VMoodle (local) : Installation

Introduction

`Warning :` Setting up a virtualized array of moodle assumes you have a good knowledge of moodle administration and installation, and a correct background in LAMP application administration.

ActiveProlearn can provide full support in helping building, setting up and training exploitation staff to master the architecture.

Installation

What you basically need to virtualize a Moodle installation is:

- Install your moodle using a non virtualized codebase as a main moodle instance.
- Add/Install the VMoodle plugin.
- Add/Install VMoodle block and report plugins.
- Configure a local/vmoodle/vconfig.php on the local/vmoodle/vconfig-dist.php base.
- Plug in to your main configration script the VMoodle trap sequence.
- Report some patchs in the /mnet implementation if you are planningn using MNET and want to get MNET stable over time.

Components installation

- Download the zips from the repositories (local_vmoodle, block_vmoodle and report_vmoodle) and extract respectively the archives in '/local', '/blocks' and '/report' directory in your moodle.
- Browse to the Administration Notifications to terminate installation.

After plugins have been installed, your moodle is still NOT capable to answer to virtual domains. Your virtualized installation keeps working a s standard moodle.

Note that we DO NOT support any more the pre-28 version.

Virtualisation file Configuration

This file is located in `/local/vmoodle` as `vconfig.php` and is responsible of the configuration switch. The configuration settings allow you to point where the vmoodle definition register is located.

Usually you wil use the database of your primary installation to store the main register. It is also possible to move this register to another DB instance as long as the name of the table `{PREFIX}_local_vmoodle` is conserved.

A template file is provided as `<moodleroot>/local/vmoodle/vconfig-dist.php`. Copy this file as `vconfig.php` and change adequate values inside. The `vconfig.php` could be relocated anywhere in your server as long as the main config.php is able to include it.

In most case, you will simply have to report your main dbsettigns in this file.

Update the main configuration file of Moodle

The virtualisation trap need to be set in the master configuration of Moodle as any entrance page of moodle will call it at startup.

Updates are needed inorder to:

- hook the configuration switch for standard use
- add special conditionnality for CLI script (VMoodle enabled scripts)
- Configure some network related behaviour when using MNET cross-services.

Hooking the vconfig.php will be the point where the current configuration (working database and moodledata volume) will switch.

Hooking the virtual configuration switch

The main point about this hooking is that it MUST be placed before calling the `setup.php` include in your `config.php` file.

```
... {CLI VMoodle Hook} ...
```

```
require(</path/to/moodleinstall>/local/vmoodle/vconfig.php');
```

```
require_once(dirname(__FILE__) . '/lib/setup.php');
```

Adding the CLI hook

What issue it solves:

Standard cli script just play the operations loading the standard configuration. VMoodle switches the configuration dynamically analysing the current HTTP required hostname or URL. In CLI mode you do NOT have this information in the execution environment and the admin needs to provide it in some way.

The CLI VMoodle hooking allows the following process to occur:

- The CLI script starts and loads an incomplete moodle configuration (just getting the vmoodle register db description),

- The script receives the command line parameters in which a –host additional option has been added.
- The script reloads the full configuration file till the end, activating the virtualistion switch.

Add the following code to your config.php file before including vconfig.ghp:

```
 // this fragment will trap the CLI scripts trying to work for a virtual
node, and
 // needing booting a first elementary configuration based on main config
 if (isset($CLI_VMOODLE_PRECHECK) && $CLI_VMOODLE_PRECHECK == true) {
     $CLI_VMOODLE_PRECHECK = false;
     return;
 }
```

Important : Only adapted CLI script will be able to process data in virtual submoodles. Most of the standard admin scripts have been adapted and are provided in the /local/vmoodle/cli directory. In additions we added for each standard script a bulk script for launching each script on the complete register.

Note : use the –help command-line attribute to see the options.

Routing the instance domains to the single moodle install

As the physical install is unique, tou will have to configure your apache or nginx virtual hosts to converge to this single location. The easiest way to do so is:

- In apache : Adding ServerAlias clauses to the VHost definition
- On nginx : Adding servername additional instructions to the server structure routing the moodle root.

If you have one single domain for all your moodles and your web server configuration and DNS supports wildcard, you can wildcard the subdomain as *.mymoodles.edu for convenience.

Global configuration keys having an influence on the VMoodle behaviour

```
$CFG->mainhostprefix = 'http://someprefixthatmatchsmymainmoodle';
```

This key allows any instance to know which host matches the master install of moodle. We use this for some features such as f.e. identifying who is the “master network admin account” of the whole plant.

```
$CFG->forced_plugins_settings['user_mnet_hosts']['admin_override'] = true;
```

This key will allow to force the User Mnet Hosts blocks to let the local admins go through the network links. Usually we will forbid local subadmins to roam, but the master global admin should be able to. A sample of this settign is possible in vconfig.php :

```
$CFG->forced_plugins_settings['user_mnet_hosts']['admin_override'] = false;
if (preg_match('#'.$CFG->mainhostprefix.'#', $CFG->wwwroot)) {
```

```
    $CFG->forced_plugins_settings['user_mnet_hosts']['admin_override'] =
true;
 }
```

Using this configuration, the master moodle admin will be allowed to roam to any submoodle. The local subadmins will not. As roaming to aénother moodle creates an image of the user account, mulitplying the admin account could lead to some confusing states.

```
$CFG->mnetsiteadmins = true;
```

This key allows remote users to become site administrators of the local moodle. This is forbidden by standard Mnet rules but VMoodle implementation propose patchs to override.

Cron setup in virtualized environements

Using unitary crons for each nodes

A Moodle site has a unique cron task that performs a lot of distinct delayed and scheduled processings in moodle.

A standard way of setting up would setup a cron line for each of the virtual moodle an installation contains.

If you use the HTTP method to run crons, f.e.

- /10 * * * * wget -O /dev/null -q http://virtual1.mymoodles.edu

then you can register as many `wget` calls pointing all your subdomains, as yo'll do for independant standard moodles. If you use the command line method you will have to run the VMoodle enabled cron script, and tell in the command line for which host you are running the script:

```
/local/vmoodle/cli/cron.php --host=http://my_virtual.mymoodledomain.com
```

You will provide the “official” wwwroot the virtual site is attached to.

Using the VCron hyper rotator

The VMoodle implementation provides a vcron scheduler that will be lauched in the master moodle host context, and will automatically rotate and schedule all submoodle crons. VCron scheduler can operate using HTTP calls or command line execs (internally configured) and can itself be invoked through HTTP:

```
wget http://main.mymoodledomain.com/local/vmoodle/vcron.php
```

or in command line

```
sudo -uwww-data php /root/to/moodle/local/vmoodle/cli/vcron.php
```

You may internally configure the vcron settings to raise the number of virtual moodle processed per cron run.

the basic calculation for vcron rotation is :

With a “run per turn” of 1 and 20 virtual moodles and a cron period of 1 minute per turn, each vmoodle will be processed once per 20 minutes. Raising the “run per turn” to 2 will divided the effective cron period to 10 minutes per vmoodle instance.

For very large moodle arrays, it may be worth having a separated machine to process the crons for all the moodle instances. Cron processes may ovelap one on each other, but will not pull down theusual user's page distribution too much.

A very advanced feature (undocumented here) allow to distribute the vcron scheduling on several clusters.

---

back to VMoodle component index - Back to plugins - Back to catalogue