



CDRouter TR-069 User Guide

Version 6.2



QA Cafe © 2010
support@qacafe.com

1.Introduction

CDRouter TR-069 Add-on for CDRouter

CDRouter TR-069 adds TR-069 testing and functionality to our industry leading CPE and router test platform, CDRouter. CDRouter TR-069 has been designed from the ground up to work with implementations based on the Broadband Forum's CPE WAN Management Protocol described by Technical Report 69 (TR-069). CDRouter TR-069 currently supports TR-069 Amendments 1 and 2.

With the CDRouter TR-069 add-on module enabled, CDRouter is enhanced to provide a built-in ACS during all testing sessions. The CDRouter TR-069 add-on module also contains several new test modules focusing on three major areas of TR-069 testing:

1. Fully automated Broadband Forum PD-128 testing
 - *pd128.tcl* test module
2. Real-world testing that addresses functionality not currently covered in PD-128
 - *tr69.tcl* test module
 - *tr111_part1.tcl* test module
 - *tr111_part2.tcl* test module
3. CWMP data model profile verification testing for Broadband Forum TR-098 (through Amendment 2), TR-106 (through Amendment 2), and custom user-defined data models
 - *tr098_profiles.tcl* test module (data mode for IGDs)
 - *tr106_profiles.tcl* test module (data model for generic Devices)
 - *cnmp_profiles.tcl* test module (user defined data models)

By offering automated versions of the PD-128 test cases in addition to the growing battery of real-world and data model profile tests, CDRouter TR-069 can help you test your TR-069 device quickly and thoroughly.

CDRouter TR-069 Extended Data Model Add-on (TR69-EDM)

The Extended Data Model add-on for CDRouter TR-069 (TR69-EDM) provides support for testing supplementary profiles defined additional Broadband Forum data models not included with CDRouter TR-069. The TR69-EDM add-on is only available on systems that already have CDRouter and the CDRouter TR-069 add-on installed.

TR69-EDM provides support for the profiles defined in Broadband Forum TR-104, TR-135, TR-140, TR-143, TR-157, and TR-196. The following six test modules are included with the TR69-EDM add-on:

- *tr104_profiles.tcl* (data model for VoIP devices)
- *tr135_profiles.tcl* (data model for STB devices)
- *tr140_profiles.tcl* (data mode for storage devices)
- *tr143_profiles.tcl* (profiles for network testing and monitoring)
- *tr157_profiles.tcl* (additional profiles)
- *tr196_profiles.tcl* (data model for Femto devices)

NOTE: This document covers the CDRouter TR-069 and TR69-EDM versions bundled with CDRouter 6.2.

2. Test Methodology

Initial Setup

CDRouter TR-069 provides a built-in ACS for CPEs to communicate with. Prior to running any tests however, the CPE must be told how to connect to the ACS. There are no methods for automatic ACS discovery currently supported. Information on how to configure CDRouter TR-069's ACS can be found in Section 4.

Start-Up

During the start-up procedure, CDRouter TR-069 expects to receive an Inform RPC from the device being tested. This Inform should be triggered by rebooting the device, or by configuring the CPE Connection Request URL in the CDRouter configuration file. Once CDRouter has received the initial Inform, the testing procedure will begin.

Running

Once testing has begun, CDRouter TR-069 will use the built-in ACS to communicate with the CPE. No additional user interaction is needed!

Seven modules are included with CDRouter TR-069:

pd128.tcl	Automated version of PD-128 test plan
tr69.tcl	Additional TR-069 tests
tr111_part1.tcl	TR-111 Part 1 test cases
tr111_part2.tcl	TR-111 Part 2 test cases
tr098_profiles.tcl	CWMP profile testing for TR-098 (IGD)
tr106_profiles.tcl	CWMP profile testing for TR-106
cwmp_profiles.tcl	CWMP profile testing for user defined profiles

Six additional modules are available with the TR69-EDM add-on:

tr104_profiles.tcl	CWMP profile testing for TR-104 (VoIP)
tr135_profiles.tcl	CWMP profile testing for TR-135 (STB)
tr140_profiles.tcl	CWMP profile testing for TR-140 (Storage)
tr143_profiles.tcl	CWMP profile testing for TR-143
tr157_profiles.tcl	CWMP profile testing for TR-157
tr196_profiles.tcl	CWMP profile testing for TR-196 (Femto)

To run only the TR-069 related test modules from the buddy command-line:

```
# buddy -module  
pd128.tcl, tr69.tcl, tr111_part1.tcl, tr111_part2.tcl,  
tr098_profiles.tcl, tr106_profiles.tcl, tr104_profiles.tcl, tr135_p  
rofiles.tcl, tr140_profiles.tcl, tr143_profiles.tcl, tr157_profiles  
.tcl, tr196_profiles.tcls, cwmp_profiles.tcl  
-trace -pt
```

For additional execution options, see the CDRouter User Guide or the CDRouter Quick Start Guide.

3. Requirements and License

In order to run the CDRouter TR-069 or TR69-EDM add-ons, your license file must be updated to enable the CDRouter TR-069 functionality. Note that the TR69-EDM add-on requires the CDRouter TR-069. Please follow the instructions from support@qacafe.com on updating your license file to enable CDRouter TR-069 and TR69-EDM (if applicable).

CDRouter will report the status of all available add-ons during the installation process and during startup. To verify that the CDRouter TR-069 or TR69-EDM add-ons are enabled on your system, execute the command `buddy -info` as root and look for the lines “TR69 is enabled” and “TR69-EDM is enabled”. If these lines are present, the CDRouter TR-069 and TR69-EDM add-ons are enabled and ready to use.

```
Copyright (c) 2001-2009 by QA Cafe  
Built on 2009-05-22 on seagull, pktsrc version 5.0 build 194
```

```
Using license installed at: /etc/cdr-mp.lic  
Registered to: qacafe  
Maintenance, Support and Upgrades until: 2010-03-10  
Licensed to run: cdr-mp  
Buddyweb is enabled  
IKE is enabled  
TR69 is enabled  
TR69-EDM is enabled
```

```
INFO(setup): Starting buddy Mon May 25 18:04:49 EDT 2009  
INFO(setup): Loaded OS version Linux-2.6.24-18-generic i686  
INFO(setup): Loaded Tcl version 8.5.0  
INFO(setup): Loaded buddy version 1.45  
INFO(setup): (buildd@terrano) (gcc version 4.2.3 (Ubuntu 4.2.3-2ubuntu7))
```

4. Configuration

ACS Configuration

The ACS has several configuration entries to control its setup.

The following table describes each entry.

acsIp	<p>The testvar acsIp is the IP address where CDRouter will create the ACS server. This IP address must be on the WAN side of the network. CDRouter will only respond to this IP address on the WAN.</p> <p>This IP address must be unique and can not be the same as other IP addresses used for the wanIspIp, remoteHost, etc.</p> <p>When configuring the ACS URL path on the CPE device, the CDRouter ACS does not have any additional path location. The URL should be configured based on the acsIp and acsPort as follows:</p> <pre>acsTransport://acsIp:acsPort</pre> <p>For example</p> <pre>http://6.0.0.1:80 https://6.0.0.1:443</pre>
acsPort	<p>The testvar acsPort is used to configure the port number of the ACS server. The default is port 80.</p>
acsTransport	<p>The testvar acsTransport value is used to configure the desired transport for the ACS server. The transport can be http or https. By default, when https is specified, CDRouter will use the certificate <code>/usr/share/doc/acs.qacafe.com.pem</code> as its server certificate.</p>
acsAuth	<p>The testvar acsAuth value is the type of HTTP authentication that will be run on the ACS server. The authentication type can be either digest, basic, or null.</p>

acsDigestQopAuth	<p>If the <code>qop=auth</code> option should be used when digest authentication is enabled on the ACS server, the testvar acsDigestQopAuth should be configured to yes. To disable the use of the <code>qop=auth</code> option, the testvar acsDigestQopAuth should be set to no. The default value is yes.</p> <p>NOTE: <code>qop=auth-int</code> is not currently supported.</p>
acsDigestMd5Sess	<p>When the ACS is acting as a HTTP client and digest authentication is requested by the CPE server, the ACS will use the algorithm MD5. The <code>MD5-sess</code> algorithm can also be used by configuring the testvar acsDigestMd5Sess to yes. The default value is no.</p> <p>NOTE: Starting in release CDRouter TR-069 3.6, the acsDigestMd5Sess testvar is obsolete. CDRouter will automatically select the correct digest algorithm based on the CPE server response. The default algorithm is MD5. MD5-sess will only be used if the CPE's server returns digest authentication requests with the <code>algorithm=md5-sess</code> option.</p>
acsCertPath	<p>The testvar acsCertPath can be used to configure the ACS with a server certificate. The certificate must be in PEM format containing both the public certificate and private key. Normally, CDRouter's default server certificate should be used which is a Verisign signed certificate. When the default server certificate is used with the ACS, the testvar acsCertPath does not need to be configured.</p>
acsCaCertPath	<p>The testvar acsCaCertPath can be used to configure the CA for the certificate defined by the testvar acsCertPath. This is true if you are using an intermediate CA that is not already installed on the CPE.</p>
acsDownloadCertPath	<p>The testvar acsDownloadCertPath can be used to configure a ACS Download Server with a server certificate. The certificate must be in PEM format containing both the public certificate and private key. Normally, CDRouter's default download server certificate should be used which is a verisign signed certificate. When the default download server certificate is used with the ACS, the testvar acsDownloadCertPath does not need to be configured.</p>

acsDownloadCaCertPath	The testvar acsDownloadCaCertPath can be used to configure the CA for the certificate defined by the testvar acsDownloadCertPath . This is true if you are using an intermediate CA that is not already installed on the CPE.
acsDefaultUser	The username the CPE device should use when connecting to the ACS. Currently, the username should be statically configured on the CPE.
acsDefaultPassword	The password the CPE device should use when connection to the ACS. Currently, the password should be statically configured on the CPE.
acsCpeDefaultUser	The username the ACS should use when connecting to the CPE for new connection requests.
acsCpeDefaultPassword	The password the ACS should use when connection to the CPE for new connection requests.
acsCpeConnReqURL	The location of the CPE Connection Request URL. If this value is not configured, CDRouter will learn this value automatically when it receives the first Inform RPC.
acsWaitForInform	The number of seconds the ACS should wait to receive the initial Inform during the start phase. The default is 120 seconds. If the CPE Connection Request URL is configured using acsCpeConnReqURL , the acsWaitForInform value can be 0. Otherwise, it should be set to an interval in which the initial Inform is expected after a reboot or restart.
acsSkipInitialInform	If the ACS should not wait for an initial Inform during the start phase, the testvar acsSkipInitialInform may be set to yes. This may be desirable for running with an ACS configuration and running other test cases without waiting for an Inform.
acsCpeKeepAlive	By default, when CDRouter initiates a CPE Connection Request to the Connection Request URL, CDRouter will try to keep the HTTP or HTTPS session open provided the server supports Keep-Alive. If CDRouter should always close the HTTP session, this value should be configured to no.
acsDeviceDiscovery	If acsDeviceDiscovery is set to yes, the CDRouter ACS will automatically configure the <code>InternetGatewayDevice.ManagementServer.ConnectionRequestUsername</code> and <code>Password</code> the first time it makes contact with a CPE device. These parameters will be configured using a <code>SetParameterValues</code> RPC.

acsCookieMode	<p>By default, the ACS server will use Set-Cookie2 to set a session cookie on the client. Older Set-Cookie style cookies can also be configured. It is also possible to turn off ACS server cookies completely. When cookies are turned off, any test cases that depend on cookie functionality are automatically skipped.</p> <p>Examples.</p> <pre># RFC 2965 Set-Cookie2 testvar acsCookieMode 2 # Use Set-Cookie testvar acsCookieMode 1 # Do not use any cookies testvar acsCookieMode none</pre>
acsChunkedEncoding	<p>HTTP chunked encoding can be used on the ACS server by configuring the testvar acsChunkedEncoding to yes. The default is no.</p>
acsChunkSize	<p>The chunk size used by the ACS server when chunked encoding is enabled can be configured using the testvar acsChunkSize. The value must be greater than 0.</p>
acsUseFoldedHeader	<p>The ACS can be configured to use HTTP folded headers for digest authentication. When folded HTTP headers are used, each field of the digest authentication header is placed on a separate line using the HTTP continuation character of 'tab' or 'space'. The continuation character can be configured by configuring the testvar acsUseFoldedHeader to tab or space.</p> <p>Examples:</p> <pre># -- default, no folded headers testvar acsUseFoldedHeader none # -- folded headers with 'tab' continuation character testvar acsUseFoldedHeader tab # -- folded headers with 'space' continuation character testvar acsUseFoldedHeader space</pre>
acsNumberCpeAttempts	<p>The ACS can be configured to attempt multiple requests to the CPE Connection Request URL using the testvar acsNumberCpeAttempts. By default, the ACS will make two attempts.</p>

acsDelayCpeAttempts	The number of seconds to delay between each attempt to the CPE Connection Request URL. The default for the CDRouter ACS is 0 seconds. Some CPE devices may rate limit the number of connections at the CPE Connection Request URL and it may be necessary to add a delay between attempts.
acsAuthOncePerSession	When configured to “yes”, the ACS will only request authentication for the first CWMP request in the same TCP session. Other requests do not require authentication. This mode allows the CDRouter ACS to act similar to other ACS vendors which only authenticate the first request in a session. The default value is no.
acsDownloadFileExt	Some CPEs will not attempt a download if the filename extension is not known. You can configure the filename extension that CDRouter will use for negative download tests using the testvar acsDownloadFileExt . This extension will be appended to file names that are created automatically. Examples: <pre>testvar acsDownloadFileExt .bin testvar acsDownloadFileExt .config</pre>
acsTcpWindowSize	The acsTcpWindowSize can be used to change the TCP window size used by the CDRouter ACS. By default, the ACS will use a TCP window size of 2048. QA Cafe has found that some devices may not correctly establish a SSL session when the window size is this small. To work around this, a larger TCP window size may be configured. <pre>testvar acsTcpWindowSize 4048</pre>
cwmpProtocolVersion	The ACS uses CWMP protocol version 1.0 by default. TR-069 Amendment 2 defines CWMP protocol version 1.1. To enable CWMP 1.1 on the ACS, configure the cwmpProtocolVersion testvar. <pre>testvar cwmpProtocolVersion 1.1</pre>

Additional TR-069 Parameters

There are a few additional parameters that control the overall behavior of CDRouter TR-069.

tr69InformTimeout	The default amount of time to wait for an Inform from the CPE device after making a connection request. The default is 120 seconds.
tr69MinPeriodicInform	This value should be set to the minimum supported periodic Inform interval. Some devices have a minimum value that is supported. The default is 60 seconds.
tr69ScheduleInformRPC	If the CPE device supports the ScheduleInform RPC, this testvar should be set to yes. The default is yes.
tr69UploadRPC	If the CPE device supports the Upload RPC, this testvar should be set to yes. The default is yes.
tr69CpeGetRPCs	If the CPE device will send a GetRPCMethods RPC during the establishment of a new TR-69 session, this testvar should be set to yes. The default is no. In order to execute pd128_7.2, the CPE device must have some way of initiating a GetRPCMethods RPC during the test. If the value of tr69CpeGetRPCs is no, CDRouter will automatically skip test pd128_7.2. If the value is yes, it is assumed that the CPE will initiate a GetRPCMethods after the Inform method has been sent.
tr69InitiateConnection	Set to no to disable the automatic connection request for PD-128 tests that normally require the CPE to initiate the connection through some local means (i.e reboot, manual request). When set to yes, CDRouter will attempt to initiate a connection using the CPE Connection Request URL rather than waiting for the CPE device. The default is yes.
tr69ForceBoolean	If set to yes , all boolean types will be sent out as 0 or 1. If set to no , all Boolean types will be sent out as true or false.
tr69XMLCapturePath	CDRouter can capture all XML messages to a specific directory by configuring the tr69XMLCapturePath. XML traces will be placed in this directory.

tr69RebootTimeout	The tr69RebootTimeout is the amount of time in seconds CDRouter will wait for a “1 BOOT” event after rebooting the device or being notified of a reboot after a SetParameterValues RPC. The default value is 150 seconds. Setting this value too low can lead to failure of the RebootRPC tests.
tr69RpcTimeout	The tr69RpcTimeout is a general timeout used for RPC methods that do not have their own specific timeout. The default value is 180 seconds, but this can be increased to support devices that take longer to return a complete RPC response. In some cases when a device is returning a large number of parameters, the tr69RpcTimeout may need to be increased so that CDRouter does not timeout the session before the complete RPC response is received.
tr69DownloadTimeout	The tr69DownloadTimeout is the amount of time in seconds to wait for a successful download and TransferComplete response from the CPE for an image download. The default is 300 seconds.
tr69DownloadImage	<p>The tr69DownloadImage is the full path name to an image file on the CDRouter Linux host that should be used for the firmware download tests. The full path to the file must be used.</p> <p>Example:</p> <pre>testvar tr69DownloadImage /home/a/fm.bin</pre> <p>For large firmware files, the tr69DownloadTimeout value should be increased.</p>
tr69DownloadConfig	<p>The tr69DownloadConfig is the full path name to a file on the CDRouter Linux host that should be used for the vendor config file download tests. The full path to the file must be used.</p> <p>Example:</p> <pre>testvar tr69DownloadConfig /home/a/config</pre> <p>For large config files, the tr69DownloadTimeout value should be increased.</p>

tr69WildcardPortMappingOnly	Set to yes if the device only supports wildcard port mapping entries for the RemoteHost Parameter of a PortMapping. If both wildcard and specific IP address mappings are supported, the testvar should be set to no . The default value is no .
tr69PortMapDeletionDelay	The tr69PortMapDeletionDelay is the amount of time to wait after deleting a port mapping and checking that it is actually deleted. By default, CDRouter will wait 15 seconds after the DeleteObject before sending traffic to verify that the port map is deleted. Example: testvar tr69PortMapDeletionDelay 120
tr69MaxUserNameLen tr69MaxPasswordLen	TR-069 defines the maximum management server user name and password to be a string of 256 characters. You can configure a lower limit and CDRouter will not create user names or passwords above this limit.
tr69MaxCpeUserNameLen tr69MaxCpePasswordLen	TR-069 defines the maximum CPE connection request user name and password to be a string of 256 characters. You can configure a lower limit and CDRouter will not create user names or passwords above this limit.
supportsTr111Part1	If the IGD device supports TR-111 Part 1, this testvar should be set to 'yes'. Otherwise, this value should be set to 'no'. This testvar must be set to yes in order to run the tr111_part1.tcl module which contains specific tests for the Part 1 functionality on an IGD device.
supportsTr111Part2	If the TR-069 device supports TR-111 Part 2, this testvar should be set to 'yes'. Otherwise, this value should be set to 'no'. When TR-111 Part 2 support is enabled, CDRouter checks for the NATDetected and UDPConnectionRequestAddress parameters when a TR-069 device is first discovered. The CDRouter ACS will also set up Active Notification on these parameters and switch to UDP style connection requests when NAT is detected.

tr69DeviceType

CDRouter TR-069 supports devices that implement the TR-104 and TR-135 Parameter models. The TR-104 and TR-135 Parameter models may either be embedded within an InternetGatewayDevice (IGD + VoIP) or standalone VoIP ATA or STB.

Currently, only the pd128.tcl and tr104_profiles.tcl or tr135_profiles.tcl test modules support TR-104 and TR-135 parameters, respectively.

The testvar **tr69DeviceType** tells CDRouter what general parameter models to expect from a device. This testvar accepts a comma separated list of IGD, VoIP, or STB device.

The first device specified will be used for the default when running PD-128 tests.

To enable a standalone TR-104 device:

```
testvar tr69DeviceType VoIP
```

To enable a standalone TR-135 device:

```
testvar tr69DeviceType STB
```

To enable TR-104 support embedded within an IGD device but defaulting to TR-104 parameters when running PD-128:

```
testvar tr69DeviceType "VoIP,IGD"
```

To enable TR-104 support embedded within an IGD device but defaulting to TR-098/IGD parameters when running PD-128:

```
testvar tr69DeviceType "IGD,VoIP"
```

tr69FakeDeviceSummary

Used to configure a fake DeviceSummary string for the device. The value must be either the keyword "all" to select all available profiles or a valid DeviceSummary string.

cwmpSkipParameters

Define a list of CWMP parameters to ignore when profile testing. See the "CWMP Profile Testing" section for an example.

cwmpModifyParameters	Define a list of CWMP parameters to modify. See the “CWMP Profile Testing” section for an example.
cwmp_profile1 cwmpProfileName cwmpProfileObject cwmpProfilePath	User defined CWMP profiles. See the “CWMP Profile Testing” section for an example.
STUNAuth	By default, the CDRouter STUN server will not require authentication for the STUN client. Authentication can be enabled by configuring the testvar STUNAuth to yes. CDRouter automatically configures a STUN username and STUN password when TR-111 Part 2 is enabled.
STUNMaximumKeepAlivePeriod	The STUNMaximumKeepAlivePeriod maps directly to the TR-111 Part 2 parameter STUNMaximumKeepAlivePeriod. CDRouter will configure this value on the TR-069 client. The default is 60 seconds.
STUNMinimumKeepAlivePeriod	The STUNMinimumKeepAlivePeriod maps directly to the TR-111 Part 2 parameter STUNMinimumKeepAlivePeriod. CDRouter will configure this value on the TR-069 client. The default is 60 seconds which matches the value of STUNMaximumKeepAlivePeriod.

Example 1. CDRouter TR-069 Basic Configuration

In the example below, CDRouter is configured with the CPE Connection Request URL so that it can initiate new TR-069 sessions from the start. The CPE Connection Request username and password are statically configured on the CPE and the CPE is also configured with the username and password of the ACS.

```
testvar acsIp 6.0.0.1
testvar acsPort 80
testvar acsTransport http
testvar acsAuth digest
testvar acsDigestQopAuth yes
testvar acsCpeConnReqURL http://192.168.200.2:80/XML/000FCC-23570008.xml
testvar acsDeviceDiscovery no
testvar acsDefaultUser 000FCC-23570008
testvar acsDefaultPassword acs123
testvar acsCpeDefaultUser cpeurl
testvar acsCpeDefaultPassword cpe123
testvar tr69InitiateConnection yes
testvar tr69DownloadImage /home/user/fw_image_ver2.bin
```

Example 2. CDRouter TR-069 Configuration with Device Discovery

In the example below, CDRouter is configured to use device discovery by configuring testvar **acsDeviceDiscovery** to yes. When the first Inform is received from the device, CDRouter will attempt to configure the CPE Connection Request URL username and password. The CPE is also configured with the username and password of the ACS.

```
testvar acsIp                6.0.0.1
testvar acsPort              80
testvar acsTransport        http
testvar acsAuth              digest
testvar acsDigestQopAuth    yes
testvar acsWaitForInform    120
testvar acsDeviceDiscovery  yes
testvar acsDefaultUser      000FCC-23570008
testvar acsDefaultPassword  acs123
testvar acsCpeDefaultUser   cpeurl
testvar acsCpeDefaultPassword cpe123
testvar tr69InitiateConnection yes
testvar tr69DownloadImage   /home/user/fw_image_ver2.bin
```

Firmware Upgrade

CDRouter-TR-069 contains support for downloading new images to the CPE as part of the PD-128 firmware download tests. To enable firmware download tests, the path to the desired firmware image must be configured with the testvar **tr69DownloadImage**. The full path to the file should be used. By default, CDRouter TR-069 will wait up to 5 minutes for a download to complete. This is generally sufficient for a 1 – 3 megabyte image file. For larger image files, the download timeout can be adjusted by configuring the testvar **tr69DownloadTimeout**.

Log Messages

CDRouter TR-069 protocol messages can be filtered out using the `-show` and `-hide` options from the command-line or the BuddyWeb Advanced Tab. The following protocols are specific to CDRouter TR-069:

ACS All ACS protocol messages

Examples:

To show only ACS log messages
% buddy -show ACS

To exclude all ACS related messages
% buddy -hide ACS

5. TR-111 Parts 1 and 2

CDRouter TR-069 supports TR-111 Parts 1 and 2. By default, support for TR-111 Part 1 and 2 is disabled.

NOTE: TR-111 Parts 1 and 2 have been incorporated into the TR-069 specification as Annexes F and G, respectively, starting with TR-069 Amendment 1. This functionality (as defined in Annexes F and G of TR-069 Amendments 1 and 2) is referenced as TR-111 Part 1 and TR-111 Part 2 within CDRouter TR-069.

TR-111 Part 1

Support for TR-111 Part 1 in an IGD device under test can be enabled by configuring the testvar **supportsTr111Part1** to yes. This testvar defaults to no. When TR-111 Part 1 is enabled, CDRouter can run the TR-111 Part 1 tests from the pd128.tcl module. There is also an additional tr111_part1.tcl module with additional coverage of TR-111 Part 1. During any TR-111 Part 1 testing, CDRouter will create LAN side devices to associate with the IGD device.

NOTE: TR-111 Part 1 testing requires at least 1 Ethernet LAN interface.

TR-111 Part 2

Support for TR-111 Part 2 in an IGD device or LAN side TR-069 device under test can be enabled by configuring the testvar **supportsTr111Part2** to yes. This testvar defaults to no. When TR-111 Part 2 is enabled, CDRouter will attempt to read the NATDetected and UDPConnectionRequestAddress parameters when it first learns about a TR-069 device. CDRouter will also enable STUN on the device and setup active notification of these parameters. If NAT is detected and the UDPConnectionRequestAddress is provided, CDRouter will switch to UDP style connection request attempts instead of TCP based requests.

By default, the ACS STUN server does not require authentication on STUN Binding Requests. When the ACS configures the STUN settings on the CPE, it automatically generates a STUN username and STUN password for the CPE. The ACS server will only require authenticated STUN Binding Requests when the testvar **STUNAuth** is test to yes. The ACS will configure the maximum and minimum STUN keep alive intervals to 60 seconds by default. These can be changed using the testvars **STUNMaximumKeepAlivePeriod** and **STUNMinimumKeepAlivePeriod**.

6.LAN Side TR-069 Devices

Starting with CDRouter TR-069 release 3.8, LAN side TR-069 aware devices can be tested using the pd128.tcl test module. When testing a LAN side TR-069 device, the device must be connected to an IGD device that provides a WAN connection and NAT. Since the LAN side device is now behind NAT, the device must either support TR-111 Part 2 or a port mapping must be created on the IGD. When working with port mappings on the IGD, CDRouter must either be configured with a URL that works with the port mapping on the IGD, or CDRouter can create a port mapping dynamically.

CDRouter TR-069 currently supports testing TR-104 VoIP and TR-135 STB devices using the pd128.tcl module.

LAN side TR-069 devices can be configured using testvar groups. Each LAN side TR-069 device must be named **tr069_device2** through **tr069_device8**. The testvar group contains configuration items unique to each TR-069 LAN side device. Only a small number of testvars are required for each LAN device.

When LAN side TR-069 devices are used, the Serial Number and OUI for all TR-069 devices must be configured. To tell CDRouter to run all test cases against a LAN side TR-069 device instead of the normal IGD device, the global testvar **acsTargetDevice** must be set to the testvar group name of the desired LAN side device. Otherwise, CDRouter will default to running all TR-069 tests against the IGD defined in the main CDRouter configuration.

The following table describes each entry available to LAN side TR-069 devices:

acsDefaultUser	The username the CPE device should use when connecting to the ACS. Currently, the username should be statically configured on the CPE. Each TR-069 device can have a unique username.
acsDefaultPassword	The password the CPE device should use when connection to the ACS. Currently, the password should be statically configured on the CPE. Each TR-069 device can have a unique password.

acsCpeConnReqURLMapped	<p>The location of the CPE Connection RequestURL reachable using a port mapping on the router. When the testvar acsCpeConnReqURLMapped is used, the configuration of the port mapping is beyond the scope of CDRouter. If you wish for CDRouter to create the port mapping automatically, please see the testvar acsCreatePortMapOnIGD below.</p> <p>CDRouter can also learn the CPE's Connection Request URL automatically when it receives it first Inform RPC. In this case, the device <i>must</i> support TR-111 Part 2. If the device supports TR-111 Part 2, this entry should not be defined.</p>
acsCreatePortMapOnIGD	<p>When set to yes, CDRouter will attempt to create a new port mapping on the IGD for the CPE Connection Request URL. You must also configure the specific port number to use using the testvar acsPortMapPublicPort.</p>
acsPortMapPublicPort	<p>This testvar is used to configure the public port of a port mapping for a LAN device's CPE Connection Request URL. The user must designate a specific port to use for this port mapping.</p>
tr69DeviceOui	<p>The expected OUI for the device. This must match the OUI field from the device's DeviceId in the device Inform.</p>
tr69DeviceSn	<p>The expected serial number for the device. This must match the SerialNumber field from the device's DeviceId in the device Inform.</p>
tr69DownloadImage	<p>The tr69DownloadImage is the full path name to a file on the CDRouter Linux host that should be used for the download image for this specific LAN device. The full path to the file must be used.</p> <p>Example:</p> <pre>testvar tr69DownloadImage /home/a/ata.bin</pre> <p>For large firmware files, the tr69DownloadTimeout value should be increased. Note that the testvar tr69DownloadTimeout is global and should not be configured as part of the testvar_group for a tr069_device.</p>

tr69DeviceType

The testvar **tr69DeviceType** tells CDRouter what general parameter models to expect from a device. This testvar accepts a comma separated list of IGD, VoIP, or STB device.

The first device specified will be used for the default when running PD-128 tests.

To enable a standalone TR-104 device:

```
testvar tr69DeviceType VoIP
```

To enable a standalone STB TR-135 device:

```
testvar tr69DeviceType STB
```

Example 1. LAN side TR-104 device with TR-111 Part 2

In the example below, CDRouter is assumed to already have a working configuration for an IGD device with TR-069 enabled. A new device with TR-104 and TR-111 support has been added to the LAN side of the IGD. The following configuration will allow CDRouter to run all TR-069 related tests against the LAN side TR-104 device.

```
testvar acsTargetDevice          tr069_device2
testvar supportsTr111Part2      yes

testvar_group tr069_device2 {
  testvar acsDefaultUser        myata
  testvar acsDefaultPassword    qacafe123
  testvar tr69DeviceOui         001122
  testvar tr69DeviceSn          123
  testvar tr69DownloadImage     /home/files/myata.bin
  testvar tr69DeviceType        VoIP
}
```

Example 2. LAN side TR-135 device with static Port Mapping on IGD

In the example below, CDRouter is assumed to already have a working configuration for an IGD device with TR-069 enabled. A new device with TR-135 support has been added to the LAN side of the IGD. The device does not support TR-111 and a port mapping has been created on the IGD device with a public IP address of 192.168.200.2. The LAN side device must define a URL that can be used through the IGD for the normal TCP based connection request mechanism. The following configuration will allow CDRouter to run all TR-069 related tests against the LAN side TR-135 device.

```
testvar acsTargetDevice          tr069_device3
testvar supportsTr111Part2      no
```

```

testvar_group tr069_device3 {
    testvar acsDefaultUser          mySTB
    testvar acsDefaultPassword      qacafe123
    testvar acsCpeConnReqURLMapped  http://192.168.200.2:1234/mySTB_request.htm
    testvar tr69DeviceOui           001144
    testvar tr69DeviceSn            124
    testvar tr69DownloadImage       /home/files/mySTB.bin
    testvar tr69DeviceType          STB
}

```

Example 3. LAN side TR-135 device with dynamic Port Mapping on IGD

In the example below, CDRouter is assumed to already have a working configuration for an IGD device with TR-069 enabled. A new device with TR-135 support has been added to the LAN side of the IGD. The device does not support TR-111. CDRouter will automatically configure a port mapping on the IGD device with a public IP address of 192.168.200.2. The following configuration will allow CDRouter to run all TR-069 related tests against the LAN side TR-135 device.

```

testvar acsTargetDevice            tr069_device4
testvar supportsTr111Part2        no

testvar_group tr069_device4 {
    testvar acsDefaultUser          mySTB
    testvar acsDefaultPassword      qacafe123
    testvar acsCreatePortMapOnIGD   yes
    testvar acsPortMapPublicPort    5000
    testvar tr69DeviceOui           001144
    testvar tr69DeviceSn            124
    testvar tr69DownloadImage       /home/files/mySTB.bin
    testvar tr69DeviceType          STB
}

```

7.CWMP Profile Testing

TR-106 defines the structure and use of CWMP profiles to allow the ACS to quickly detect the data model parameters supported by a device. The “DeviceSummary” parameter is used to provide a list of all CWMP profiles supported by a device. If a device claims support for a profile, it MUST support all the required parameters of that profile. This allows the ACS to have some commonality among devices.

CDRouter TR-069 contains profile definitions for all CWMP profiles defined in TR-098 (including Amendments 1 and 2) and TR-106 (including Amendments 1 and 2). The TR69-EDM add-on adds support supplemental CWMP profiles defined in TR-104, TR-135, TR-140, TR-143, TR-157, and TR-196. It is also possible for users to define their own CWMP profiles. For each profile, CDRouter defines several generic tests that verify the consistency of the profile:

1. Verify all required parameter names are returned using GetParameterNames
2. Verify parameter name instances can be walked using GetParameterNames
3. Verify all parameter names and values are returned using GetParameterValues
4. Verify the read and write status of each parameter
5. Verify all writable parameters can be set using SetParameterValues
6. Verify the creation and deletion of all creatable objects using AddObject and DeleteObject

New Test Modules

There are new test modules specifically for CWMP profile testing. All of the modules can be selected together or individual modules and test cases can be selected. The following new test modules are included with the CDRouter TR-069 add-on:

```
tr098_profiles.tcl
tr106_profiles.tcl
cwmp_profiles.tcl (user defined profiles)
```

The following additional test modules are available with the TR69-EDM add-on:

```
tr104_profiles.tcl
tr135_profiles.tcl
tr140_profiles.tcl
tr143_profiles.tcl
tr157_profiles.tcl
tr196_profiles.tcl
```

Getting Started with CWMP Profile Testing

If the device under test supports the “DeviceSummary” parameter, CDRouter will automatically select the list of CWMP profiles to test. When CDRouter receives the first TR-069 Inform from the device, the CDRouter ACS will parse the DeviceSummary string and automatically deselect any CWMP profiles that are not supported. If the DeviceSummary parameter is not supported, CDRouter will not automatically exclude any test cases.

It is possible to configure a “fake” DeviceSummary string for devices that do not support the DeviceSummary parameter or to override the real DeviceSummary from the device. Two new testvars have been introduced to control this behavior. A fake DeviceSummary string can be configured using the `tr69FakeDeviceSummary` testvar. The contents of this testvar should be included within “{“ and “}”. The `tr69FakeDeviceSummary` testvar also accepts the special keyword “all” to designate all available CWMP profiles. This can be used to select all available CWMP profiles regardless of the DeviceSummary reported by the device under test.

Examples:

```
testvar tr69FakeDeviceSummary {InternetGatewayDevice:1.1[] (Baseline:1) }
testvar tr69FakeDeviceSummary {InternetGatewayDevice:1.1[] (Baseline:1,ADSLWAN:1) }
testvar tr69FakeDeviceSummary all
```

By default, CDRouter uses the real DeviceSummary from a device and will only use the fake DeviceSummary if the `tr69FakeDeviceSummary` testvar is configured.

Profile Version Numbers

Each CWMP profile reported in the DeviceSummary string contains a version number. When testing a CWMP profile, CDRouter will try to match the profile version number against the CDRouter library of CWMP profile definitions. By default, CDRouter uses “strict” matching and will skip a profile unless an exact match is made. For example, if the device is reporting Baseline:3 and CDRouter only knows about Baseline:2, CDRouter will not attempt to test the Baseline profile. CDRouter can also be configured for “best” case matching which allows CDRouter to use the closest version available. With “best” matching in the scenario above, CDRouter would proceed to verify the Baseline profile using Baseline:2. NOTE: While most CWMP profiles are backwards compatible, there are some unfortunate exceptions. Some care must be exercised when using the “best” option.

Examples:

Use strict CWMP profile matching (the default behavior)

```
testvar tr69ProfileMatch strict
```

Use best case CWMP profile matching

```
testvar tr69ProfileMatch best
```

Excluding CWMP Parameters

In some cases, you may wish to exclude a parameter from the CWMP profile check because it is not supported by the device under test. CDRouter allows you do to this by configuring a list of parameters to skip in the `cwmpSkipParameter` testvar. A list of parameters can be specified by enclosing the list members between “{“ and “}”. The list of parameters can be either full instances or generic instances that use “{i}” for each instance index. Note that each individual parameter within the list must be specified using a full instance or partial path, or a fully generic instance using “{i}” for each index. Parameters cannot be specified using a mix of generic and full instances. Here is an example:

```
testvar cwmpSkipParameters {
  InternetGatewayDevice.DeviceInfo.ModelName
  InternetGatewayDevice.LANDevice.{i}.USBLANConfigurationNumberOfEntries
}
```

If you are skipping an entire object path and all parameters contained below it, you only need to specify the top object path. All top object paths (partial paths) must end with a ‘.’ character. For example, if the device under test does not support the `LANConfigSecurity` object from the TR-098 Baseline profile, you could skip the object by configuring the following `cwmpSkipParameters` testvar:

```
testvar cwmpSkipParameters {
  InternetGatewayDevice.LANConfigSecurity.
}
```

Modifying CWMP Parameters

If you wish to change the requirement (read or write status) of a parameter, you may configure a list of parameters to modify. CDRouter normally uses all parameters with a requirement field of ‘W’ for the `SetParameterValues` test. The defined requirement from the original CWMP profile can be changed. It may be helpful to find CDRouter’s original definition for a CWMP parameter in the `/usr/share/doc/cdrouter/cwmp-profiles` directory. Here is an example:

```
testvar cwmpModifyParameters {
  string R InternetGatewayDevice.LANDevice.{i}.LANEthernetInterfaceConfig.{i}.MaxBitRate
}
```

When modifying parameters that are part of a service object such as `VoiceService` or `STBService`, the root object should be specified as ‘root’, since it can be either `InternetGatewayDevice` or `Device`. Here is an example:

```
testvar cwmpModifyParameters {
```

```

string R root.Services.VoiceService.{i}.VoiceProfile.{i}.SignalingProtocol
unsignedInt R root.Services.VoiceService.{i}.VoiceProfile.{i}.MaxSessions
}

```

Creating Your Own Custom CWMP Profiles

New CWMP profiles may be created by creating a file that defines all of the included parameters. Each line of the file should contain one parameter.

The new file containing the CWMP profile can be included in the CDRouter setup by defining a testvar_group using the cwmp_profile naming convention. You can create up to 128 unique profiles using testvar_group names cwmp_profile1 through cwmp_profile128. For example:

```

testvar_group cwmp_profile1 {
    testvar cwmpProfileName      myVendorParameters
    testvar cwmpProfileObject    InternetGatewayDevice
    testvar cwmpProfilePath      /home/dev/myVendorParameters.cwmp
}

```

The file /home/dev/myVendorParameters.cwmp should contain a CWMP parameter on each line. The format for each parameter is <type> <requirement> <path>. For example:

```

boolean W InternetGatewayDevice.X_Vendor.featureA.Enabled
string  W InternetGatewayDevice.X_Vendor.featureA.foo

```

The type field comes directly from the CWMP parameter definition. The following requirement options are available:

- P** The parameter is a required object
- PC** The parameter is a required object and creatable
- PO** The parameter is an optional object
- POC** The parameter is an optional object and creatable
- R** The parameter is read-only
- W** The parameter is writable and readable
- WO** The parameter is writable but can not be read (password, keys, etc)

Using the Pktsrc/CWMP API

It is also possible to create new profiles using the PktSrc/CWMP API. Please see the API reference for an overview of the Tcl procedures available.

Profile Testing Issues

When starting CWMP profile testing, it is common to find unsupported or misspelled parameter names. The SetParameterValues testing can also detect issues with the data model that may need to be resolved. The following issues may occur:

The initial value of the parameter is not a valid value. CDRouter will attempt to read the initial value and write back this same value. If the initial value is not valid, the SetParameterValues test will fail.

The initial value of a parameter may be empty until some additional operation occurs. In some cases, it may be more useful to run the CWMP profile tests after running other TR-069 test cases. For example, some devices may not populate the IPPing parameters with valid values until the IPPing diagnostics have been run at least once.

Any parameters causing the testing to fail can either be skipped or modified.

8. Testing Exercises

Besides running each test case in CDRouter TR-069, other testing scenarios can be created. The following test scenarios are recommended.

Vary the ACS Configuration

The ACS has several different configuration modes that impact its behavior. It is worthwhile to test the ACS using several different configurations. Besides the options for http or https and the different authentication schemes (basic, digest), chunked transfer mode can be enabled or disabled. See the listing of ACS testvars in Chapter 4.

Long Duration Testing

Besides running through the CDRouter TR-069 test cases one time, it is recommended to set up long duration runs of the pd128.tcl and tr69.tcl modules using the `-repeat` option or a BuddyWeb package with the repeat option enabled. Frequent CWMP sessions will help verify that the TR-069 client will continue to operate over time.

XML Validation with Help from CDRouter

CDRouter TR-069 can help validate your device's XML against the TR-069 CWMP Schema. This schema is provided by CDRouter in `/usr/share/doc/cdrouter/cwmp.xsd`. While CDRouter TR-069 is running, it can be configured to save all XML input and output for later validation.

The testvar `tr69XMLCapturePath` should point to an empty directory that will hold this XML. If this directory doesn't exist, CDRouter will attempt to create it for you. While the test is running, every XML payload will be written to this directory in individual files prefixed by either 'inbound' or 'outbound'. Inbound files will have been sent by the DUT where Outbound will be payloads sent by CDRouter TR-069.

Once the test run is complete, you can use a third-party tool to validate your XML against the provided schema.

QA Cafe recommends using `xmllint` for schema validation. `xmllint` is part of `libxml2` and is installed by default on most Linux distributions. To run `xmllint` against the saved XML and compare it to the CWMP schema, use the following syntax:

```
# xmllint --noout /path/to/xml/*.xml --schema
/usr/share/doc/cdrouter/cwmp.xsd
```

For more information on using `xmllint`, please visit <http://xmlsoft.org/xmllint.html>

Check Folded HTTP Header Behavior

HTTP 1.1 allows long headers to be folded onto multiple lines using a continuation character. Most HTTP clients and servers do not use folded headers by default. However, QA Cafe has seen cases of clients using folded headers and it is possible that special applications could use them in CWMP headers.

The CDRouter ACS can be configured to use folded HTTP headers for digest authentication by configuring the testvar `acsUseFoldedHeader`. You can quickly check that folded HTTP headers are supported for the CPE's client and server side by enabling folded headers and then running through a set of TR-069 tests.

Please see section 4 on Configuration for a description of how to set the testvar `acsUseFoldedHeader` to enable folded HTTP headers. Folded HTTP headers are not used by default. However, they will be accepted if the CPE sends them.

9. Possible Problems

No Initial Inform Received from the CPE

If you configure the testvar **acsWaitForInform**, CDRouter will wait up to this interval in order to receive the first Inform from the CPE. If the Inform is not received, CDRouter will fail the startup phase and end the test session. There are a couple of things to check if the Inform is not being received during this interval.

NTP – Some devices will hold off sending the initial Inform while they try to synchronize the system clock using NTP. You can enable the built-in CDRouter NTP server to speed up this process.

Short wait interval – If you are using a small value for **acsWaitForInform**, you may need to increase this value.

CPE configuration – Verify that the CPE is configured with the correct IP or domain name of the ACS server. By default, CDRouter will populate the name **acs.qacafe.com** in all of the CDRouter DNS servers. This domain name maps to the testvar **acsIp**.

Periodic Inform

When starting a testing session with CDRouter TR-069, the Periodic Inform should be disabled. CDRouter TR-069 will specifically turn on the Periodic Inform during specific test cases.

Invalid Certificate Using HTTPS

When using HTTPS, the ACS configuration on the CPE may need to use the domain name of the ACS rather than the IP address in order to validate the ACS server certificate. The default server certificate for the ACS included with CDRouter is registered to the domain name **acs.qacafe.com**. CDRouter will automatically configure this DNS name in all DNS servers used in CDRouter.

The ACS URL configured on the CPE should use the domain name.

For example:

```
https://acs.qacafe.com
```

The CDRouter configuration file would be as follows.

```
testvar acsIp          6.0.0.1
testvar acsPort        443
testvar acsTransport  https
```

CDRouter will automatically map the testvar **acsIp** (6.0.0.1 in the example above) to the domain name `acs.qacafe.com`.

ACS is Unable to Request New CWMP Sessions

When the CDRouter ACS needs to communicate with the TR-069 CPE, it will attempt to initiate a new CWMP session using the CPE Connection Request URL. CDRouter will use the username and password from the **acsDefaultCpeUser** and **acsDefaultCpePassword** when trying to access this URL.

```
testvar acsCpeDefaultUser      cpeurl
testvar acsCpeDefaultPassword  cpe123
```

If the testvar **acsDeviceDiscovery** is set to yes, CDRouter can automatically set the CPE Connection Request URL username and password during the first CWMP session. This will insure that the username and password match.

Tests Fail when `tr69InitiateConnection` is Set to No

When setting up CDRouter TR-069 to run automatically, the testvar `tr69InitiateConnection` should be set to yes. When `tr69InitiateConnection` is set to yes, CDRouter will always initiate a connection using the CPE Connection Request URL. However, some PD-128 test cases specifically require the CPE to initiate a new connection by some local means. This may be a reboot or local action. By setting the testvar `tr69InitiateConnection` to no, you can execute these tests using the exact PD-128 test procedure. This may require some level of manual intervention to make sure the local action is executed when expected.

We recommend that you start out running with `tr69InitiateConnection` set to yes.

Working Around a Failing CWMP Client

In some cases, you may need to have the CDRouter ACS enabled and configured, but not wait for the initial Inform during the start phase. If the CWMP client does not respond consistently to the CPE Connection Request URL or experiences other failures, the start phase can be configured to skip the initial Inform by configuring the testvar **acsSkipInitialInform** to yes.

10. PD-128 Test Case Notes

The test cases in the PD-128 test module (pd128.tcl) are based on the test case procedures outlined in PD-128 Interoperability Test Plan for TR-069 Plugfests (Version 9). In some cases, the test cases have been modified slightly to allow for a fully automated test run. In other cases, the parameter names have been changed to parameter names that should always be supported. For example, PD-128 does reference specific parameter names for wireless interfaces which may not always be available.

The following notes have been included for each implemented PD-128 test case.

Test	Supported	Notes
1.1-1.3	yes	In PD-128, these tests all require the CPE to initiate the TR-069 session. By default, CDRouter will attempt to request a new session from the CPE using the CPE Connection Request URL. If you want to force CDRouter to wait for the new session, the testvar tr69InitiateConnection should be set to no. In this case, the TR-069 will need to be manually initiated outside of CDRouter.
1.4	yes	
2.x	yes	When using https, the ACS URL on the CPE should use the domain name acs.qacafe.com. NOTE: These tests are only run when the testvar acsTransport is set to https.
3	yes	TR-111 Part 1 is supported.
4	yes	TR-111 Part 2 is supported.
5	yes	These tests are somewhat redundant since CDRouter TR-069 will also attempt a CWMP session for Tests 1.x
6.1	yes	
6.2	yes	
7.1	yes	
7.2	yes	By default, this test case is disabled unless the testvar tr69CpeGetRPCs is set to yes.
8.x	yes	The location of the firmware image to download should be configured using the testvar tr69DownloadImage . Likewise, the location of the vendor config file to download should be configured using the testvar tr69DownloadConfig .
9.x	yes	
10.x	yes	For test case 10.4, the SSID parameter is not used in order to support devices that do not support wireless. Instead the LANEthernetInterfaceNumberOfEntries parameter is used.

11.x	yes	
12 & 13	yes	These two PD-128 test cases have been combined into a single CDRouter test case.
14	yes	
15	yes	For test case 15.2 and 15.4, no wireless parameters are used in order to support devices that do not support wireless. Instead the LANEthernetInterfaceNumberOfEntries parameter is used.
16.1	no	CDRouter currently does not support Active Notification tests. In order for an Active Notification to be sent on the parameter PortMappingNumberOfEntries, the port mapping must be created through some other means besides the CDRouter ACS.
16.2-16.5	yes	Tests 16.2 through 16.5 use the Parameter InternetGatewayDevice.DeviceInfo.UpTime instead of the Parameter PortMappingNumberOfEntries.
17	yes	This test uses a wildcard value for the RemoteHost. CPEs are required to support wildcard values for the RemoteHost.
18	yes	This test assumes the device supports a wireless LAN interface.
19 & 20	yes	PD128 test cases 19 & 20 have been combined into a single CDRouter test case. Currently, this test should only be run on a DSL device since the WAN interface setup is DSL specific.
21	yes	
22	yes	
23	yes	
24	yes	This test case actually calls test Tests 5, 9, and 10.
25	yes	This test is supported using Basic HTTP authentication ONLY. Section 3.4.5 of WT-121 draft 8 recommends using a different nonce for each TCP session. Once the specific nonce behavior of HTTP closed connections is clarified, this test will be supported using digest authentication.
26.x	yes	Only tests 26.1 – 26.2 are currently supported.
27	yes	

11. More Help

For more information on CDRouter and a listing of CDRouter TR-069 test cases, please visit the CDRouter TR-069 page at <http://www.qacafe.com/show/tr069>.

For general help running CDRouter, please check the CDRouter User Guide on-line at <http://www.qacafe.com/cdrouter>

Additional support notes on TR-069 can be found in the QA Cafe knowledge base at <http://www.qacafe.com/help>

For answers to common CDRouter and CDRouter TR-069 questions, see our Knowledge Base at <http://www.qacafe.com/kb>

For additional help, contact support@qacafe.com