ACP Test Server
User Guide

CONFIDENTIAL

Reference: 13003_ACPSRV_UG
Revision: 2.3
Date: 2009-09-23

Revisions

1	03/23/09	Initial Release.
2	07/13/09	Updated to ACP Test Gateway release.
2.1	07/28/09	Added documentation of the virtual hosts configuration file format.
2.2	08/12/09	Added ZPL license information for RestrictedPython.
2.3	09/23/09	Emphasis user interface usage instead of server administration.

Copyright Notice and Trade marks

ACP Test Server

User Guide

Document Version: 2.3

Document, Release Date, September 23th, 2009

EDANTECH (ILWICK S.A.),

Av. Libertador 1807,

11800, Montevideo, URUGUAY

Phone#: +598 2 929 0029

Fax#: +598 2 924 8013

e-mail: info@edantech.com

Contents

1 References

[ACP245]	ACP 245 – Application Communication Protocol v 1.2.1 http://www.denatran.gov.br/download/ACP_245_V1_2_1_19_05_09.pdf
[ACP_SPEC]	ACP Test Server Specification, Technical Specification, EDANTECH 13001_ACP_SPEC_2_1.pdf, available on Edantech Web Site.
[ACP_HD]	ACP Test Server , High level PDU Library Documentation, EDANTECH 13005_ACPSRV_HD_1_4.pdf, available on Edantech Web Site.
[ACP_HTTP]	ACP Test Server, HTTP Interface Specification, EDANTECH 13006_ACPSRV_HTTP_1_3.pdf, available on Edantech Web Site.

2 Usage Scenarios

The ACP Test Server can be used on the following scenarios:

- ✔ Testing of Service Operator (SO) servers protocol implementation.
- ✔ Testing of Telematic Control Units (TCU) client protocol implementation.
- ✔ Automated testing, driven by external tool or program.
- ✔ Supervised testing, by execution of predefined or customized test scripts.
- ✔ Regression tests, by customized test scripts and storage of previous test results with different server or client versions.
- ✔ Analysis of ACP245 message flows and binary format.

3 Overview

The ACP Test Server can be used to test both TCU (Telematic Control Units) and SO (Service Operators) from the same interface, allowing to test both clients and servers that use the ACP245 protocol.

The suite includes different interfaces to perform automated and operator assisted ACP245 protocol tests:

- To run simple tests, examine the message flow between server and clients, and test the behavior of each system as they respond to different ACP245 messages, the Test Console is the suggested interface.
- To execute a test driven by an external program or tool, the HTTP Interface is the best approach. Using it, you can integrate the ACP Test Server features with your custom testing tools.

This document describes the Test Console interface and the web interface that can be used to monitor the use of the HTTP interface. For a detailed specification of the HTTP interface itself, you can read the HTTP Interface Specification document on [ACP_HTTP].

! IMPORTANT: The system was developed for Firefox 3.5. Internet Explorer 7 and 8 are also supported, other browser will probably also work, but we don't support them and encourage our users to use Firefox 3.5 whenever possible.

4 Test Console

The Test Console is used to execute predefined or custom user-defined scripts against a ACP245 client or server. These scripts include control logic to check that the client/server behaves as expected.

The Test Console provides means to start/stop the scripts and examine the resulting message interchange, including a decoded view of the binary ACP messages sent and received.

It also provides some basic script editing capabilities, to modify the provided scripts and suit them

to your specific requirements. However, if preferred, the scripts can be modified using more powerful editors and then copied to the Console Script directory or pasted into the web editor.

The scripts are written on Python, using some built-in features provided by the ACP245 Test Server. For additional information on writing scripts, see the High-Level PDU Library documentation on [ACP_HD].

4.1 Accessing the Test Console

The test console can be accessed using a browser with the following URL:

`http://<server>/console/`

If configured, the test console can also be accessed securely using SSL through:

`https://<server>/console/`

The server name is provided by Edantech, if running using Edantech infrastructure, or by your system administrator if your company hosts the tools.

4.2 Predefined scripts

The Console allows you to run a set of predefined scripts that are included by Edantech. Some of them are just script examples that you can use to create your own scripts, other represent standard interactions for the ACP245 protocol.

The script collection **includes all the examples presented on Section 13.3 of ACP245 Protocol Specification** [ACP245], both for the server and client side, which constitute a minimum message set that must be supported by every TCU and Service Operator.

Each of these examples are named using the following convention:

`standard_<section_number>`

For example, the scripts that run the exchange described on Section 13.3.1, *Activation attempt without version element*, is called `standard_13.3.1`. All these scripts are completely documented on the script itself to describe what exactly they are testing.

You can use the ACP245 server to see how the message flow must proceed for each of this cases, by running a client script against a server script, as described on section 4.5, *Running a script against another script*.

4.3 Running a script against a Service Operator Server

To run a script against a Service Operator server you should first configure the server IP and port on the *client* section of the **Configuration Tab** of the Web Console Interface.

Once configured, you must go to the **Console Tab**,. Notice that the **Client Status Label** says *'stopped'* because the client has not yet started it's execution. Select your *client* script, and click on the *client* start button.

When you do so, the test client will try to connect to the Service Operator Server, and will inform you on the **Log Panel** if there was an error doing so. Once connected, the **Client Status Label** will

read *'started'* and the script will start it's execution and you will see the message flow and test result on the **Log Panel**.

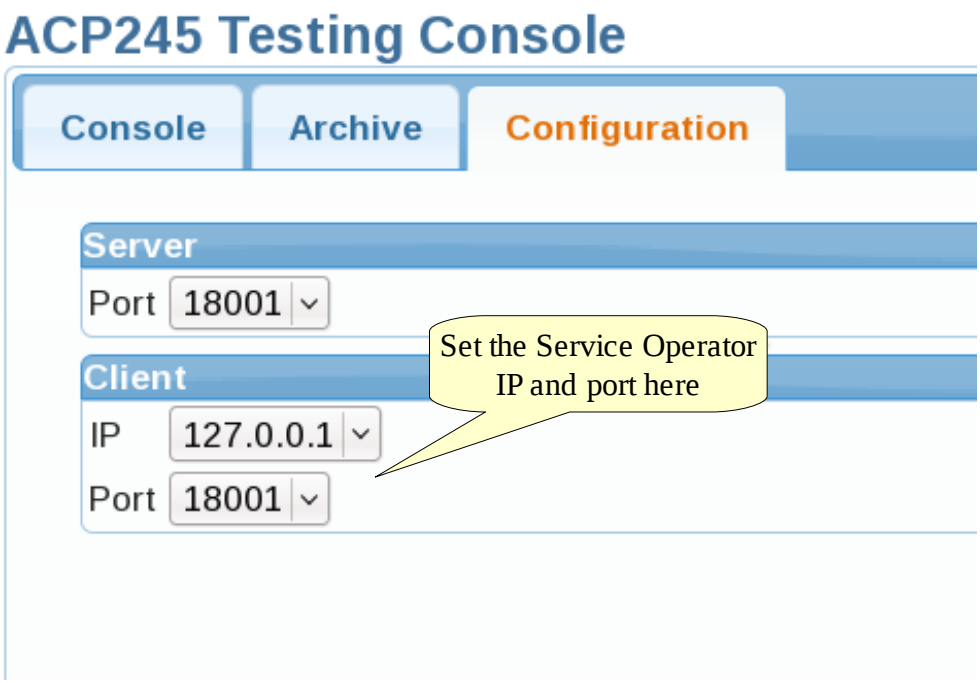

Figure 1: Client Configuration

To stop the client, you can click on the same button you used to start it. The client will also disconnect automatically when the script fails or comes to an end, or if the server closes the connection.

! IMPORTANT: For security reasons, the IP and port that can be selected on the **Configuration Tab** must be previously configured on the ACP Test Server by an administrator. If you need to connect to another IP and port, you must contact your administrator.

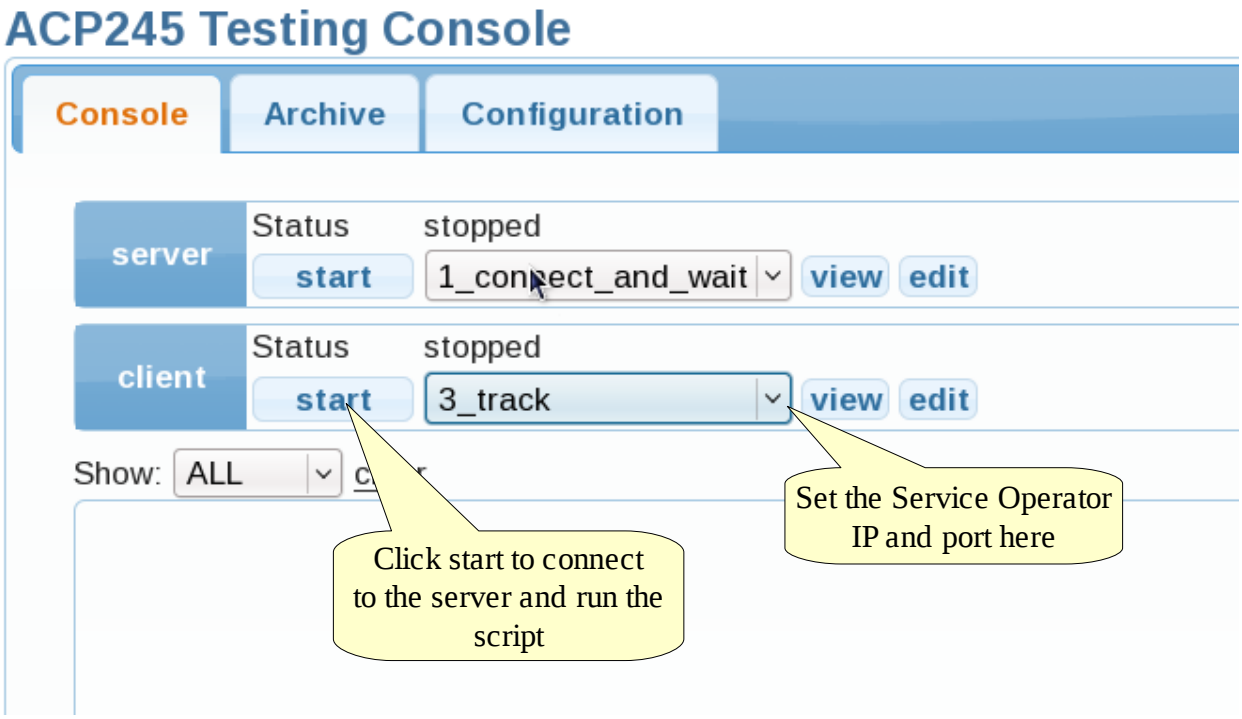

Figure 2: Using the ACP Test Server client

4.4 Running a script against a Telematic Control Unit (TCU)

To run a script against a TCU client, you should first configure the port to which the ACP Test Server will listen and the TCU will try to connect to on the *server* section of the **Configuration Tab** of the Web Console Interface.

Once configured, you must go to the **Console Tab**,. Notice that the **Server Status Label** says *'stopped'* because the server has not yet started to listen for connections. Select your *server* script, and click on the *server* start button.

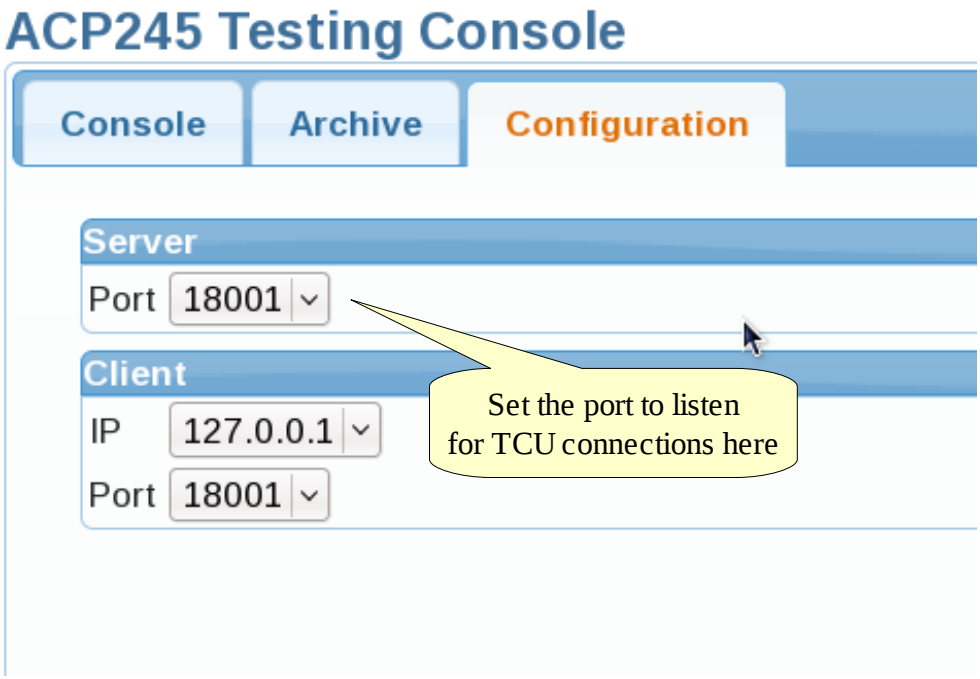

Figure 3: Server Configuration

When you do so, the server will start to listen to the configured port for a TCU connection and the **Server Status Label** will read *'started'*. When a TCU connects, the script you selected for the server will start executing and its output will be displayed on the **Log Panel**.

To stop the server, you can click on the same button you used to start it. Be aware that when the TCU disconnects from the server, the server will keep listening for new connections until you stop it. Each new connection will cause the script to start from the start. A test server only accepts one TCU connection at a time.

! IMPORTANT: For security reasons, the port on which you can listen for connections that can be selected on the **Configuration Tab** must be previously configured on the ACP Test Server by an administrator. If you need to use another port, you must contact your administrator.

4.5 Running a script against another script

You can use both the client and server of the ACP245 Test Server so they connect to each other.

For this, you only need to set the IP where the client will connect to 127.0.0.1 (the ACP245 Server local IP), and set the client port and server port to the same value.

In this scenario, you can execute a server script against a client script and examine the output. It's useful to see how a client/server must behave to comply with the given script, and also if you are trying to learn or understand how to pack a particular ACP245 binary message.

This is also useful to test a custom script that you are writing or modifying before running it against the real Service Operator server or TCU.

4.6 Analyzing the message flow and decoding messages

During a test script execution, you can monitor the output of the test using the **Log Panel**. For each event, a line will be displayed on the panel, including the time (in UTC), the origin of the message (server or client) and a general description.

Some lines includes a link to a decode ACP245 message, for example:

```
19:51:28,602 [server] Sent message "TrackReply" to 127.0.0.1:38261 [View MSG]
```

Indicates that a Track Reply was sent from the server to the address 127.0.0.1:38261 (from where there's a client TCU connected), and the [View MSG] is a link that allows you to see the decoded sent message, including the value for each one of it's fields.

Depending on the detail level you select for the log, some lines may include the binary message in hexadecimal or other low level information that you can use to diagnose problems. You can change the detail level of the log using the combo box included on the top of the **Log Panel**.

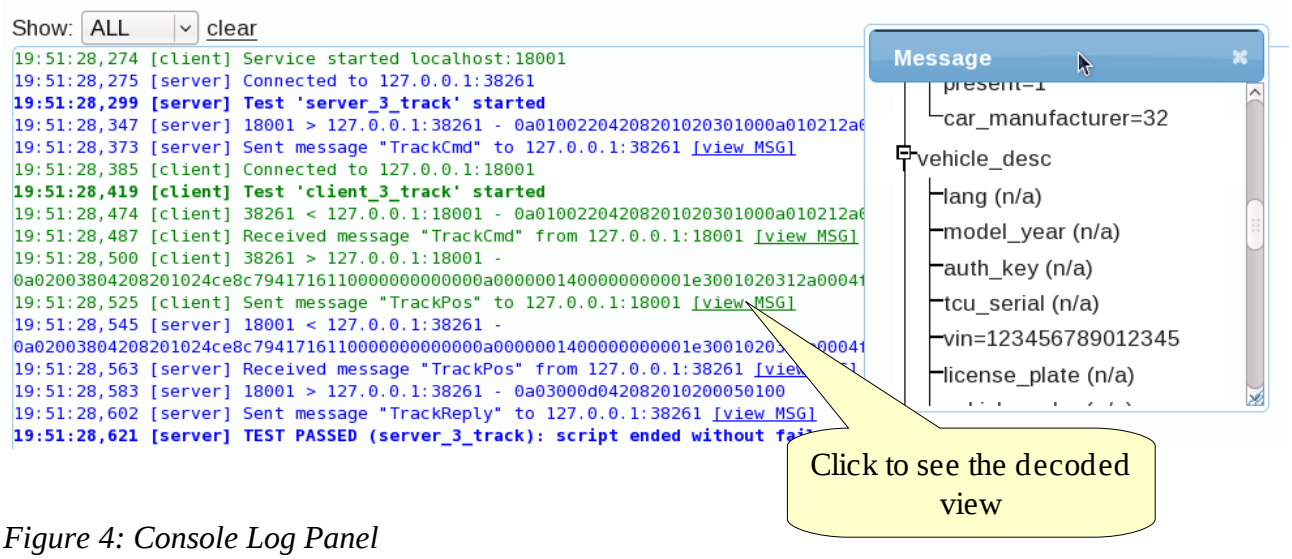

Figure 4: Console Log Panel

4.7 Reviewing previously executed tests

The console provides an **Archive Tab** where you can review the output of previously executed tests. Also, you can export the test in KML[1] format, so you can see the positions sent or received during the execution of the test on Google Earth, Google Maps, or other tools supporting KML.

You can click on each test to it's output. The output looks just like what it's displayed on the console **Log Panel**, you can even click on a message to see it's decoded representation just as you do on the panel.

On the bottom of the screen, you have icons to delete selected tests cases and to search among the test archive for tests performed in a range of dates or with a certain name.

1 http://code.google.com/apis/kml/

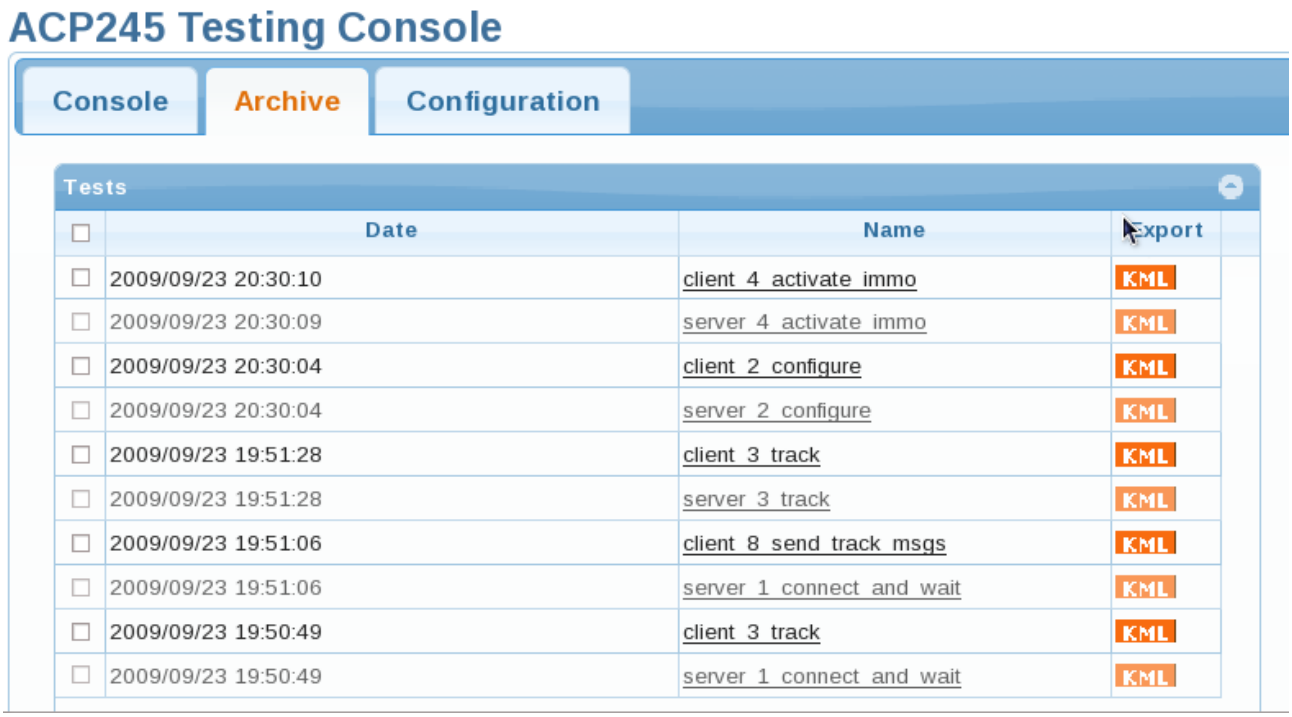

Figure 5: Console Archive Panel

4.8 Viewing and editing a test script

You can view and edit a test script by using the Console Interface.

To view a script, click on the **View Button** next to the script. It will display a box that includes the script with syntax highlighting, which makes it easy to read and understand it.

To edit a script, click on the **Edit Button** next to the script. It will display a text area where you can edit the script text. Once you are done, click on the **Save** or **Save As** button to save your changes.

Scripts are written using the Python programming language and built-in functions provided by the ACP245 Test Server. If you want to program a new script, you should read the High-Level PDU Library Documentation [ACP_HD], available on Edantech Web Site.

5 Gateway Console

The Gateway Console allows an operator to monitor and review previous logs of tests performed using the suite HTTP to ACP245 Gateway. This gateway has been designed to integrate with a test bench, and is therefore not designed to be particularly expressive or efficient, but easy to use.

The Gateway Console provides an status panel to monitor the tests that are being performed in real time, and an archive panel that includes a report of previously executed test with all the interactions with the gateway performed while running that test.

The tests are grouped based on indications from the test bench; if the test bench does not uses this functionality all gateway interactions will be grouped under the same test name.

5.1 Accessing the Gateway Console

The test gateway can be accessed with the following URL:

`http://<server>/gateway/`

If configured, the test gateway can also be accessed securely using SSL through:

`https://<server>/gateway/`

5.2 Running an automated test

To run an automated test you must use the HTTP interface as described on the HTTP Interface Specification [ACP_HTTP] from an external tool (or just using the browser if you are only trying it out).

You don't need to configure the listen port of the server or the IP and port to which a client will connect, since these can be specified on the HTTP command you use to start and stop a client or server. However, the same security restriction applies as in the console, the IP and ports specified must have been previously allowed by an administrator.

5.3 Monitoring the progress of an automated test

You can access the Gateway Web Interface to monitor the progress of an automated test that is using the HTTP interface.

During the test, the Log Panel will display when a test start, stops, all the messages exchanged during the test, and client/server connections and disconnections.

As in the case of the Test Console, you can click on some of the messages to see it's decoded view.

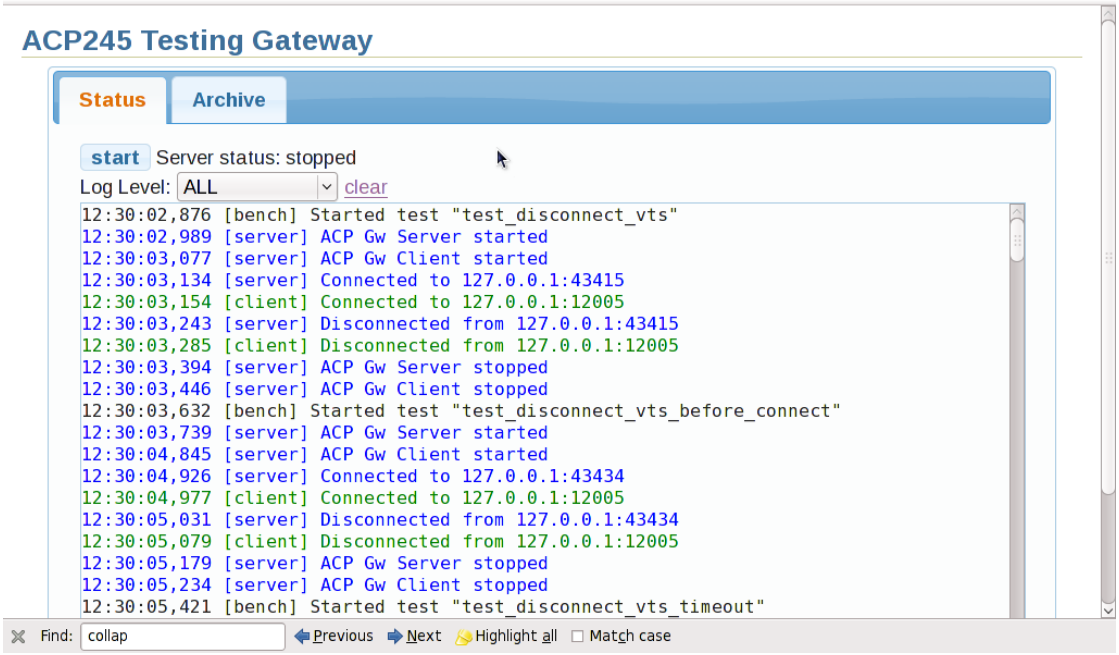

Figure 6: Gateway Status Panel

The **Server Start Button** of the Gateway **Status Panel** should not be used since the server can be started/stopped using HTTP gateway commands, as described on the HTTP Interface Specification [ACP_HTTP]. However, it can be used in case the server want to be forcefully stopped for some reason. In any case, the button will reflect the *current* server status (ie. if started, the button will display *'stop'* and the server status will display *'started'*).

5.4 Reviewing previously executed tests

As in the case of the console, the Gateway Interface provides an **Archive Tab** that displays all the tests previously executed using the HTTP interface.

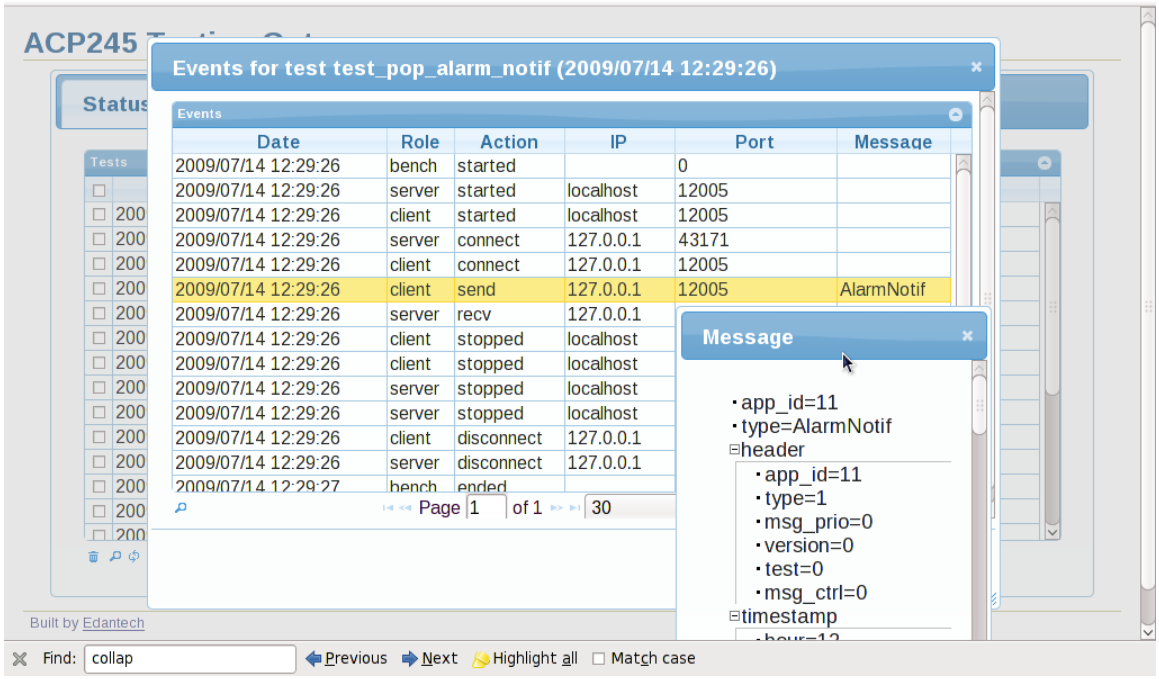

Figure 7: Gateway Archive Panel

6 Required Software Environment

This section describes the requirements to host your own ACP245 Test Server Suite in your own infrastructure. You don't need this if you are using Edantech infrastructure to host the ACP245 Test Server Suite. To just use the suite, you only need a standards compliant web browser.

6.1 Base software

The ACP 245 Test Server Suite has been designed to work on a Linux based operating system. Specifically, it has been tested on the following Linux distributions:

- Fedora Core 8, 10 and 11, for 32 and 64 bit processors.

The base system should be installed with only the minimum software packages. To follow the installation procedure correctly, the base system should include `yum` 3.2 (which are normally installed on the required systems by default).

6.2 Dependencies

Besides the base software, the system has the following dependencies, normally included on the target Linux distributions:

- Python 2.5
- Python setuptools
- Twisted Core 2.5
- Divmod Nevow 0.9.29
- Divmod Epsilon 0.50 (provided by Edantech in RPM format)
- Divmod Axiom 0.50 (provided by Edantech in RPM format)
- Python SimpleJSON

7 Required Hardware Platform

The system requires a hardware platform compatible with the selected base software. The hardware minimum requirements, besides the ones required by the base software, are the following:

- 256MB RAM of memory.
- Intel Celeron Processor 430 or better.
- 100MB of hard drive storage for system and required dependencies.

The system has been tested successfully on 32 and 64 bit processors.

8 Installation

To install the ACP Server Suite it's necessary to have a software and hardware environment

working and installed as specified above.

Once that the base system has been installed, the following procedure must be performed.

1. Copy the provided RPM packages to the `/root` directory. This can be performed by copying it from installation media or by using SCP protocol.
2. Install the provided RPM using `yum`, this will install required dependencies also.

   ```
   yum install e_libs*.rpm acp245-*.rpm acpyd-*.rpm python-axiom-*.rpm python-epsilon-*.rpm
   ```

   `python-axiom` and `python-epsilon` are dependencies that are provided by Edantech in RPM format, since they are not normally available for the target Linux distributions.
3. Check the system configuration of the ACP245 server (*acpyd*) as specified on the Server Management section.
4. Start the acpyd server

   ```
   service acpyd start
   ```

   The server should start producing the following output:

   ```
   Starting acpyd:                                            [  OK  ]
   ```
5. If default configuration options were used, the server will answer HTTPS requests on port 12002 for the Test Console and on 12004 for the Gateway Console.

9 Service Management

This section describes ACP245 Administrator management options. You don't need to read this if you are using the ACP245 Test Server hosted by Edantech.

9.1 Starting the service

To start the service, use the following command as `root` user:

```
service acpyd start
```

The service should start producing the following output:

```
Starting acpyd:                                            [  OK  ]
```

9.2 Stopping the service

To start the service, use the following command as `root` user:

```
service acpyd stop
```

The service should start producing the following output:

```
Stopping acpyd:                                            [  OK  ]
```

9.3 Checking service logs

Service logs are stored on `/var/log/acpyd/acpyd.log`.

Each virtual domain will keep a separate test log on `/var/log/acpyd/<domain>.log`.

You can use standard system tools to read the service log. The logs will be rotated automatically, and a copy of the old log will be stored with a different suffix.

9.4 Service Execution Options

The service execution options can be configured by editing a configuration file stored on `/etc/sysconfig/acpyd`

The file is documented to show how to specify the different available options. At the time of these documentation the options are:

Option	Description
`--config`	The filename of the server configuration file.
`--log-dir`	Directory to store per domain acpyd log files.
`--certificate`	Certificate to use for HTTPS SSL connections.
`--privkey`	Private key to use for HTTPS SSL connections.

The server uses `/var/acpyd/` as it's data directory, using it to store scripts, script event logs, templates, and static files which are served through the HTTPS connection.

9.5 Configuration File

The basic options and Virtual Host support is configured using the configuration file located in `/etc/acpyd/acpyd.conf`.

Be aware that, at the time, if the configuration is changed the server must be restarted.

The configuration file has the following format:

9.5.1 Basic Options

```
skel-dir    = string(default='vhost-skel')
```

Directory to use as base for new virtual hosts.

```
http-port   = integer(1, 65535, default=80)
```

HTTP port of the ACP Server web interface

```
https-port  = integer(1, 65355, default=443)
```

HTTPS port of the ACP Server web interface

```
privkey     = string(default='server.pem')
certificate = string(default='server.pem')
```

Private key and certificate for SSL connections

```
acp-console-port =  integer(1, 65535, default=12001)
```

Default ACP port where the console server listens and client connects to. This value is used if virtual host support is disabled.

```
acp-gateway-port =  integer(1, 65535, default=12005)
```

Default ACP port where the gateway server listens and client connects to. This value is used if virtual host support is disabled.

```
use-vhosts  = boolean(default=True)
```

If set to True, enables support for virtual hosts.

9.5.2 Virtual Hosts Options

The ACP Test Server can be configured to provide different server instances to different domains, so that users of one domain can perform tests without conflicting with other users.

To enable virtual host support, `use-vhosts` option of the basic configuration options, must be set to `True`.

These options follow the basic options in the configuration file.

Per domain section:

```
[domains]
```

Indicates the start of the per-domain section.

Include an additional [[...]] for each domain.

For example, to add the virtual host test.acptest.edantech.net, add:

```
# [[test.acptest.edantech.net]]
# .... configuration option as described below ...
```

Per domain options:

For each virtual host you can provided the following options:

```
debug-level = option('debug', 'error',default='debug')
```

Debug level.

```
acp-console-port =  integer(1, 65535, default=12001)
```

Default ACP port where the console server listens and client connects to.

```
acp-gateway-port =  integer(1, 65535, default=12005)
```

Default ACP port where the gateway server listens and client connects to.

```
acp-gateway-allowed-ports = int_list(default=list())
```

ACP ports that the gateway server can be commanded to listen and connect to.

```
acp-gateway-allowed-ips = ip_addr_list(default=list())
```

ACP IPs that the gateway server can be commanded to connect to (using gateway commands).

```
user-pass = string_list(default=list())
```

A comma separated list of <user>:<password> tuples with the user names that are allowed to connect to the server. If no value is provided, the server will not require authentication.

10 Contacting support

Support is provided under the following conditions:

- An incident must be reported by the Customer to Edantech using email, phone or an alternative mechanism proposed by Edantech (ie. Support. Web site). Contact information will be provided on the commercial proposal.

  Email contact information: support@edantech.com

- After the incident is reported, Edantech is bound to answer and solve the incident in the terms specified on the commercial proposal (supplier statement of work or services contract document).
- Edantech will only resolve incidents relating to the ACP Server or the base software used on the Server (operating system, database systems, etc.).
- All support will be provided remotely, by establishing a SSH connection to the server running the ACP Sever.
- Once the Server is installed at the Customer site, network and hardware issues that impede Edantech for connecting to the server must be fixed by the Customer (by reinstalling or replacing the server, or fixing their network configuration).

RestrictedPython is provided under ZPL license.

ZPL 2.1

Zope Public License (ZPL) Version 2.1

A copyright notice accompanies this license document that identifies the copyright holders.

This license has been certified as open source. It has also been designated as GPL compatible by the Free Software Foundation (FSF).

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions in source code must retain the accompanying copyright notice, this list of conditions, and the following disclaimer.
2. Redistributions in binary form must reproduce the accompanying copyright notice, this list of conditions, and the following disclaimer in the documentation and/or other materials provided with the distribution.
3. Names of the copyright holders must not be used to endorse or promote products derived from this software without prior written permission from the copyright holders.
4. The right to distribute this software or to use it for any purpose does not give you the right to use Servicemarks (sm) or Trademarks (tm) of the copyright holders. Use of them is covered by separate agreement with the copyright holders.
5. If any files are modified, you must cause the modified files to carry prominent notices stating that you changed the files and the date of any change.

Disclaimer

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS ``AS IS'' AND ANY EXPRESSED OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT HOLDERS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.