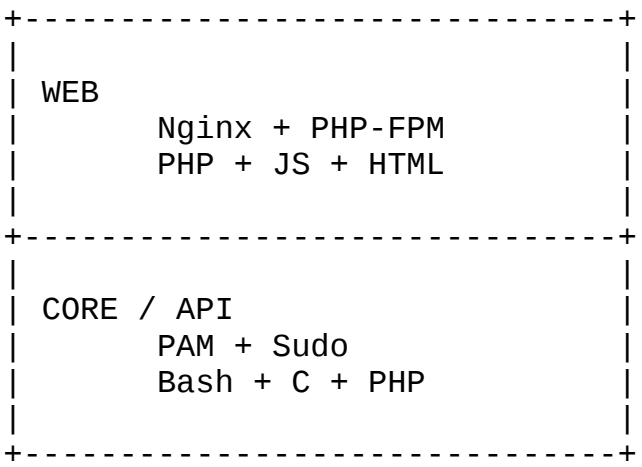
SIMPLE & CLEVER HOSTING CONTROL PANEL

VESTA CONTROL PANEL

While Vesta is arguably one of the most popular open source control panel, its history is actually quite short considering the timeline of control panels. It all started back in 2010 as small project for our own use. In 2012 we have publicly released an initial version. Now we are registering about 15k of new installations each month.

ARHHITECTURE

Vesta Control Panel is a stack of software components which is roughly divided into two main layers as shown below in the architecture diagram.



At the bottom of the layers is mainly Bash. It covers all the essential calls for hosting management needs. Bash scripts provide a level of abstraction between the OS and system services.

However, some scripts are written in C and PHP. Especially when it comes to password hashing or mailing functions. This layer itself is independent and can used as fully featured CLI interface.

Scripts are located in the /usr/local/vesta/bin directory and should be executed by root. Since root user have PATH variable configured it can run command without defining full path

```
[root@r7 ~]# v-add-user
Usage: v-add-user USER PASSWORD EMAIL [PACKAGE] [FNAME] [LNAME]
```

```
[root@r7 ~]# v-add-web-domain
Usage: v-add-web-domain USER DOMAIN IP [RESTART] [ALIASES] [PROXY_EXTENTIONS]
```

```
[root@r7 ~]# v-add-database
Usage: v-add-database USER DATABASE DBUSER DBPASS [TYPE] [HOST] [CHARSET]
```

As you can see we follow specific name convention. Every single script starts with v prefix this way we separate system calls like adduser from vesta calls. Then we define action add, delete, change, list and so on and then objects. Sometime object can be nested like domain for example because there are dns,web and mail domains too. It helps to get used to CLI and API.

On top of Bash scripts there is a set of rules and some logic imposed by PAM and Sudo. Web stack (nginx+php fpm) is running under admin user account. This user account is not privileged and it needs sudo to run vesta commands. However admin can execute any script that is located in /usr/local/vesta/bin via sudo without password.

PHP scripts are located in the /usr/local/vesta/web directory and follow almost the same convention. Index.php script in add/user directory creates new user accounts. Script that is located in delete/user supposed to delete it. There is very thin layer of logic in php script. They are basically wrappers for shell scripts.

When you want to create new user via CLI you run:

```
[admin@r7 ~]# sudo /usr/local/vesta/bin/v-add-user demo s3cp4ssw0rd demo@gmail.com
```

When you want to do the same via web stack you should write something like this:

```
<?php

exec ('sudo /usr/local/vesta/bin/v-add-user demo s3cp4ssw0rd demo@gmail.com')

?>
```

We obey UNIX philosophy that states that when a program has nothing surprising, interesting or useful to say, it should say nothing. Every vesta command returns an exit status (sometimes referred to as a return status or exit code). A successful command returns a 0, while an unsuccessful one returns a non-zero value that usually can be interpreted as an error code.

VALUE	NAME	COMMENT
0	OK	Command has been successfully executed
1	E_ARGS	Not enough arguments provided
2	E_INVALID	Object or argument is not valid
3	E_NOTEXIST	Object does not exist
4	E_EXISTS	Object already exists
5	E_SUSPENDED	Object is suspended
6	E_UNSUSPENDED	Object is already unsuspended
7	E_INUSE	Object can't be deleted because is used by the other object
8	E_LIMIT	Object cannot be created because of hosting package limits
9	E_PASSWORD	Wrong or short password
10	E_FORBIDEN	Object cannot be accessed by the user
11	E_DISABLED	Subsystem is disabled
12	E_PARSING	Configuration is broken
13	E_DISK	Not enough disk space to complete the action
14	E_LA	Server is to busy to complete the action
15	E_CONNECT	Connection failed / Host is unreachable
16	E_FTP	FTP server is not responding
17	E_DB	Database server is not responding
18	E_RRD	RRDtool failed to update the database
19	E_UPDATE	Update operation failed
20	E_RESTART	Service restart failed

PHP code should be aware of this

```
exec ('/usr/bin/sudo /usr/local/vesta/bin/v-add-user demo p4ssw0rd
demo@gmail.com', $output, $return_var);

if ($return_var != 0) {
    echo 'Error code: ', $return_var;
    exit($return_var);
}
```

If you want to fetch object information you should use one of v-list-* commands. Vesta currently supports several output formats: shell, raw, plain, csv, json. JSON format is preferred for web communication.

```
[root@r7 ~]# v-list-user
Usage: v-list-user USER [FORMAT]

[root@r7 ~]# v-list-user admin json
{
        "admin": {
                "FNAME": "System",
                "LNAME": "Administrator",
                "PACKAGE": "default",
                "WEB_TEMPLATE": "default",
                "BACKEND_TEMPLATE": "default",
                "PROXY_TEMPLATE": "default",
                "DNS_TEMPLATE": "default",
                "WEB_DOMAINS": "100",
                "WEB_ALIASES": "100",
                "DNS_DOMAINS": "100",
                "DNS_RECORDS": "100",
                "MAIL_DOMAINS": "100",
                "MAIL_ACCOUNTS": "100",
                "DATABASES": "100",
                "CRON_JOBS": "100",
                "DISK_QUOTA": "10000",
                "BANDWIDTH": "100000",
                "NS": "ns1.localhost.ltd, ns2.localhost.ltd",
                "SHELL": "bash",
                "BACKUPS": "3",
                "CONTACT": "admin@r7.vestacp.com",
                "CRON_REPORTS": "yes",
                "RKEY": "gHbppf4poe",
                "SUSPENDED": "no",
                "SUSPENDED_USERS": "0",
                "SUSPENDED_WEB": "0",
                "SUSPENDED_DNS": "0",
                "SUSPENDED_MAIL": "0",
                "SUSPENDED_DB": "0",
                "SUSPENDED_CRON": "0",
                "IP_AVAIL": "3",
                "IP_OWNED": "3",
                "U_USERS": "1",
                "U_DISK": "2",
                "U_DISK_DIRS": "1",
                "U_DISK_WEB": "1",
                "U_DISK_MAIL": "0",
                "U_DISK_DB": "0",
                "U_BANDWIDTH": "0",
                "U_WEB_DOMAINS": "1",
                "U_WEB_SSL": "0",
                "U_WEB_ALIASES": "1",
                "U_DNS_DOMAINS": "1",
                "U_DNS_RECORDS": "11",
                "U_MAIL_DOMAINS": "1",
                "U_MAIL_DKIM": "0",
                "U_MAIL_ACCOUNTS": "1",
                "U_DATABASES": "2",
                "U_CRON_JOBS": "7",
                "U_BACKUPS": "3",
                "LANGUAGE": "en",
                "HOME": "/home/admin",
                "NOTIFICATIONS": "yes",
                "TIME": "21:39:41",
                "DATE": "2015-07-04"
        }
}
```

```
<?php
define ('VESTA_CMD', '/usr/bin/sudo /usr/local/vesta/bin/');
$user = 'demo';
exec (VESTA_CMD . "v-list-user " . $user . " json", $output, $return_var);
if ($return_var != 0) {
    echo 'Error code:',$return_var;
    exit($return_var)
}
$data = json_decode(implode('', $output), true);
echo 'email: '.$data[$v_username]['CONTACT'];
?>
```

GENERAL DESIGN GUIDELINES

1. Please consider reading the link below about BASH Conding convention
https://github.com/serghey-rodin/vesta/blob/master/src/bash_coding_style.txt

2. Please store your code in /usr/local/vesta/web/inc directory and since we don't have centralized router make handlers in /usr/local/vesta/web/add/app, /usr/local/vesta/web/edit/app, /usr/local/vesta/web/delete/app and other relevant directories.

3. Template files should be stored in /usr/local/vesta/web/templates

4. We don't use any frameworks but if you are please feel free to apply necessary changes in hierarchy described above.

5. Vesta supports internationalization (i18n). Please use existing word-base in /usr/local/vesta/web/inc/i18n/en.php and add missing keys.

REQUESTED API CALLS

1. User data : Logged in User's email, domain, homedir, username

```
<?php

// Get username and protect resource for unauthorized use
include ($_SERVER['DOCUMENT_ROOT'] . '/inc/main.php');

// Print user name
echo $user;

// Get user parameters
exec (VESTA_CMD . "v-list-user " . escapeshellarg($user) . " json", $output,
$return_var);
```

```
// Check command status
check_return_code ($return_var, $output);

// Parse json array
$data = json_decode (implode('', $output), true);

// Get email
$email = $data[$user]['CONTACT'];

// Get current domain
// GET for edits and deletion POST for adding and updates
$domain = escapeshellarg($_GET['domain']);

// Get home directory
$home = $data[$user]['HOME'];
```

2. Domains list : List of domains owned by the user and the path to those domains

```
<?php

// Get username and protect resource for unauthorized use
include ($_SERVER['DOCUMENT_ROOT'] . '/inc/main.php');

// Get user domains
exec (VESTA_CMD . "v-list-web-domains " . escapeshellarg($user) . " json",
$output, $return_var);

// Check command status
check_return_code ($return_var, $output);

// Parse json array
$data = json_decode (implode('', $output), true);

// Print user domains
foreach ($data as $key => $value) {
    $domain = $key;
    echo 'Domain: ' . $domain;
    echo 'Path: ' . $home . '/' . $user . '/web/' . $domain . '/public_html';
}
```

3. Space left : The Disk space remaining for the user

```
<?php

// Get username and protect resource for unauthorized use
include ($_SERVER['DOCUMENT_ROOT'] . '/inc/main.php');
```

```
// Get user parameters
exec (VESTA_CMD . "v-list-user " . escapeshellarg($user) . " json", $output,
$return_var);

// Check command status
check_return_code ($return_var, $output);

// Parse json array
$data = json_decode (implode('', $output), true);

// Define disk space limit in Mb
$disk_quota = $data[$user]['DISK'];

// Define current disk space usage in Mb
$disk_used = $data[$user]['U_DISK'];

// Check remaining disk space
if ($disk_quota != 'unlimited') {
    $disk_free = $disk_quota - $disk_used;
    if ($disk_free < 10) {
        echo 'Error: not enough disk space to complete operation';
        exit(8);
    }
}
```

4. Database host : If you support remote database host we will need API to determine the database host the user is using. If you support only localhost we will not need this

There are four possible use cases:

>>> disabled
Please note that Vesta Control Panel supports MySQL, PostgreSQL, MongoDB and DB-less stacks.

```
<?php

// Get environment variables and protect resource for unauthorized use
include ($_SERVER['DOCUMENT_ROOT'] . '/inc/main.php');

// Define available database types
$database_types = split (",",$_SESSION['DB_SYSTEM']);
if (!in_array("mysql", $database_types)) {
    echo 'Error: MySQL subsystem is disabled';
    exit(11);
}
```

>>> default
Local MySQL server. Wast majority of installations are using it this way.

```
[root@r7 ~]# v-list-database-hosts mysql json
{
      "localhost": {
            "CHARSETS": "UTF8,LATIN1,WIN1251,WIN1252,WIN1256,WIN1258,KOI8",
            "MAX_DB": "500",
            "U_SYS_USERS": "0",
            "U_DB_BASES": "0",
            "TPL": "",
            "SUSPENDED": "no",
            "TIME": "02:08:13",
            "DATE": "2016-02-18"
        }
}
```

>>> sharding
Two small vps could cost you less than one powerful. It is also quite flexible since you can upgrade them separately depending on application demands and bottlenecks. So the idea is that you can run Nginx with php on one server and use MySQL on another.

```
<?php

// Get environment variables and protect resource for unauthorized use
include ($_SERVER['DOCUMENT_ROOT'] . '/inc/main.php');

// Get database hostname
exec (VESTA_CMD."v-list-database-hosts mysql json'", $output, $return_var);

// Parse json array
$data = json_decode (implode('', $output), true);

// Print database hostname
echo 'Hostname: ' . key ($output);
```

>>> cluster
This is kind of setup was created specifically for one big customer. Chances that someone else is using this implementation are quite small.

```
<?php

// Get environment variables and protect resource for unauthorized use
include ($_SERVER['DOCUMENT_ROOT'] . '/inc/main.php');

// Get database hostname
exec (VESTA_CMD."v-list-database-hosts mysql json'", $output, $return_var);

// Parse json array
$data = json_decode (implode('', $output), true);
```

```
// List database hostnames
foreach ($data as $key => $value) {
    echo 'Hostname: ' . $key;
}
```

5. Maximum databases : Maximum databases the user can create

```
<?php

// Get username and protect resource for unauthorized use
include ($_SERVER['DOCUMENT_ROOT'] . '/inc/main.php');

// Get user parameters
exec (VESTA_CMD . "v-list-user " . escapeshellarg($user) . " json", $output,
$return_var);

// Check command status
check_return_code ($return_var, $output);

// Parse json array
$data = json_decode (implode('', $output), true);

// Define database limit
$database_quota = $data[$user]['DATABASES'];

// Define current database usage counter
$database_used = $data[$user]['U_DATABASES'];

// Check remaining
if ($database_quota != 'unlimited') {
    $database_free = $database_quota - $database_used;
    if ($database_free < 1) {
        echo 'Error: database limit is reached please upgrade hosting package';
        exit(8);
    }
}
```

6. Database list : The list of existing databases of the user

```
<?php

// Get username and protect resource for unauthorized use
include ($_SERVER['DOCUMENT_ROOT'] . '/inc/main.php');

// Get user databases
exec (VESTA_CMD . "v-list-databases " . escapeshellarg($user) . " json", $output,
$return_var);

// Check command status
check_return_code ($return_var, $output);
```

```
// Parse json array
$data = json_decode (implode('', $output), true);

// Print user databases
foreach ($data as $key => $value) {
    echo 'Database: ' . $key;
    echo 'Type: ' . $data[$key]['TYPE'];
    echo 'Hostname: ' . $data[$key]['HOST'];
    echo 'Username: ' . $data[$key]['DBUSER'];
}
```

7. Database users list : The list of existing databases users of the user

Vesta Control Panel currently support only one user per database. We don't plan to implement multi-user support in near future, but if this is critical for you please let us know. We will rearrange our roadmap.

```
<?php

// Get username and protect resource for unauthorized use
include ($_SERVER['DOCUMENT_ROOT'] . '/inc/main.php');

// Get current database
$database = $_GET['database'];

// Get database parameters
exec (VESTA_CMD . "v-list-database " . escapeshellarg($user) . " " .
escapeshellarg($datatbase) . " json", $output, $return_var);

// Check command status
check_return_code ($return_var, $output);

// Parse json array
$data = json_decode (implode('', $output), true);

// Print user databases
echo 'Database: ' . $database;
echo 'Type: ' . $data[$database]['TYPE'];
echo 'Hostname: ' . $data[$database]['HOST'];
echo 'Username: ' . $data[$database]['DBUSER'];
```

8. Create Database : Create a database

Arguments in the square brackets could be omitted. It means that you can rely on default values and in 99.999% of cases it will work just fine. I guess for first release this is sort of OK

```
[root@r7 ~]# v-add-database
Usage: v-add-database USER DATABASE DBUSER DBPASS [TYPE] [HOST] [CHARSET]
```

Anyway here is the full coverage:

```
<?php

// Get environment variables and protect resource for unauthorized use
include ($_SERVER['DOCUMENT_ROOT'] . '/inc/main.php');

// Define available database types
$database_types = split (",",$_SESSION['DB_SYSTEM']);
if (!in_array("mysql", $database_types)) {
    echo 'Error: MySQL subsystem is disabled';
    exit(11);
}

// Get database hostname
exec (VESTA_CMD."v-list-database-hosts mysql json'", $output, $return_var);

// Parse json array
$data = json_decode (implode('', $output), true);

// Define database hostname
$db_host = key ($output);

// In this example $user = 'admin'

// Define database name
$db_name = 'main';

// Define database user
$db_user = 'user';

// Define database password
$db_pass = 'p4sw0rd';

// Create database
exec (VESTA_CMD . "v-add-database " . $user . " " . $db_name . " " . $db_user . "
" . $db_pass . " mysql " . $db_host, $output, $return_var);

// Check result
check_return_code($return_var,$output);

// If everything is ok control panel will create database admin_main
// user admin_user with password p4sword
// As you may notice database names and user names are prefixed
// based on username
```

9. Create Database User : Create a database user

Vesta Control Panel currently support only one user per database. We don't plan to implement multi-user support in near future, but if this is critical for you please let us know. However you can replace existing database user using next call:

```
v-change-database-user USER DATABASE DBUSER [DBPASS]
```

This call call can be also used to reset dbuser's password. Just pass exiting username and new password control panel will change only password.

10. Delete Database : Delete Database

```
<?php

// Get environment variables and protect resource for unauthorized use
include ($_SERVER['DOCUMENT_ROOT'] . '/inc/main.php');

// Define database name
$database = escapeshellarg($_GET['database']);

// Delete database
exec (VESTA_CMD . "v-delete-database " . $user . " " . $db_name, $output,
$return_var);
```

11. Delete Database user : Delete database user

Vesta Control Panel currently support only one user per database. We don't plan to implement multi-user support in near future, but if this is critical for you please let us know. Database user will be deleted automatically on database deletition.

12. Cron jobs list : List of Cron jobs of the user

```
<?php

// Get username and protect resource for unauthorized use
include ($_SERVER['DOCUMENT_ROOT'] . '/inc/main.php');

// Get user cron jobs
exec (VESTA_CMD . "v-list-cron-jobs" . escapeshellarg($user) . " json", $output,
$return_var);

// Parse json array
$data = json_decode (implode('', $output), true);

// Print user cron jobs
foreach ($data as $key => $value) {
    echo 'ID: ' . $key;
    echo 'Minute: ' . $data[$key]['MIN'];
    echo 'Hour: ' . $data[$key]['HOUR'];
    echo 'Day: ' . $data[$key]['DAY'];
    echo 'Month: ' . $data[$key]['MONTH'];
    echo 'Day of the week: ' . $data[$key]['WDAY'];
    echo 'Command: ' . $data[$key]['CMD'];
}
```

13. Add a Cron Job : Adding a CRON job for the user

```
<?php

// Get environment variables and protect resource for unauthorized use
include ($_SERVER['DOCUMENT_ROOT'] . '/inc/main.php');

// Define schedule
$min = '*';
$hour = '*';
$week = '*';
$month = '*';
$wday = '*';
$cmd = 'echo test';

// Add cron job
exec (VESTA_CMD."v-add-cron-job ".$user." ".$min." ".$hour." ".$day." ".$month."
".$wday." ".$cmd, $output, $return_var);
```

14. Delete a Cron Job : Deleting a CRON job for the user

```
<?php

// Get environment variables and protect resource for unauthorized use
include ($_SERVER['DOCUMENT_ROOT'] . '/inc/main.php');

// Define cron job id
$job = escapeshellarg($_GET['job']);

// Delete cron job
exec (VESTA_CMD."v-delete-cron-job ".$user." ".$job, $output, $return_var);
```