Tiki and Virtualmin interop

To easily install and update Tiki within Virtualmin: https://gitlab.com/wikisuite/wikisuite-packages

See also Tikis remote management tools and Tiki Manager within Tiki

Context: You want several Tiki instances on a same Virtualmin server. How to organize? How to segment data? What are naming conventions? Default settings?

Most of the previous testing of Tiki Manager was done on ClearOS. And ClearOS is not designed for shared hosting but instead to have different servers per project. When we do have multiple Tiki instances (clones and test upgrades, or even completely different projects), they are on the same level.

- /var/www/virtual/example.org/html
- /var/www/virtual/next.example.org/html
- /var/www/virtual/example.com/html

Virtualmin is designed for shared hosting and designed to have different users with different permissions. This is great and we will use this. But there are many permutations and will require some TLC.

There are virtual servers and sub-servers:
https://www.virtualmin.com/documentation/tutorial/virtual-server-basics
https://www.virtualmin.com/node/40081

Now how should we handle various use cases?

- An org that needs just one Tiki
- A Tiki shared hosting service
- A dev server (needs many Tikis)
- A show server: https://dev.tiki.org/show.tiki.org+Overview
- https://tiki.org/Pre-Dogfood-Server
- MultiTiki
- development, testing, staging production environments
- etc.

Working scenario is that we must always plan for more than one Tiki instance. At the very least to test upgrades.

- I dislike adding extra Tikis as example.org and example.org/21x/ and want example.org and 21x.example.org
    - But in a shared hosting context, there can be a difference in price between the two.

I never want to install Tiki directly anymore. Everything should be done via Tiki Manager.

Someone could want a different access level for dev. Ex.: So some developers don't have access to production. So production would be:
/home/project1/domains/example.org/public_html/
But dev would be:
/home/devuserforproject1/domains/dev.example.org/public_html/

But you still want Tiki Manager to permit a clone from prod to dev.

About usernames and domain names:

- A same site can have several domain names
- Until DNS is pointing real domain, it's nice to be able to use a wild card subdomain. Ex. example.org is the future site on a dev1.example.com server. So we'd use example.dev1.example.com
- Virtualmin also has a Preview Website feature.

There are many recent changes to Tiki Manager to make it work well with Virtualmin
https://gitlab.com/tikiwiki/tiki-manager/-/commits/master
These changes generally also make Tiki Manager better even without Virtualmin (ex.: running as non-root)

I look forward to your ideas on how we can make this great and have all the community dogfooding

Virtual Servers vs Sub-Servers

Virtualmin can do this for Virtual Servers or Sub Servers: use relative paths for all data not stored in the DB. Like https://doc.tiki.org/File-Storage#Ideal_scenario

What is overhead of RAM and disk space to have a new Virtual Server for each Tiki?

Each Tiki or Tiki Manager instance is in its own Virtual Server

Niel suggested to not use Sub-Servers and have a pattern of
home/user1/public_html
home/user2/public_html

But then, we'll end up with hundreds on a same server. Listing and sorting becomes important, so we need to think of a pattern, such as:

Usernames as domain names

This is simple and less error-prone, but offers poor sorting

- home/manager.example.org/public_html
- home/dev.example.org/public_html
- home/example.org/public_html

Concat domain name and username

Good sorting but looks odd

- home/example.orgmanager/public_html
- home/example.orgdev/public_html
- home/example.org/public_html

Concat part of domain name and usernames

Good sorting. Could work. On long domain names, it's not always clear what to use for username, so two users could do it differently

- home/examplemanager/public_html
- home/exampledev/public_html
- home/example/public_html

Usernames as inverted domain

Good sorting and predictable pattern. Can be quite long.

- home/org.example.manager/public_html
- home/org.example.dev/public_html
- home/org.example/public_html

Other concerns

- Non-root users can't create new Tiki instances. We want non-root users to be autonomous (clones, test upgrades, etc.)

1.1.2. Virtual Server for Tiki Manager and Sub-Servers for all Tiki instances

So perhaps the 1st application of all Virtualmin Virtual Servers should be Tiki Manager with a pattern as follows:

user: project1
domain: example.org

1st domain is manager.example.org

Tiki Manager CLI:
/home/project1/manager/tiki-manager.php

If web interface is activated:
/home/project1/public_html/

Be aware that "A sub-server cannot have MySQL enabled unless the parent server does" so you need to let Virtual create a MySQL DB for Tiki Manager even if it won't be used.

1st site:
/home/project1/domains/example.org/public_html

Then, all sites follow the pattern. Ex.:
/home/project1/domains/clone.example.org/public_html/
/home/project1/domains/dev.example.org/public_html/
/home/project1/domains/next.example.org/public_html/

Odd that Sub-Servers are not subdomains of main server.

This is scenario in use at: https://wikisuite.org/How-to-install-WikiSuite

1.1.3. Empty Virtual Server and Sub-Servers for Tiki Manager and all Tiki instances

Create a first Virtual Server with a random name (iwillneverusethis.example.org)

Then, all sites follow the pattern. Ex.:
/home/project1/domains/manager.example.org/public_html/
/home/project1/domains/example.org/public_html/
/home/project1/domains/clone.example.org/public_html/
/home/project1/domains/dev.example.org/public_html/
/home/project1/domains/next.example.org/public_html/

Benefits

- All websites (including manager) at the same level in domains

1.1.4. CLI in base Virtual Server, and put manager web files in Sub-Server

Base site remains empty from web files: /home/project1/public_html/

Tiki Manager CLI:
/home/project1/tiki-manager.php

Tiki Manager Web:
/home/project1/domains/manager.example.org/public_html/

Then, all sites follow the pattern. Ex.:
/home/project1/domains/clone.example.org/public_html/
/home/project1/domains/dev.example.org/public_html/
/home/project1/domains/next.example.org/public_html/

A little confusing
All web-accessible files are visible from one place
Tiki Manager is in a predictable path

One or many Tiki Managers per server

There will be recipes (probably at least 2) to get Tiki Manager working on Virtualmin:

1.1.1. A single instance of Tiki Manager running as root

Manages Tiki instances in various web spaces (which Virtualmin calls Virtual Servers)

An idea from Jonny: Similar to phpMyAdmin, have one instance of Tiki Manager and depending of your permissions in the web interface, you see certain Tiki instances, but not others. But not clear how to handle for CLI.

1.1.2. A Tiki Manager per Virtual Server

Which then opens a new question for clones / and cloneandupdate: do we push or pull?

When cloning from prod to dev (without root and when they are in two different virtual servers), it will require using SSH to same server. /home/project1/manager/tiki-manager.php will connect as devuserforproject1 to push clone.

We brainstormed on the idea that Tiki Manager could be used to install other Tiki Managers. Perhaps the root Tiki Manager could launch Virtualmin CLI/API to create a new webspace (virtual server), and install a Tiki Manager, which in turn installs a Tiki.