Realtime

New in Tiki25. Use built-in websockets server within Tiki to enable realtime communication features of the app.

This is optional. If your server doesn't support this, it's OK. If you have it, supported features will be faster and smoother. For now, it is used for Tiki Manager Package, and will later be added to other Tiki Features.

Powered by https://packagist.org/packages/cboden/Ratchet and https://reactphp.org

This will require a special server configuration (possibly root access). tiki-check.php will indicate if server is correctly set up.

In this documentation, we are going to configure Virtualmin (but any other server can be used), and then we will check if it worked.

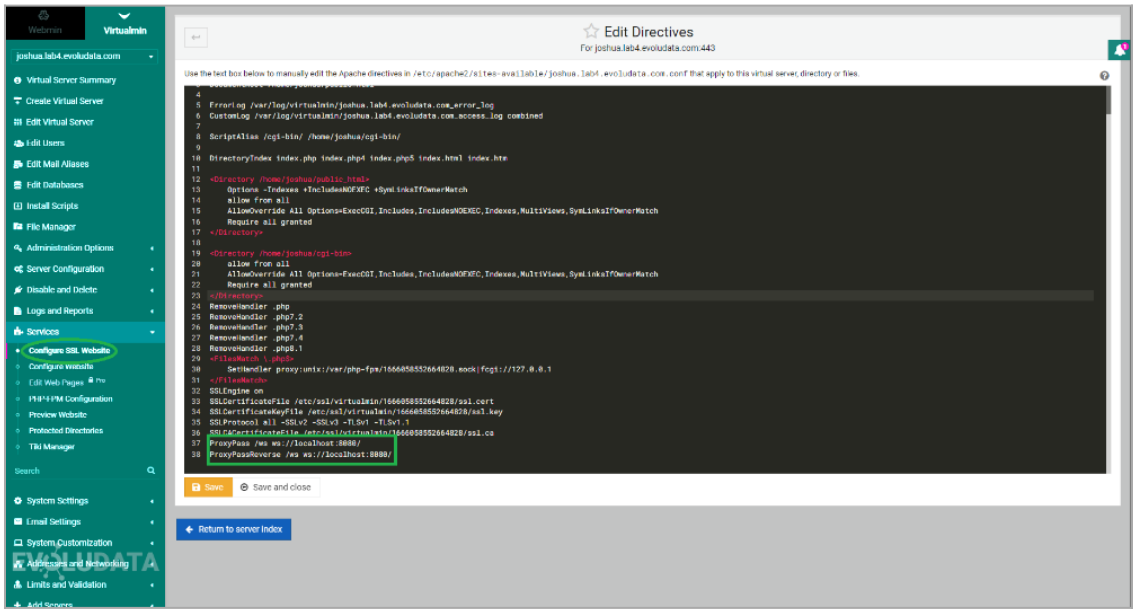
Configuration

Requirements

We already have a Tiki instance up and running, on Virtualmin, now we need to take into account a few more things: **Proxy Settings**, **Realtime Service**, **proxy_wstunnel**, **Realtime** feature, and **Realtime port** in Tiki in order to get Realtime working.

1. Proxy Settings

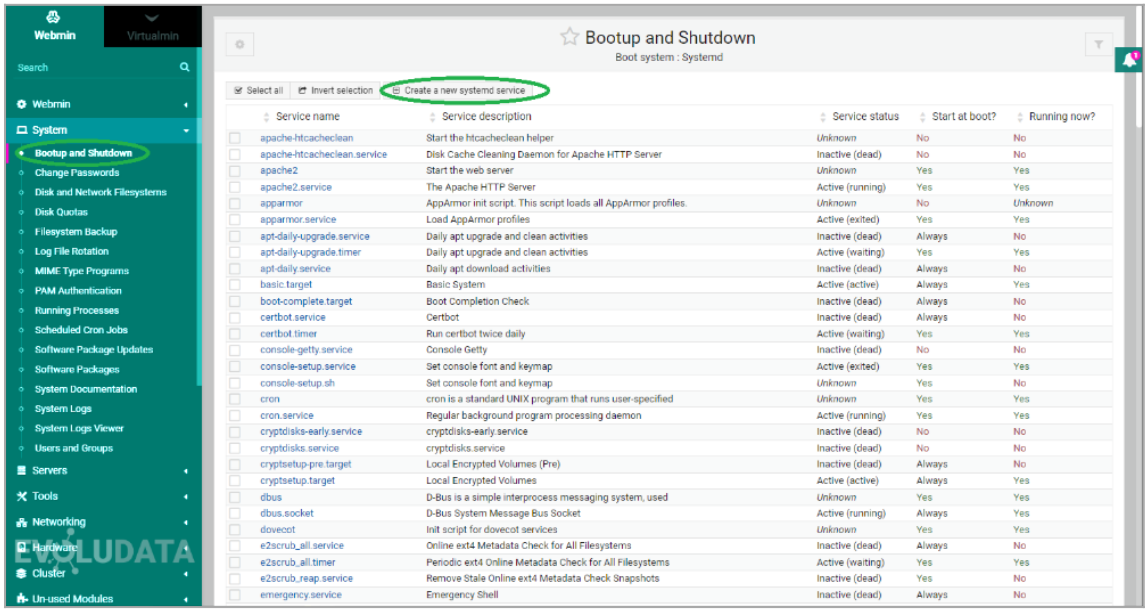
First, let us edit Apache directives and add the ProxyPass and the ProxyPassReverse settings in the SSl virtual host. We make both point to */ws* *ws://localhost:8080/*
To do this, we go to ***Virtualmin*** -> ***Services*** -> ***Configure SSL Website*** -> ***Edit Directives***.
In the text box that opens, we add `ProxyPass /ws ws://localhost:8080/` and `ProxyPassReverse /ws ws://localhost:8080/` at the end, like shown in the picture below:



Click to expand

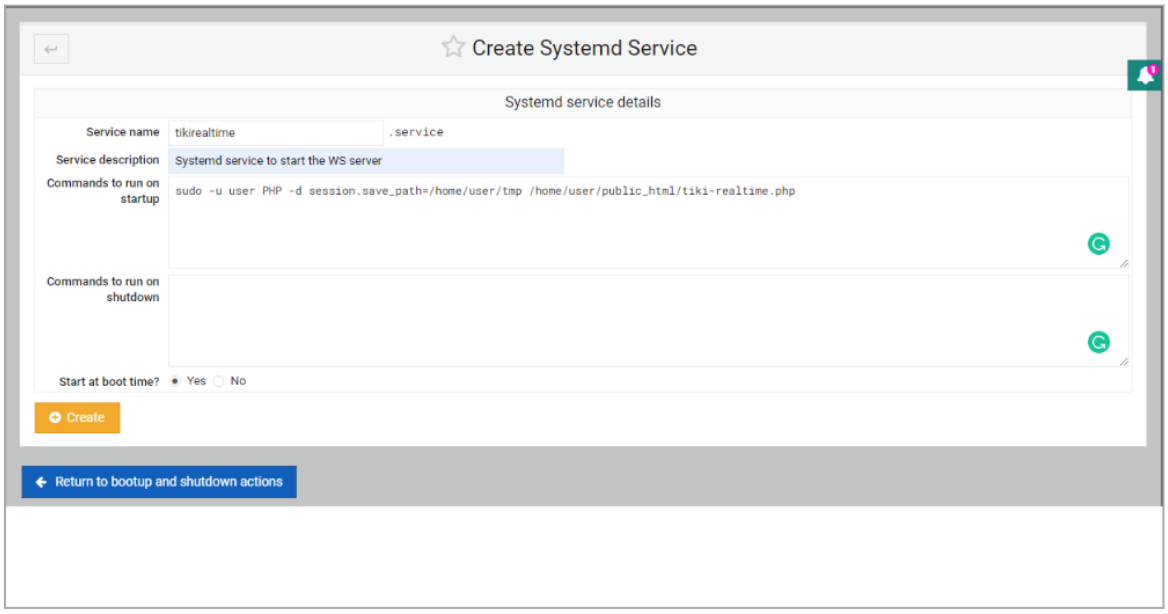
2. Realtime Service

At this step, we need to create a Systemd service to start the WS server.
N.B: Start the WS server with the same user that Tiki web requests run as (to avoid permission issues) - e.g. `sudo -u www-data php tiki-realtime.php`
The process is simple. Just go to ***Webmin*** -> ***System*** -> ***Bootup and Shutdown*** -> ***create a new systemd service***.



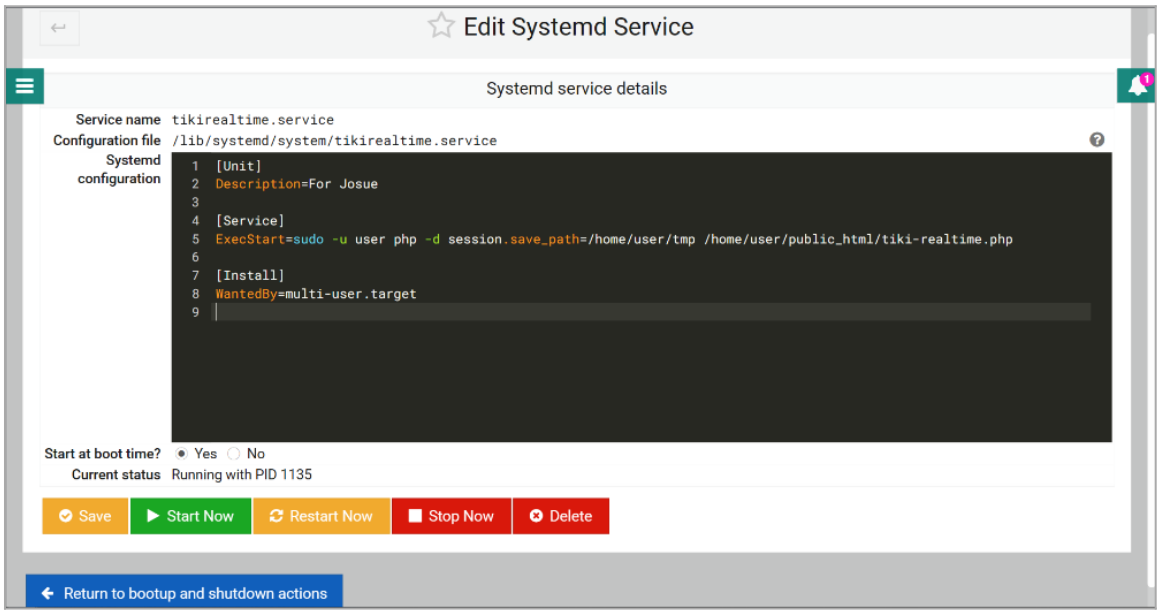
Click to expand

In the opening form, we fill in the **Service name**, the **service description**, and the **commands to run on startup**. As we don't need to shut this down, we are going to leave empty the "Commands to run on shutdown" field, check **Yes** for Start at boot time option, and click **create**.
The command to run on startup is: `sudo -u user PHP -d session.save_path=/home/user/tmp /home/user/public_html/tiki-realtime.php`
Replace user with your user



Click to expand

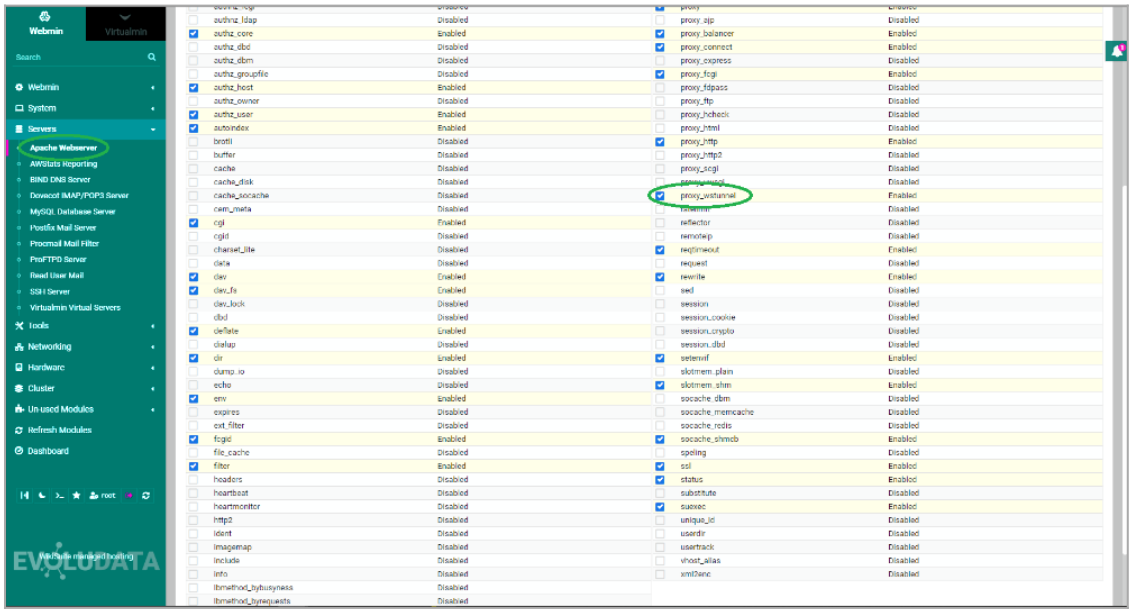
After this, you can check the content of your system service, and it should look something similar to the image below:



Click to expand

3. The proxy_wstunnel

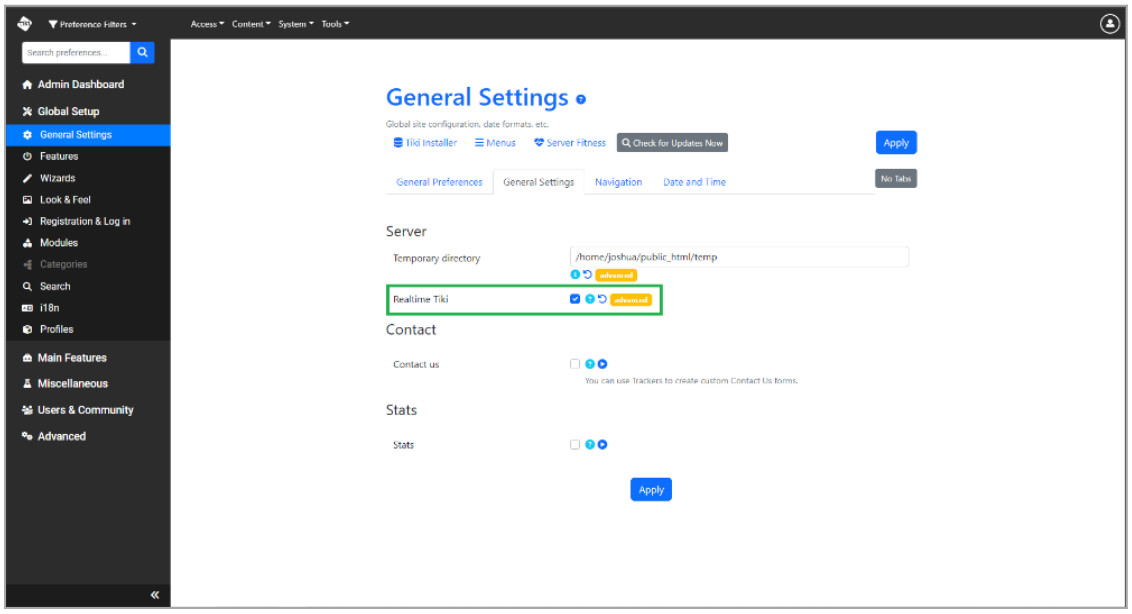
The last step here in Virtualmin is to make sure the proxy_wstunnel module is enabled.
To do this, go to ***Webmin*** -> ***Servers*** -> ***Apache Webserver*** -> ***Global configuration*** -> ***Configure Apache Modules***, and check the **proxy_wstunnel** module checkbox, like in the figure below:



Click to expand

4. Realtime feature

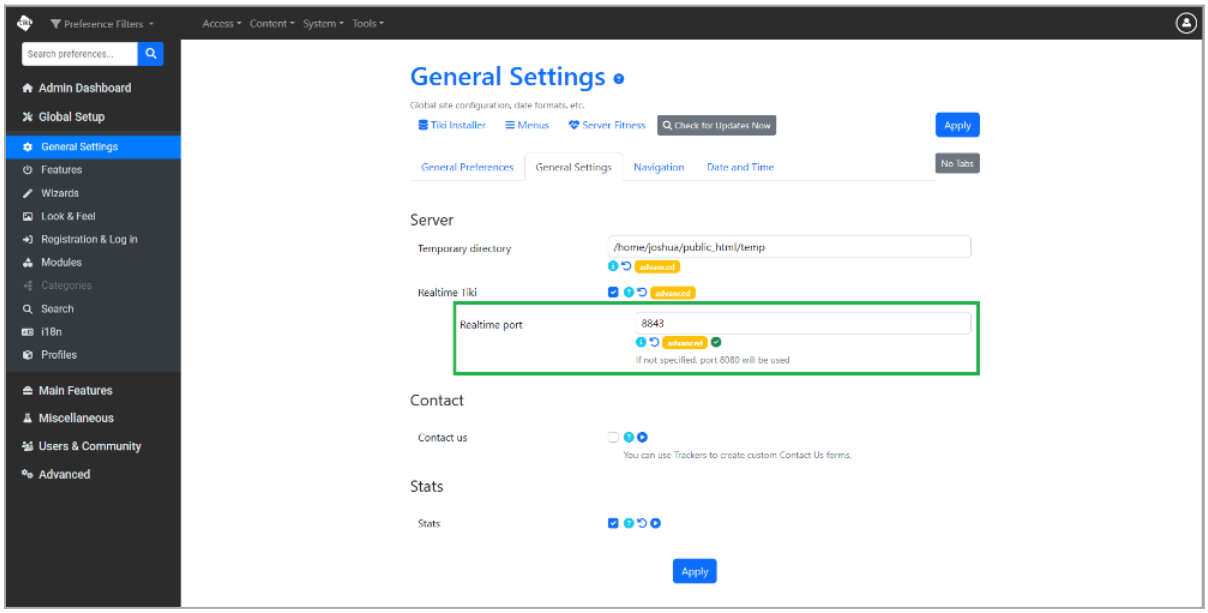
At this step, we need to enable Realtime feature in Tiki. When not enabled, this feature is visible only upon search. Under ***Settings*** -> ***Control Panels***, in the "*Search preferences*" search box, type a keyword, "*realtime*", for example.



Click to expand

5. The Realtime port

Let's note that this not required, but it is an interesting preference and usefull in some cases (default port is not available), because it allows you to set the port on which the server will be listening to. 8080 is the port number by default...



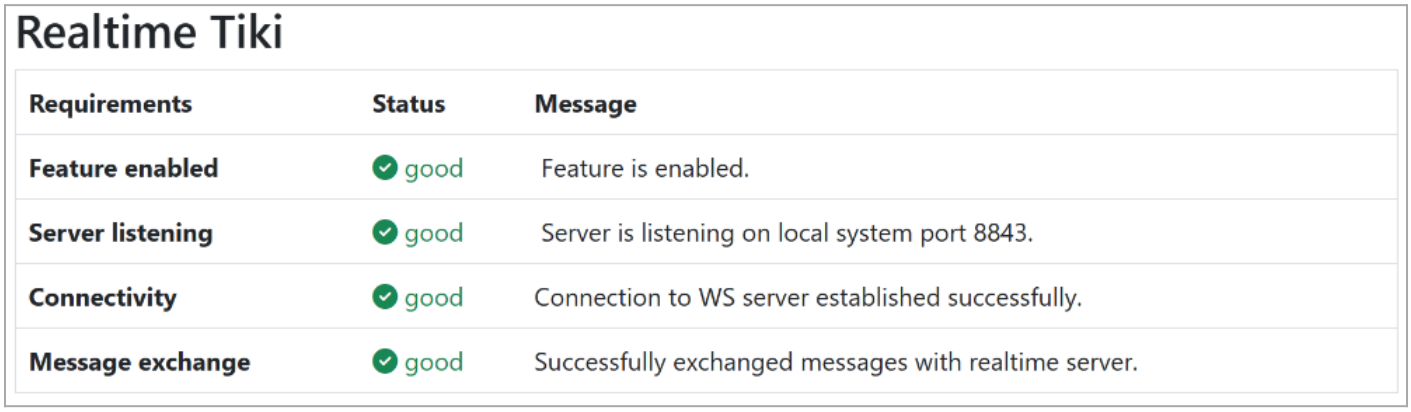
Click to expand

Let's note also that Tiki Realtime can also be started from CLI by running, from the project root, the command ***php tiki-realtime.php***, and, here too, the port can be specified by adding a port parameter to the command: ***php tiki-realtime.php -p 8843*** which means port number will be set to **8843**.

After this is done, Tiki Realtime should be well set up and working. There is a way to check that. So let's see if everything went well...

Check that Realtime is working

This task is as simple as opening our Tiki and running "**Server Check**"
To do this you need to go under ***Settings*** -> ***Control Panels*** -> ***Global Setup*** -> ***General Settings*** and find "***Server Fitness***", click on it, and on the page that opens, scroll down until you find the "***Tiki Realtime***" section. If Tiki Realtime is working, all the requirements status should be "Good" as shown in the image below:



Click to expand

For developers

- Initial commit: https://gitlab.com/tikiwiki/tiki/-/merge_requests/1638
- If you'd like some guidance to deploy this to various Tiki features, please reach out to Marc Laporte

Related links

- Chatbots