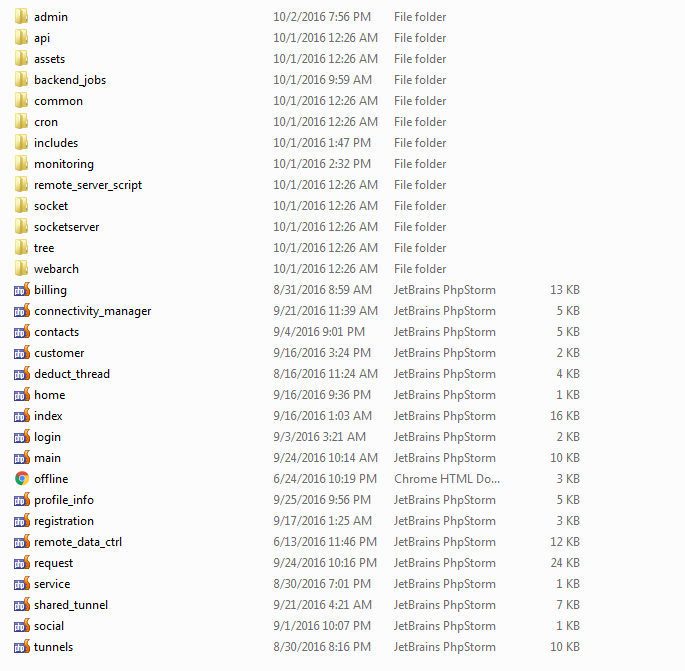
1. Description of source files

* File structure



* **admin**

This folder is subfolder for admin panel.

**。elements**

In this folder there are header file, footer file, sidebar file and session check file for admin panel.

**。cust\_info.php**

This file is used to manage customers info.

**。cust\_admin\_info.php**

This file is used to manage customers who is admin.

**。remote\_data.php**

This file is used to manage daemon of main server and remote servers.

**。cost\_ctrl.php**

This file is used to set cost of basic and premium tunnels.

Each tunnel cost contains some costs. Ex:tunnel, route tag, internet tag, server tag, getway, bidirection, real ip, route path

**。voucher\_ctrl.php**

This file is used to manage voucher.

**。vpn\_ip.php**

This file is used to manage vpn ips.

**。real\_ip.php**

This file is used to manage real ips.

**。global\_settings.php**

This file is used to manage website status and registration status.

* **api**

**。api\_function.php**

This file contains main functions that is used in this site.

* **assets**

This folder contains main css and js files and resources that is used in this site.

**。js/custom.js**

This file contains most of javascript functions of this site.

And this file is used for websocket.

**。js/panel\_layout.js**

This file contains the javascript functions of this website layout.

**。css/panel\_layout.css**

This file contains the css of website layout.

…

* **backend\_jobs**

**。resmon.php**

This file is used to get remote server info (cpu, memory info).

**。test.php**

This file is the main script file of remote server.

**。thread.php**

This file is the thread for php.

* **common**

This folder contains the header, footer, sidebar and session check files of this site.

**。head.php**

This file contains the css and js files of this site.

**。header.php**

This file is used for website header.

**。script.php**

This file contains js file of this site.

**。session\_check.php**

This file is used to check if session is expired.

* **cron**

**。cron.php**

This file is used to delete rows in tunnels table that has deleted fields.

This file is for cron job.

* **includes**

**。auth**

This folder contains files for google OAuth.

**。phpseclib**

This folder contains some files for ssh2,sftp.

**。config.php**

This file contains some info(database info, google oauth info).

**。connection.php**

This file is used for database connection.

* **monitoring**

**。mon.php**

This file is used to manage all daemons of main server.

* **socket**

This folder is used for php-websocket.

**。server.php**

This file is the main socket file for php-websocket file.

* **tree**

。This folder contains some js and css files for this site.

* **webarch**

。This folder contains some js and css files for this site.

* **billing.php**

。This file is used to manage points.

* **connectivity\_manager.php**

。This file is used to manage all tunnels.

* **customer.php**

。This file is used to show friends and contacts.

* **deduct\_thread.php**

。This file is deduct\_runner daemon.

* **login.php**

。This file is for login page.

* **profile\_info.php**

。This file is for profile page.

* **registration.php**

。This file is for registration page.

* **request.php**

。This file contains all requests of ajax call.

1. How the remote servers are working.

* **Remote server script( test.php)**

Remote server works as thread.

It checks “job\_queue” table of main server database every 5 seconds.

If there is any rows with “is\_complete\_action” field that is 0, then remote server parsing this row and process matched jobs and set “is\_complete\_action” as 2.

Then main server(php websocket) and search this row and show the result.

* **Main server scripts**

。server\_php\_runner.

This is php deamon for websocket server and this uses server.php.

This file is the main server script and contains several functions as following.

get completed job data from database,

connect websocket and send data.

。deduct \_cash\_runner

This is php deamon for cash of customers.

This daemon uses socket\_trigger1.py, and this daemon is used to calculate total points of each customers every 5 seconds.

。process\_complete\_runner

This is php deamon for completed jobs.

This daemon also uses socket\_trigger1.py, and this daemon is used to send request to the socket server to get completed jobs data.

。mon\_runner

This is php daemon to manage all other deamons of main server.

This daemon uses mon.php and this file contains some functions that start or stop other deamons and get status data of remote server.

1. Websock Messages

**dologin (Post Request):**

Post Fields: email, password

Response: {“status”:0/1, “data”: “Message”,”type”:”login”,”message\_type”:”reply”,”value”:”User info”}

**user\_register (Post Request):**

Post Fields: name, email, password,display\_name

Response: {“status”:0/1, “data”: “Message”} status = 0 not success, status = 1 success and data containing the message.

**add\_cloud (Post Request):**

Post Fields: cloud\_name, cloud\_email

Response: {“status”:0/1, “data”: “Message”} status = 0 not success, status = 1 success and data containing the message.

**delete\_cloud (Get Request):**

Post Fields: cloud\_id

Response: {“status”:0/1, “data”: “Message”} status = 0 not success, status = 1 success and data containing the message.

**add\_contact (Post Request):**

Post Fields: contact\_mail, contact\_name

Response: {“status”:0/1, “data”: “Message”} status = 0 not success, status = 1 success and data containing the message.

**addTunnel (Post Request):**

Post Fields: This will be a 2D post array and detected by a field containing inside the post data, that type (server/client). So the values may look like this in PHP format array(0=>array(‘name’=>’’, ‘email’=>’’, ‘type’=>’’, ‘cloud\_id’=>’’, ‘bidirection’=>’’, ‘gateway’=>’’, ‘group’=>’’), 1=>array(‘name’=>.....).....)

bidirection => 0/1/2/3

gateway=>0/1

group=>0-25 (i.e. 0=A, 1=B etc.)

cloud\_id=>numeric id of cloud

Response: {“status”:0/1, “data”: “Message”} status = 0 not success, status = 1 success and data containing the message.

**delete\_tunnel (Get Request):**

Get Fields: id

Response: {“status”:0/1, “data”: “Message”} status = 0 not success, status = 1 success and data containing the message.

Now Following All has Similar Request Type –

**gateway\_change, bidirection\_change, internet\_change, route\_change, plan\_change, group\_change:**

Get Fields: id, val (id=>id of the tunnel, val=>new value)

Response :

{"status" :1, 'data':'Your request under process, please wait...', "type":"gateay\_change/bidirection\_change/internet\_change/route\_change/group\_change ", "value":{‘id’:’’, ‘val’:’’}}

Now send this response to connected socket server, use the response from API to send as the value of ‘message’ attribute.

Now Following All has Similar Request Type –

**add\_server\_clone, add\_client\_clone:**

Get Fields: id (id=>id of the tunnel)

Response :

{"status" :1, 'data':'Your request under process, please wait...', “uid”:{customer-id}, "type":" add\_server\_clone/add\_client\_clone", "value":””}

Now send this response to connected socket server, use the response from API to send as the value of ‘message’ attribute.

**change\_tunnel:**

Get Fields: id, type (id=>id of the tunnel, type=>server/client)

Response :

{"status" :1, 'data':'Your request under process, please wait...', “uid”:{customer-id}, "type":" add\_server\_clone/add\_client\_clone", "value":””}

Now send this response to connected socket server, use the response from API to send as the value of ‘message’ attribute.

**request\_real\_ip:**

Get Fields: id (id=>id of the tunnel)

Response :

{"status" :1, 'data':'Your request under process, please wait...', "type":" request\_real\_ip ", "value":””}

Now send this response to connected socket server, use the response from API to send as the value of ‘message’ attribute.

**save\_a\_tunnel (Post Request):**

This action is for apply button on a particular tunnel.

Post Fields: id (id of the tunnel)

Response: {“status”:0/1, “data”: “Message”} status = 0 not success, status = 1 success and data containing the message.

**save\_a\_tunnel (Get Request):**

This action is for apply button for all tunnels.

Response: {“status”:0/1, “data”: “Message”} status = 0 not success, status = 1 success and data containing the message.

**get subnet from desktop app:**

{"type":"get\_subnet", "message\_type":"reply", "data":{"id":id, "value":subnet\_value}}

**get DeV from desktop app:**

{"type":"get\_DeV", "message\_type":"reply", "data":{"id":id, "value":dev\_value}}

here dev\_value are:

-1 = disconnected

0 = connecting

1 = connected

**get tunnels**

{"type":"get\_tunnels", "message\_type":"reply"}

**create new acl:**

{"type":"create\_new\_acl", "message\_type":"request", "data":tunnel\_id}

**get\_acl\_info:**

{"type":"get\_acl\_info", "message\_type":"request", "data":tunnel\_id}

**get acl value by button:**

{"type":"get\_acl\_val", "message\_type":"request", "data":{"id":id, "type":database\_table\_name, "name":database\_field\_name}}

**update acl value:**

{"type":"acl\_update", "message\_type":"request", "data":{"id":id, "type":database\_table\_name, "name":database\_field\_name, "val":value}}

**clear acl value:**

{"type":"clear\_acl\_values", "message\_type":"request", "data":{"tid":id}}

**set default acl:**

{"type":"set\_default\_acl", "message\_type":"request", "data":{"id":id, “tid”:tunnel id}

**get available real ip :**

{"type":"get\_available\_real\_ip ", "message\_type":"request"}

**check acl result stored in database**

{"type":"chk\_res", "message\_type":"request", "data":{"id":acl\_id, "type":database\_table\_name, "val":database\_field\_name}}

**send point to friend:**

{"type":"send\_point\_to\_friend", "message\_type":"request", "data":{"friend\_id":friend id, "point":point}}

**shared tunnel search:**

{"type":"shared\_tunnel\_search", "message\_type":"request", “shared\_with”:shared user id,”user\_id”:user id}

**shared tunnel:**

{"type":"shared\_tunnel", "message\_type":"request", "shared\_with”:shared user id,”user\_id”:user id,”t\_id”:tunnel id,”c\_id”: cloud id}

**remove sharing:**

{"type":"remove\_sharing", "message\_type":"request", "shared\_with":shared user id,”user\_id”:user id,”tunnel\_id”:tunnel id}

**install acl:**

{"type":"install\_acl ", "message\_type":"request",”acl\_id”:acl id,”tunnel\_id”:tunnel id}

<https://docs.google.com/document/d/1-4ZLk3mjBY5PTm0qFIYj-4ehrpRCCZRAXJ5ajr9WErA/edit>