ANTlabs
Connectivity Made Easy

INNGATE 3

API DEVELOPER'S GUIDE

DOCUMENT RELEASE 1.01

Powered by
ezxcess™
GATEWAY

Incorporating
tru'connect™
TECHNOLOGY

InnGate 3 API Developer's Guide

This guide covers applications development using the InnGate Application Programming Interface (API) and is intended for web programmers and developers who intend to perform customizations on the InnGate.

TRADEMARKS AND ACKNOWLEDGEMENTS

DISCLAIMER

No part of this manual may be copied, distributed, transmitted, transcribed, stored in a retrieval system or translated into any human or computer language, in any form or by any means, electronic or otherwise, without the express written permission of ANT*labs*.

The software and accompanying written materials (including instructions for use and this document) are provided "as is" without warranty of any kind.

ANT*labs* does not warrant, guarantee or make any representations regarding the use, or the results of the use, of the software or written materials in terms of correctness, accuracy, reliability, trend or otherwise. ANT*labs* reserves the right to make changes without further notice to any products described herein to improve reliability, function or design. This documentation is copyrighted and may not be altered without written consent from ANT*labs*.

ANT*labs* reserves the right to prosecute companies or individuals who make, distribute or use illegal copies of this software system and its accompanying documentation.

Release Date: July 2010
Document Reference No: IG3-API-1.01

CONTENTS

PREFACE

AUDIENCE

This manual is intended for administrators who will be responsible for the installation and configuration of the InnGate.

This manual will explain how first-time installation and configuration should be done as well as the tasks involved in performing regular maintenance and configuration.

Administrators are expected to have a good working knowledge of networks and TCP/IP. Knowledge of the operating environment and characteristics of the systems used in the deployed networks are also useful. Basic knowledge of HTML and HTTP will also allow the administrator to customize the user-facing web pages.

RELATED DOCUMENTATION

You may refer to the ANT*labs* homepage at **http://www.antlabs.com/** for other related materials and documents released by ANT*labs*.

FEEDBACK AND COMMENTS

ANT*labs* welcomes all comments and suggestions on the quality and usefulness of this document. Our users' feedback is an important component of the information used for improvement of this document.

Please include in your feedback:

- Name
- Title
- Company
- Department
- E-Mail
- Postal Address
- Telephone Number
- Document Title & Release No
- Document Reference No.
- Comments/Feedback

Also, please include the chapter, section and/or page number when referring to specific portions of the document.

Send your comments via email to **documentation@antlabs.com**

Chapter 1

INTRODUCTION

1.1 Overview

The InnGate API is a set of modules that give developers access to system services and resources. This enables developers to program the behavior of the InnGate, allowing flexibility to customize many aspects of the user experience.

The API is most commonly used for:

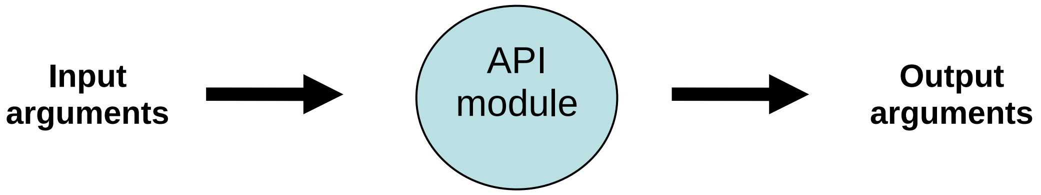
1. **On-demand services** – The API provides functions that can dynamically update user information, enabling on-demand services such as buying additional online time by updating the user's session duration or upgrade of access privileges by changing the user group.

2. **Applications integration** – The API allows the retrieval of information about the connected devices, enabling external applications to act on the status and events of client devices that are managed by the gateway.

Developers can thus leverage the InnGate API to create value-added services whose transactions can span a variety of existing or new application servers. This allows businesses to roll-out new service offerings to its customers.

1.2 What is the InnGate API?

The API consists of modules, each providing a particular service or function. Most modules require a set of input arguments to execute and once the operation is completed, a set of output arguments is generated which contain the results of the operation.



There are 2 methods of using the API:

1. **Making calls to the API in PHP scripts** – PHP scripts can be uploaded to the gateway and executed by the built-in PHP application server. The InnGate API can be accessed from within these scripts.

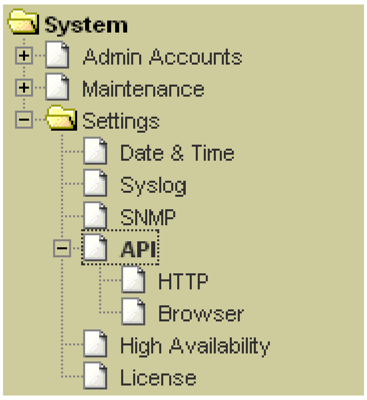
2. **Invoking the API via HTTP** – The InnGate API can also be invoked via HTTP. This is especially useful when external applications need to communicate and share information with the gateway over the network.

Each of these methods will be covered separately in the subsequent chapters.

A reference of all the API modules can be found in Appendix A. The modules are sorted alphabetically and provide developers with a reference of the functionality, input and output and parameters for each module.

1.3 Checking API Versions

The version of the InnGate API and a list of the installed API modules and their respective versions can be viewed in the Admin GUI.

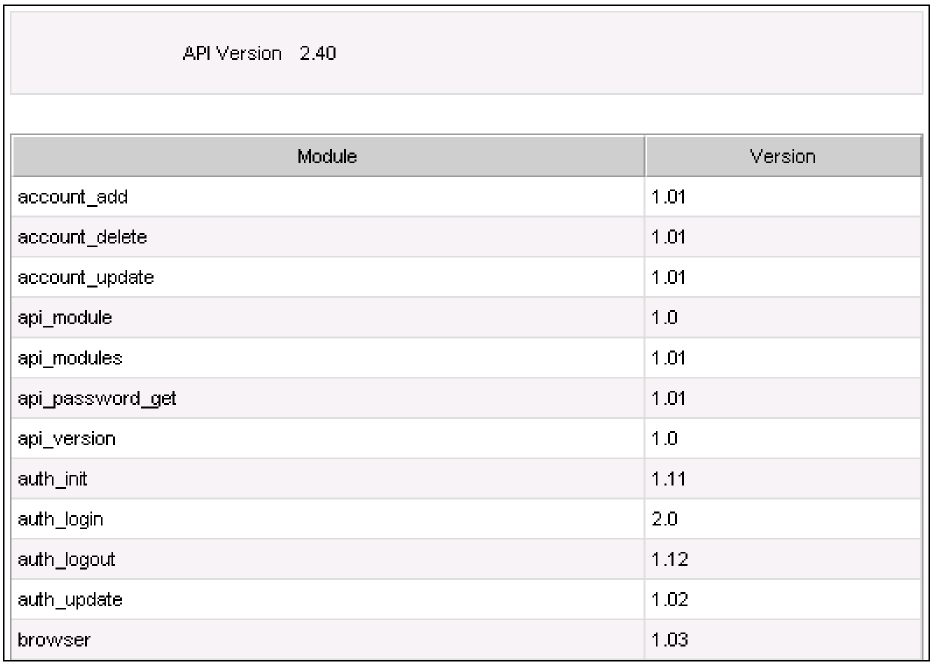


To see API versions:

1. Click on **Settings**.

2. Click on **API**.

The API version is displayed, along with a list of all API modules installed in the gateway.



API Version 2.40

Module	Version
account_add	1.01
account_delete	1.01
account_update	1.01
api_module	1.0
api_modules	1.01
api_password_get	1.01
api_version	1.0
auth_init	1.11
auth_login	2.0
auth_logout	1.12
auth_update	1.02
browser	1.03

Figure 1-1 API Version and Installed API Modules

Chapter 2

USING THE INNGATE API

2.1 Invoking the API in PHP

The gateway has a built-in PHP application server allowing custom PHP scripts to be uploaded and executed. Developers can then make API calls within these scripts to invoke the necessary functionality for their applications.

This chapter explains how the API can be invoked from within a PHP script hosted on the gateway.

2.1.1 Basic Steps for using the API in PHP

The following are the basic programming steps for using the API in PHP scripts:

1. **Include the API class at the start of the PHP script.**

   ```
   require_once($_SERVER['DOCUMENT_ROOT'] .
   '/api/api.php');
   ```

   If you do not do this, the functionality of the API will not be accessible to the script and calls to the API will generate errors.

2. **Instantiate an API Object.**

   ```
   $api = new API();
   ```

   This will create an API object that you will use to invoke the necessary functionality.

3. **Set the Input Arguments required by the module.**

   ```
   $api->SetArg('inputargument1', 'inputvalue');
   $api->SetArg('inputargument2', 'inputvalue');
   ```

   The input arguments needed for each API module can be found in Appendix A.

4. **Execute the API module.**

   ```
   $api->Execute('modulename');
   ```

Use the `Execute` class method and specify the name of the API module to invoke. The module will use the arguments set in the previous step.

5. **Get the Output Arguments of the operation.**

```
$api->GetResult('outputargument1');
$api->GetResult('outputargument2');
```

When a module is executed, output arguments contain the results of the operation. The `GetResult` method allows you to retrieve the results by passing the name of the output argument. The output arguments generated by each module found in Appendix A.

The final result of the script will be as shown below:

```
require_once($_SERVER['DOCUMENT_ROOT'] . '/api/api.php');

$api = new API();
$api->SetArg('inputargument1', 'inputvalue');
$api->SetArg('inputargument2', 'inputvalue');

$api->Execute('modulename');
$api->GetResult('outputargument1');
$api->GetResult('outputargument2');
```

2.1.2 Example API Call in PHP

As a practical example of the steps above, the following section of code illustrates the use of the API to obtain the status of a given device on the network.

```
require_once($_SERVER['DOCUMENT_ROOT'].'/api/api.php');

$api = new API();
$mac_address = $_GET[‘mac’];
$api->SetArg('client_mac', $mac_address);
$api->Execute('device_status');

if ($api->GetResult('result') == 'ok')
{
   if ($api->GetResult('connected') == 'yes')
   {
      $ip_address = $api->GetResult('client_ip');
      echo 'Client IP address: ' . $ip_address;
   }
}
```

This example illustrates:

1. Including the functionality of the API in the PHP script with the **`require_once`** statement.

2. Creating a new API object with the **`new API()`** statement.

3. Getting the MAC address of the client.

4. Using the **`SetArg`** method to set the **`client_mac`** input argument needed by the **`device_status`** module which is used to retrieve information about a device connected to the network identified by its MAC address.

5. Using the **`Execute`** method to invoke the **`device_status`** module.

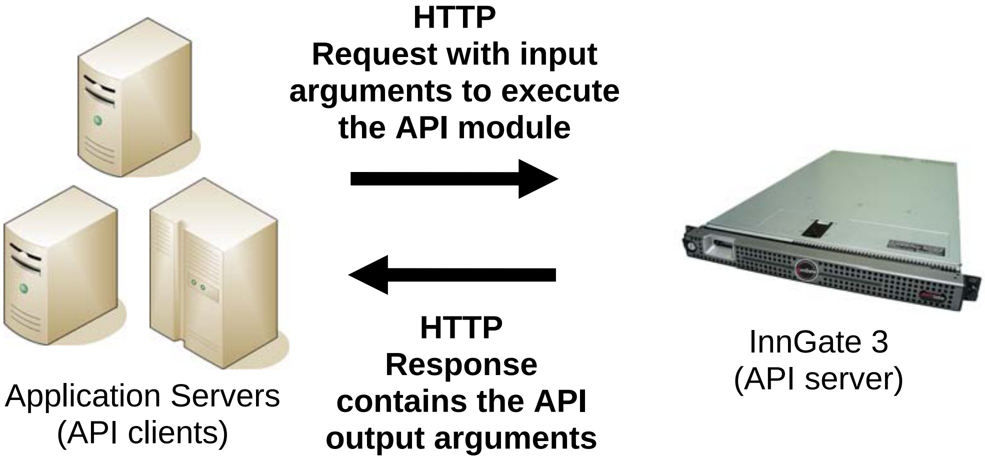
6. Using the **`GetResult`** method to retrieve the **`result`** output argument to find out if the operation is successful.

7. Using the **`GetResult`** method to retrieve the **`client_ip`** output argument so as to display the IP address of the device with the **`echo`** statement.

2.2 Invoking the API via HTTP

HTTP provides a standardized communications protocol through which the functionality of the API can be exposed to external entities over the network.

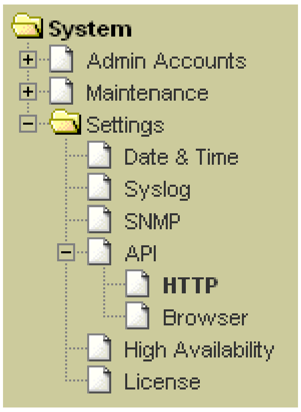
External application servers that need to interface with the InnGate API will customize their scripts to send HTTP API Requests to the gateway and parse the HTTP Response.


HTTP
Request with input arguments to execute the API module
HTTP
Response contains the API output arguments
Application Servers
(API clients)
InnGate 3
(API server)

This means existing applications distributed on the network can interface easily with the gateway to leverage on the functionality of the Inngate API as a building block to create integrated services with a higher value-add.

2.2.1 System Setup for HTTP API Access

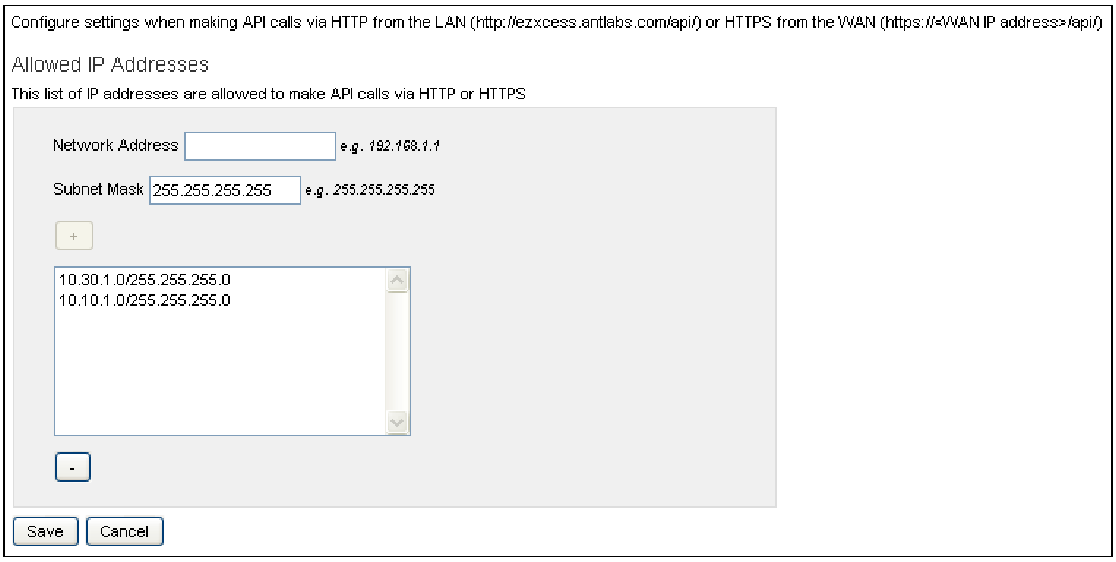
Due to the open nature of HTTP, access to the API via HTTP is restricted and secured through the Admin GUI (see InnGate 3 Administrator's Manual). By default the access to the API via HTTP is blocked.



To configure API access via HTTP:

1. Click on **Settings**.
2. Click on **API**.
3. Click on **HTTP**.

A list of IP addresses that are allowed to access the API will be shown, if any.



Configure settings when making API calls via HTTP from the LAN (http://ezxcess.antlabs.com/api/) or HTTPS from the WAN (https://<WAN IP address>/api/)

Allowed IP Addresses

This list of IP addresses are allowed to make API calls via HTTP or HTTPS

Network Address e.g. 192.168.1.1

Subnet Mask 255.255.255.255 e.g. 255.255.255.255

10.30.1.0/255.255.255.0
10.10.1.0/255.255.255.0

Figure 2-1 HTTP API Allowed IP Addresses

The fields are described as follows:

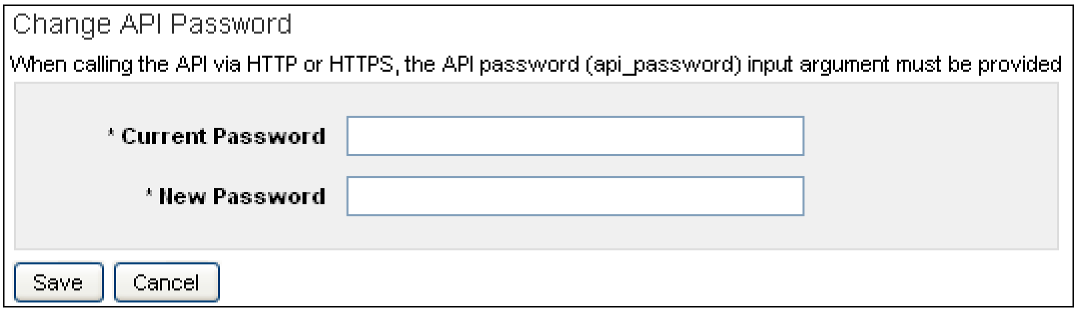
1. **IP Address** and **Subnet Mask** – Specify the host IP or network IP address allowed to access the API.

Click + to add the entry to the list or - to delete selected entries.

Click Save to commit the list.

In addition, you should change the password that is required to be sent as a HTTP Request parameter for all HTTP API Requests as shown in Figure 2-2.

⚠ The default API password is **admin**.



Change API Password

When calling the API via HTTP or HTTPS, the API password (api_password) input argument must be provided

* Current Password

* New Password

Figure 2-2 Change API Password

2.2.2 Example HTTP API Calls

An API client will send an HTTP API Request to the gateway. The gateway then processes the API call and sends the results back in the HTTP Response.

Some examples are shown here to illustrate the process.

Example 1

An HTTP API URL that calls **`api_module`** to get information about a particular API module (see Appendix A) might look like this:

```
https://192.168.123.21/api/?op=api_module&api_password=
admin&module=device_status
```

The above HTTP API Request comprises of the following elements:

1. **`https://192.168.123.21/api/?`** is the portion of the URL which addresses gateway and invokes the API from the **upstream**. The IP address is thus the WAN interface IP of the gateway.

   The API can also be accessed from the downstream. For LAN access, the URL should be:

   `http://ezxcess.antlabs.com/api/?`

   ⚠ Note that HTTP API access from the WAN uses the HTTPS protocol while access from the LAN uses HTTP.

2. **`op=api_module`** is the HTTP Request parameter that identifies the module to execute.

3. **`api_password=admin`** is the HTTP Request parameter that supplies the password needed to invoke the API from an external source. The API password is configured through the Admin GUI (see Section 2.2.1).

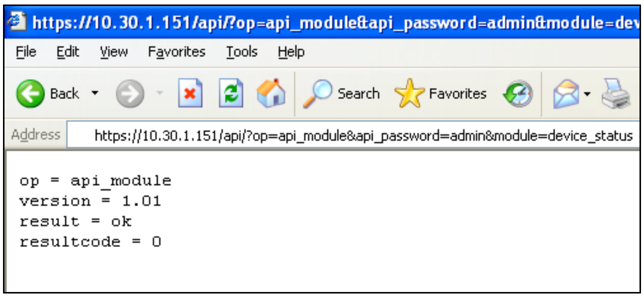
4. **`module=device_status`** is the HTTP Request parameter that provides the input argument required to execute `api_module` (see Appendix A).

Once the API has completed its execution, the output arguments generated are sent back to the client in the HTTP Response in clear text format:

```
op = api_module
version = 1.01
result = ok
resultcode = 0
```

The client is then expected to parse the text with standard string functions to obtain and interpret the output arguments.

You can try this out by entering the URL in the Address field of your Internet browser. The result will be sent back and presented in your browser as shown in Figure 2-3.



https://10.30.1.151/api/?op=api_module&api_password=admin&module=dev

https://10.30.1.151/api/?op=api_module&api_password=admin&module=device_status

```
op = api_module
version = 1.01
result = ok
resultcode = 0
```

Figure 2-3 Browser initiated HTTP API Request

Example 2

Here is another example of an HTTP API request that calls the **`device_status`** module to get information about a connected device on the network:

```
http://ezxcess.antlabs.com/api/?op=device_status&api_pa
ssword=admin&client_mac=00:11:25:87:0B:7D
```

The above HTTP API Request may generate the following HTTP Response in clear text format:

```
op = device_status
version = 1.01
```

```
result = ok
resultcode = 0
connected = yes
failed_probes = 0
internet_access = no
logged_in = no
client_ip = 10.128.250.254
ppli = eth0
vlan =
vlan_moved = no
location_index = 2
url =
```

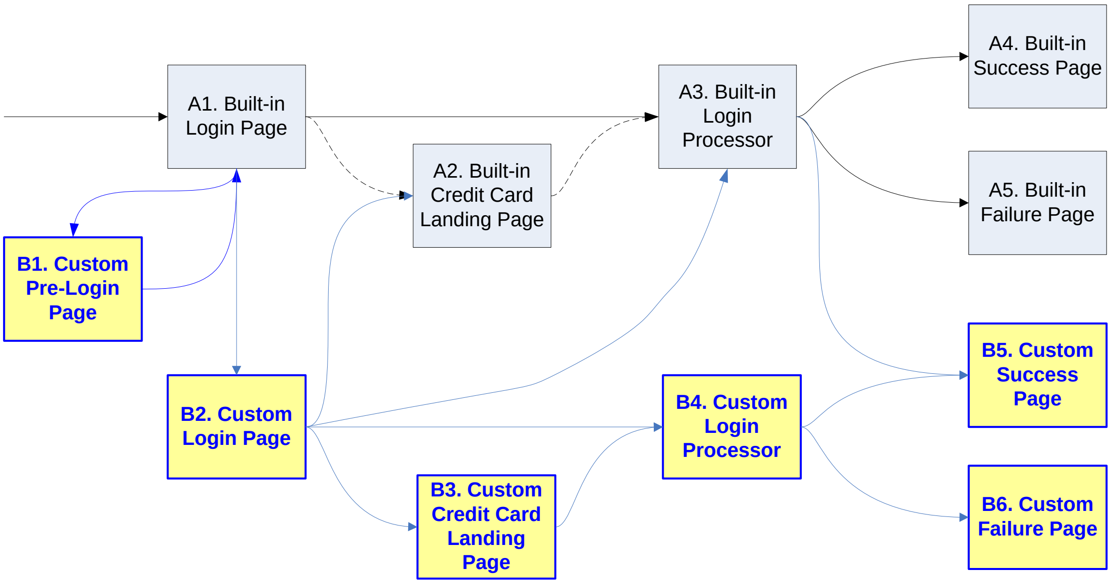
The client can then parse the text to obtain the IP address of the specified connected device.

Chapter 3

CUSTOMIZING THE LOGIN EXPERIENCE

3.1 Overview

In this chapter, we will explore how the API can be used to customize the standard login process as shown in Figure 3-1.



A4. Built-in Success Page
A1. Built-in Login Page
A3. Built-in Login Processor
A2. Built-in Credit Card Landing Page
A5. Built-in Failure Page
B1. Custom Pre-Login Page
B5. Custom Success Page
B2. Custom Login Page
B4. Custom Login Processor
B3. Custom Credit Card Landing Page
B6. Custom Failure Page

Figure 3-1 Customized Login Process

The diagram above shows the various standard built-in pages of InnGate and also the different custom pages that can be created using the standard APIs provided.

The standard built-in pages are:

A1. **Built-in Login page** – The standard login page that is shown to the user before authentication. The look and feel of this page is configurable using the Login Policy Generator in the Admin GUI under *Policies -> Locations*.

A2. **Built-in Credit Card Landing page** – The standard credit card landing page when the user selects credit card payment.

A3. **Built-in Login Processor** – The standard login processor.

A4. **Built-in Success page** – The standard success page shown to the user after successful authentication. This page is configurable using the Login Policy Generator.

A5. **Built-in Failure page** – The standard failure page shown to the user when authentication is not successful.

The custom pages that can be written are:

B1. **Custom Pre-Login Page** – Useful for showing ads, customized terms and conditions page.

B2. **Custom Login Page** – Useful for defining custom login options

B3. **Custom Credit Card Landing Page** – Custom Credit Card payment methods

B4. **Custom Login Processor** – self-explanatory

B5. **Custom Success Page** – self-explanatory

B6. **Custom Failure Page** – self-explanatory

3.2 Different Customization Options

InnGate supports different levels of customizations, depending on the customer's needs. The 4 common options of customization available are:

1. Customizing the Pre-login page.

This can be achieving by defining a Custom Pre-Login page (B1) which directs the user back to the Built-in (A1) Login page.

This option is useful for delivering simple Ads, customized Terms and Conditions and messages to the users.

2. Customizing the Login Page(s).

This can be achieved by defining a set of Custom Login Pages (B2) which is used to specify the various login types and collect the necessary inputs. If Credit card payment is required, these pages can include a Custom Credit Card Landing Page (B3). Once all the necessary information is collected, the custom page will post to the Built-in Login Processor Page (A2) to process the login request and display the results.

This option is applicable if you only want to customize the login page look and feel and want to reuse the built-in Login Processor for handling the login request.

3. Customizing the Success Page.

This can be achieved by defining a Custom Success page (B5) which the Built-in Login Processor (A3) will direct the user to after successful authentication.

This option is applicable if you want to customize the look and feel of the success page to display extra information that is not available via the Built-in Success page (A4)

4. Full customization.

This can be achieved by defining Custom Login pages (B2) which can optionally include a Custom Credit card Landing Page (B3). These pages will eventually post to a Custom Login Processor (B4) which will process the authentication request and redirect the user to either the Custom Success page (B5) or the Custom Failure page (B6) based on the authentication result

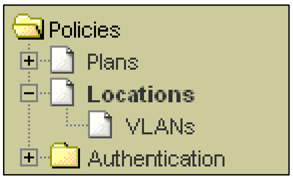
This option provides the most flexibility and allows you to create complex user login process flows to handle special business requirements.

3.3 Step by Step Configuration

The following subsections will define how to configure the InnGate GUI to support the following customization:

1. Custom Pre-login Page
2. Custom Login Page(s)
3. Custom Success Page
4. Full Customization

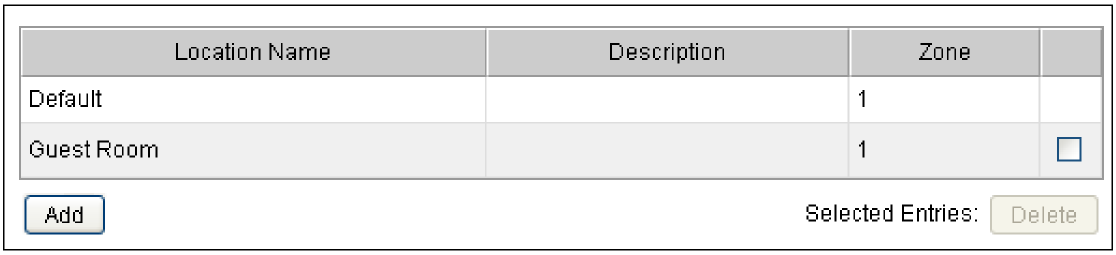
3.3.1 Custom Pre-login Page Configuration



To configure Custom Pre-login Page:

1. Click on **Locations.**

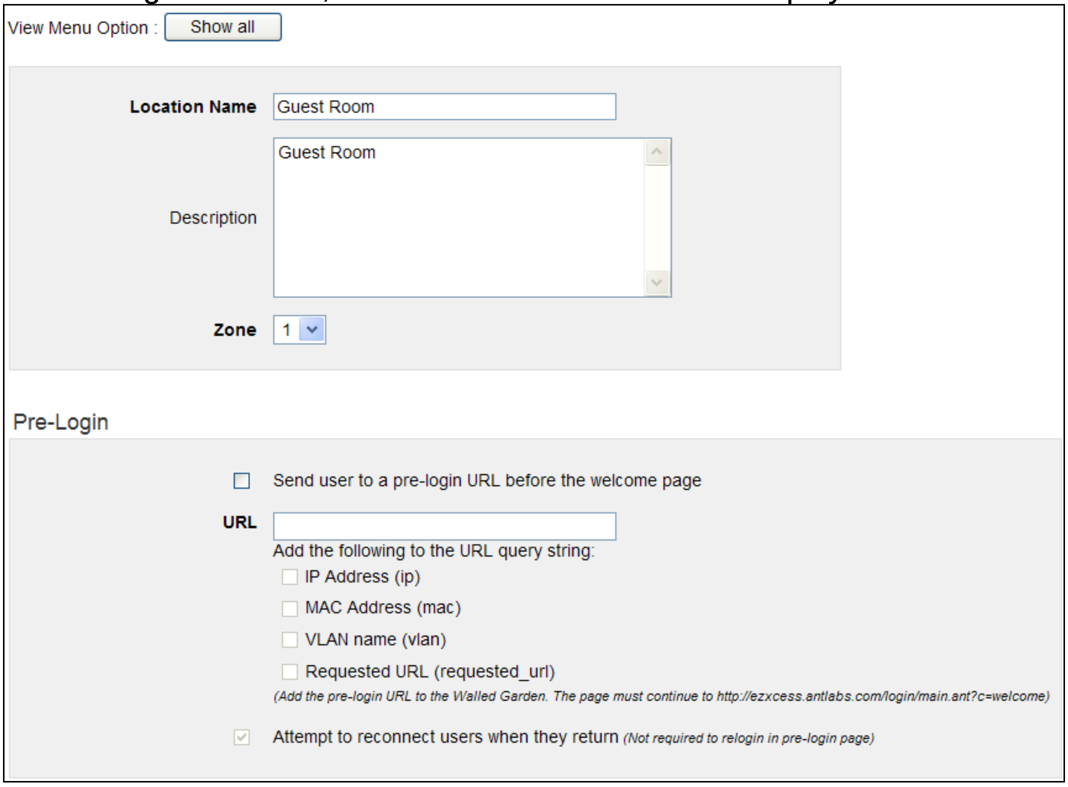
A list of existing locations will be displayed. Click on an entry to modify it or click Add to create one.



Location Name	Description	Zone	
Default		1	
Guest Room		1	□

Figure 3-2 List of Locations

After making a selection, details about the Location is displayed.



Location Name Guest Room

Description Guest Room

Zone 1

Pre-Login

□ Send user to a pre-login URL before the welcome page

URL

Add the following to the URL query string:

□ IP Address (ip)

□ MAC Address (mac)

□ VLAN name (vlan)

□ Requested URL (requested_url)

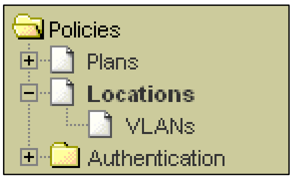
(Add the pre-login URL to the Walled Garden. The page must continue to http://ezxcess.antlabs.com/login/main.ant?c=welcome)

☑ Attempt to reconnect users when they return *(Not required to relogin in pre-login page)*

Figure 3-3 Location details

To configure a Custom Pre-login Page, enable the option '**Send user to a pre-login URL before the welcome page**'.

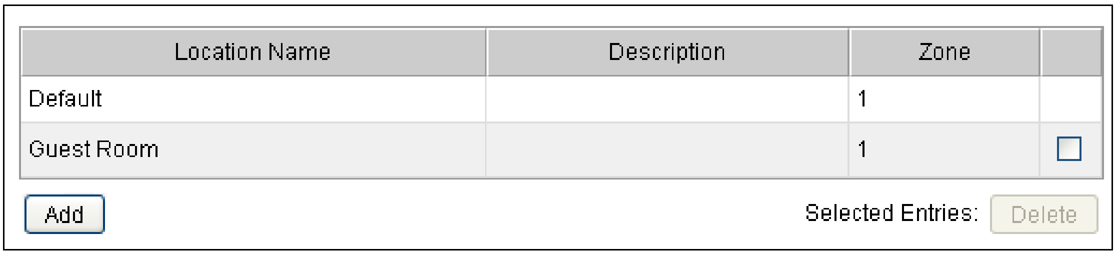
3.3.2 Custom Login Page(s) Configuration



To configure Custom Login Pages:

1. Click on **Locations.**

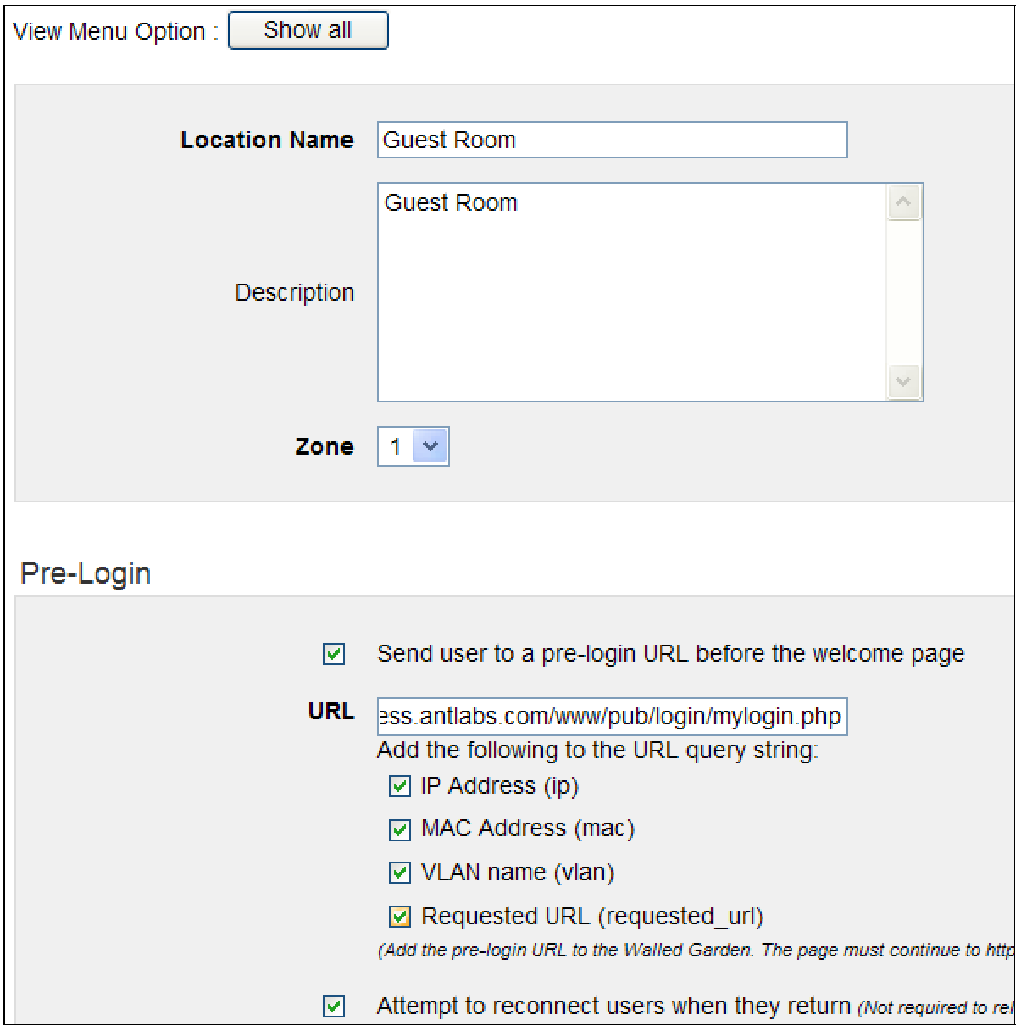
A list of existing locations will be displayed. Click on an entry to modify it or click Add to create one.



Location Name	Description	Zone	
Default		1	
Guest Room		1	□

Figure 3-4 List of Locations

After making a selection, details about the Location is displayed.



Location Name Guest Room

Description Guest Room

Zone 1

Pre-Login

☑ Send user to a pre-login URL before the welcome page

URL ɘss.antlabs.com/www/pub/login/mylogin.php

Add the following to the URL query string:

☑ IP Address (ip)

☑ MAC Address (mac)

☑ VLAN name (vlan)

☑ Requested URL (requested_url)

(Add the pre-login URL to the Walled Garden. The page must continue to http

☑ Attempt to reconnect users when they return *(Not required to re*

Figure 3-5 Location details

To configure a Custom Login Page, enable the option ‘**Send user to a pre-login URL before the welcome page**’.

1. **URL** – This is the URL of the Custom Login page to send the user to.

2. **ip, mac, vlan, requested_url** – Check on the option **MAC address** (mac) to pass the MAC address to the external Custom Login page.

3. **Attempt to reconnect users ...** - When this option is checked the InnGate will automatically do re-login check before redirecting the user to the custom pre-login page.

You can now fully customize the Login Page based on your preference. To obtain the various plans that are available, you can call the **plan_get_all** API to obtain the necessary information.

To continue with the login process, your Custom Login pages must eventually post to the Built-in Login Processor.

The URL of the Built-in Login processor is:

For complimentary access, access code authentication, local user id and password authentication and PMS authentication

```
http://ezxcess.antlabs.com/login/main.ant?c=proc
```

For credit card authentication

```
http://ezxcess.antlabs.com/login/main.ant?c=cc
```

The following POST parameters are supported for the Built-in Login Processor.

1. **p** – payment type. Acceptable values are:
    a. **complimentary** – complimentary access
    b. **code** – access code authentication
    c. **local** – local user id and password authentication
    d. **pms** – PMS authentication
    e. **cc** – credit card authentication

2. **code** – access code (p=code only)

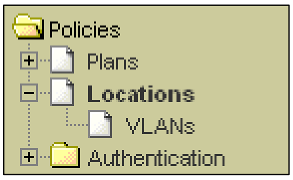
3. **uid** – user id (p=local)
    PMS Room number (p=PMS, guest based authentication)

4. **pwd** – password (p=local)
    PMS password (p=PMS, guest based authentication)

5. **plan** – plan id (p=PMS or p=cc only)

⚠ If your Custom Login Pages reside on an external server, you will need to create a **Walled Garden** entry, specifying the IP address of the external authentication server. This will allow the client to communicate with the authentication server prior to login.

3.3.3 Custom Success Page Configuration

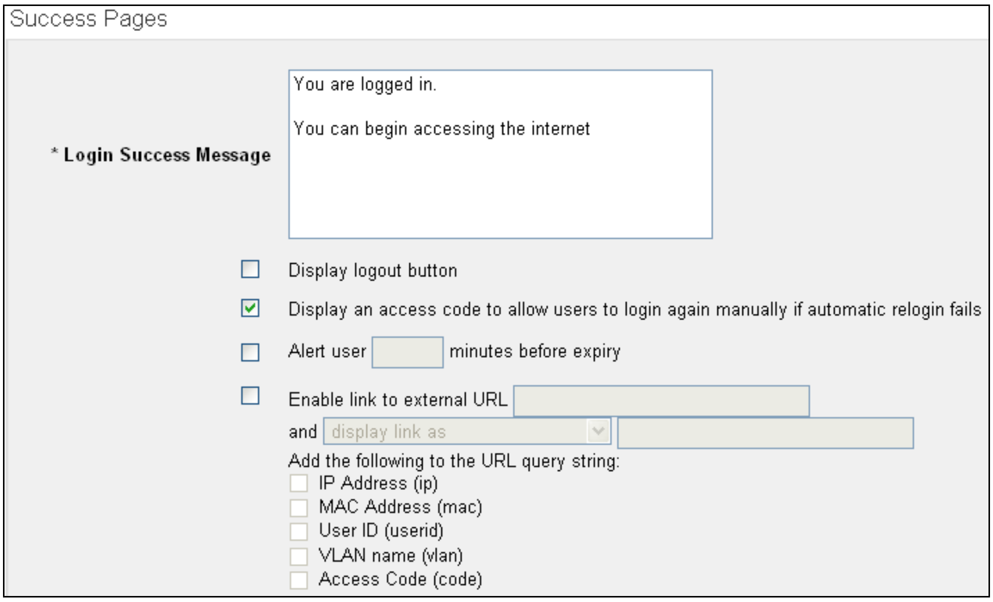


To configure Custom Success Page:

1. Click on **Locations.**

A list of existing locations will be displayed. Click on an entry to modify it or click Add to create one.

After making a selection, details about the Location is displayed. Click Next Step > button until you see the **Success Pages** screen.



Success Pages

* **Login Success Message**: You are logged in.

You can begin accessing the internet

□ Display logout button

☑ Display an access code to allow users to login again manually if automatic relogin fails

□ Alert user ____ minutes before expiry

□ Enable link to external URL ____
and display link as ____

Add the following to the URL query string:

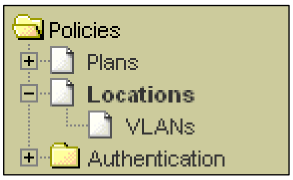
□ IP Address (ip)
□ MAC Address (mac)
□ User ID (userid)
□ VLAN name (vlan)
□ Access Code (code)

Figure 3-6 Success Page Configuration

Enable the checkbox '**Enable link to external URL**', type in the Custom Success Page URL and select the option 'use link as login success page'.

You can also choose to pass the zero-configuration variables, such as IP address, MAC address, User ID, VLAN or access code to the Custom Success Page for further customization.

3.3.4 Full Customization Configuration

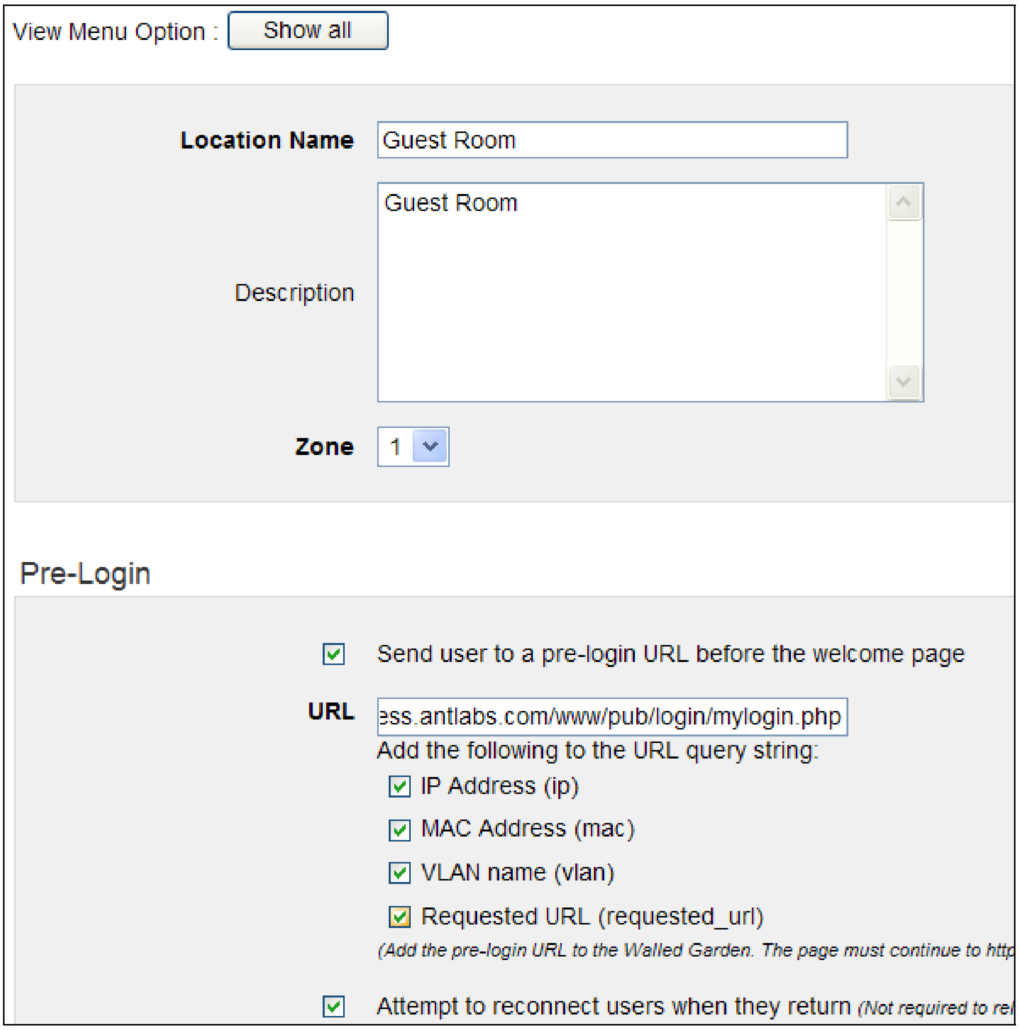


To configure for Full Customization:

1. Click on **Locations.**

A list of existing locations will be displayed. Click on an entry to modify it or click Add to create one.

After making a selection, details about the Location is displayed.



Location Name Guest Room

Description Guest Room

Zone 1

Pre-Login

☑ Send user to a pre-login URL before the welcome page

URL ess.antlabs.com/www/pub/login/mylogin.php

Add the following to the URL query string:

☑ IP Address (ip)

☑ MAC Address (mac)

☑ VLAN name (vlan)

☑ Requested URL (requested_url)

(Add the pre-login URL to the Walled Garden. The page must continue to http

☑ Attempt to reconnect users when they return *(Not required to rel*

Figure 3-7 Location details

To configure for Full Customization, enable the option '**Send user to a pre-login URL before the welcome page**'

1. **URL** – This is the URL of the Custom Login page to send the user to.

2. **ip, mac, vlan, requested_url** – Check on the option **MAC address** (mac) to pass the MAC address to the Custom Login page.

3. **Attempt to reconnect users ...** - When this option is checked the InnGate will automatically do re-login check before redirecting the user to the custom login page.

   ⚠ Uncheck this option if you want to fully customize the login process including the relogin portion

You can now fully customize the whole login process based on your preference.

⚠ If your Custom Login Pages reside on an external server, you will need to create a **Walled Garden** entry, specifying the IP address of the external authentication server. This will allow the client to communicate with the authentication server prior to login.

3.4 Uploading Customization Files

To upload customized portal pages, a graphical FTP client is recommended.

1. Enable Remote Access

Make sure FTP is enabled on the gateway (Network > Services > Remote Access) and that the WAN connection is connected to the Internet. FTP security requires a reverse lookup of your IP before allowing an FTP connection.

2. Login via FTP Client

Login to the gateway with your FTP client. Default userid and password are *ftponly* and *antlabs*. You will be in the /www/pub/ location as the FTP home directory. All further instructions regarding directory assumes this base directory.

3. Create Location Directory

Create your own subdirectory under the /login directory.
E.g. for a guest room, you would create a subdirectory called “guest_room”. This would allow you to access this customized portal page at http://ezxcess.antlabs.com/www/pub/login/guest_room/.

4. Upload All Files

Finally upload all your custom pages into that directory.

3.5 API Customization Logs

To troubleshoot API issues and PHP script problems, you can download the API logs from the ftp directory /log/php/php.log.

Below are samples of PHP errors or warnings:

```
[25-Jun-2009 15:27:39] PHP Warning:  Missing argument 5 for
formDateTimeString() in /home/httpd/modules/standard/auth.local-2.0/page-
local.php on line 1641
```

Samples of API errors:

```
Thu, 25 Jun 2009 14:59:00 +0800 [auth_login 2.0] Cookie not found [166]
(INPUT:
mode=relogin|client_mac=00:22:41:86:DE:AD|client_ip=10.10.1.244|location_ind
ex=6|ppli=eth0.210|api_interface=php) (OUTPUT: none)
```

```
Thu, 25 Jun 2009 15:28:54 +0800 [auth_login 2.0] Invalid argument: password
[160] (INPUT:
userid=test1|password=***|type=local|client_mac=00:13:E8:A3:D0:1D|client_ip=
10.10.1.249|location_index=6|ppli=eth0.210|api_interface=php) (OUTPUT: none)
```

Chapter 4

EXTERNAL AUTHENTICATION INTEGRATION

4.1 Overview

Common authentication methods such as Access Codes, User ID and password, PMS, Credit card, etc, are natively supported by the gateway. However, there may be instances when there is a need to integrate the gateway with an authentication method that is not natively supported.

The ability to invoke the InnGate API via HTTP is a feature commonly used to support external integration with existing authentication servers.

This chapter illustrates how external integration can be achieved.

4.2 Authentication Protocol Sequence

Figure 4-1 shows a typical protocol sequence between a client and an authentication server in a web-based authentication system without the gateway in place.

⚠ This protocol sequence may vary depending on the systems used, but the general principles apply.

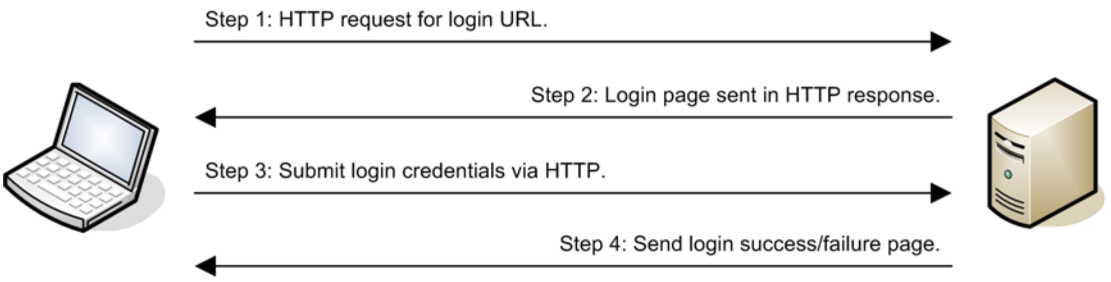

Step 1: HTTP request for login URL.
Step 2: Login page sent in HTTP response.
Step 3: Submit login credentials via HTTP.
Step 4: Send login success/failure page.

Figure 4-1 UAM Authentication

When the gateway is introduced, the authentication server must be customized to make use of the Session ID (SID) that the gateway generates to track the client's session.

This SID is sent to the authentication server, who must then embed it into the login page to be sent to the client.

When the client submits the login form, the SID is passed back to the authentication server along with the submitted credentials, who then uses it to invoke an HTTP API call to the **auth_login** module on the gateway.

Figure 4-2 shows the protocol sequence with the integration of the gateway.

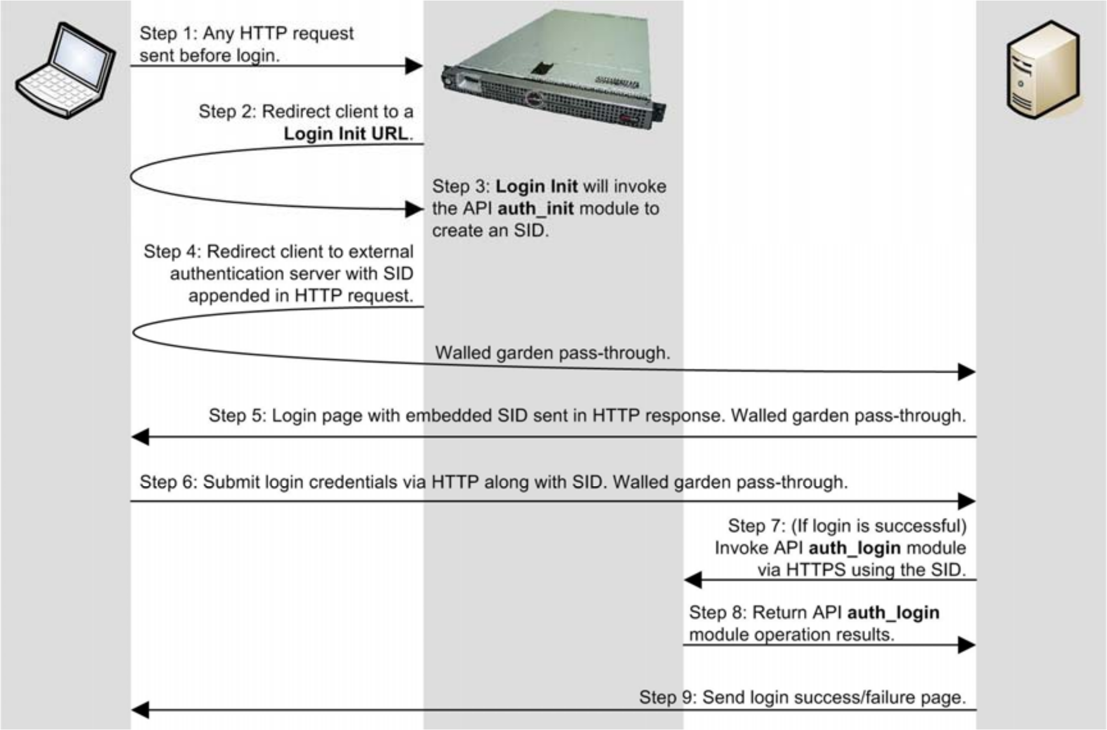

Step 1: Any HTTP request sent before login.
Step 2: Redirect client to a **Login Init URL**.
Step 3: **Login Init** will invoke the API **auth_init** module to create an SID.
Step 4: Redirect client to external authentication server with SID appended in HTTP request.
Walled garden pass-through.
Step 5: Login page with embedded SID sent in HTTP response. Walled garden pass-through.
Step 6: Submit login credentials via HTTP along with SID. Walled garden pass-through.
Step 7: (If login is successful) Invoke API **auth_login** module via HTTPS using the SID.
Step 8: Return API **auth_login** module operation results.
Step 9: Send login success/failure page.

Figure 4-2 Integrated UAM Authentication

The following sections describe the steps required to achieve the integration with the external UAM-based Authentication Server:

1. **Gateway Configuration** – Refer to Section 4.2.1.

2. **Systems Integration** – Refer to Section 4.2.2.

4.2.1 Gateway Configuration

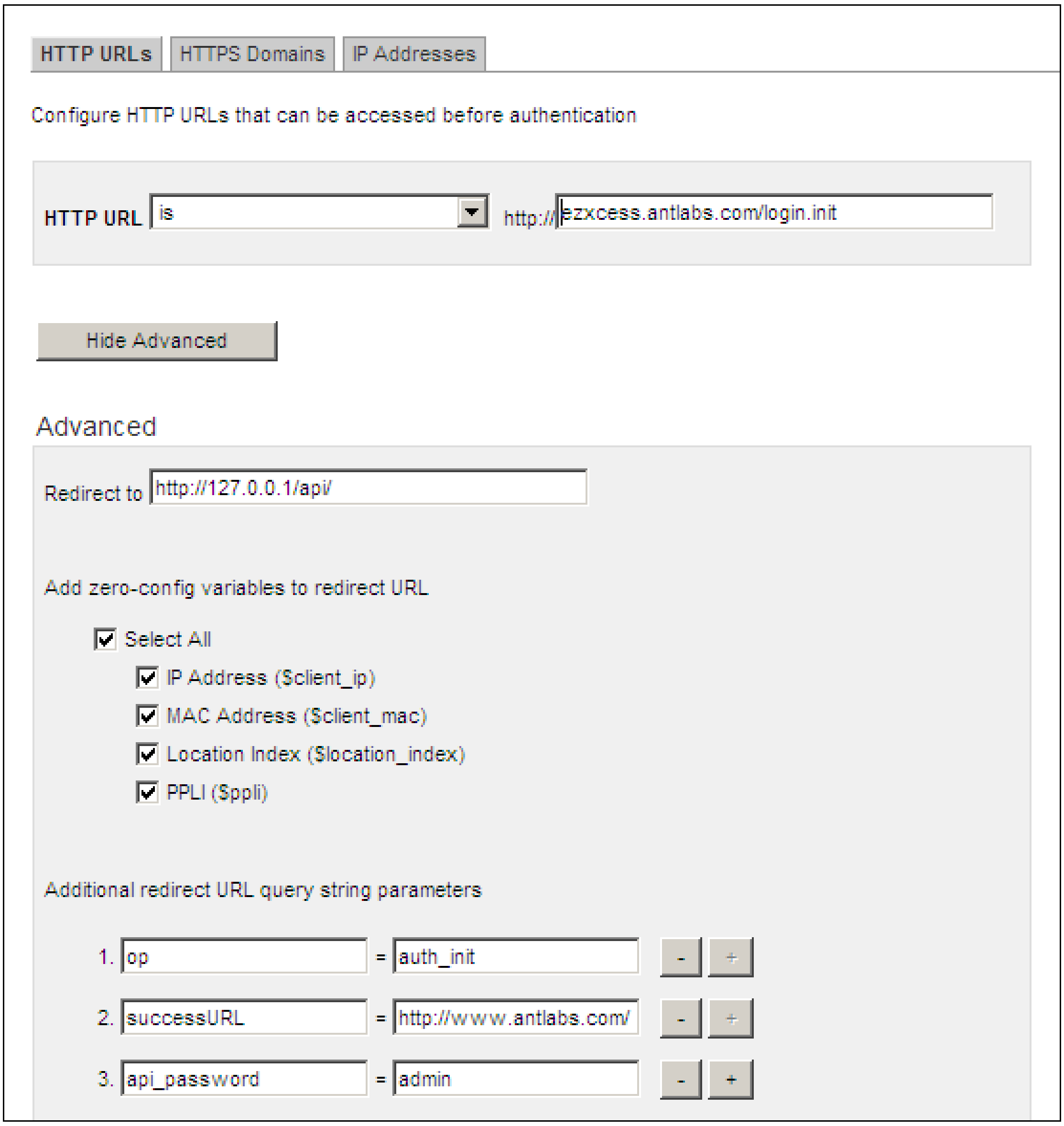
These are the steps to setup the gateway for integration with an external UAM-based authentication server. You should refer to the Administrator's Manual if you are unsure of how to perform any of these steps via the Admin GUI.

1. Refer to Section 3.3.2, configure a Custom Login Page, sending user to the following URL:

```
http://ezxcess.antlabs.com/login.init
```

⚠ You can use your own naming convention in place of “login.init” but make sure that it tallies with the next step.

2. Create a **HTTP URL Walled Garden Rule** that will make an API call when the **Login Init URL** in the previous step is encountered. See Figure 4-3.



Configure HTTP URLs that can be accessed before authentication

HTTP URL: is ▾ http:// ezxcess.antlabs.com/login.init

Advanced

Redirect to: http://127.0.0.1/api/

Add zero-config variables to redirect URL

☑ Select All
- ☑ IP Address ($client_ip)
- ☑ MAC Address ($client_mac)
- ☑ Location Index ($location_index)
- ☑ PPLI ($ppli)

Additional redirect URL query string parameters

1. op = auth_init
2. successURL = http://www.antlabs.com/
3. api_password = admin

Figure 4-3 Invoking the API with a URL Rule

The settings for the various fields are as follows:

i. **URL** is **http://ezxcess.antlabs.com/login.init**

ii. Click on Advanced > button, and enter the API URL into the **Redirect to** textbox:

```
http://127.0.0.1/api/
```

iii. **Add zero-config variables...** – Select the **Select All** checkbox.

iv. **Additional URL query string parameters...** – Create the following parameters:

```
op = auth_init
successURL = http://auth.server/login/
api_password = [api password]
```

⚠ The **successURL** is the URL of the login page on the external authentication server. The one shown here is just an example and you should replace it accordingly.

3. Create a **Walled Garden** entry, specifying the IP address of the external authentication server. This will allow the client to communicate with the authentication server prior to login.

4.2.2 Systems Integration

Apart from configuring the gateway, the external authentication server must be configured to perform the following tasks:

1. The Authentication Server must read the SID sent in the URL query string when the client sends the HTTP request for the login page. See Step 4 in Figure 4-2.

2. The Authentication Server must embed the SID within the login page as a hidden HTML Form element. This is so that the client will send the SID along with the login credentials when it submits the form. See Step 5 in Figure 4-2.

3. The Authentication Server must use the SID submitted along with the login credentials and use it to tell the gateway that the login was successful by invoking the **auth_login** API module via HTTP. See Step 6 – 8 in Figure 4-2.

   ⚠ The Authentication Server should also parse the result of the API module execution as illustrated in Section 2.2.2.

Appendix A

INNGATE API REFERENCE SUMMARY

Account

- **account_add** – Create new local account(s)
- **account_delete** – Delete local account
- **account_get** – Retrieve details of local account(s)
- **account_get_all –** Retrieve the account details based on some filters
- **account_update** – Update local account

API

- **api_module -** Display installed API modules version
- **api_modules** - Display installed API modules
- **api_password_get** - Retrieve the API password
- **api_version -** Display Installed API version

Authentication

- **auth_authenticate –** Perform verification of local and RADIUS account
- **auth_init** – Initialize a Session ID
- **auth_login** – Perform Login for client
- **auth_logout** – Perform Logout for client
- **auth_update** – Perform Session update for client
- **sid_get** – Retrieve Session ID data
- **publicip_get** – Get a public IP address for a client device

Plan

- **plan_get_all -** Retrieve all plan configured on InnGate
- **plan_get_id –** Retrieve the plan's ID

Data

- **data_get**
- **data_set**
- **data_get_keys**
- **data_get_names**
- **data_delete**

Property Management System (PMS)

- **pms_billing_log -** Retrieve data from the PMS billing log
- **pms_guest_status -** Retrieve guest status information
- **pms_post_check -** Send a posting to the PMS system
- **pms_post** - Check PMS posting details
- **pms_room_status -** Retrieve guest room status information

Network

- **vlan_get** – Returns information on a VLAN
- **vlan_update** – Update VLAN information
- **device_status** – Retrieve the network status of a client device

Credit Card

- **cc_payflowpro_post** – Payflow Pro credit card payment

Miscellaneous

- **browser -** Detect the type of browser used

account_add

Creates a new user account

Required Input

creator	20 chars, e.g. ‘admin’, ‘pms’, ‘printer’ For radius authentication value **must be** ‘radius’.
plan_id or plan_name	valid plan_id value or plan_name

Optional Input

type	‘userid’ or ‘code’ If not set, defaults to userid.
userid	90 chars, min 3, valid chars: A-Z a-z 0-9 - _ @
If userid is blank (all optional)	
userid_format	'alpha', 'alnum', 'num' . Default value is alpha
userid_length	Default value 5, minimum value is 3
userid_prefix	Default blank, max 20
userid_suffix	Default blank, max 20
userid_start	starting number, or 'auto'* to continue from last highest used number. Default value is auto.
password	unlimited chars, default blank
If password is blank (all optional)	
password_length	Default value 5, minimum value 3
password_format	'alpha', 'alnum', 'num'. Default value is alnum.
code	10 chars, min 3, valid chars: a-z 0-9

if code is blank (all optional)	
either one:	
code_format	'alpha', 'alnum', 'num'. Default value is alnum
code_start	starting number, or 'auto'* to continue from last highest used number
if code_format is set (optional)	
code_length	Default value is 5, minimum value is 3
code_prefix	Default value is blank, minimum value is 4
code_suffix	Default value is blank, minimum value is 4
count	number of accounts to create, default 1, max 100
description	255 chars
valid_from	'now' or Unix time.
valid_until	Unix time or blank • If not set, defaults to blank
login_max	Value >= 1, default unlimited
sharing_max	Value >= 1, default 1
billing_id	100 char, default blank
allowed_login_zone	Smallint, default value is 0

Output

op	The name of this module: **account_add**
version	The version of this module
result	The result of the execution: ok if successful, or error when the module failed (pipe separated list of result for each account: 'ok' or error message)
resultcode	The result code matching the **result**: 0 if **result** is ok or one of the result codes in the "Result Codes" section below (pipe separated list of result for each account: 'ok' or error message)
error	If **result** is error, contains a description of the error
created	Number of accounts created successfully
userids	Pipe separated list of created userids

passwords	Pipe separated list of created passwords
codes	Pipe separated list of created codes

Result Codes

0	Execution successful
1	More input arguments required
2	Incorrect **api_password**. For HTTP API calls only.
3	Incorrect **op**. For HTTP API calls only.
90	An invalid value was provided for an input argument
98	Database error

For Radius

When creating a local account that requires Radius Accounting during auth_login, the following attributes must be set accordingly:

1. creator must be set as ‘radius’ .
2. billing_id must be the same as the radius user id used for authentication. It will be used for radius accounting.

account_delete

Remove user accounts

Required Input

userid or code	- single userid, or pipe-separated list of userids - single code, or pipe-separated list of codes

Output

op	The name of this module: `account_delete`
version	The version of this module
result	The result of the execution: `ok` if successful, or `error` when the module failed
resultcode	The result code matching the **result**: `0` if **result** is `ok` or one of the result codes in the "Result Codes" section below
error	If **result** is `error`, contains a description of the error
deleted	Number of account deleted successfully

Result Codes

0	Execution successful
1	More input arguments required
2	Incorrect **api_password**. For HTTP API calls only.
3	Incorrect **op**. For HTTP API calls only.
98	Database error

account_get

Retrieve the details of specific account.

Required Input

userid or code or client_mac	Valid **userid** or **code** or **client_mac**

Output

op	The name of this module: `account_get`
version	The version of this module
result	The result of the execution: `ok` if successful, or `error` when the module failed
resultcode	The result code matching the **result**: `0` if **result** is `ok` or one of the result codes in the "Result Codes" section below
error	If **result** is `error`, contains a description of the error
userid	Userid of the account, pipe separated if sharing max value more than 1
code	Code of the account, pipe separated if sharing max value more than 1
sharing_index	Sharing index value for each pipe separated, pipe separated if sharing max value more than 1
client_mac	Client_mac for the account, pipe separated if sharing max value more than 1
description	Account description, pipe separated if sharing max value more than 1
enabled	Account status , pipe separated if sharing max value more than 1
valid_from	Account valid_from value, pipe separated if sharing max value more than 1
valid_until	Account expired date and time, pipe separated if sharing max value more than 1
login_limit	Login limit value, pipe separated if sharing max value more than 1
login_max	Maximum login allowed, pipe separated if sharing max value more than 1
login_count	each time being used will increase the value, pipe separated if sharing max value more than 1
sharing_max	Maximum sharing allowed for the account, pipe separated if

	sharing max value more than 1
plan	Plan name, pipe separated if sharing max value more than 1
duration_balance	Duration, pipe separated if sharing max value more than 1
volume_balance	Volume value in bits that still can be use (if the account plan is volume based) , pipe separated if sharing max value more than 1
create_time	Time and date account being created, pipe separated if sharing max value more than 1
update_time	Time and date account being updated, pipe separated if sharing max value more than 1

Result Codes

0	Execution successful
1	More input arguments required
2	Incorrect **api_password**. For HTTP API calls only.
3	Incorrect **op**. For HTTP API calls only.
90	An invalid value was provided for an input argument
98	Database error

account_get_all

Retrieve the details of all accounts.

Filter

creator	e.g. ‘admin’, ‘pms’, ‘printer’, ‘radius’, ‘cc’, ‘complimentary’
description	
type	‘userid’ or ‘code’
valid_from_start	Start timestamp of validfrom filter
valid_from_end	End timestamp of validfrom filter
valid_until_start	Start timestamp of validuntil filter
valid_until_end	End timestamp of validuntil filter
created_start	Start time of createdtime filter Format: - ‘Y-m-j H:i:s’ e.g. ‘2010-03-24 17:14:35’ - ‘Y-m-j’ will be treated as ‘Y-m-j 00:00:00’ e.g. ‘2010-03-15’ - ‘j M Y’ will be treated as ‘j M Y 00:00:00’ e.g. ‘7 Jun 2010’
created_end	End time of createdtime filter Format: - ‘Y-m-j H:i:s’ e.g. ‘2010-03-24 17:14:35’ - ‘Y-m-j’ will be treated as ‘Y-m-j 00:00:00’ e.g. ‘2010-03-15’ - ‘j M Y’ will be treated as ‘j M Y 00:00:00’ e.g. ‘7 Jun 2010’
plan_name	Usergroupname filter

Output

Op	The name of this module: `account_get_all`
version	The version of this module
result	The result of the execution: `ok` if successful, or `error` when the module failed
resultcode	The result code matching the **result**: `0` if **result** is `ok` or one of the result codes in the "Result Codes" section below
error	If **result** is `error`, contains a description of the error
count	Number of records found
header	1. Type 2. Creator

	3. Userid 4. Code 5. Description 6. Enable 7. Validfrom 8. Validuntil 9. Loginlimit 10. Loginmax 11. Logincount 12. Sharingmax 13. Usergroupname 14. Createtime 15. Updatetime 16. Accounting 17. billingID

Result Codes

0	Execution successful
1	More input arguments required
2	Incorrect **api_password**. For HTTP API calls only.
3	Incorrect **op**. For HTTP API calls only.
90	An invalid value was provided for an input argument
98	Database error

account_update

Change user account information

Required Input

userid or code	The field to be used to match the entry to be updated. Valid fields: `userid` or `code`

Optional Input

password	Password. Must be set if you are adding a built-in account. Set to blank to auto generate password.
password_length	Default value is 5, minimum is 3
password_format	‘alpha’ (alphabet), ‘alnum’ (alphanumeric), ‘num’ (numeric). Default is alnum
description	255 chars
valid_until	Unix time or blank
valid_from	Required is valid_until is set to a unix time Start date/time of the user account. In Unix time format. • Set to ‘`now`’ to use the current time • Set to blank to remove the start time
login_limit	‘on’ or ‘off’
login_max	Value >= 1
sharing_max	value >= 2 and bigger than previous value.
plan_id or plan_name	Only if the account never login.
allowed_login_zone	Smallint, default value is 0

Output

Op	The name of this module: `account_update`
version	The version of this module
result	The result of the execution: `ok` if successful, or `error` when the module failed
resultcode	The result code matching the **result**: `0` if **result** is `ok` or one of the result

	codes in the "Result Codes" section below
error	If **result** is `error`, contains a description of the error
password	Generated password

Result Codes

0	Execution successful
1	More input arguments required
2	Incorrect **api_password**. For HTTP API calls only.
3	Incorrect **op**. For HTTP API calls only.
90	An invalid value was provided for an input argument
98	Database error

api_module

Return the version of the specified API module

Required Input

module	Name of the module (op)

Output

op	The name of this module: `api_module`
version	The version of this module
result	The result of the execution: `ok` if successful, or `error` when the module failed
resultcode	The result code matching the **result**: `0` if **result** is `ok` or one of the result codes in the "Result Codes" section below
error	If **result** is `error`, contains a description of the error
version	Version of the specified **module**

Result Codes

0	Execution successful
1	More input arguments required
2	Incorrect **api_password**. For HTTP API calls only.
3	Incorrect **op**. For HTTP API calls only.
90	Invalid **module**
98	Module has no version number

api_modules

Returns a list of installed API modules

Output

op	The name of this module: `api_modules`
version	The version of this module
result	The result of the execution: `ok` if successful, or `error` when the module failed
resultcode	The result code matching the **result**: `0` if **result** is `ok` or one of the result codes in the "Result Codes" section below
error	If **result** is `error`, contains a description of the error
modules	List of installed API modules. Modules are separated by a pipe `|` character. The module name and module version is separated by a space character. `<module 1 name> <module 1 version>|<module 2 name> <module 2 version>|<module 3 name> <module 3 version>|...`
count	Total number of installed API modules

Result Codes

0	Execution successful
1	More input arguments required
2	Incorrect **api_password**. For HTTP API calls only.
3	Incorrect **op**. For HTTP API calls only.
98	API modules could not be found

api_password_get

- Returns the configured API password for the given API interface type
- This module only works when executed from the PHP API interface using the API class

Required Input

type	The API interface type. Set to `http` to get the API password for HTTP API calls.

Output

op	The name of this module: `api_password_get`
version	The version of this module
result	The result of the execution: `ok` if successful, or `error` when the module failed
resultcode	The result code matching the **result**: `0` if **result** is `ok` or one of the result codes in the "Result Codes" section below
error	If **result** is `error`, contains a description of the error
api_password	The API password
type	The interface type for the API password. Matches the **type** input argument.

Result Codes

0	Execution successful
1	More input arguments required
2	Incorrect **api_password**. For HTTP API calls only.
3	Incorrect **op**. For HTTP API calls only.
3	The module must be executed from a PHP API interface
90	The API password cannot be retrieved successfully

api_version

Returns the version of the API installed in the gateway

Output

op	The name of this module: `api_version`
api_version	The version of the API
result	The result of the execution: `ok` if successful, or `error` when the module failed
resultcode	The result code matching the **result**: `0` if **result** is `ok` or one of the result codes in the "Result Codes" section below
error	If **result** is `error`, contains a description of the error

Result Codes

0	Execution successful
1	More input arguments required
2	Incorrect **api_password**. For HTTP API calls only.
3	Incorrect **op**. For HTTP API calls only.
98	API version could not be determined

auth_authenticate

Perform verification of local and RADIUS accounts. This API does not perform the actual login.

Required Input

code or (userid and password)	Access code or (userid and password) depending on the authentication method

Optional Input

mode	local or radius (Default : local)

Output

op	The name of this module: `auth_authenticate`
version	The version of this module
result	The result of the execution: `ok` if successful, or `error` when the module failed
resultcode	The result code matching the **result**: `0` if **result** is `ok` or one of the result codes in the "Result Codes" section below
error	If **result** is `error`, contains a description of the error
radiusattrs	Will have radius attributes being return by radius server A pipe-delimited list of keys

Result Code

0	Execution successful
1	More input arguments required
2	Incorrect **api_password**. For HTTP API calls only.
3	Incorrect **op**. For HTTP API calls only.
90	Argument values incorrect
150	Authentication error
151	Authentication rejected. Gateway not in Radius client list or incorrect shared secret
153	Password must be provided
159	Invalid access code

160	Invalid userid and/or password
170	Radius module license not found

Radius attributes:

1. Session-Timeout - integer
   Radius session time out

ANTlabs Vendor specific attributes:

1. Antlabs-User-Group-Name (12902:1) – string
   Plan name of account to be created

2. Acct-Session-Octets (12902:21) – integer
   Radius account volume

3. Acct-Session-Gigawords (12902:22) – integer
   Radius account volume (giga)

Sample output:

```
op = auth_authenticate
version = 1.0
result = ok
resultcode = 0
radiusattrs = Antlabs-User-Group-Name=stored_volume|Antlabs-Acct-
Session-Octets=12345678|Framed-Protocol=1|Service-Type=2|Session-
Timeout=86400
```

auth_init

- Initializes and returns a unique session ID
- Session ID is associated with the provided input arguments

Required Input

client_mac **client_ip** **location_index** **ppli**	The 4 zero-config variables provided by a URL rewrite rule

Optional Input

new_sid	• When set to `1`, the module will not attempt to reuse and issue the same **sid** to the same device • A new **sid** will always be issued • This is useful when **auth_init** is used more than once during a login procedure, and with different or changing **[extra-fields]** o In such cases, the **[extra-fields]** configured may be inconsistent when the same sid is reused on subsequent initializations o This may cause logins to fail, since it could not reference the correct **[extra-fields]** set during a specific **auth_init** execution • However, note that if this feature is used, a DoS attack that executes **auth_init** many times consecutively might cause the system to get overloaded with data
[extra-fields]	You can set any other arguments of any name, and they will be stored and associated with the session ID. These arguments can be retrieved using the **sid_get** module. • Arguments beginning with `login-` will be used by the **auth_login** module as input • Arguments beginning with `logout-` will be used by the **auth_logout** module as input • Arguments beginning with `update-` will be used by the

	auth_update module as input For example, if login-userid is set to abc, the **auth_login**'s **userid** input argument will automatically be set to abc when the **auth_login** is executed with the session ID provided as input.

Output

op	The name of this module: auth_init
version	The version of this module
result	The result of the execution: ok if successful, or error when the module failed
resultcode	The result code matching the **result**: 0 if **result** is ok or one of the result codes in the "Result Codes" section below
error	If **result** is error, contains a description of the error
sid	A unique 32-character session ID
client_mac **client_ip** **ppli** **vlan**	Zero-config variables associated with the session ID

Result Codes

0	Execution successful
1	More input arguments required
2	Incorrect **api_password**. For HTTP API calls only.
3	Incorrect **op**. For HTTP API calls only.
102	Input arguments are invalid or insufficient
141	Failed to create a new session ID

auth_login

Login and create a new session for a device on the LAN

Required Input

sid or (**client_mac**, **client_ip**, **location_index** and **ppli**)	The session ID or 4 zero-config variables

Optional Input

mode	login or relogin. default: login for normal login **mode**=login for attempting relogin of the user via cookie **mode**=relogin
code or (userid and password)	Access code or (userid and password) depending on the login method
secret	Ensures that the provided secret code matches the **secret** input argument provided to **auth_init**

Output

op	The name of this module: `auth_login`
version	The version of this module
result	The result of the execution: `ok` if successful, or `error` when the module failed
resultcode	The result code matching the **result**: `0` if **result** is `ok` or one of the result codes in the "Result Codes" section below
error	If **result** is `error`, contains a description of the error
requestedURL	The URL that the device last tried to access. Will be blank if there is no HTTP request.
preloginURL	The URL that the device tried to access before logging in and hitting **auth_init** to get a sid. The URL is usually the browser's home page. Will be blank if the user did not access any URL before hitting the login page.
publicip	Result of the public IP allocation: `ok` when a public IP address is allocated successfully, or `error` when it fails. Only output when **publicip** is set to `1`, and when the **result** is `ok`.
sid	The session ID, if **sid** is set as an input argument
client_mac **client_ip** **ppli** **vlan**	Zero-config variables used for login

Result Codes

0	Execution successful
1	More input arguments required
2	Incorrect **api_password**. For HTTP API calls only.
3	Incorrect **op**. For HTTP API calls only.
110	The authentication **type** is not found, or is incorrectly
150	Authentication error
151	Authentication rejected
152	Sharing limit exceeded
153	Password must be provided
155	Cannot login when access control is not set to charged access
156	userid is blacklisted
157	Maximum number of users for this location exceeded
158	Secret does not match the secret provided to auth_init
159	Invalid access code
160	Invalid userid and/or password
161	Account disabled
162	Account not yet valid or expired
163	Not allowed to login at this point of this time
164	Maximum number of login reached
165	Cookie data not found
166	Cookie not found
167	Account cannot be used with this device
168	Account usage duration and/or volume has expired
169	Stored volume accounting not available

Usage Example

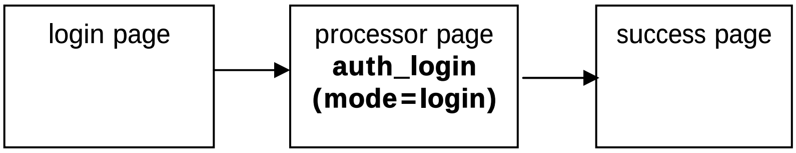
There are 2 ways of using auth_login API:

a. to perform normal authentication or
b. attempt relogin of a user with an automatic relogin plan

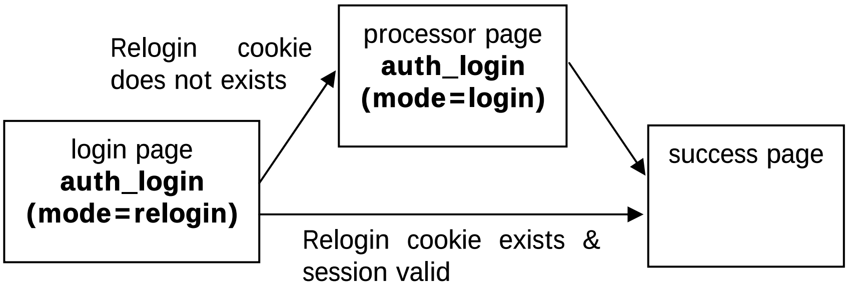
Below are the minimum parameters required for the 2 modes of usage:

1. Login authentication
   - sid or (client_mac, client_ip, location_index, ppli)
   - code or (userid & password)
   - mode=login
2. Relogin authentication
   - sid or (client_mac, client_ip, location_index, ppli)
   - mode=relogin

If your customization does not need to support automatic relogin, then the process flow is as follow:



If your customization requires automatic relogin support, then the login page will need to check for cookie. If the cookie exists and session is valid, the user will be automatically login and redirected to the success page. If cookie does not exist, relogin user based on MAC address and Zone. Else, the login page will be shown, and the normal login process will follow.



auth_logout

Logout a device on the LAN

Required Input

sid (or **client_mac**)	Either the session ID or client_mac zero-config variable must be provided

Output

op	The name of this module: `auth_logout`
version	The version of this module
result	The result of the execution: `ok` if successful, or `error` when the module failed
resultcode	The result code matching the **result**: `0` if **result** is `ok` or one of the result codes in the "Result Codes" section below
error	If **result** is `error`, contains a description of the error
accounting	Result of the accounting operation: `ok` when there is no accounting error, or `error` when the accounting server could not be contacted during logout. This argument is only output when the logout is successful (i.e. **result** is `ok`). When this is `error`, the logout is likely to be successful, but the accounting stop packet could not be sent, so the user will be in a `pending_close` status in the Session Monitor until the server is able to retry and send the accounting stop packet successfully.
sid	The session ID, if **sid** is set as an input argument
client_mac	MAC address of the device used for logout

Result Codes

0	Execution successful
1	More input arguments required
2	Incorrect **api_password**. For HTTP API calls only.
3	Incorrect **op**. For HTTP API calls only.
110	The authentication **type** set in **auth_login** is incorrectly configured
122	The device's MAC address is not found on the LAN network
190	Logout error

auth_update

Update a device's session

- Used to change the usage duration
  - Either **duration** or **volume** must be set
  - The device must have a MAC address

Required Input

client_mac	client_mac variable must be provided

Optional Input

duration	Change the number of **minutes** of network access. Overwrites the current value.
volume	Change the number of **bytes** to the current volume balance of the device

Output

op	The name of this module: `auth_update`
version	The version of this module
result	The result of the execution: `ok` if successful, or `error` when the module failed
resultcode	The result code matching the **result**: `0` if **result** is `ok` or one of the result codes in the "Result Codes" section below
error	If **result** is `error`, contains a description of the error

Result Codes

0	Execution successful
1	More input arguments required
2	Incorrect **api_password**. For HTTP API calls only.
3	Incorrect **op**. For HTTP API calls only.
90	Argument values incorrect
98	Critical error

sid_get

Returns the variables associated with a session ID created by **auth_init**

Required Input

sid	The session ID

Output

op	The name of this module: `sid_get`
version	The version of this module
result	The result of the execution: `ok` if successful, or `error` when the module failed
resultcode	The result code matching the **result**: `0` if **result** is `ok` or one of the result codes in the "Result Codes" section below
error	If **result** is `error`, contains a description of the error
sid	The retrieved session ID
client_mac **ppli** **vlan** **client_ip** **location_index**	Zero-config variables associated with the session ID
[extra-fields]	Any other extra fields set during **auth_init** will also be returned

Result Codes

0	Execution successful
1	More input arguments required
2	Incorrect **api_password**. For HTTP API calls only.
3	Incorrect **op**. For HTTP API calls only.
105	Invalid **sid**

publicip_get

Get a public IP address for a logged in user

Required Input

sid (or **client_mac** & **ppli**)	The session ID or 2 zero-config variables must be provided

Output

op	The name of this module: `publicip_get`
version	The version of this module
result	The result of the execution: `ok` if successful, or `error` when the module failed
resultcode	The result code matching the **result**: `0` if **result** is `ok` or one of the result codes in the "Result Codes" section below
error	If **result** is `error`, contains a description of the error

Result Codes

0	Execution successful
1	More input arguments required
2	Incorrect **api_password**. For HTTP API calls only.
3	Incorrect **op**. For HTTP API calls only.
4	The **sid** could not be found
98	Failed to get a public IP address

plan_get_all

Retrieve all plan configured in the InnGate 3.

Required Input

There is no input needed for calling plan_get_all.

Output

op	The name of this module: `plan_get_all`
version	The version of this module
result	The result of the execution: `ok` if successful, or `error` when the module failed
resultcode	The result code matching the **result**: `0` if **result** is `ok` or one of the result codes in the "Result Codes" section below
error	If **result** is `error`, contains a description of the error
[record_X] **X is value for each record**	Each plan will be returned as individual output argument. Within each output argument, the pipe | character is used to separate the value for each field, if there are more than one values within the field. Do not assume that this is a single value without pipe. example below : record_1 = 4|0.00|unlimited|off|0|off|0|logout|on|256|kbps|on|128|kbps|off|off|off|Throttled Below are field being return by API plan_get_all in sequence : 1. plan ID 2. Price 3. Authentication type (unlimited, fixed_duration, stored_duration, stored_volume) 4. duration limit (on, off) 5. valid duration in minutes 6. volume limit status (on, off) 7. valid volume limit in megabytes 8. Action if stored volume plan is expired (change, logout) 9. download limit (on, off) 10. download bandwidth 11. download bandwidth unit (bps, kbps, mbps) 12. upload limit (on, off) 13. upload bandwidth 14. bandwidth unit (bps, kbps, mbps) 15. Public IP status (on, ask, off) 16. When user comes back attempt user to relogin (on, off) 17. fair_use value (on, off) 18. Plan name

Result Codes

0	Execution successful
1	More input arguments required
2	Incorrect **api_password**. For HTTP API calls only.
3	Incorrect **op**. For HTTP API calls only.
90	An invalid value was provided for an input argument
98	Could not read plan data.

plan_get_id

Retrieve the plan’s ID.

Required Input

plan_name	The name of the plan

Output

op	The name of this module: `plan_get_id`
version	The version of this module
result	The result of the execution: `ok` if successful, or `error` when the module failed
resultcode	The result code matching the **result**: `0` if **result** is `ok` or one of the result codes in the "Result Codes" section below
error	If **result** is `error`, contains a description of the error

Result Codes

0	Execution successful
1	More input arguments required
2	Incorrect **api_password**. For HTTP API calls only.
3	Incorrect **op**. For HTTP API calls only.
90	An invalid value was provided for an input argument
98	Could not read plan data.
401	Plan not found

data_get

Retrieves a single entry of data matching the specified criteria

Required Input

name	A unique string identifying the set of data being stored
key	Unique key of the entry

Optional Input

timestamp	If set, the **timestamp** output argument will be formatted using PHP's date() function

Output

op	The name of this module: `data_get`
version	The version of this module
result	The result of the execution: `ok` if successful, or `error` when the module failed
resultcode	The result code matching the **result**: `0` if **result** is `ok` or one of the result codes in the "Result Codes" section below
error	If **result** is `error`, contains a description of the error
name	A unique string identifying the set of data being stored
key	Unique key of the entry
timestamp	Time that the data was created or updated with **data_set**. In Unix time format if the **timestamp** input argument is not set.
[extra-fields]	Other extra fields stored with **data_set**

Result Codes

0	Execution successful
1	More input arguments required
2	Incorrect **api_password**. For HTTP API calls only.
3	Incorrect **op**. For HTTP API calls only.
90	Criteria from **name** and **key** doesn't match any data
98	Data could not be retrieved

data_set

Stores a unique entry of data consisting of one or more data arguments

- If the **name** and **key** input arguments match an existing entry, that entry will be updated
- Each stored entry is identified uniquely by a combination of **name** and **key**
- Setting the **name** argument differently allows you to store different sets of data by giving it different names (e.g. cookies, userids)
- **key** can be similar across multiple data sets, if necessary (e.g. the **name**s userids_allowed and userids_free_access can use a **key** which is the user ID)

Required Input

name	A unique string identifying the set of data being stored. 32 characters maximum.
key	Unique key of the entry. 64 characters maximum.
[extra-fields]	One or more fields to be stored as part of the entry's data. These argument names cannot be used: name, key, timestamp, op, api_interface, version, result, resultcode

Output

op	The name of this module: data_set
version	The version of this module
result	The result of the execution: ok if successful, or error when the module failed
resultcode	The result code matching the **result**: 0 if **result** is ok or one of the result codes in the "Result Codes" section below
error	If **result** is error, contains a description of the error

Result Codes

0	Execution successful
1	More input arguments required
2	Incorrect **api_password**. For HTTP API calls only.
3	Incorrect **op**. For HTTP API calls only.
98	Data could not be set

data_get_keys

Retrieves a list of keys matching the specified criteria

- If no optional input arguments are specified, all available keys will be output

Optional Input

name	Get entries belonging to this name
after_timestamp	Get entries created/set on or after this time. In Unix time format.
before_timestamp	Get entries created/set on or before this time. In Unix time format. Set to `now` to use the current time.

Output

op	The name of this module: `data_get_keys`
version	The version of this module
result	The result of the execution: `ok` if successful, or `error` when the module failed
resultcode	The result code matching the **result**: `0` if **result** is `ok` or one of the result codes in the "Result Codes" section below
error	If **result** is `error`, contains a description of the error
keys	A pipe-delimited list of keys `<key1>|<key2>|<key3>|...`
count	Number of keys output

Result Codes

0	Execution successful
1	More input arguments required
2	Incorrect **api_password**. For HTTP API calls only.
3	Incorrect **op**. For HTTP API calls only.
98	Data could not be retrieved

data_get_names

Retrieves a list of names matching the specified criteria

- If no optional input arguments are specified, all available names will be output

Optional Input

key	Get entries with this key
after_timestamp	Get entries created/set on or after this time. In Unix time format.
before_timestamp	Get entries created/set on or before this time. In Unix time format. Set to `now` to use the current time.

Output

op	The name of this module: `data_get_names`
version	The version of this module
result	The result of the execution: `ok` if successful, or `error` when the module failed
resultcode	The result code matching the **result**: `0` if **result** is `ok` or one of the result codes in the "Result Codes" section below
error	If **result** is `error`, contains a description of the error
names	A pipe-delimited list of names `<name1>|<name2>|<name3>|...`
count	Number of names output

Result Codes

0	Execution successful
1	More input arguments required
2	Incorrect **api_password**. For HTTP API calls only.
3	Incorrect **op**. For HTTP API calls only.
98	Data could not be retrieved

data_delete

Removes data matching the specified criteria

- At least one of the optional input arguments must be set
- If both **after_timestamp** and **before_timestamp** are not set, entries that match the other criteria will be removed, irregardless of the time the entry is created/set
- If **name** is set and **key** is not set, all entries (of any **key**) matching **name** will be removed
- If **key** is set and **name** is not set, all entries (of any **name**) matching **key** will be removed

Optional Input

Name	A unique string identifying the set of data being stored
Key	Unique key of the entry
after_timestamp	Delete entries created/set on or after this time. In Unix time format.
Before_timestamp	Delete entries created/set on or before this time. In Unix time format. Set to `now` to use the current time.

Output

op	The name of this module: `data_delete`
version	The version of this module
result	The result of the execution: `ok` if successful, or `error` when the module failed
resultcode	The result code matching the **result**: `0` if **result** is `ok` or one of the result codes in the "Result Codes" section below
error	If **result** is `error`, contains a description of the error
count	Number of entries deleted

Result Codes

0	Execution successful

1	More input arguments required
2	Incorrect **api_password**. For HTTP API calls only.
3	Incorrect **op**. For HTTP API calls only.
98	Data could not be deleted

pms_billing_log

Retrieve entries from the PMS billing log

Required Input

type	The field to be used to match the pms type (fcs , mf, hobic)

Optional Input

start_time	Starting date and time of the log entries to be retrieved. In GNU standard date and time format.
end_time	Ending date and time of the log entries to be retrieved. In GNU standard date and time format.
room_no	When specified, limits the entries to the specified room number. When left empty, all room numbers are considered.
sort	Field to be used for sorting the retrieved log entries : date(default) or room_no
order	Sort order of the sort field chosen : asc (ascending) or desc (descending)
count	When specified, limits the number of log entries retrieved. Use in conjunction with the **page** argument. When left empty, all matching entries will be retrieved.
page	The logical “page” number to be retrieved, taking into account the number of entries retrieved limited by the **count** argument. Use in conjunction with the **count** argument. Defaults to page 1.

Output

op	The name of this module: `pms_billing_log`
version	The version of this module
result	The result of the execution: `ok` if successful, or `error` when the module failed
resultcode	The result code matching the **result**: `0` if **result** is `ok` or one of the result codes in the "Result Codes" section below
error	If **result** is `error`, contains a description of the error
count	The total of how many record being shown
record_x	Each record will contain pms billing log details for each transaction example :

	record_1 = 9|2009-06-25 16:34:57|0|VLAN 210||21600|2009-06-25 16:34:57|2009-06-25 16:34:57|1000|S||Fixed Duration 6 hours Below are field being return by API pms_billing_log in sequence : 1. Billing ID 2. date of billing transaction 3. guest number 4. room number 5. original room number (if guest ever change room) 6. usage time 7. start time 8. charge start time 9. amount 10. status 11. hardware address (MAC address) 12. description

Result Codes

0	Execution successful
1	More input arguments required
2	Incorrect **api_password**. For HTTP API calls only.
3	Incorrect **op**. For HTTP API calls only.
90	An invalid value was provided for an input argument
98	Could not read PMS data

pms_guest_status

Retrieve guest status information

Required Input

room_no or guest_name or guest_no	Retrieve entries either by guest name or room number. If both are specified, guest_name will be used. For Galaxy, it uses either combination of room_no and guest_name, room_no and guest_no or room_no and guest_name and guest_no.
type	The PMS system installed.Valid value : • fcs – FCS • mf – Micros Fidelio • galaxy - Galaxy

Output

op	The name of this module: pms_guest_status
version	The version of this module
result	The result of the execution: ok if successful, or error when the module failed
resultcode	The result code matching the **result**: 0 if **result** is ok or one of the result codes in the "Result Codes" section below
error	If **result** is error, contains a description of the error
count	The number of entries retrieved.
[field]	If count is 1 or more, fields will be returned as individual output argument. Within each output argument, the pipe | character is used to separate the value for each field, if there are more than one values within the field. Do not assume that this is a single value without a pipe |, because it is possible for a guest room to have two guest.
guest_status_id	
guest_no	guest number
guest_name	guest name
room_no	room number where the guest checks in
date	date of check in (timestamp)

status	
guest_vip_status	guest VIP status (Y or N)
guest_payment_type	guest payment type (NO POST or ALLOW POS)
guest_departure	date of check out

Result Codes

0	Execution successful
1	More input arguments required
2	Incorrect **api_password**. For HTTP API calls only.
3	Incorrect **op**. For HTTP API calls only.
90	An invalid value was provided for an input argument
98	Database error

pms_post_check

Check PMS posting and double_posting protection information.

Required Input

bill_mode	Billing mode for double-posting protection. Valid values : guest_no, guest_name, client_mac, ppli, or vlan.

Optional Input

client_mac or sid	Session ID or client_mac to identify the user. Required if bill_mode is client_mac.
guest_name	Required if bill_mode is guest_name.
guest_no	Required if bill_mode is guest_no.
ppli or vlan	Required if bill_mode is ppli or vlan

Output

op	The name of this module: `pms_post_check`
version	The version of this module
result	The result of the execution: `ok` if successful, or `error` when the module failed
resultcode	The result code matching the **result**: `0` if **result** is `ok` or one of the result codes in the "Result Codes" section below
error	If **result** is `error`, contains a description of the error
start_time	The time the posting was made
start_timestamp	The time the posting was made in unix time format
end_time	The time the billing will expire and cause the user to be charged again.
end_timestamp	The time the billing will expire in unix time format.
balance	Remaining time in seconds.

Result Codes

0	Execution successful
1	More input arguments required
2	Incorrect **api_password**. For HTTP API calls only.
3	Incorrect **op**. For HTTP API calls only.
90	An invalid value was provided for an input argument
98	Could not read PMS data.

pms_post

Sends a posting to the hotel PMS system to charge a specified amount to a room.

Commas in input arguments will be removed. To enable double-posting protection, bill_mode and duration (seconds) must be set.

Required Input

room_no	The room number to send the posting.
amount	The amount to charge to the specified room, usually in the currecy configured in the PMS system (e.g. cents). Accept whole number 0 or higher.
type	The PMS system installed.Valid value : • fcs – FCS • mf – Micros Fidelio • galaxy – Galaxy The default value is ‘mf’.

Optional Input

bill_mode	Billing mode for double-posting protection. Valid values: guest_no, guest_name, client_mac, ppli, or vlan. The specified value will be used to decide if the customer should be charged again. If this is not set, double-posting protection will be turned off and every call will result in a posting
client_mac or sid	Session ID or client_mac to identify the user. Required for double-posting protection.
desc	An arbitrary description field.
duration	The duration that the billing will be effective for, in seconds. Required for double-posting protection.
guest_name	For double-posting protection based on guest_name.
guest_no	For double-posting protection based on guest_no. Required for Galaxy PMS.
label	This must be set to T, if you are posting to FCS.
ppli or vlan	For double-posting protection based on VLANs.
time	in unix time format.
sales_outlet	Sales outlet identification number.

folioid	Required for Galaxy PMS.

Output

op	The name of this module: `pms_post`
version	The version of this module
result	The result of the execution: `ok` if successful, or `error` when the module failed
resultcode	The result code matching the **result**: `0` if **result** is `ok` or one of the result codes in the "Result Codes" section below
error	If **result** is `error`, contains a description of the error
post	Yes – Posting was sent successfully. No – No posting sent due to double-posting protection The following arguments will only be output when double-posting protection is turned on : **start_time** - The time the posting was made. **start_timestamp** – The time the posting was made in Unix time format. **end_time** – The time the billing will expire and cause the user to be charged again. **end_timestamp** – The time billing will expire in Unix time format. **balance –** Remaining time in seconds. guest_no guest_name client_mac ppli vlan Input arguments, as provided. For verification purposes.

Result Codes

0	Execution successful
1	More input arguments required

2	Incorrect **api_password**. For HTTP API calls only.
3	Incorrect **op**. For HTTP API calls only.
90	An invalid value was provided for an input argument
98	Posting Failed.

pms_room_status

Retrieves guest room status information.

Required Input

type	The PMS system installed. Valid values : • fcs – FCS. • mf – Micros Fidelio • hobic – Hobic • prologic – Prologic
room_no	Room number to retrieve.

Output

op	The name of this module: `pms_room_status`
version	The version of this module
result	The result of the execution: `ok` if successful, or `error` when the module failed
resultcode	The result code matching the **result**: `0` if **result** is `ok` or one of the result codes in the "Result Codes" section below
error	If **result** is `error`, contains a description of the error
room_no	room number being inputed.
date	unix timestamp inputed date
guest_no	guest number
num_guest	Number of guest staying in the room
[field]	Each field will be returned as individual output argument. Within each output argument, the pipe | character is used to separate the value for each field, if there are more than one values within the field. Do not assume that this is a single value without pipe.

Result Codes

0	Execution successful
1	More input arguments required

2	Incorrect **api_password**. For HTTP API calls only.
3	Incorrect **op**. For HTTP API calls only.
90	An invalid value was provided for an input argument
98	Could not read PMS data.

vlan_get

Returns information on a VLAN

- If **vlan** is not provided, information for the `No VLAN` entry is returned

Optional Input

vlan or **ppli**	VLAN ID or ppli zero-config variable of the VLAN to get

Output

op	The name of this module: `vlan_get`
version	The version of this module
result	The result of the execution: `ok` if successful, or `error` when the module failed
resultcode	The result code matching the **result**: `0` if **result** is `ok` or one of the result codes in the "Result Codes" section below
error	If **result** is `error`, contains a description of the error
name	VLAN name
description	VLAN description
vlangroup	VLAN Group name that the VLAN belongs to
maxlogins	Maximum number of logins allowed for this VLAN
accesscontrol	One or more Access Control names that the VLAN belongs to. This is not displayed if **vlangroup** is not displayed. This may be blank if the VLAN has no associated access controls. `<accesscontrol1>|<accesscontrol2>|<accesscontrol3>|...`

Result Codes

0	Execution successful
1	More input arguments required
2	Incorrect **api_password**. For HTTP API calls only.
3	Incorrect **op**. For HTTP API calls only.
90	An invalid value was provided for an input argument
98	Database error

341	Could not find the **vlan** / **ppli**

vlan_update

Update VLAN information

- If **vlan** is not provided, the `No VLAN` entry is updated
- The **name** of the `No VLAN` entry cannot be changed
- At least one of the following input arguments should be provided: **name**, **description**, **vlangroup** or **maxlogins**

Optional Input

vlan or **ppli**	VLAN ID or ppli zero-config variable of the VLAN to be updated
name	VLAN name. Cannot be blank.
description	VLAN description. Set to blank to remove the description.
vlangroup	VLAN Group name that the VLAN belongs to. Cannot be blank.
maxlogins	Maximum number of logins allowed for this VLAN • Set to blank to disable • Set to an integer (`0` or larger) to enable

Output

op	The name of this module: `vlan_update`
version	The version of this module
result	The result of the execution: `ok` if successful, or `error` when the module failed
resultcode	The result code matching the **result**: `0` if **result** is `ok` or one of the result codes in the "Result Codes" section below
error	If **result** is `error`, contains a description of the error

Result Codes

0	Execution successful
1	More input arguments required
2	Incorrect **api_password**. For HTTP API calls only.
3	Incorrect **op**. For HTTP API calls only.

90	An invalid value was provided for an input argument
98	Database error
341	Could not find the **vlan** / **ppli**

device_status

Retrieves the network status of a particular device connected to the gateway

- If **connected** is `no`, all other output arguments will not be available

Required Input

client_mac	MAC address of the network device

Output

op	The name of this module: `device_status`
version	The version of this module
result	The result of the execution: `ok` if successful, or `error` when the module failed
resultcode	The result code matching the **result**: `0` if **result** is `ok` or one of the result codes in the "Result Codes" section below
error	If **result** is `error`, contains a description of the error
connected	• `yes` Device is found on the network • `no` A device with this MAC address does not exist on the network
failed_probes	Number of times the gateway has failed to probe the network device. A value more than `1` indicates that the device did not respond to probes and is likely to have disconnected from the network.
internet_access	• `yes` Device can access the Internet/WAN network • `no` Device does not have network access because he has not logged in, or the device is not allowed to access the internet
logged_in	• `yes` Device has a Charged Access Network Policy, and has already logged in. • `no` Device has not logged in or does not have a policy which

	allows logins
client_ip	IP address of the device
ppli	ppli zero-config variable of the device
vlan	VLAN of the device. This will be blank if the device is not in a VLAN.
vlan_moved	• yes The device has moved from one VLAN to another • no
location_index	location_index zero-config variable
url	The HTTP URL that the device last tried to request

Result Codes

0	Execution successful
1	More input arguments required
2	Incorrect **api_password**. For HTTP API calls only.
3	Incorrect **op**. For HTTP API calls only.
90	An invalid value was provided for an input argument

cc_payflowpro_post

Retrieves guest room status information.

Required Input

paymentserver_host	Payflow Pro payment server host name
vendor_id	Registered vendor ID
vendor_password	Password for the registered Vendor ID
partner	Partner name whom the vendor ID is registered with
cc_number	Credit Card Number
cc_expiry	Credit Card Expiry date (MMYY format)
amount	Amount to be charged

Optional Input

paymentserver_port	Payflow Pro payment server port number (default: 443)
user_id	Registered user ID (Default: Will use the value of vendor)
timeout	Max amount of seconds to wait for reply from Payflow Pro payment server (Default: 30)
invoice	Invoice number
cc_csc	Card Security Code
cc_name	Card holder's name
cc_street	Card holder's street address
cc_postalcode	Card holder's ZIP/postal code

Output

op	The name of this module: `cc_payflowpro_post`
version	The version of this module
result	The result of the execution: `ok` if successful, or `error` when the module failed
resultcode	The result code matching the **result**: `0` if **result** is `ok` or one of the result codes in the "Result Codes" section below
error	If **result** is `error`, contains a description of the error
RESULT	The outcome of the attempted trasaction. A result of 0 (zero) indicates the transaction was approved. Any other number indicates

	a decline or error
PNREF	Payflow Transaction ID, a unique number that identifies the transaction
CVV2MATCH	Result of the card security code (CVV2) check. The issuing bank may decline the transaction if there is a mismatch. In other cases, the transaction may be approved despite a mismatch
RESPMSG	The response message returned with the transaction result. Exact wording varies. Sometimes a colon appears after the initial RESPMSG followed by more detailed information
AUTHCODE	approval code obtained over the telephone from the processing network. AUTHCODE is required when submitting a Force (F) transaction
AVSADDR	Address Verification Service address response returned if you are using Address Verification Service. Address Verification Service address responses are for advice only. This process does not affect the outcome of the authorization
AVSZIP	Address Verification Service zip code response returned if you are using Address Verification Service. AVSZIP responses are for advice only. This process does not affect the outcome of the authorization
IAVS	International Address Verification Service address responses may be returned if you are using Address Verification Service. IAVS responses are for advice only. This value does not affect the outcome of the transaction
PROCAVS	Address Verification Service response from the processor when you use Address Verification Service and send a VERBOSITY request parameter value of MEDIUM
PROCCVV2	CVV2 response from the processor when you send a VERBOSITY request parameter value of MEDIUM.
AMEXID	Unique transaction ID returned when VERBOSITY = medium or high for tracking American Express CAPN transactions
AMEXPOSDATA	Value returned when VERBOSITY = medium or high

Result Codes

0	Execution successful
1	More input arguments required
2	Incorrect **api_password**. For HTTP API calls only.
3	Incorrect **op**. For HTTP API calls only.
90	An invalid value was provided for an input argument
98	Could not execute pfpro binary or transaction unsuccessful

browser

Detects the type of browser used, based on the provided HTTP user agent string.

Required Input

useragent	The HTTP user agent string from the browser

Output

op	The name of this module: `browser`
version	The version of this module
result	The result of the execution: `ok` if successful, or `error` when the module failed
resultcode	The result code matching the **result**: `0` if **result** is `ok` or one of the result codes in the "Result Codes" section below
error	If **result** is `error`, contains a description of the error
browser	The type of browser detected • `pda` for PDA browsers • `phone` for phone browsers with very small screens • `other` for other (non-detected) browsers o Usually standard browsers like Netscape and Internet Explorer

Result Codes

0	Execution successful
1	More input arguments required
2	Incorrect **api_password**. For HTTP API calls only.
3	Incorrect **op**. For HTTP API calls only.

Usage Example

Instead of detecting the browser by executing this API module, you can also use the `BrowserType()` function within PHP code on the gateway.

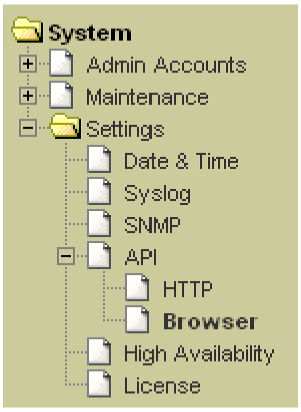
When the `BrowserType()` function is used, you can omit passing the user agent string as it will be automatically detected.

Using this, a single PHP page can be used to output two different sets of HTML content, depending on the browser type.

```
<?php
// include the InnGate API
require once($ SERVER['DOCUMENT ROOT'] . '/api/api.php');

// get the type of browser the user is using
$browserType = BrowserType();

if ($browserType == 'pda' || $browserType == 'phone')
{
// HTML FOR SMALL DEVICES ------------------------------------------------
?>

<html>
<body>
This is a tiny web page
</body>
</html>

<?php
}
else
{
// HTML FOR STANDARD BROWSERS --------------------------------------------
?>

<html>
<body>
This is a standard web page
</body>
</html>

<?php
}
?>
```

You can also adapt this code to output three different types of pages depending on the browser type, or even redirect the browser to other pages if necessary.

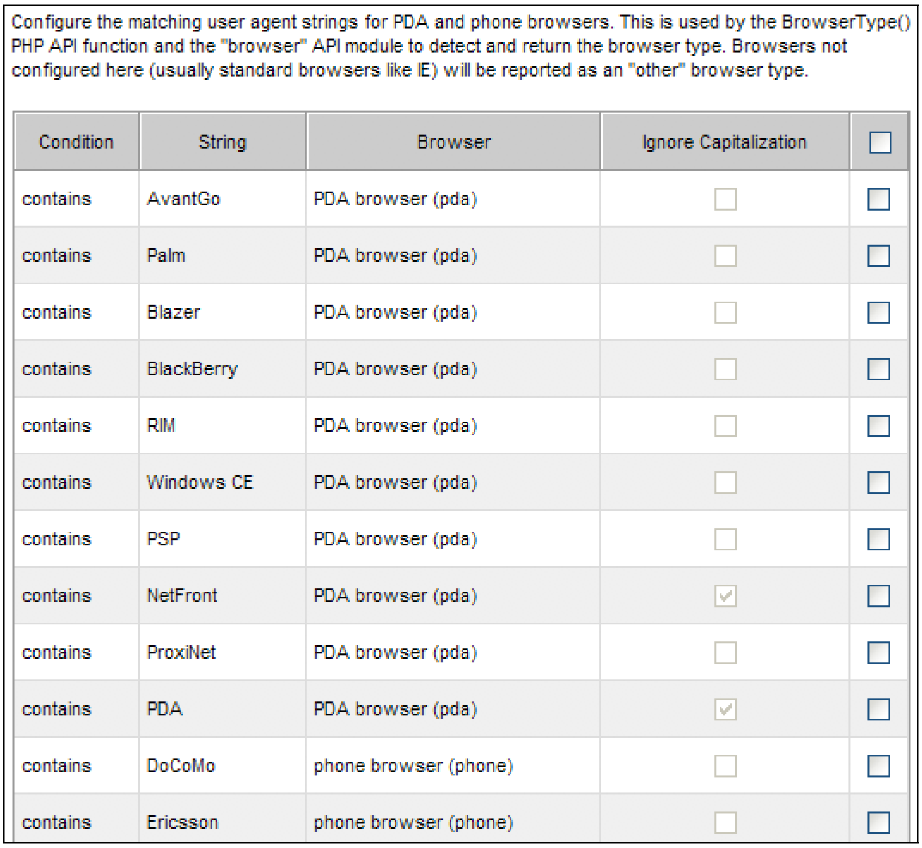
The browser strings supported by this API module can be configured in the Admin GUI.



To configure browser strings:

1. Click on **Settings**.
2. Click on **API**.
3. Click on **Browser**.

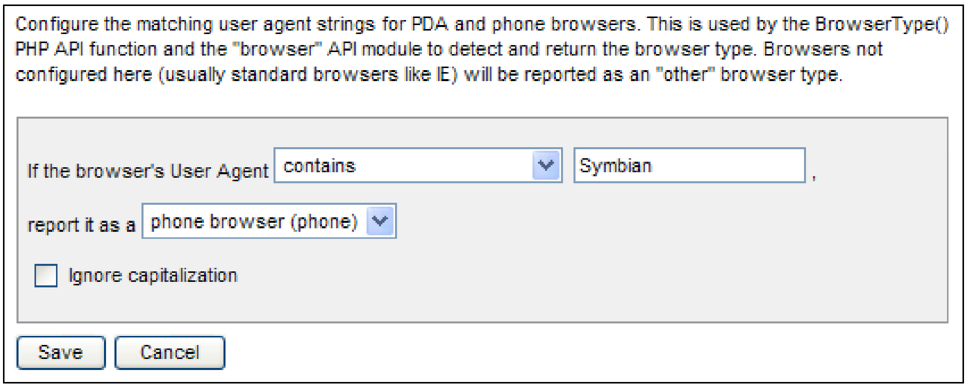
A list of recognized browser strings and resulting browser type is displayed. The module will use sub-string matching on the provided user agent string to determine the browser type. Capitalization can be ignored, if necessary.



Configure the matching user agent strings for PDA and phone browsers. This is used by the BrowserType() PHP API function and the "browser" API module to detect and return the browser type. Browsers not configured here (usually standard browsers like IE) will be reported as an "other" browser type.

Condition	String	Browser	Ignore Capitalization	☐
contains	AvantGo	PDA browser (pda)	☐	☐
contains	Palm	PDA browser (pda)	☐	☐
contains	Blazer	PDA browser (pda)	☐	☐
contains	BlackBerry	PDA browser (pda)	☐	☐
contains	RIM	PDA browser (pda)	☐	☐
contains	Windows CE	PDA browser (pda)	☐	☐
contains	PSP	PDA browser (pda)	☐	☐
contains	NetFront	PDA browser (pda)	☑	☐
contains	ProxiNet	PDA browser (pda)	☐	☐
contains	PDA	PDA browser (pda)	☑	☐
contains	DoCoMo	phone browser (phone)	☐	☐
contains	Ericsson	phone browser (phone)	☐	☐

Click on an existing browser string to modify it, or add new ones to be recognized by the API module.



Configure the matching user agent strings for PDA and phone browsers. This is used by the BrowserType() PHP API function and the "browser" API module to detect and return the browser type. Browsers not configured here (usually standard browsers like IE) will be reported as an "other" browser type.

If the browser's User Agent: contains, Symbian ,

report it as a: phone browser (phone)

☐ Ignore capitalization

ANTlabs

Connectivity Made Easy