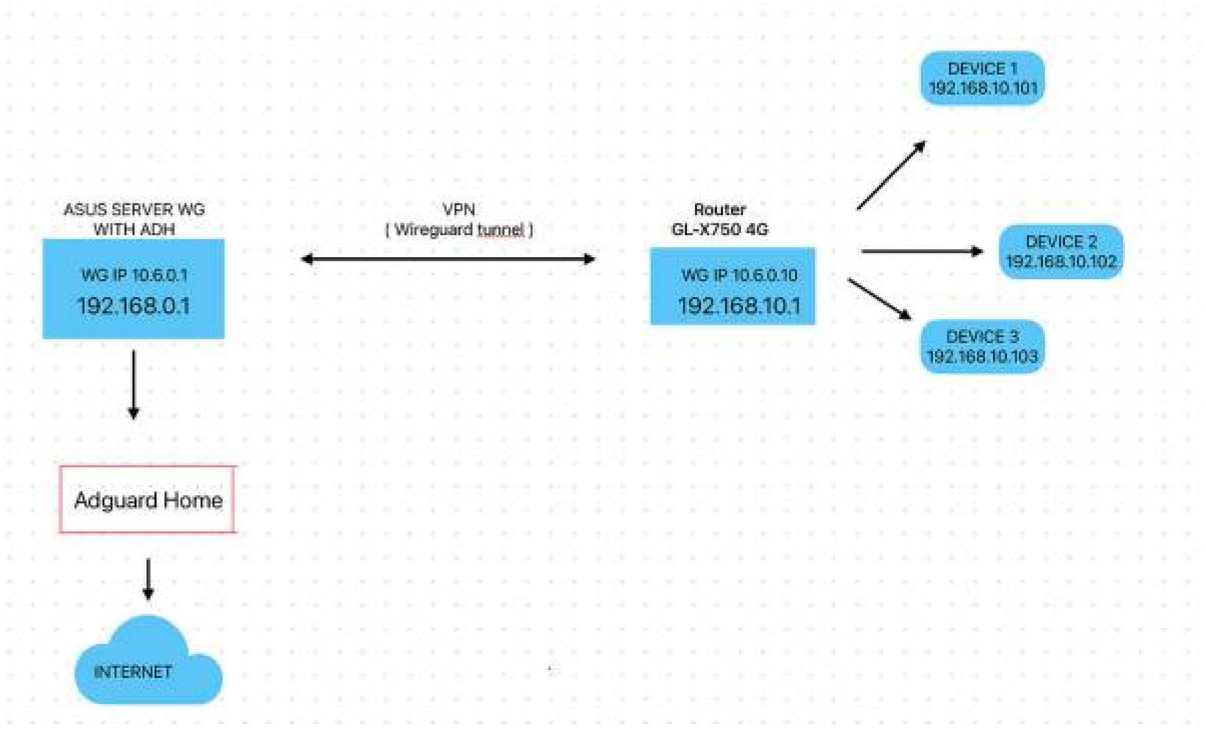
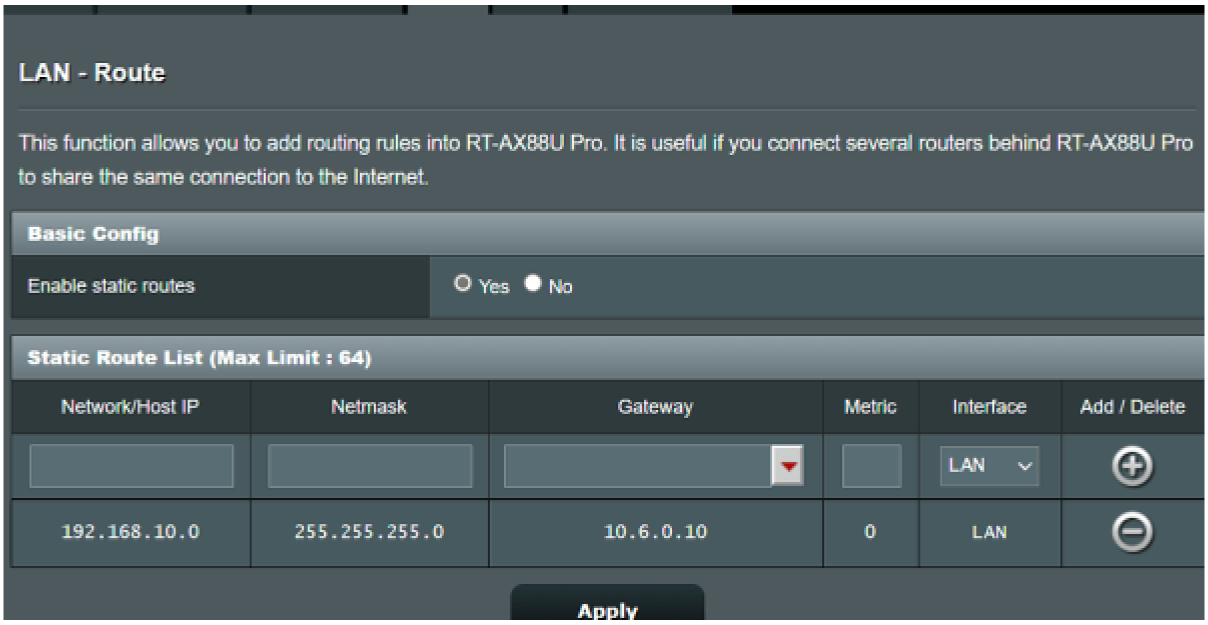


Our solution bases on the GL router, including the VPN Server and Client. Only for reference.

1. Enable Lan Access on the server and client.
2. Disable IP masq on the client. And configure the server to route to the client's subnet.
3. In the client, DNS is configured as the wg server's address, and custom DNS override vpn dns is turned off.
4. The server firewall is configured with a redirection rule that redirects traffic from port 53 of the wgserver to port 3053 (adguard).
5. The client subnet DNS traffic will be directed to server.
6. Client Luci add firewall rule



LAN - Route

This function allows you to add routing rules into RT-AX88U Pro. It is useful if you connect several routers behind RT-AX88U Pro to share the same connection to the Internet.

Basic Config

Enable static routes: Yes [✓ No]

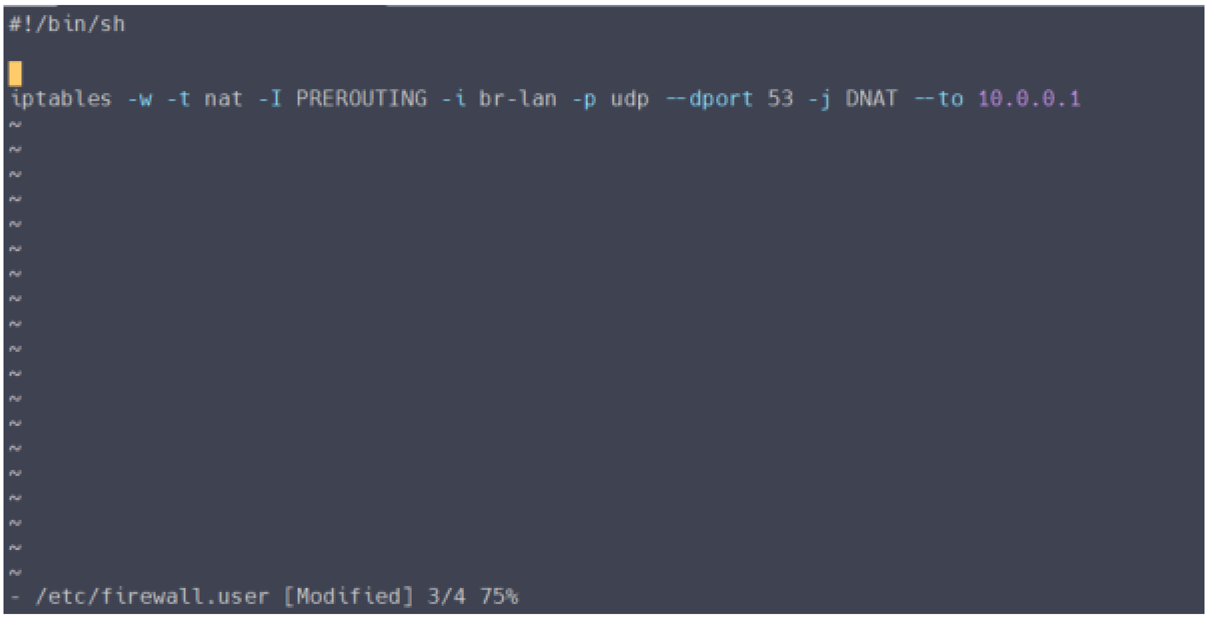
Static Route List (Max Limit : 64)

Network/Host IP	Netmask	Gateway	Metric	Interface	Add / Delete
				LAN	⊕
192.168.10.0	255.255.255.0	10.6.0.10	0	LAN	⊖

Since the firewall menu has not the custom rule, please manual add in the SSH: Login the SSH, Create the file:

```
vi /etc/firewall.user
```

```
iptables -w -t nat -I PREROUTING -i br-lan -p udp --dport 53 -j DNAT --to
10.6.0.1
```



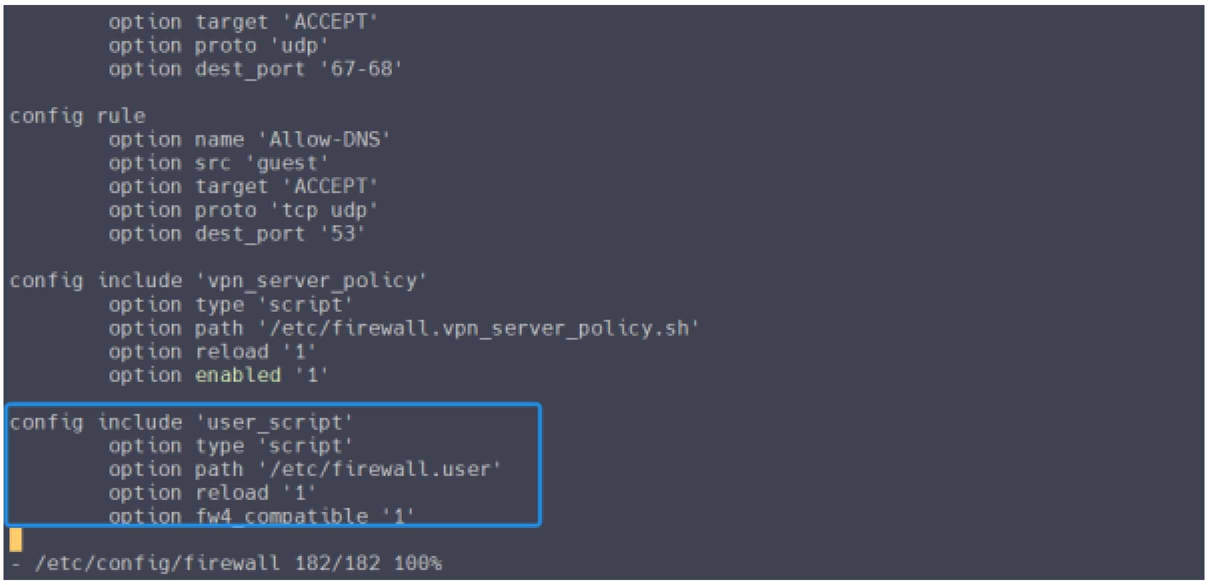
```
#!/bin/sh

iptables -w -t nat -I PREROUTING -i br-lan -p udp --dport 53 -j DNAT --to 10.0.0.1
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
- /etc/firewall.user [Modified] 3/4 75%
```

```
chmod 755 /etc/firewall.user
```

add the custom rule in the /etc/config/firewall:

```
config include 'user_script'
        option type 'script'
        option path '/etc/firewall.user'
        option reload '1'
        option fw4_compatible '1'
```



```
        option target 'ACCEPT'
        option proto 'udp'
        option dest_port '67-68'

config rule
        option name 'Allow-DNS'
        option src 'guest'
        option target 'ACCEPT'
        option proto 'tcp udp'
        option dest_port '53'

config include 'vpn_server_policy'
        option type 'script'
        option path '/etc/firewall.vpn_server_policy.sh'
        option reload '1'
        option enabled '1'

config include 'user_script'
        option type 'script'
        option path '/etc/firewall.user'
        option reload '1'
        option fw4_compatible '1'

- /etc/config/firewall 182/182 100%
```

reboot OR

```
/etc/init.d/firewall restart
```

https://forum.gl-inet.com/t/vpn-wireguard-and-adguard-home/45861

To Start vi To use vi on a file, type in vi filename. If the file named filename exists, then the first page (or screen) of the file will be displayed; if the file does not exist, then an empty file and screen are created into which you may enter text.

vi filename	edit filename starting at line 1
vi -r filename	recover filename that was being edited when system crashed
:x<Return>	quit vi, writing out modified file to file named in original invocation
:q<Return>	quit (or exit) vi
:q!<Return>	quit vi even though latest changes have not been saved for this vi call

To Exit vi Usually the new or modified file is saved when you leave vi. However, it is also possible to quit vi without saving the file. Note: The cursor moves to bottom of screen whenever a colon (:) is typed. This type of command is completed by hitting the <Return> (or <Enter>) key. * :x<Return> :wq<Return> quit vi, writing out modified file to file named in original invocation :q<Return> quit (or exit) vi * :q!<Return> quit vi even though latest changes have not been saved for this vi call Moving the Cursor Unlike many of the PC and MacIntosh editors, the mouse does not move the cursor within the vi editor screen (or window). You must use the the key commands listed below. On some UNIX platforms, the arrow keys may be used as well; however, since vi was designed with the Qwerty keyboard (containing no arrow keys) in mind, the arrow keys sometimes produce strange effects in vi and should be avoided. If you go back and forth between a PC environment and a UNIX environment, you may find that this dissimilarity in methods for cursor movement is the most frustrating difference between the two. In the table below, the symbol ^ before a letter means that the <Ctrl> key should be held down while the letter key is pressed. * j or <Return>

```
[or down-arrow] move cursor down one line
```

* k [or up-arrow] move cursor up one line * h or <Backspace>

```
[or left-arrow] move cursor left one character
```

* l or <Space>

```
[or right-arrow]    move cursor right one character
```

* 0 (zero) move cursor to start of current line (the one with the cursor) * $ move cursor to end of current line w move cursor to beginning of next word b move cursor back to beginning of preceding word :0<Return> or 1G move cursor to first line in file :n<Return> or nG move cursor to line n :$<Return> or G move cursor to last line in file Screen Manipulation The following commands allow the vi editor screen (or window) to move up or down several lines and to be refreshed. ^f move forward one screen ^b move backward one screen ^d move down (forward) one half screen ^u move up (back) one half screen ^l redraws the screen ^r redraws the screen, removing deleted lines Adding, Changing, and Deleting Text Unlike PC editors, you cannot replace or delete text by highlighting it with the mouse. Instead use the commands in the following tables. Perhaps the most important command is the one that allows you to back up and undo your last action. Unfortunately, this command acts like a toggle, undoing and redoing your most recent action. You cannot go back more than one step. * u UNDO WHATEVER YOU JUST DID; a simple toggle The main purpose of an editor is to create, add, or modify text for a file. Inserting or Adding Text The following commands allow you to insert and add text. Each of these commands puts the vi editor into insert mode; thus, the <Esc> key must be pressed to terminate the entry of text and to put the vi editor back into command mode. * i insert text before cursor, until <Esc> hit I insert text at beginning of current line,

until <Esc> hit * a append text after cursor, until <Esc> hit A append text to end of current line, until <Esc> hit * o open and put text in a new line below current line, until <Esc> hit * O open and put text in a new line above current line, until <Esc> hit Changing Text The following commands allow you to modify text. * r replace single character under cursor (no <Esc> needed) R replace characters, starting with current cursor position, until <Esc> hit cw change the current word with new text, starting with the character under cursor, until <Esc> hit cNw change N words beginning with character under cursor, until <Esc> hit;

```
e.g., c5w changes 5 words
```

C change (replace) the characters in the current line, until <Esc> hit cc change (replace) the entire current line, stopping when <Esc> is hit Ncc or cNc change (replace) the next N lines, starting with the current line, stopping when <Esc> is hit Deleting Text The following commands allow you to delete text. * x delete single character under cursor Nx delete N characters, starting with character under cursor dw delete the single word beginning with character under cursor dNw delete N words beginning with character under cursor;

```
e.g., d5w deletes 5 words
```

D delete the remainder of the line, starting with current cursor position * dd delete entire current line Ndd or dNd delete N lines, beginning with the current line;

```
e.g., 5dd deletes 5 lines
```

Cutting and Pasting Text The following commands allow you to copy and paste text. yy copy (yank, cut) the current line into the buffer Nyy or yNy copy (yank, cut) the next N lines, including the current line, into the buffer p put (paste) the line(s) in the buffer into the text after the current line Other Commands Searching Text A common occurrence in text editing is to replace one word or phase by another. To locate instances of particular sets of characters (or strings), use the following commands. /string search forward for occurrence of string in text ?string search backward for occurrence of string in text n move to next occurrence of search string N move to next occurrence of search string in opposite direction Determining Line Numbers Being able to determine the line number of the current line or the total number of lines in the file being edited is sometimes useful. :.= returns line number of current line at bottom of screen := returns the total number of lines at bottom of screen ^g provides the current line number, along with the total number of lines, in the file at the bottom of the screen Saving and Reading Files These commands permit you to input and output files other than the named file with which you are currently working.

:r filename<Return> read file named filename and insert after current line (the line with cursor) :w<Return> write current contents to file named in original vi call :w newfile<Return> write current contents to a new file named newfile :12,35w smallfile<Return> write the contents of the lines numbered 12 through 35 to a new file named smallfile :w! prevfile<Return> write current contents over a pre-existing file named prevfile