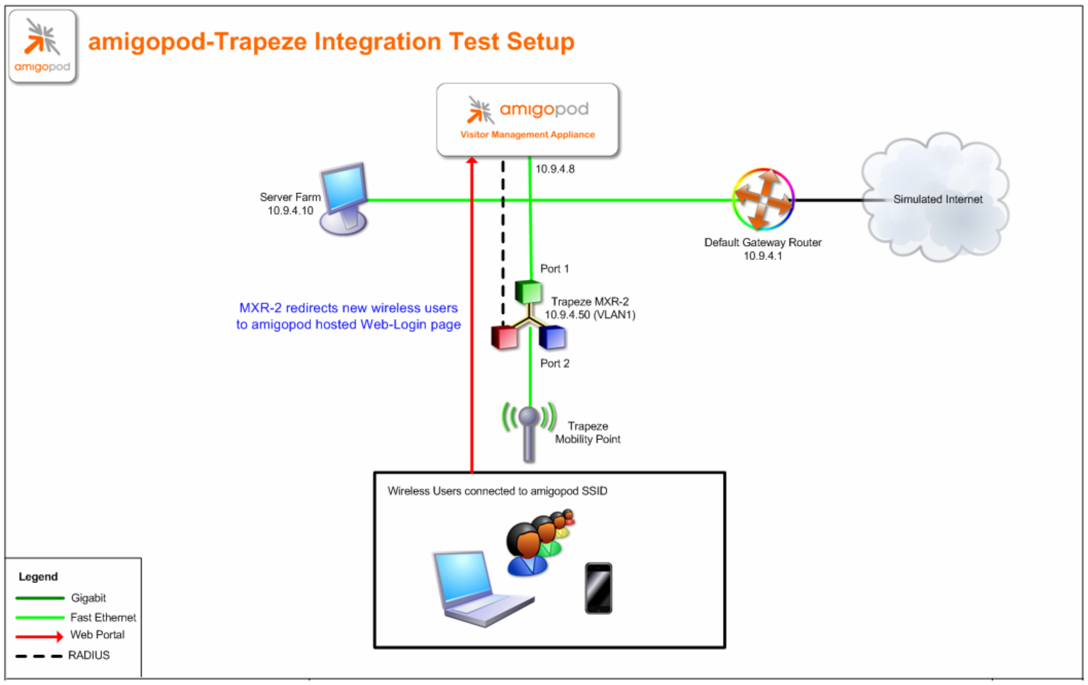
The following table shows the software versions used during the integration testing. This document will be updated in the future if changes in either amigopod or Trapeze subsequent releases affect the stability of this integration. It is advised that the customer always check for the latest integration guide available from either amigopod or Trapeze.

Date Tested:	May 2009
AmigoPod Version:	Kernel→1.9.6, Radius Services→ 1.9.5
Plugins Required:	Standard build only
MSS Version:	7.0.9.6
Integration:	HTTP Captive Portal

Amigopod was deployed locally on the LAN interface of the Trapeze controller as a VMWare image running on a test laptop. Although the VMWare image has been used the integration is equally valid for the amigopod appliance and self installing DVD deployment variants.

MXR-2 IP Address	10.9.4.50
Internet Gateway Address	10.9.4.1
amigopod IP Address	10.9.4.8
amigopod RADIUS port	Auth 1812 Acc 1813 (default settings)

The following diagram provides a high level overview of the test lab topology:



Integration

Although the MXR-2 MSS supports both internal and external Captive portal functionality, this integration guide will focus on the later as the internal Web-Portal dictates the use of the internal Login Page resident on the controller itself. The Login page is very basic and doesn’t allow for significant customization as is possible with the amigopod Web Logins feature.

Note: Trapeze now allows for fully customised Captive portal pages to be uploaded to the controller but this process requires a significant amount of web design and javascript experience to produce a professional result. One of amigopod’s strongest selling points is the Skin Plugin technology where the presentation of the User Interface is separated from the mechanics of the underlying application. This allows amigopod to supply end users with a ready branded Skin for all amigopod interaction (both Visitor and Administrators) for a small nominal fee at time of purchase.

The integration will also leverage the MSS’ ability to define and reference external RADIUS servers for the authentication and accounting of visitor accounts. In the standalone Trapeze Guest provisioning solution the local database in each controller is used to store user credentials, limiting the solution to the scope of the local deployment. With the introduction of amigopod, all visitor accounts are created, authenticated and accounted for on the amigopod internal RADIUS Server.

Trapeze do offer their SmartPass solution for centralizing Guest Management functions but this does not address the branding, interactive self registration & extensive reporting functions found natively in the amigopod solutions.

Trapeze has strong support for the RADIUS extensions defined in RFC3576 which allows dynamic authorization of wireless users from the configured RADIUS servers. This allows amigopod administrators to list and disconnect users at will or even deploy automated policies to disconnect users that exceed some pre-determined traffic profiles.

This deeper integration is detailed in Appendix A.

Amigopod Configuration

The following configuration procedure assumes that the amigopod software or appliance has been powered up and a basic IP configuration has been applied through the setup wizard to allow the administrator to access the Web User Interface. The following table again reviews the IP Addressing used in the test environment but this would be replaced with the site specific details of each customer deployment:

MX IP Address	10.9.4.50
Internet Gateway Address	10.9.4.1
amigopod IP Address	10.9.4.8
amigopod RADIUS port	Auth 1812 Acc 1813 (default settings)

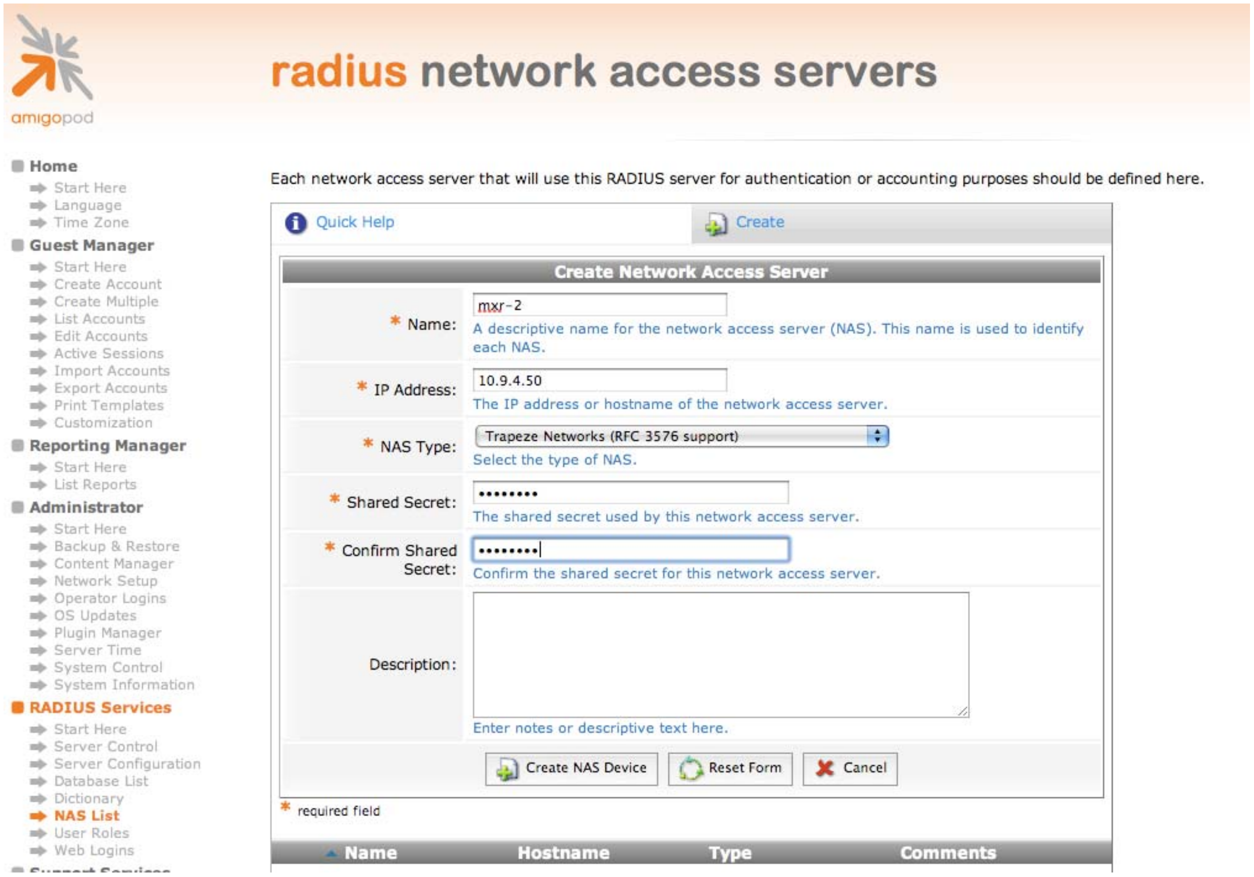
Please refer to the amigopod Quick Start Guide for more information on the basic configuration of the amigopod software.

Step 1 – Create RADIUS NAS for Trapeze Controller

In order for the Trapeze controller to authenticate users it needs to be able to communicate with the amigopod RADIUS instance. This step configures the amigopod NAS definition for the Trapeze Controller. The RADIUS key used here needs to be configured exactly the same as what will be configured on the MXR-2 for the RADIUS transactions to be successful.

For simplicity we will use a shared secret of ***wireless***. Please note this as it will be required in the first step of the Trapeze configuration.

From the *RADIUS Services→Network Access Servers* screen click on the *Create* button to add a new NAS device. Enter the IP Address of the Trapeze Controller, set the *NAS Type* as *Trapeze Networks* (*RFC 3576 Support)* and enter the key of *wireless* in the *Shared Secret* field.



Each network access server that will use this RADIUS server for authentication or accounting purposes should be defined here.

Create Network Access Server

* Name:	mxr-2 A descriptive name for the network access server (NAS). This name is used to identify each NAS.
* IP Address:	10.9.4.50 The IP address or hostname of the network access server.
* NAS Type:	Trapeze Networks (RFC 3576 support) Select the type of NAS.
* Shared Secret:	•••••••• The shared secret used by this network access server.
* Confirm Shared Secret:	•••••••• Confirm the shared secret for this network access server.
Description:	Enter notes or descriptive text here.

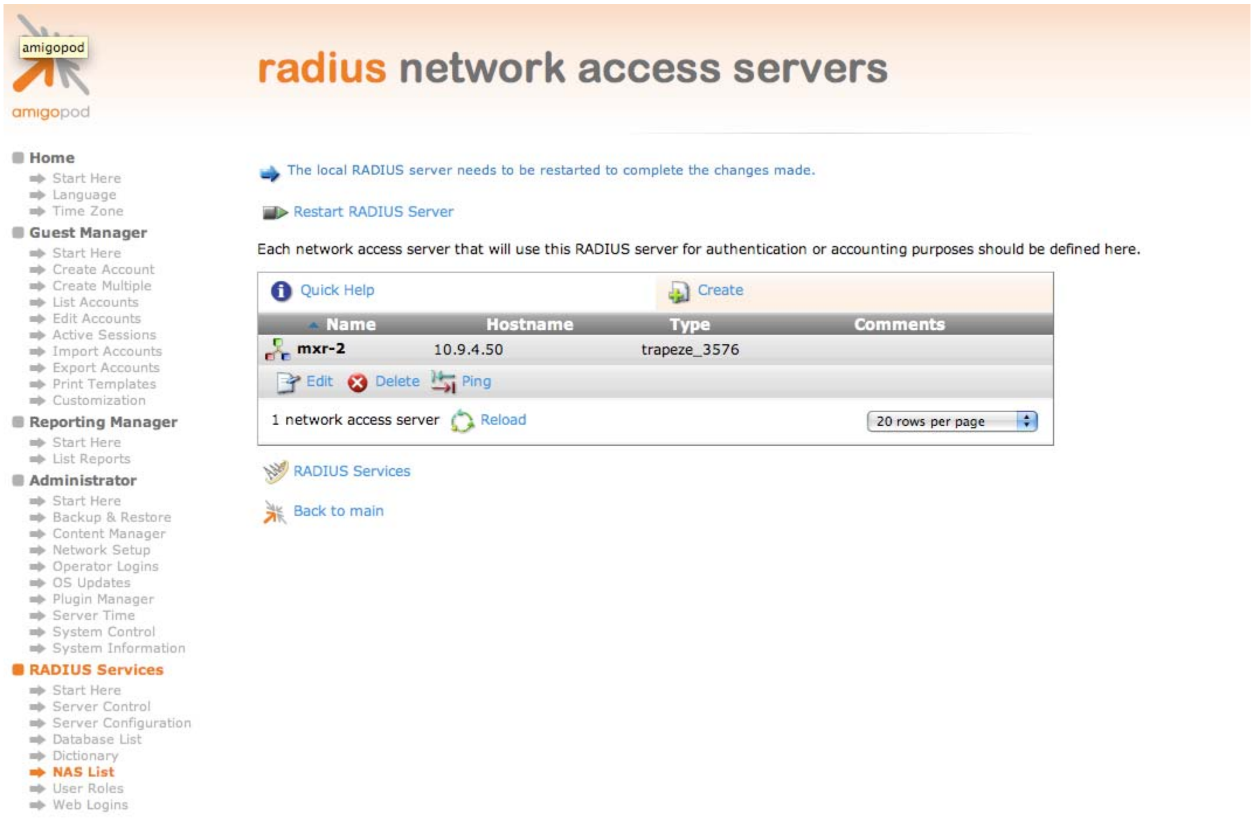
* required field

Click the *Create NAS* button to commit the change to the RADIUS database.

Step 2 – Restart RADIUS Services

A restart of the RADIUS Service is required for the new NAS configuration to take effect.

Click the *Restart RADIUS Server* button shown below and wait a few moments for the process to complete.



The local RADIUS server needs to be restarted to complete the changes made.

Restart RADIUS Server

Each network access server that will use this RADIUS server for authentication or accounting purposes should be defined here.

Quick Help   Create

Name	Hostname	Type	Comments
mxr-2	10.9.4.50	trapeze_3576	

Edit Delete Ping

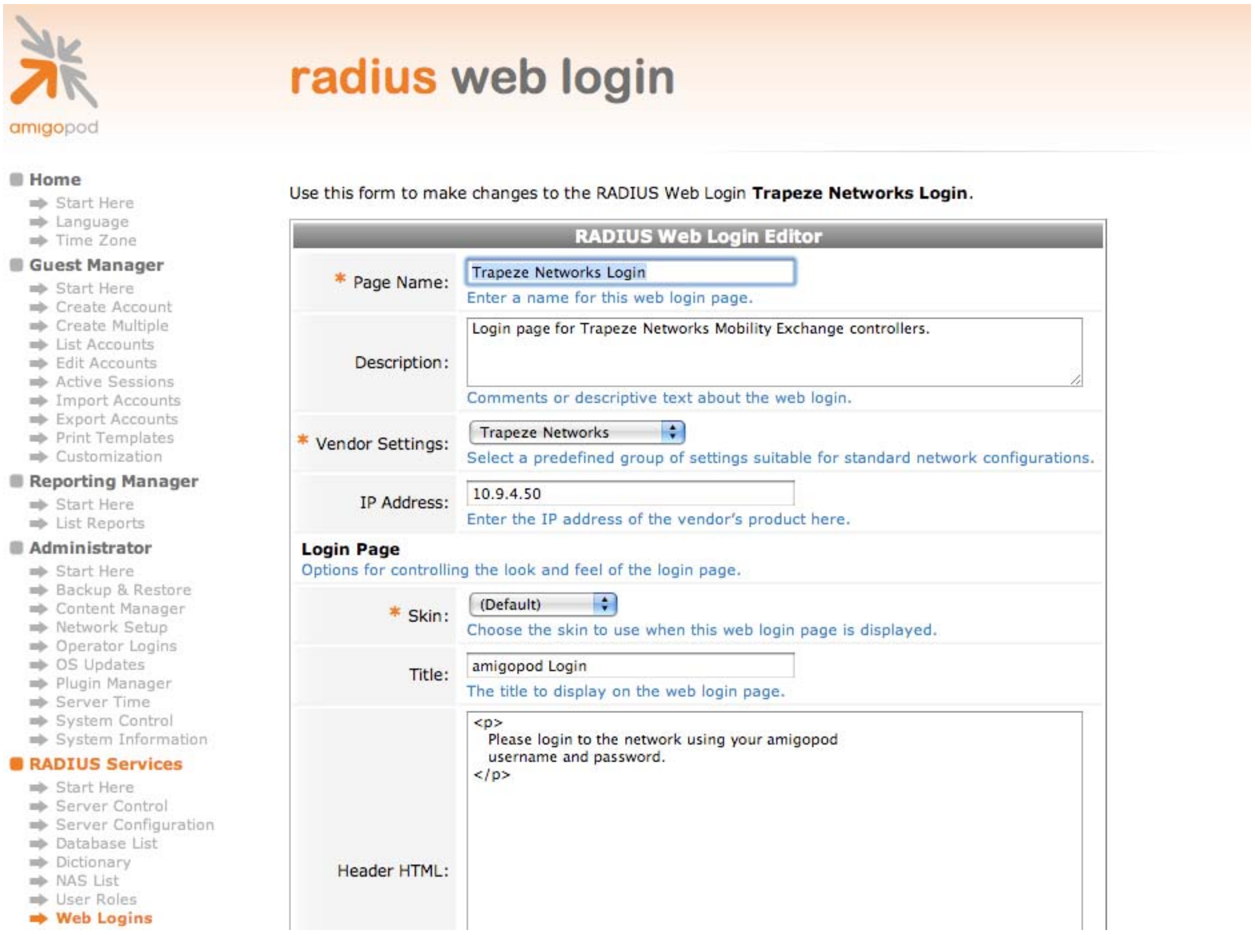
1 network access server Reload   20 rows per page

RADIUS Services

Back to main

Step 3 – Create a Web-Login Page

From the *RADIUS Services→Web Logins* page select the *Trapeze Networks Login* entry and Click the *Edit* button. From the *RADIUS Web* Login page enter the IP Address of the Trapeze MXR-2 and select the *Skin* that you would like presented as the branding for the Captive Portal page.



Use this form to make changes to the RADIUS Web Login **Trapeze Networks Login**.

RADIUS Web Login Editor

* Page Name:	Trapeze Networks Login Enter a name for this web login page.
Description:	Login page for Trapeze Networks Mobility Exchange controllers. Comments or descriptive text about the web login.
* Vendor Settings:	Trapeze Networks Select a predefined group of settings suitable for standard network configurations.
IP Address:	10.9.4.50 Enter the IP address of the vendor's product here.

Login Page
Options for controlling the look and feel of the login page.

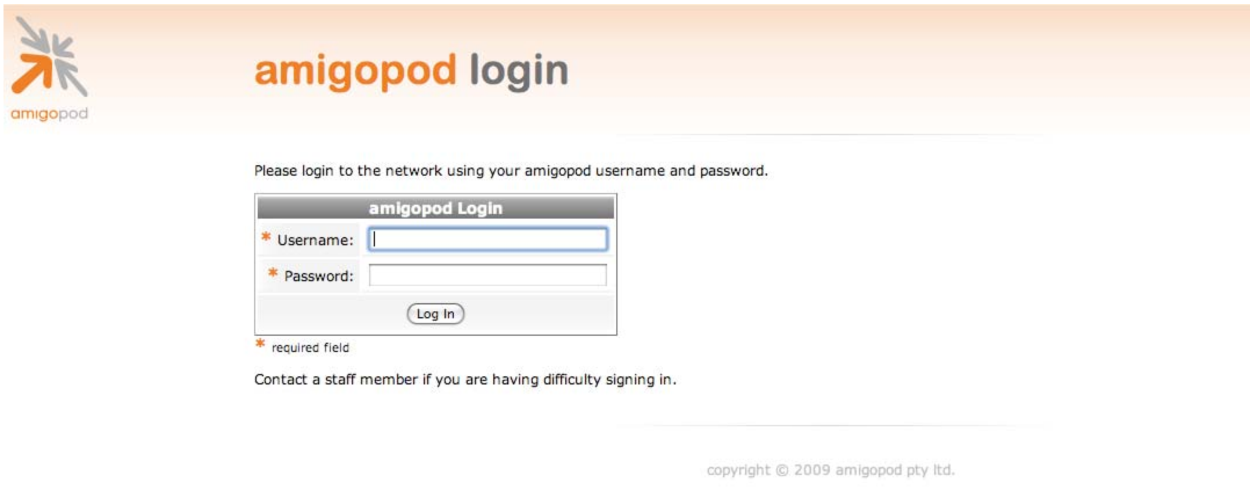
* Skin:	(Default) Choose the skin to use when this web login page is displayed.
Title:	amigopod Login The title to display on the web login page.
Header HTML:	see below

```
<p>
  Please login to the network using your amigopod
  username and password.
</p>
```

Modify the sample HTML in the *Header HTML, Footer HTML* and *Login Message* section to customize for your local environment. Click the *Save Changes* button to commit the changes.

Step 4 - Review to Web Login Captive Portal page

Returning to the *Web Logins* page, select the *Trapeze Networks Login* entry and Click the *Test* button and in a new window the configured captive portal page will be displayed as shown below:



Please login to the network using your amigopod username and password.

amigopod Login	
* Username:	
* Password:	

* required field

Contact a staff member if you are having difficulty signing in.

copyright © 2009 amigopod pty ltd.

Click the Back button in the web browser to return to the amigopod configuration screen.

Note: Make note of the URL presented in the web browser after the *Test* button has been clicked. This URL will be required in the configuration of the Web Portal settings on the Trapeze controller. An example of the URL is shown below:

http://10.9.4.8/weblogin.php/6

Trapeze MSS Configuration

The following configuration procedure assumes that the Trapeze Mobility Exchange has been powered up and a basic IP configuration has been applied through the Quick Start CLI to allow the administrative access. The following table again reviews the IP Addressing used in the test environment but this would be replaced with the site specific details of each customer deployment:

MXR-2 IP Address	10.9.4.50
Internet Gateway Address	10.9.4.1
amigopod IP Address	10.9.4.8
amigopod RADIUS port	Auth 1812 Acc 1813 (default settings)

Below is the configuration snippet of the basic IP configuration that is assumed for the test lab environment:

```
set ip dns domain amigopod.com
set ip dns enable
set ip route default 10.9.4.1 1
set ip dns server 10.9.4.1 PRIMARY
set system name mxr-2
set system ip-address 10.9.4.50
set port poe 2 enable
set vlan 1 port 1
set interface 1 ip 10.9.4.50 255.255.255.0
set interface 1 ip dhcp-server enable
```

Step 1 – Create RADIUS Definition for amigopod

From the Trapeze CLI ensure you are in enable mode by checking the # suffix on the hostname as shown below:

```
 mxr-2#
```

Enter the following two set commands to create firstly a RADIUS server definition for amigopod including the IP address and shared secret and then a server group called for example *radius* with the new amigopod RADIUS definition as a member.

```
set radius server amigopod address 10.9.4.8 key wireless
set server group radius members amigopod
```

You should receive the following confirmation message after each set command:

```
success: change accepted.
```

Note: The key above needs to be the same as the one defined in Step 1 of the amigopod configuration. For example, ***wireless***.

Step 2 – Create the Captive Portal service-profile

A service profile within the context of the Trapeze configuration represents a set of options that may be configured and deployed on the wireless network. Services define networking specifics such as SSID, authentication type, local or RADIUS authentication, encryption and VLAN mappings.

Below are the set commands used to create the basic service definition for our Captive Portal test environment:

```
set service-profile captive-portal ssid-name amigopod
set service-profile captive-portal ssid-type clear
set service-profile captive-portal auth-fallthru web-portal
set service-profile captive-portal web-portal-acl portalacl
set service-profile captive-portal attr vlan-name default
```

You can see from the above commands the following wireless configuration settings have been applied to the *captive-portal* service profile:

- The SSID has been defined as *amigopod*
- There wireless authentication is set to Open (clear)
- The fall through authentication is based on *web-portal*
- A default Access Control List (ACL) has been assigned to redirect traffic (*portalacl*)
- The wireless SSID has been mapped to the default VLAN (this is likely to change in production deployment for security and separation reasons).

Step 3 – Enable Logout function for wireless users

By default the Trapeze controller will not display a Logout pop-up window to allow the user to manually terminate their captive portal session. The session is either ended by the expiry of their session timer or idle timer.

To enable the Logout pop-up, enter the following set command from the enable prompt:

```
set service-profile captive-portal web-portal-logout mode enable
```

Step 4 – Enable RADIUS Authentication & Accounting

The next step is to enable both RADIUS Authentication and Accounting for the newly create *amigopod* SSID. This is done by entering the following two set commands from the enable prompt:

```
set authentication web ssid amigopod ** radius
set accounting web ssid amigopod ** start-stop radius
```

Please note if you are not familiar with the ** notation above, refer to the Trapeze documentation regarding User Glob definitions. Essentially the ** is indicating any user at any domain will apply to this configuration. Globs allow for advanced configurations such as proxy RADIUS using configuration such as @realm for user differentiation.

Step 5 – Modify default *portalacl* to allow traffic to amigopod

The default ACL created by Trapeze when configuring web-portal only allows DHCP traffic and all other traffic is captured and redirected to the defined web-portal page as shown below:

```
set security acl name portalacl permit udp 0.0.0.0 255.255.255.255 eq 68
0.0.0.0 255.255.255.255 eq 67
set security acl name portalacl deny 0.0.0.0 255.255.255.255 capture
```

Given we intend on hosting the Captive Portal on the amigopod to leverage a fully branded Web Login environment, traffic needs to be also permitted to the IP address of the amigopod. In our test environment the amigopod IP address is *10.9.4.8* as shown in the summary diagram.

To modify the ACL from the CLI the ACL must first be removed and then re-added in the correct sequence of entries as shown below:

```
clear security acl name portalacl
set security acl name portalacl permit udp 0.0.0.0 255.255.255.255 eq 68
0.0.0.0 255.255.255.255 eq 67
set security acl name portalacl permit tcp 0.0.0.0 255.255.255.255 10.9.4.8
0.0.0.0 eq 80
set security acl name portalacl deny 0.0.0.0 255.255.255.255 capture
commit security acl portalacl
```

You can see from above an additional entry has been added to the *portalacl* to allow TCP port 80 traffic (HTTP for the basic Web-Login experience) to the amigopod on 10.9.4.8. The commit command at the end is required to save ACL changes as a whole.

Please note that when making modifications to ACLs using the Trapeze Ringmaster management suite, you do not need to remove and reapply the ACL to make changes. Inline changes can be performed live on the ACL.

Step 6 – Configure Trapeze to redirect new users to amigopod

Now that we have created the new amigopod Web-Login in the previous section, we need to configure the MXR-2 to redirect any unauthenticated users to the amigopod to display the login page. Based on the URL, presented in the last section, enter the following set command to configure the redirect process:

```
set service-profile captive-portal web-portal-form
http://10.9.4.8/weblogin.php/6
```

Step 7 – Apply new SSID to radio profile

In order for the new SSID to be made available for wireless users, the newly created SSID configuration needs to be applied to either the default radio profile or a specific radio profile based on your network design. If you are not familiar with radio profiles, please refer to the Trapeze documentation for further information on how radio profiles allow you to partition your wireless network into functional groups.

For simplicity the following example applies the new SSID configuration to the default radio profile. This may not be the desired operation in your production network.

```
set radio-profile default service-profile captive-portal
```

Step 8 – Save new configuration

Now that the new Web-Portal configuration changes are complete, please enter the following command to save the configuration:

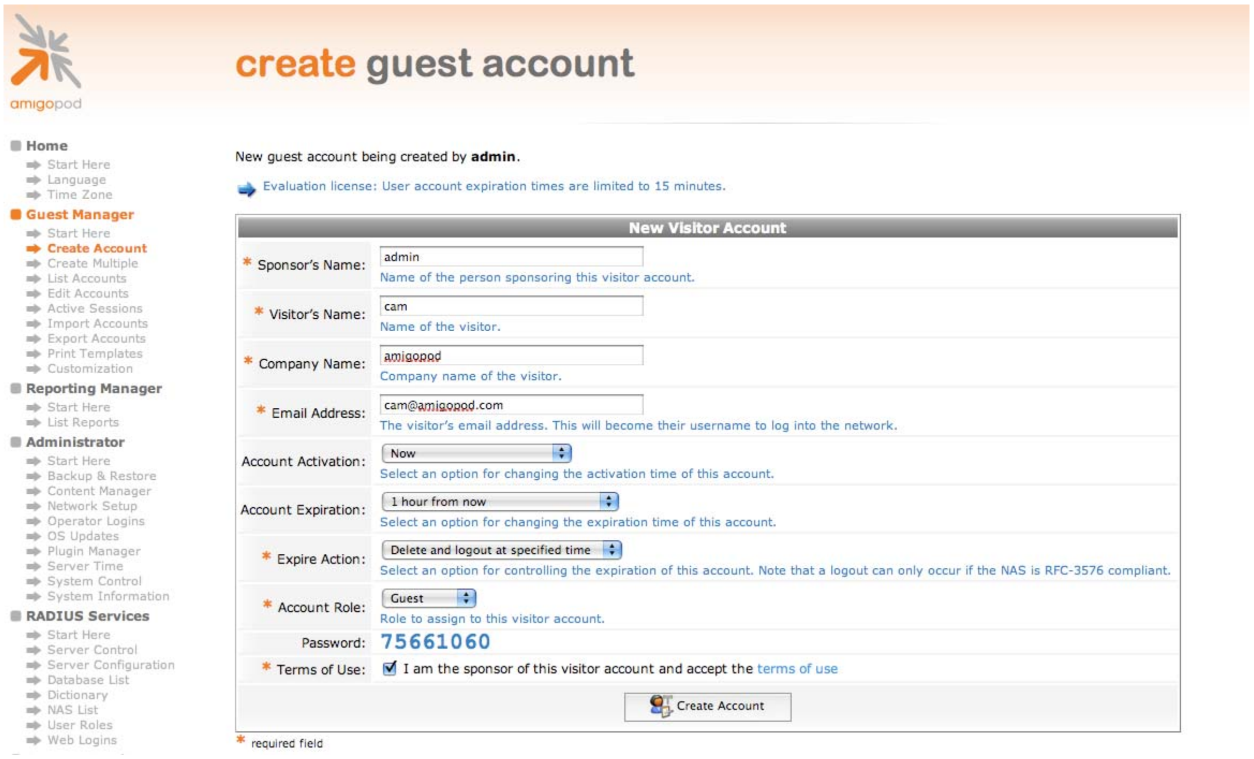
```
save configuration
```

Testing the Configuration

Now that the configuration of both the Trapeze Controller and the amigopod solution is complete, the following steps can be followed to verify the setup.

Step 1 – Create a test user account

Within the amigopod RADIUS Server a test user account can be created using the amigopod *Guest Manager.* From the *Guest Manager* menu, select the *Create New Guest Account* option. Enter the test user details as detailed on the form below and click the *Create Account* button to save the new test user account.



New guest account being created by **admin**.

Evaluation license: User account expiration times are limited to 15 minutes.

New Visitor Account

* Sponsor's Name:	admin Name of the person sponsoring this visitor account.
* Visitor's Name:	cam Name of the visitor.
* Company Name:	amigopod Company name of the visitor.
* Email Address:	cam@amigopod.com The visitor's email address. This will become their username to log into the network.
Account Activation:	Now Select an option for changing the activation time of this account.
Account Expiration:	1 hour from now Select an option for changing the expiration time of this account.
* Expire Action:	Delete and logout at specified time Select an option for controlling the expiration of this account. Note that a logout can only occur if the NAS is RFC-3576 compliant.
* Account Role:	Guest Role to assign to this visitor account.
Password:	75661060
* Terms of Use:	✓ I am the sponsor of this visitor account and accept the terms of use

* required field

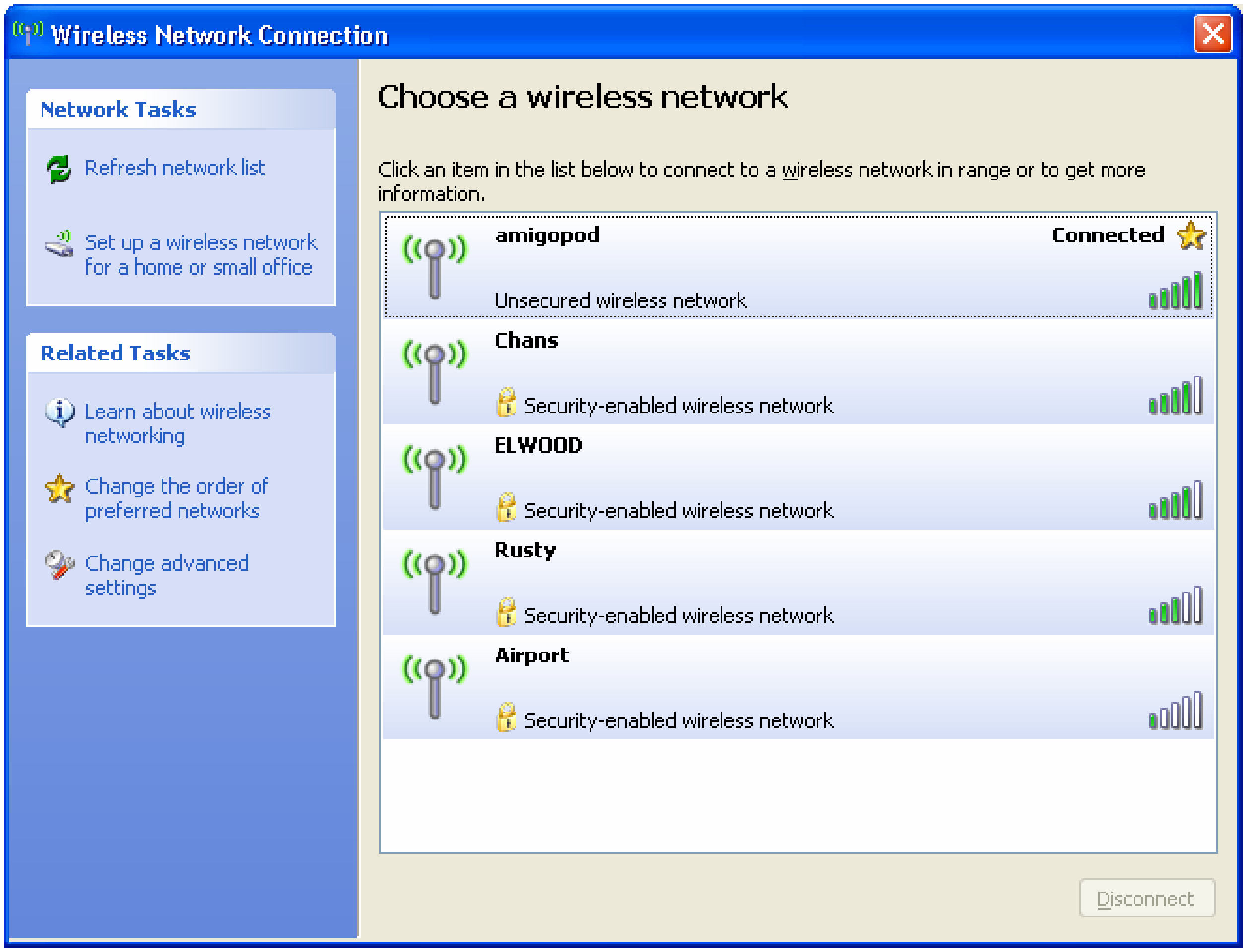
Note: Make note of the randomly generated *Visitor Password* as this will be required during the integration testing. If this password is proving difficult to remember during testing you can use the *List guest accounts* option on the screen to then edit the account and change the password to a more user friendly string.

For simplicity during our testing we took this option and changed the username to ***cam*** and password to ***wireless.*** All subsequent screenshots and debugs will reflect this change.

Step 2 - Connect to the amigopod wireless network

Using a test laptop with a compatible 802.11 based wireless card attempt to connect to the advertised *amigopod* wireless network. The screen capture below shows the interface used on a Windows XP SP2 based laptop. Although the process differs from laptop to laptop depending on the wireless card drivers installed and different operating systems in use, the basic premise of connecting to the unsecured Guest Wireless network should be fundamentally the same. Refer to your laptop manufacturer's documentation on the procedure for connecting to wireless networks if you experience basic connectivity.

Note: If the *amigopod* wireless network is not visible from the test laptop, double check the configuration of the Trapeze Controller and potentially source a second wireless test device to see if the problem is laptop specific.


Wireless Network Connection
Network Tasks
Refresh network list
Set up a wireless network for a home or small office
Related Tasks
Learn about wireless networking
Change the order of preferred networks
Change advanced settings
Disconnect

Choose a wireless network

Click an item in the list below to connect to a wireless network in range or to get more information.

amigopod — Connected — Unsecured wireless network

Chans — Security-enabled wireless network

ELWOOD — Security-enabled wireless network

Rusty — Security-enabled wireless network

Airport — Security-enabled wireless network

Step 2 – Confirm DHCP IP Address received

Using the Windows Command Prompt or equivalent in the chosen operating system, confirm that a valid IP Address has been received from the DHCP server configured on the Trapeze Controller.

Issue the *ipconfig* command from the Windows Command Prompt to display the IP information received from the DHCP process. By checking on the Wireless adaptor you should be able to confirm an IP Address in the range of *10.9.4.x* has been received.

Note: On Mac OS X and Linux operating system variants use a Terminal window and enter the *ifconfig* command to display the same information.

Step 3 – Confirm session detected by Trapeze Controller

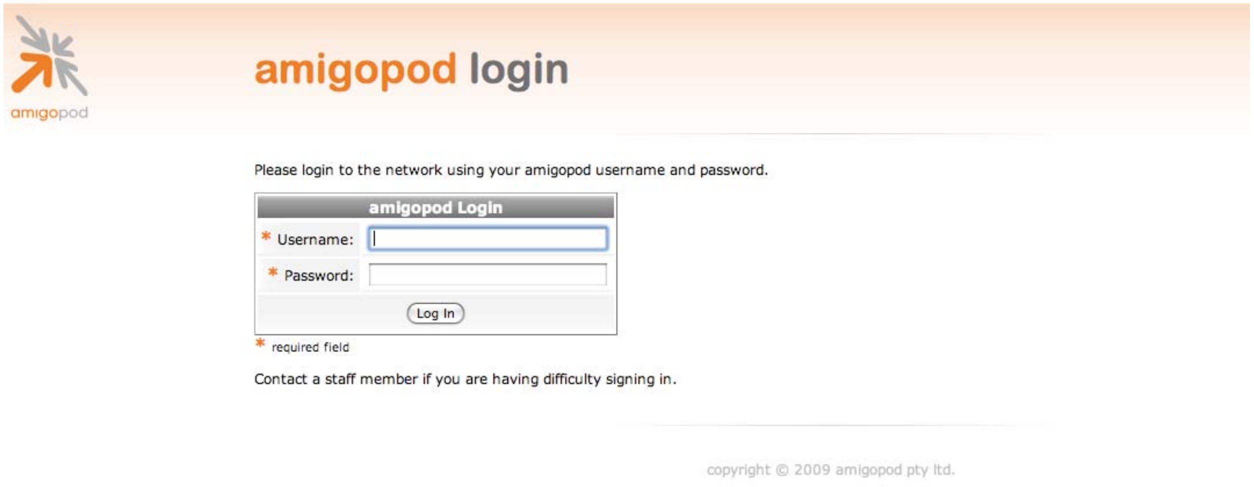
Once you have received an IP address, the Trapeze controller should have detected the new user and placed them in an unauthenticated state which can be verified by issuing the *show sessions* command from the CLI as shown below:

```
mxr-2# show sessions

1 session total

User Name             SessID  Type  Address           VLAN
AP/Radio
--------------------- ------  ----- ----------------- -------------- -------
-
web-portal-amigopod        4  -     10.9.4.12         default        1/2

mxr-2#
```

As you can see above, the test laptop IP address of *10.9.4.12* is currently under the control of amigopod based Web-Portal process.

Step 4 – Launch Web Browser and login

When the web browser on the test laptop is launched the Trapeze *portalacl* will automatically capture the session and redirect the user to the amigopod hosted login page as shown below:



Please login to the network using your amigopod username and password.

amigopod Login

* Username:	
* Password:	

* required field

Contact a staff member if you are having difficulty signing in.

copyright © 2009 amigopod pty ltd.

Enter the test user details entered and recorded in Step 1 above and click the *Login* button.

At this point the test user should be successfully authenticated and allowed to transit through the controller and onto the Internet or Corporate network.

Note: If the web browser fails to redirect check that the DNS server configured in the base Trapeze configured defined before Step 1 is available and successfully resolving domain names. Without name resolution working the web browser will never attempt to connect to the website defined in web browser home page and therefore there is no session for the Trapeze controller to redirect. Other situations that can cause issues with the captive portal include but are not limited to:

- Web browser home page set to intranet site not available in current DNS
- Proxy Server configuration in browser using non standard HTTP ports

Step 5 – Confirm the login successful from Trapeze

From the Trapeze CLI if you issues the *show sessions* command again you will now see the test user name and the star indicating that the user has been successfully authenticated:

```
mxr-2# show sessions

1 session total

User Name             SessID  Type  Address           VLAN
AP/Radio
--------------------- ------  ----- ----------------- -------------- -------
-
cam                        4* web   10.9.4.12         default         1/2

mxr-2#
```

Step 6 – Confirm RADIUS debug messages on amigopod

Once the test laptop has successfully authenticated and now able to browse the Internet, an entry should appear in the RADIUS logs confirming the positive authentication of the test user – in this example, *cam*.

Select the *RADIUS Services→Server Control* menu option and the screen displayed will show the status of the RADIUS server and a tail of the log file, including an entry for the positive authentication transaction.

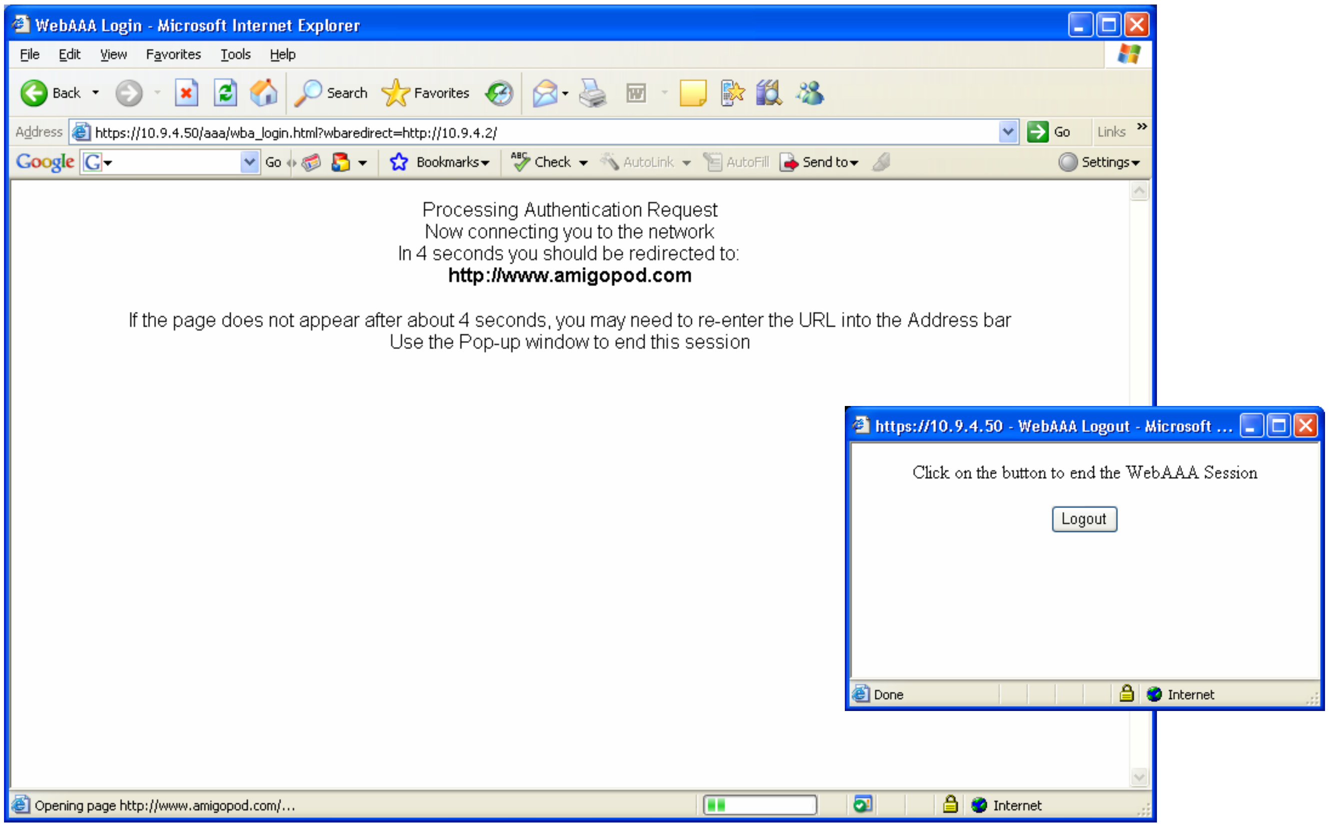
This is a useful tool to remember when troubleshooting user authentication issues. A more advanced debugging tool is also available from this screen using the *Debug RADIUS Server* button. The following output is an example from the RADIUS debugs for this transaction:

```
Ready to process requests.
rad_recv: Access-Request packet from host 10.9.4.50:20000, id=3, length=117
User-Name = "cam"
Calling-Station-Id = "00-40-96-A1-F3-99"
Called-Station-Id = "00-0B-0E-90-B8-83:amigopod"
NAS-Port = 6
NAS-Port-Type = Wireless-802.11
NAS-IP-Address = 10.9.4.50
NAS-Identifier = "Trapeze"
User-Password = "wireless"
rlm_sql (sql): Reserving sql socket id: 3
rlm_sql_postgresql: query: SELECT id, UserName, Attribute, Value, Op FROM
radcheck WHERE Username='cam' ORDER BY id
```

```
rlm_sql_postgresql: Status: PGRES_TUPLES_OK
rlm_sql_postgresql: affected rows =
rlm_sql_postgresql: query: SELECT radgroupcheck.id, radgroupcheck.GroupName,
radgroupcheck.Attribute, radgroupcheck.Value,radgroupcheck.Op ??FROM
radgroupcheck, usergroup WHERE usergroup.Username = 'cam' AND
usergroup.GroupName = radgroupcheck.GroupName ??ORDER BY radgroupcheck.id
rlm_sql_postgresql: Status: PGRES_TUPLES_OK
rlm_sql_postgresql: affected rows =
rlm_sql_postgresql: query: SELECT id, UserName, Attribute, Value, Op FROM
radreply WHERE Username='cam' ORDER BY id
rlm_sql_postgresql: Status: PGRES_TUPLES_OK
rlm_sql_postgresql: affected rows =
rlm_sql_postgresql: query: SELECT radgroupreply.id, radgroupreply.GroupName,
radgroupreply.Attribute, radgroupreply.Value, radgroupreply.Op ??FROM
radgroupreply,usergroup WHERE usergroup.Username = 'cam' AND
usergroup.GroupName = radgroupreply.GroupName ??ORDER BY radgroupreply.id
rlm_sql_postgresql: Status: PGRES_TUPLES_OK
rlm_sql_postgresql: affected rows =
rlm_sql (sql): Released sql socket id: 3
Exec-Program: /usr/bin/php /opt/amigopod/www/amigopod_request.php 2 280
Exec-Program: returned: 0
Login OK: [cam] (from client mxr-2 port 6 cli 00-40-96-A1-F3-99)
rlm_sql (sql): Processing sql_postauth
rlm_sql (sql): Reserving sql socket id: 2
rlm_sql_postgresql: query: INSERT INTO radpostauth (username, pass, reply,
authdate) VALUES ('cam', 'wireless', 'Access-Accept', NOW())
rlm_sql_postgresql: Status: PGRES_COMMAND_OK
rlm_sql_postgresql: affected rows = 1
rlm_sql (sql): Released sql socket id: 2
Sending Access-Accept of id 3 to 10.9.4.50 port 20000
Reply-Message = "Guest"
```

Step 7 – Check User Experience

After successful login the user web browser should be displayed with a holding page informing them that they are about to be redirected to their original requested page (in our example www.amigopod.com) and also the Logout pop-up box should be displayed as shown below:


WebAAA Login - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites
Address https://10.9.4.50/aaa/wba_login.html?wbaredirect=http://10.9.4.2/
Go Links
Bookmarks Check AutoLink AutoFill Send to Settings
Opening page http://www.amigopod.com/...
Internet
https://10.9.4.50 - WebAAA Logout - Microsoft ...
Logout
Done

Processing Authentication Request
Now connecting you to the network
In 4 seconds you should be redirected to:
http://www.amigopod.com

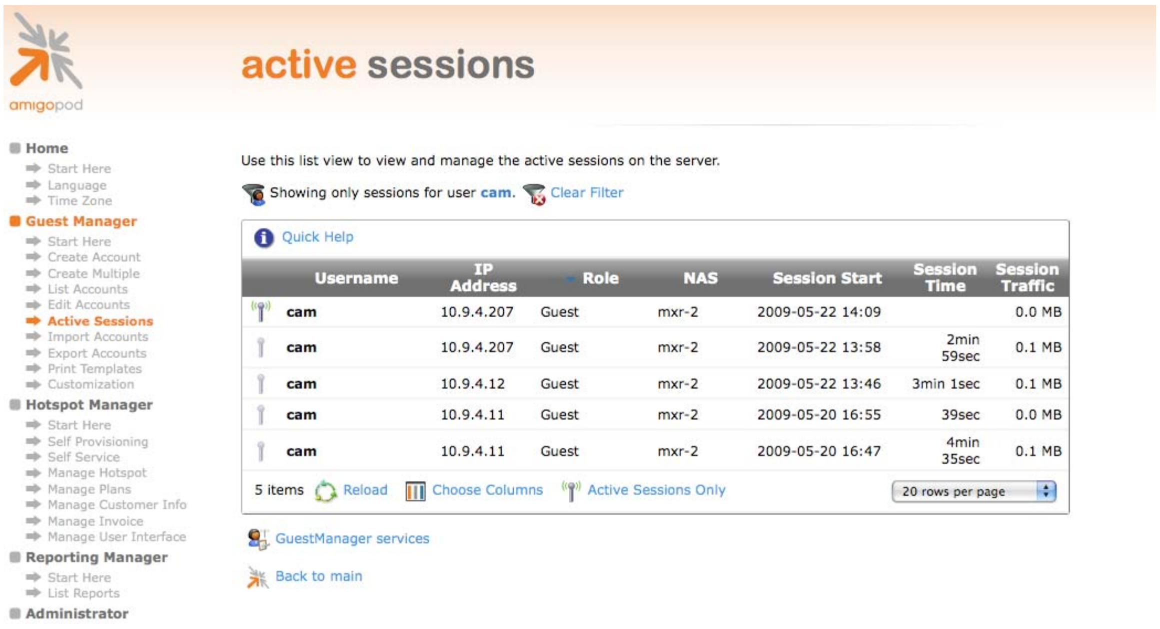
If the page does not appear after about 4 seconds, you may need to re-enter the URL into the Address bar
Use the Pop-up window to end this session

Click on the button to end the WebAAA Session

Appendix A – Dynamic Authorisation (RFC 3576)

The Trapeze Mobility Exchanges have strong in built support for RFC 3576 which is an extension of the RADIUS standard that allows RADIUS servers to participate in the dynamic disconnect or reauthorization of authenticated users.

This is of particular interest in some customer environments where they may wish to use the amigopod to disconnect users on an ad-hoc basis by listing the *Guest Manager → Active Sessions* as shown below:



Use this list view to view and manage the active sessions on the server.

Showing only sessions for user **cam.** Clear Filter

Username	IP Address	Role	NAS	Session Start	Session Time	Session Traffic
cam	10.9.4.207	Guest	mxr-2	2009-05-22 14:09		0.0 MB
cam	10.9.4.207	Guest	mxr-2	2009-05-22 13:58	2min 59sec	0.1 MB
cam	10.9.4.12	Guest	mxr-2	2009-05-22 13:46	3min 1sec	0.1 MB
cam	10.9.4.11	Guest	mxr-2	2009-05-20 16:55	39sec	0.0 MB
cam	10.9.4.11	Guest	mxr-2	2009-05-20 16:47	4min 35sec	0.1 MB

5 items

This functionality can also be extended to dynamically disconnect or potentially rate limit users that breach a particular traffic profile.

The following are the additional configuration steps required on the Trapeze controller to enable the Dynamic Authorisation support as per RFC 3576. As you will recall from the amigopod configuration section earlier, we have already defined the Trapeze NAS definition to be RFC3576 Support compatible and therefore no further configuration of the amigopod is required.

Note: RADIUS accounting must be enabled as per Step 4 in the Trapeze Configuration section earlier to ensure the Active Sessions screen above is displayed correctly.

Step 1 – Configure amigopod as a DAC entry

Enter the following set command at the enable prompt of the CLI to enable the amigopod on *10.9.4.8* to be able to send RFC3576 messages to the Trapeze. Please note that the key is still the same as the entry configured in Step 1 of the Trapeze configuration so it matches the NAS definition on the amigopod.

```
set radius dac amigopod address 10.9.4.8 replay-protect disable key wireless
```

Note: RFC 3576 uses UDP port 3799 by default so if your deployment places the amigopod on the other side of a router or firewall with Access Control Lists you will also need to permit this port along with the standard RADIUS ports of 1812 and 1813.

Step 2 – Enable authorization for the amigopod SSID

In order for Dynamic Authorisation to succeed the SSID in question on the Trapeze needs to have *authorization* enabled. Enter the following set command on the CLI to enable authorization on the amigopod SSID.

```
set authorization dynamic ssid amigopod amigopod
```

The first reference to *amigopod* is the SSID and the second reference is to the amigopod RADIUS server definition. Please modify these to suit your deployment.

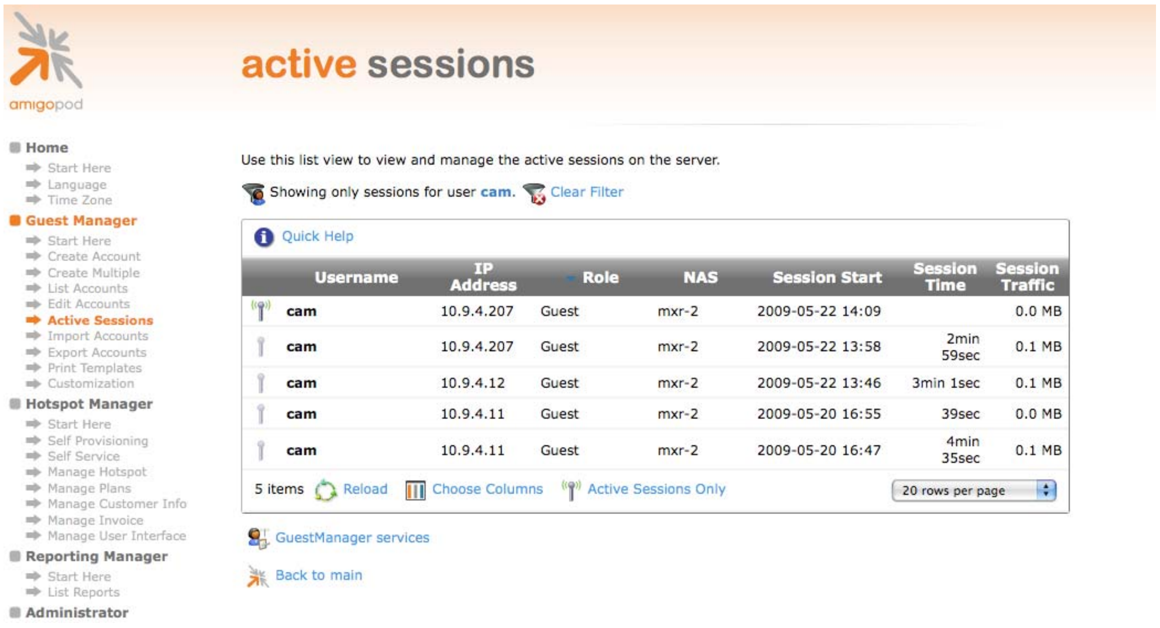
Step 3 – Test Disconnect of authenticated user

Now that all of the required Dynamic Authorisation configuration is complete, we can perform a quick test of the disconnect procedure as specific in RFC 3576.

Assuming we have already got a wireless test user connected to the amigopod SSID and has successfully authenticated via the amigopod hosted Web-Login (as per the previous section), we should see a valid entry in the *show sessions* table from the Trapeze CLI:

```
mxr-2# sh sessions

1 session total

User Name            SessID  Type  Address          VLAN
AP/Radio
-------------------- ------  ----- ---------------- -------------- -------
-
cam                      17* web   10.9.4.207       default        1/2

mxr-2#
```

From the *Guest Manager → Active Sessions* as shown below we can also see the entry for the authenticated wireless user:



Use this list view to view and manage the active sessions on the server.

Showing only sessions for user **cam**. Clear Filter

Username	IP Address	Role	NAS	Session Start	Session Time	Session Traffic
cam	10.9.4.207	Guest	mxr-2	2009-05-22 14:09		0.0 MB
cam	10.9.4.207	Guest	mxr-2	2009-05-22 13:58	2min 59sec	0.1 MB
cam	10.9.4.12	Guest	mxr-2	2009-05-22 13:46	3min 1sec	0.1 MB
cam	10.9.4.11	Guest	mxr-2	2009-05-20 16:55	39sec	0.0 MB
cam	10.9.4.11	Guest	mxr-2	2009-05-20 16:47	4min 35sec	0.1 MB

5 items

To disconnect the wireless user, click on the top Active Session entry for your test user (depicted by the coloured wireless icon in the left hand column) and click the *Disconnect* button below.

Now returning to the Trapeze CLI, if we issue the *show sessions* command again we will see that the user has been successfully disconnected and returned to the unauthenticated state:

```
mxr-2# sh sessions

1 session total

User Name              SessID  Type  Address           VLAN
AP/Radio
--------------------- ------  ----- ----------------- --------------- -------
-
web-portal-amigopod        18  -     10.9.4.207        default         1/2

mxr-2#
```

Appendix B – Testing additional RADIUS attributes

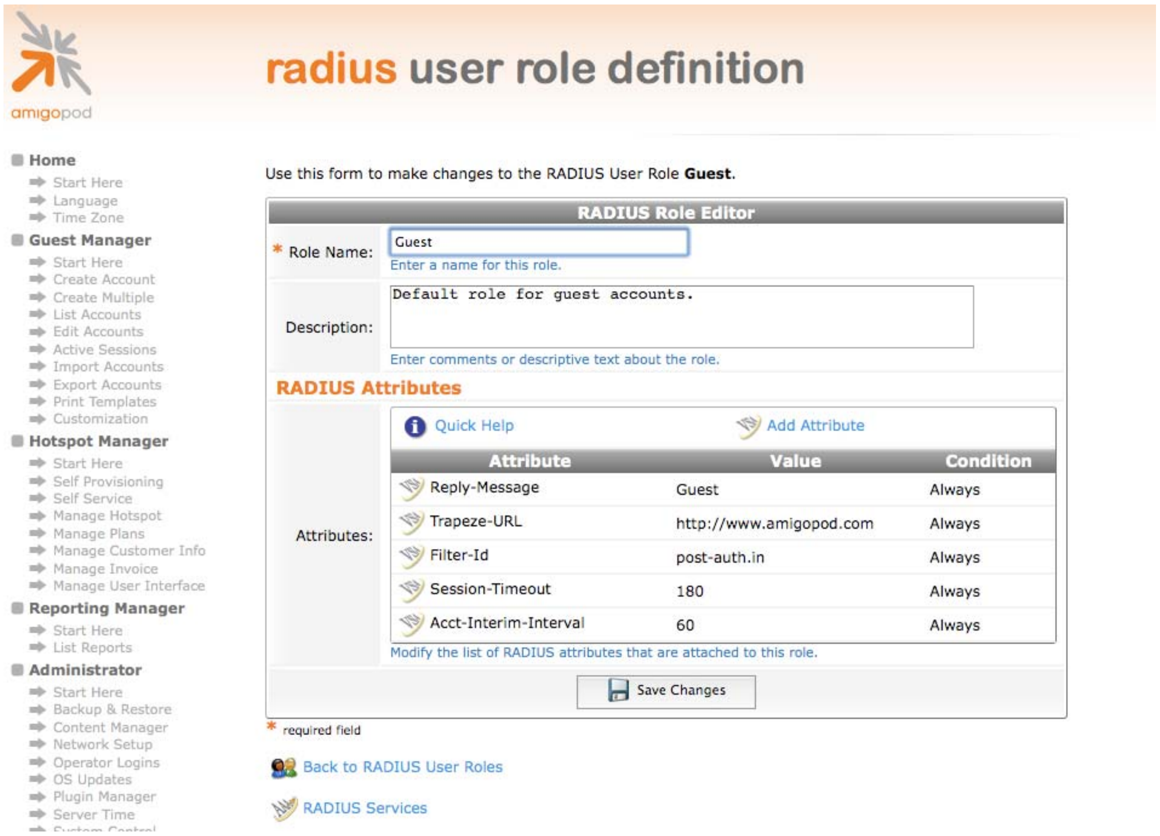
As with all amigopod deployments, *User Roles* can be configured to implement a wireless policy for each user once they have been authenticated. These roles definitions can be made up of both Standard RADIUS attributes as per RFC 2865 and also Vendor Specific Attributes (VSA) that enable vendors such as Trapeze to extend their functionality and apply policies based on their value-add features.

Amigopod has an extensive RADIUS dictionary of vendors and includes the full list of supported VSAs from Trapeze. For more details on the definition and use of the Trapeze VSA attributes please refer to the latest Trapeze MSS documentation.

Some basic testing has been performed to confirm that both Standard Attributes and Trapeze VSA's when configured on the amigopod are honored by the Trapeze MX.

Test Setup

The following screenshot from the amigopod *RADIUS Services → Users Roles* shows how several RADIUS attributes we added to the default Guest Role:



Use this form to make changes to the RADIUS User Role **Guest**.

RADIUS Role Editor

* Role Name: Guest
Enter a name for this role.

Description: Default role for guest accounts.
Enter comments or descriptive text about the role.

RADIUS Attributes

Attributes:

Attribute	Value	Condition
Reply-Message	Guest	Always
Trapeze-URL	http://www.amigopod.com	Always
Filter-Id	post-auth.in	Always
Session-Timeout	180	Always
Acct-Interim-Interval	60	Always

Modify the list of RADIUS attributes that are attached to this role.

* required field

These included the following attributes:

- A hard coded *Session-Timeout* value to ensure that account durations would be honored.
- An *Acct-Interim-Interval* was set to make sure additional accounting information can be drawn from the MX if required for accounting purposes or dynamic billing
- A *Filter-Id* which allows a local defined ACL on the Trapeze MX to be invoked post authentication essentially controlling where and what the wireless user can access once authenticated.
- Finally a Trapeze VSA of *Trapeze-URL* which hard codes the redirect URL overriding the users originally requested URL.

Note: Be aware of the syntax required to make the *Filter-ID* work as expected. The name of the ACL on the Trapeze MX (shown below) needs to be post-pended *with* a *.in* or *.out* notation signifying the direction of the traffic for the ACL to be applied.

```
set security acl name post-auth deny tcp 0.0.0.0 255.255.255.255 10.9.4.10
0.0.0.0 eq 80
set security acl name post-auth deny tcp 0.0.0.0 255.255.255.255 10.9.4.10
0.0.0.0 eq 443
set security acl name post-auth permit 0.0.0.0 255.255.255.255
commit security acl post-auth
```

As you can see from the *post-auth* ACL above, we have explicitly denied HTTP and HTTPS access to the test server (*10.9.4.10*) shown in our original diagram as Server Farm to test the *Filter-ID* functionality. All other traffic is permitted for the test.

Test Result

As expected all RADIUS attributes were sent by the amigopod and received and honored by the Trapeze MX. The RADIUS debug below shows the details of the RADIUS attributes sent as part of the RADIUS Accept message, the Interim accounting details were received each 60 seconds and the session was terminated successfully after 180 seconds.

From the test client perspective, even though they originally requested www.google.com as their initial web page, their session was redirected to www.amigopod.com as per the *Trapeze-URL* VSA.

Based on the application of the *post-auth Filter-ID*, the test user was not able to web browse to the test server on *10.9.4.10* but access to other hosts and the simulated Internet was permitted.

Detailed RADIUS Debug

```
rad_recv: Access-Request packet from host 10.9.4.50:20000, id=14, length=117
User-Name = "cam"
Calling-Station-Id = "00-40-96-A1-F3-99"
Called-Station-Id = "00-0B-0E-90-B8-83:amigopod"
NAS-Port = 13
NAS-Port-Type = Wireless-802.11
NAS-IP-Address = 10.9.4.50
NAS-Identifier = "Trapeze"
User-Password = "wireless"
rlm_sql (sql): Reserving sql socket id: 2
rlm_sql_postgresql: query: SELECT id, UserName, Attribute, Value, Op FROM
radcheck WHERE Username='cam' ORDER BY id
rlm_sql_postgresql: Status: PGRES_TUPLES_OK
rlm_sql_postgresql: affected rows =
rlm_sql_postgresql: query: SELECT radgroupcheck.id, radgroupcheck.GroupName,
radgroupcheck.Attribute, radgroupcheck.Value,radgroupcheck.Op ??FROM
radgroupcheck, usergroup WHERE usergroup.Username = 'cam' AND
usergroup.GroupName = radgroupcheck.GroupName ??ORDER BY radgroupcheck.id
rlm_sql_postgresql: Status: PGRES_TUPLES_OK
rlm_sql_postgresql: affected rows =
rlm_sql_postgresql: query: SELECT id, UserName, Attribute, Value, Op FROM
radreply WHERE Username='cam' ORDER BY id
rlm_sql_postgresql: Status: PGRES_TUPLES_OK
rlm_sql_postgresql: affected rows =
rlm_sql_postgresql: query: SELECT radgroupreply.id, radgroupreply.GroupName,
radgroupreply.Attribute, radgroupreply.Value, radgroupreply.Op ??FROM
radgroupreply,usergroup WHERE usergroup.Username = 'cam' AND
usergroup.GroupName = radgroupreply.GroupName ??ORDER BY radgroupreply.id
rlm_sql_postgresql: Status: PGRES_TUPLES_OK
rlm_sql_postgresql: affected rows =
rlm_sql (sql): Released sql socket id: 2
Exec-Program: /usr/bin/php /opt/amigopod/www/amigopod_request.php 2 280
Exec-Program-Wait: value-pairs: Reply-Message = "Guest", Trapeze-URL =
"http://www.amigopod.com", Filter-Id = "post-auth.in", Acct-Interim-Interval
= 60, Session-Timeout = 180,
Exec-Program: returned: 0
Login OK: [cam] (from client mxr-2 port 13 cli 00-40-96-A1-F3-99)
rlm_sql (sql): Processing sql_postauth
rlm_sql (sql): Reserving sql socket id: 1
rlm_sql_postgresql: query: INSERT INTO radpostauth (username, pass, reply,
authdate) VALUES ('cam', 'wireless', 'Access-Accept', NOW())
rlm_sql_postgresql: Status: PGRES_COMMAND_OK
rlm_sql_postgresql: affected rows = 1
```

```
rlm_sql (sql): Released sql socket id: 1
Sending Access-Accept of id 14 to 10.9.4.50 port 20000
Reply-Message = "Guest"
Trapeze-URL = "http://www.amigopod.com"
Filter-Id = "post-auth.in"
Acct-Interim-Interval = 60
Session-Timeout = 180
rad_recv: Accounting-Request packet from host 10.9.4.50:20000, id=15,
length=211
Acct-Status-Type = Start
Acct-Authentic = RADIUS
Acct-Multi-Session-Id = "SESS-13-6c470a-609225-67c56c"
Acct-Session-Id = "SESS-13-6c470a-609225-67c56c"
User-Name = "cam"
Event-Timestamp = "Dec 31 1999 14:08:48 EST"
Trapeze-VLAN-Name = "default"
Calling-Station-Id = "00-40-96-A1-F3-99"
NAS-Port-Id = "AP1/2"
Called-Station-Id = "00-0B-0E-90-B8-83:amigopod"
NAS-Port = 13
Framed-IP-Address = 10.9.4.207
NAS-Port-Type = Wireless-802.11
NAS-IP-Address = 10.9.4.50
NAS-Identifier = "Trapeze"
Acct-Delay-Time = 0
rlm_sql (sql): Reserving sql socket id: 0
rlm_sql_postgresql: query: INSERT INTO radacct ??(AcctSessionId,
AcctUniqueId, UserName, Realm, NASIPAddress, NASPortId, NASPortType,
AcctStartTime, AcctAuthentic, ??ConnectInfo_start, CalledStationId,
CallingStationId, ServiceType, FramedProtocol, FramedIPAddress,
AcctStartDelay, RoleName) ??VALUES('SESS-13-6c470a-609225-67c56c',
'1116020d0ab83e90', 'cam', '', '10.9.4.50', ??'AP1/2', 'Wireless-802.11',
('2009-05-22 13:58:26'::timestamp - '0'::interval), 'RADIUS', '', ??'00-0B-
0E-90-B8-83:amigopod', '00-40-96-A1-F3-99', '', '', ??'10.9.4.207', '0',
(SELECT roledef.name FROM useraccount LEFT JOIN roledef ON
useraccount.role_id=roledef.id WHERE useraccount.username='cam'))
rlm_sql_postgresql: Status: PGRES_COMMAND_OK
rlm_sql_postgresql: affected rows = 1
rlm_sql (sql): Released sql socket id: 0
Sending Accounting-Response of id 15 to 10.9.4.50 port 20000
rad_recv: Accounting-Request packet from host 10.9.4.50:20000, id=16,
length=241
Acct-Status-Type = Interim-Update
Acct-Authentic = RADIUS
Acct-Multi-Session-Id = "SESS-13-6c470a-609225-67c56c"
Acct-Session-Id = "SESS-13-6c470a-609225-67c56c"
```

```
User-Name = "cam"
Event-Timestamp = "Dec 31 1999 14:09:48 EST"
Trapeze-VLAN-Name = "default"
Calling-Station-Id = "00-40-96-A1-F3-99"
NAS-Port-Id = "AP1/2"
Called-Station-Id = "00-0B-0E-90-B8-83:amigopod"
NAS-Port = 13
Framed-IP-Address = 10.9.4.207
Acct-Session-Time = 60
Acct-Output-Octets = 26247
Acct-Input-Octets = 7760
Acct-Output-Packets = 127
Acct-Input-Packets = 636
NAS-Port-Type = Wireless-802.11
NAS-IP-Address = 10.9.4.50
NAS-Identifier = "Trapeze"
Acct-Delay-Time = 0
rlm_sql (sql): Reserving sql socket id: 4
rlm_sql_postgresql: query: UPDATE radacct SET ??FramedIPAddress='10.9.4.207',
??AcctSessionTime=(EXTRACT(EPOCH FROM('2009-05-22 13:59:26'::timestamp with
time zone - AcctStartTime::timestamp with time zone -
'0'::interval)))::BIGINT, ??AcctInputOctets=(('0'::bigint << 32) +
'7760'::bigint), ??AcctOutputOctets=(('0'::bigint << 32) + '26247'::bigint)
??WHERE AcctSessionId='SESS-13-6c470a-609225-67c56c' AND UserName='cam' ??AND
NASIPAddress='10.9.4.50' AND AcctStopTime IS NULL
rlm_sql_postgresql: Status: PGRES_COMMAND_OK
rlm_sql_postgresql: affected rows = 1
rlm_sql (sql): Released sql socket id: 4
Sending Accounting-Response of id 16 to 10.9.4.50 port 20000
rad_recv: Accounting-Request packet from host 10.9.4.50:20000, id=17,
length=241
Acct-Status-Type = Interim-Update
Acct-Authentic = RADIUS
Acct-Multi-Session-Id = "SESS-13-6c470a-609225-67c56c"
Acct-Session-Id = "SESS-13-6c470a-609225-67c56c"
User-Name = "cam"
Event-Timestamp = "Dec 31 1999 14:10:48 EST"
Trapeze-VLAN-Name = "default"
Calling-Station-Id = "00-40-96-A1-F3-99"
NAS-Port-Id = "AP1/2"
Called-Station-Id = "00-0B-0E-90-B8-83:amigopod"
NAS-Port = 13
Framed-IP-Address = 10.9.4.207
Acct-Session-Time = 120
Acct-Output-Octets = 33397
Acct-Input-Octets = 11840
```

```
Acct-Output-Packets = 196
Acct-Input-Packets = 797
NAS-Port-Type = Wireless-802.11
NAS-IP-Address = 10.9.4.50
NAS-Identifier = "Trapeze"
Acct-Delay-Time = 0
rlm_sql (sql): Reserving sql socket id: 3
rlm_sql_postgresql: query: UPDATE radacct SET ??FramedIPAddress='10.9.4.207',
??AcctSessionTime=(EXTRACT(EPOCH FROM('2009-05-22 14:00:26'::timestamp with
time zone - AcctStartTime::timestamp with time zone -
'0'::interval)))::BIGINT, ??AcctInputOctets=(('0'::bigint << 32) +
'11840'::bigint), ??AcctOutputOctets=(('0'::bigint << 32) + '33397'::bigint)
??WHERE AcctSessionId='SESS-13-6c470a-609225-67c56c' AND UserName='cam' ??AND
NASIPAddress='10.9.4.50' AND AcctStopTime IS NULL
rlm_sql_postgresql: Status: PGRES_COMMAND_OK
rlm_sql_postgresql: affected rows = 1
rlm_sql (sql): Released sql socket id: 3
Sending Accounting-Response of id 17 to 10.9.4.50 port 20000
rad_recv: Accounting-Request packet from host 10.9.4.50:20000, id=18,
length=241
Acct-Status-Type = Stop
Acct-Authentic = RADIUS
Acct-Multi-Session-Id = "SESS-13-6c470a-609225-67c56c"
Acct-Session-Id = "SESS-13-6c470a-609225-67c56c"
User-Name = "cam"
Event-Timestamp = "Dec 31 1999 14:11:47 EST"
Trapeze-VLAN-Name = "default"
Calling-Station-Id = "00-40-96-A1-F3-99"
NAS-Port-Id = "AP1/2"
Called-Station-Id = "00-0B-0E-90-B8-83:amigopod"
NAS-Port = 13
Framed-IP-Address = 10.9.4.207
Acct-Session-Time = 179
Acct-Output-Octets = 40258
Acct-Input-Octets = 16072
Acct-Output-Packets = 263
Acct-Input-Packets = 956
NAS-Port-Type = Wireless-802.11
NAS-IP-Address = 10.9.4.50
NAS-Identifier = "Trapeze"
Acct-Delay-Time = 0
rlm_sql (sql): Reserving sql socket id: 2
rlm_sql_postgresql: query: UPDATE radacct SET ??AcctStopTime=('2009-05-22
14:01:25'::timestamp - '0'::interval), ??AcctSessionTime=NULLIF('179',
'')::bigint, ??AcctInputOctets=(('0'::bigint << 32) + '16072'::bigint),
??AcctOutputOctets=(('0'::bigint << 32) + '40258'::bigint),
??AcctTerminateCause='', ??AcctStopDelay='0', ??FramedIPAddress='10.9.4.207',
??ConnectInfo_stop='' ??WHERE AcctSessionId='SESS-13-6c470a-609225-67c56c'
AND UserName='cam' ??AND NASIPAddress='10.9.4.50' AND AcctStopTime IS NULL
rlm_sql_postgresql: Status: PGRES_COMMAND_OK
rlm_sql_postgresql: affected rows = 1
rlm_sql (sql): Released sql socket id: 2
Sending Accounting-Response of id 18 to 10.9.4.50 port 20000
```