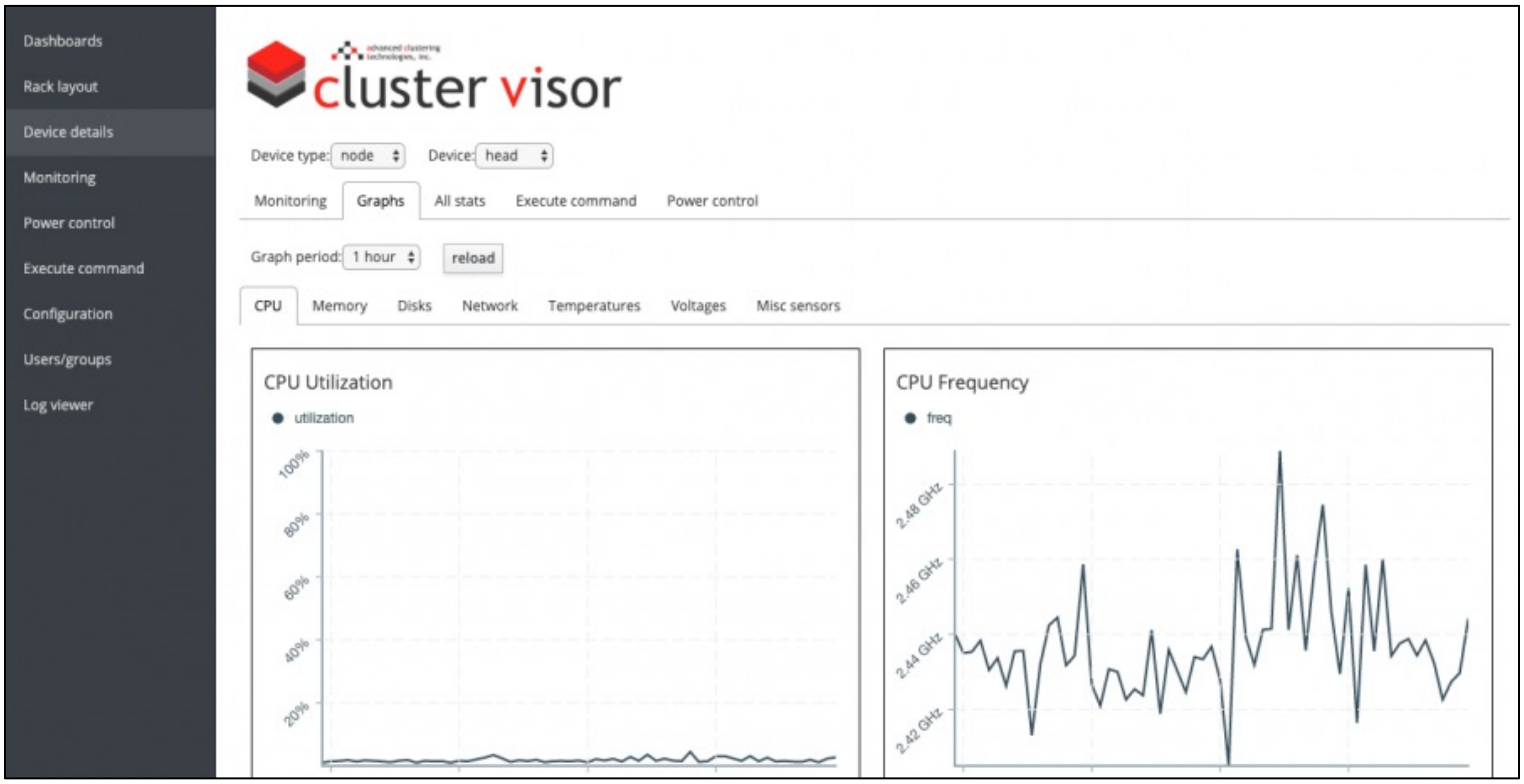
advanced clustering technologies, inc.

clustervisor MANUAL

User's Guide for Understanding and Utilizing our Cluster Management Solution



Updated April 2021

advanced clustering technologies, inc.

(866) 802-8222
advancedclustering.com

1 1. Accessing ClusterVisor

There are two ways to interact with ClusterVisor:

- Using its command line utilities
- Through a web browser

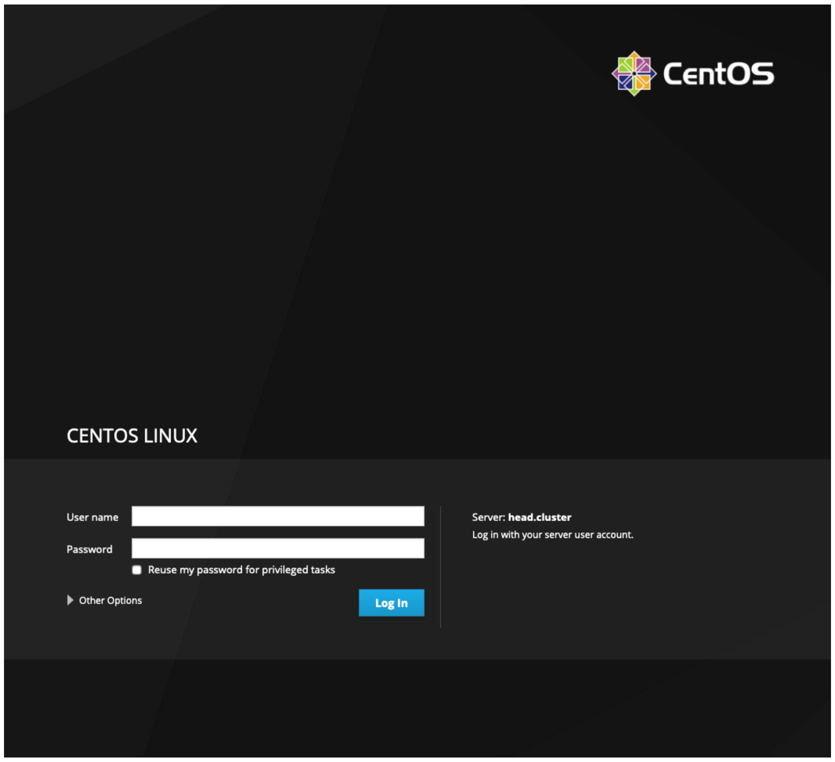
Getting to the command line utilities is as simple as logging into one of the nodes on the cluster, and the utilities will all begin with the prefix "cv-" (e.g. cv-conf and cv-commit). By default, these will be installed under /usr/bin and /usr/sbin.

Accessing the web interface from a web browser can be done by:

1. Obtaining the IP address of the node running the ClusterVisor server (this will typically be the head node). If this is not known ahead of time, it can be obtained from the command line by running:

   ```
   $ ip a
   ```

2. From a web browser, navigate to https://<*IPADDRESS*>:9090/clustervisor where <*IPADDRESS*> will be replaced with the IP address from the previous step.
3. It will then present a login prompt that can be logged in using the username and password of any account on the node.



Server: **head.cluster**
Log in with your server user account.

From here, everything in ClusterVisor can be accessed from this page using the links in the sidebar (e.g. Dashboard, Rack layout, Device details, etc). Do note that while a non-root user can log into ClusterVisor, only the root user and admin users will be able to perform any actions while all other users will only be able to view the allowed pages (for more on this see section 6. Managing ClusterVisor permissions(see page 29) of this guide).

1.1 Using the command line utilities

The web interface maximizes ClusterVisor's usefulness by being more user-friendly, so this guide primarily focuses on accessing ClusterVisor via the web web interface. Nonetheless, the command line alternative is shown at the end of each guide section.

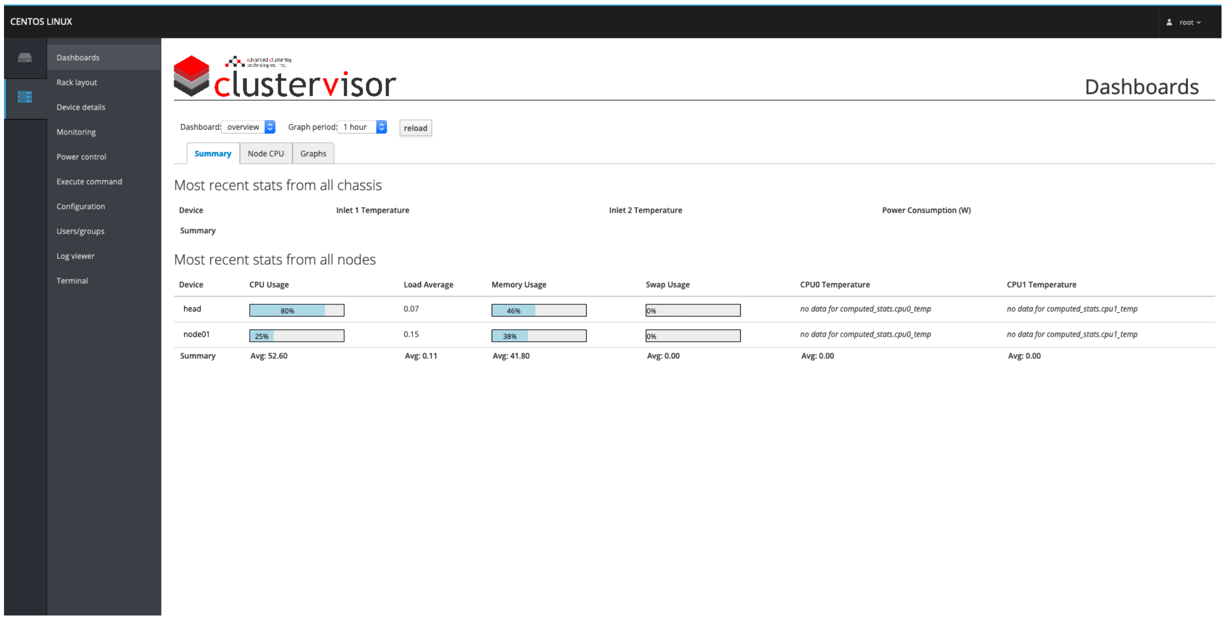
All command line utilities used for managing and/or monitoring the nodes will share the same flags for selecting the desired nodes:

- **--nodes** <*nodenames*> - A list of nodes, used to select individual nodes
- **--groups** <*groups*> - A list of node groups, used to select nodes in bulk by a group they are in
- **--exclude** <*nodenames*> - A list of nodes, used to exclude individual nodes from the selection
- **--racks** <*racks*> - A list of racks, used to select all nodes in a specific rack
- **--chassis** <*chassis*> - A list of chassis, used to select all nodes in a specific chassis
- **--all** - Will select all nodes known by ClusterVisor
- **--config-file** <*config_file*> - A config file holding all of the node names desired for the selection

With this in mind, for any section below mentioning that a command line utility can be used to select a node, this is how to do so.

2 2. Monitoring the cluster

2.1 Node stats



Dashboard: overview   Graph period: 1 hour

Most recent stats from all chassis

Device	Inlet 1 Temperature	Inlet 2 Temperature	Power Consumption (W)
Summary			

Most recent stats from all nodes

Device	CPU Usage	Load Average	Memory Usage	Swap Usage	CPU0 Temperature	CPU1 Temperature
head	80%	0.07	46%	0%	*no data for computed_stats.cpu0_temp*	*no data for computed_stats.cpu1_temp*
node01	25%	0.15	38%	0%	*no data for computed_stats.cpu0_temp*	*no data for computed_stats.cpu1_temp*
Summary	Avg: 52.60	Avg: 0.11	Avg: 41.80	Avg: 0.00	Avg: 0.00	Avg: 0.00

When ClusterVisor is first opened, it will display the **Dashboard** page that provides a way to monitor the general state of the cluster at a glance. Under the **Summary** tab on this page, all of the nodes in the cluster will be listed alongside their most recent stats. If any of the nodes stats have crossed a critical threshold, the node's name in the **Device** column will be highlighted in red to make problematic nodes easy to distinguish among the other nodes.

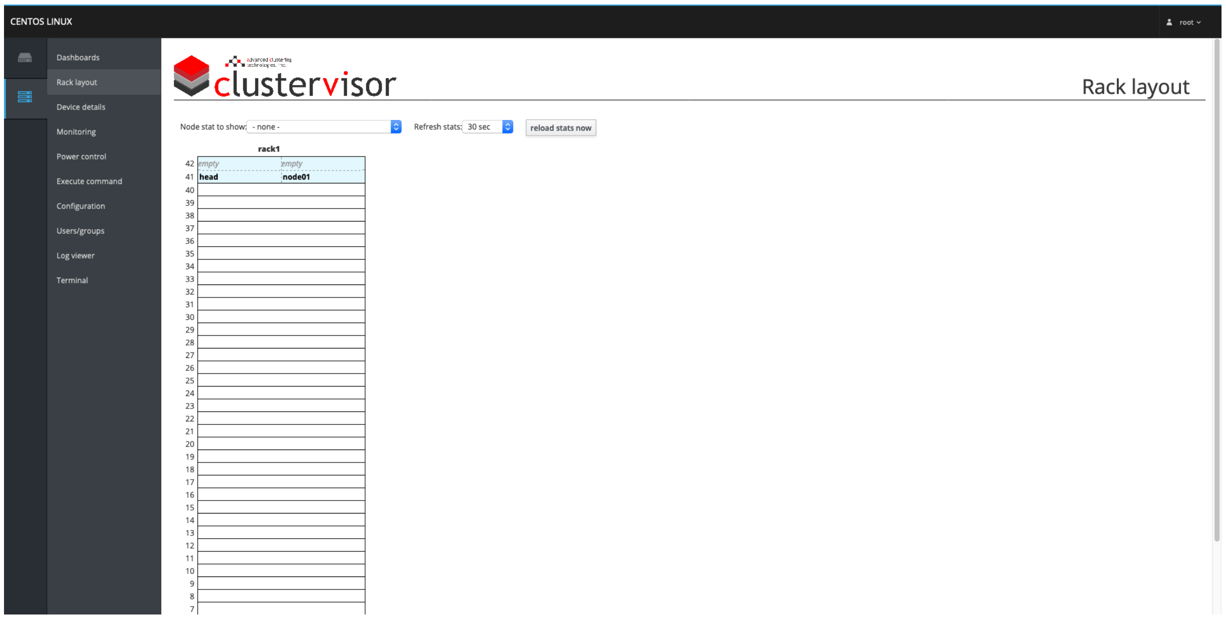
If a node is highlighted in red to indicate an issue has occurred, the cause can be determined by clicking on the highlighted device name. This will present a pop-up showing more details about the selected node. The first tab, **Monitoring**, presents the monitored stats along with indicating which one crossed its threshold under the **Status** column. The other tabs for monitoring the node are the following:

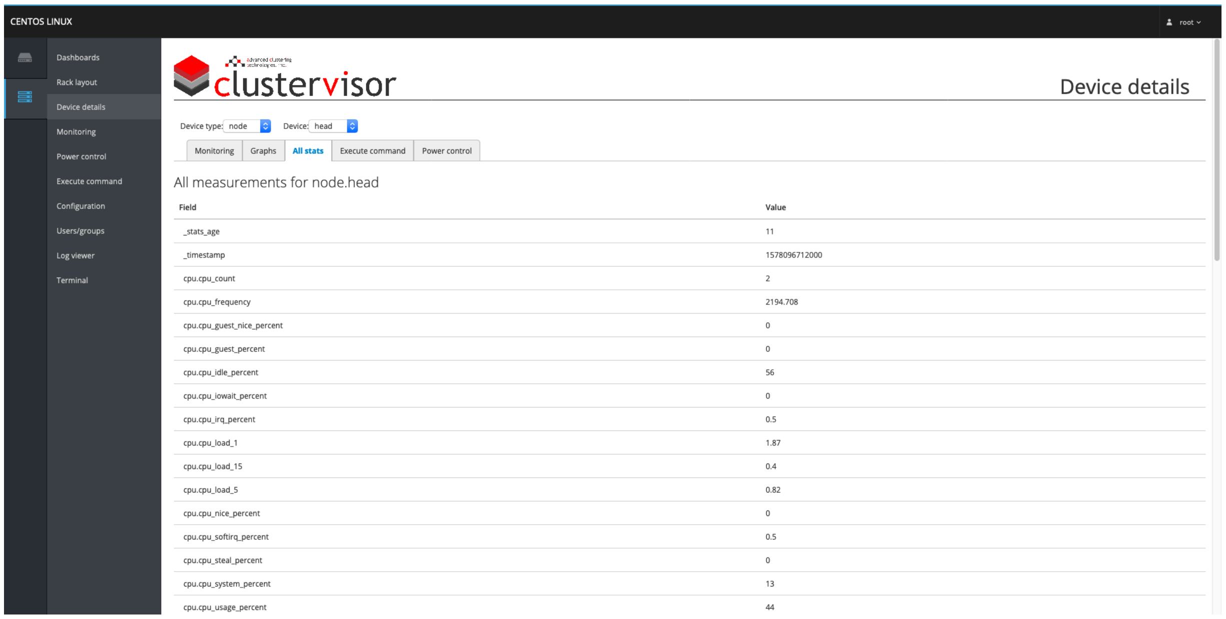
- **Graphs** - Used to display graphs of various statistics of the node over time. These statistics include CPU utilization, disk usage, and network I/O.
- **All stats** - This lists all of the current values of the stats that ClusterVisor collects.

ⓘ The names of the stats are the ones used in the Influx[1] database and are also names that should be used later on for creating "Monitoring Rules" to get notified about specific events (which can be done from the **Configuration** page under the **Monitoring Rules** tab, see more about this from section Configurations in ClusterVisor(see page 0) of this guide).

This same window is not just accessible from the **Dashboard** page, but can also be accessed from the **Rack layout** page by clicking on the names of the nodes presented or from the **Device detail** page by selecting the desired node from the **Device** drop-down menu. As well, the status of the monitored stats under the **Monitoring** tab mentioned earlier can be found on the **Monitoring** page, but will display the status for all the nodes in the cluster.

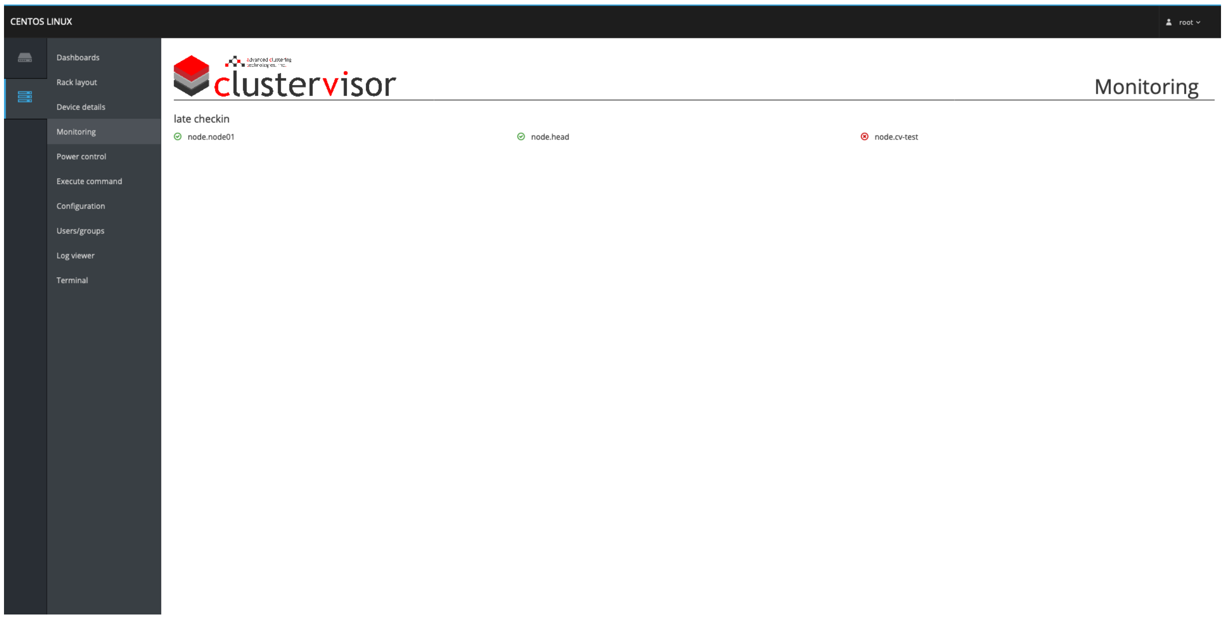
1 https://docs.influxdata.com/





All measurements for node.head

Field	Value
_stats_age	11
_timestamp	1578096712000
cpu.cpu_count	2
cpu.cpu_frequency	2194.708
cpu.cpu_guest_nice_percent	0
cpu.cpu_guest_percent	0
cpu.cpu_idle_percent	56
cpu.cpu_iowait_percent	0
cpu.cpu_irq_percent	0.5
cpu.cpu_load_1	1.87
cpu.cpu_load_15	0.4
cpu.cpu_load_5	0.82
cpu.cpu_nice_percent	0
cpu.cpu_softirq_percent	0.5
cpu.cpu_steal_percent	0
cpu.cpu_system_percent	13
cpu.cpu_usage_percent	44



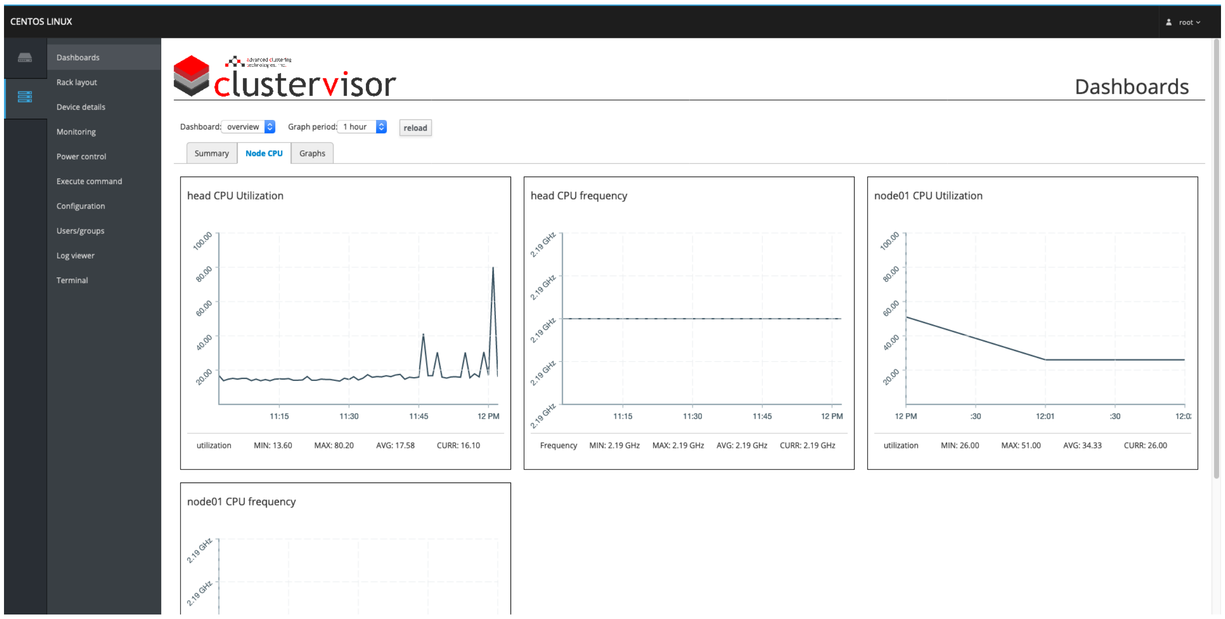
late checkin

node.node01 node.head node.cv-test

2.1.1 Using the command line

While the graphs are exclusively available in the web interface, the stats for the nodes can be queried from the command line using cv-stats. By default, the stats of the node(s) will be listed in brackets with the node names listed below beside the value of each stat. As well, the status of the monitored stats can be queried from the command line using cv-monitoring, and is displayed by default in the same form as cv-stats.

2.2 Cluster stats



Dashboard: overview Graph period: 1 hour

head CPU Utilization

utilization MIN: 13.60 MAX: 80.20 AVG: 17.58 CURR: 16.10

head CPU frequency

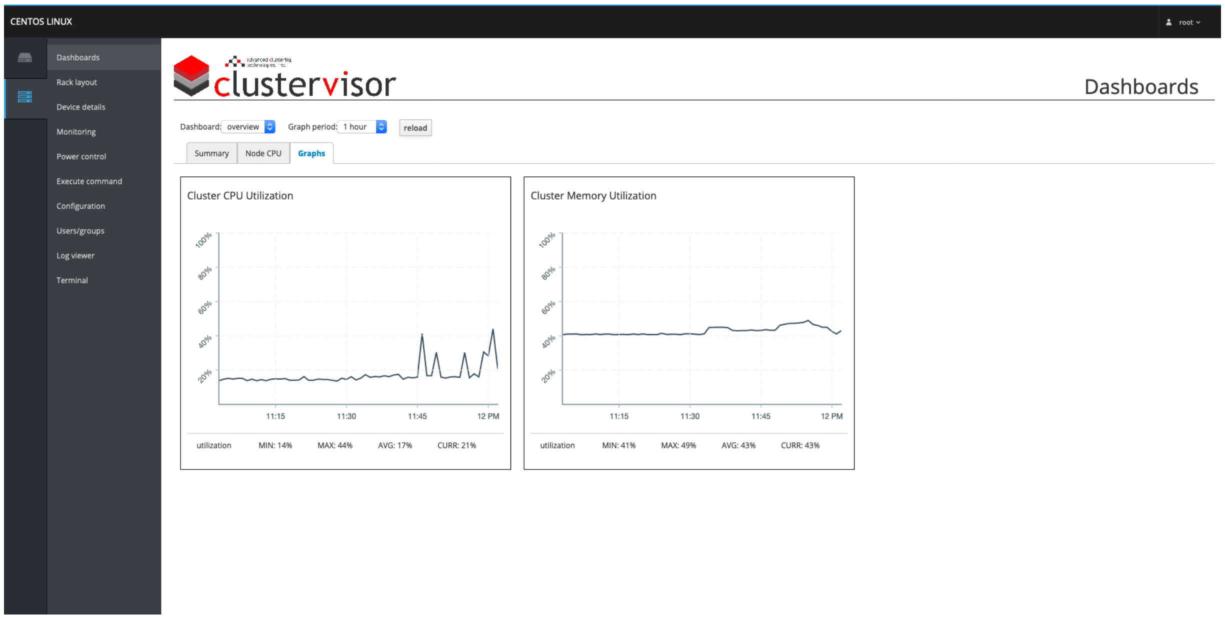
Frequency MIN: 2.19 GHz MAX: 2.19 GHz AVG: 2.19 GHz CURR: 2.19 GHz

node01 CPU Utilization

utilization MIN: 26.00 MAX: 51.00 AVG: 34.33 CURR: 26.00

node01 CPU frequency

The **Summary** tab on the **Dashboards** page will display information and graphs about the nodes. The other tabs on the page provide graphs about the cluster as a whole. The **Graphs** tab will show graphs about the utilization of hardware across the cluster over time, while the **Node CPU** tab will provide a breakdown on each node's CPU stats so they all can be viewed together on the same page.



These are just the fields provided in ClusterVisor by default, but the **Dashboards** page can be customized to change the graphs that are presented and add new tabs. This can done by navigating to the **Configuration** page and selecting the **Dashboards** tab. Further explanations on how to use the **Configuration** page can be found under section Configurations in ClusterVisor(see page 0) of this guide.

2.2.1 Using the command line

The dashboards can be customized through the command line using cv-conf. The default dashboard can be modified by running:

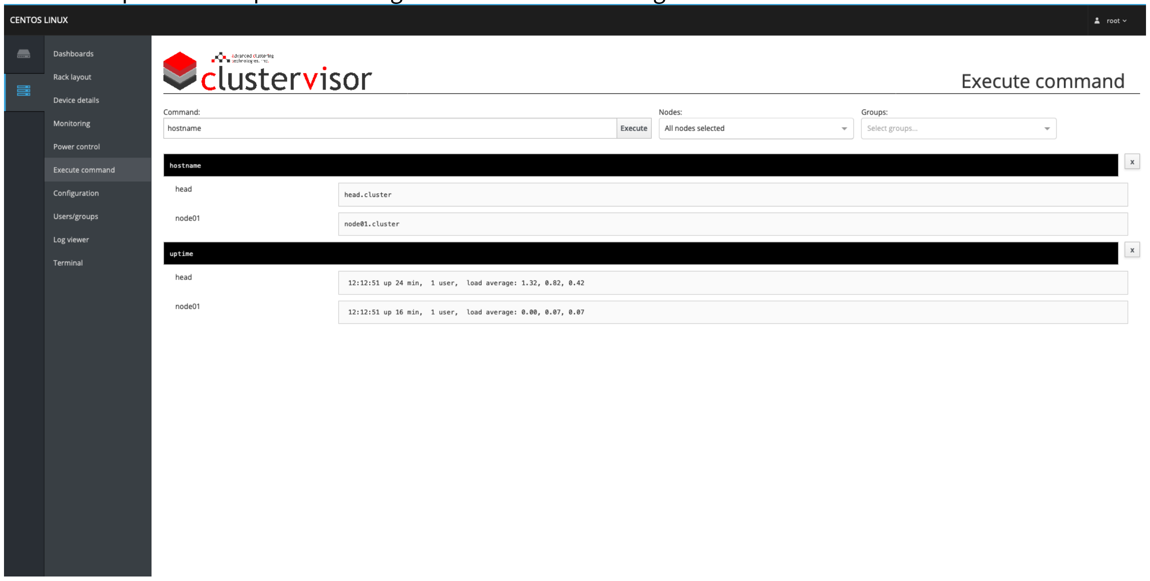
```
$ cv-conf -e dashboard.overview
```

For more information about using cv-conf, see section Configurations in ClusterVisor(see page 0) of this guide under its "Using the command line".

3 3. Remote management

The previous section mentions the tabs for monitoring the node. There are other tabs that provide controls for remotely managing the selected node along with any other node managed by ClusterVisor in the cluster. Those tabs are the following:

- **Execute command** - Provides a way to quickly run a command on itself and/or multiple other nodes and get back the result of the command. The nodes can be selected individually by name, as well as in bulk using the nodes' group name. The results will all be listed with the most recently executed command at the top and with all the previous outputs following beneath it in descending order.



Command: hostname

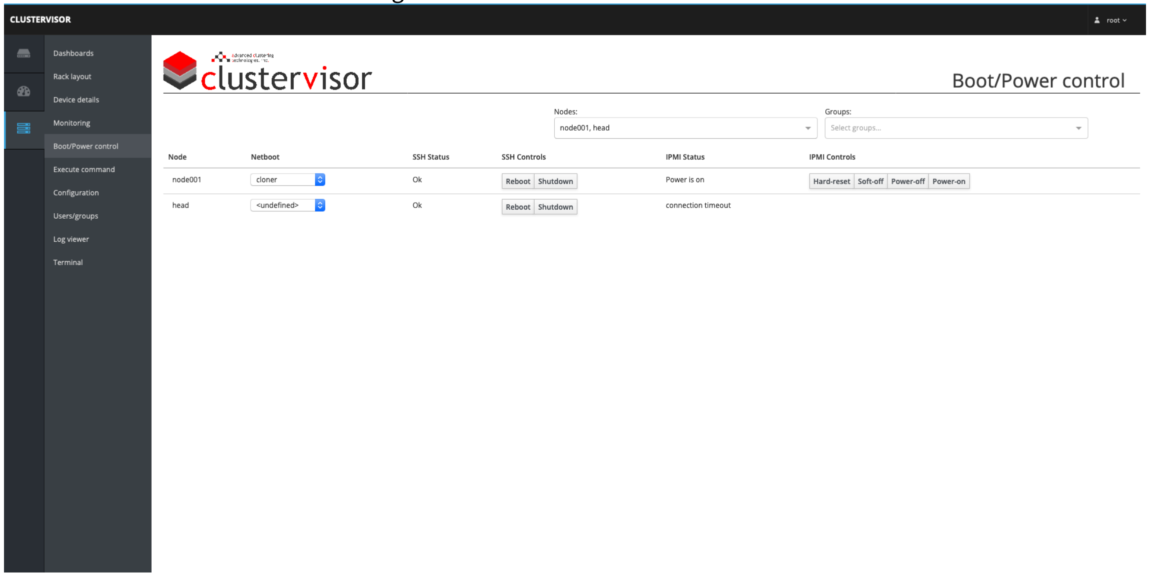
Nodes: All nodes selected

Groups: Select groups...

```
hostname
head      head.cluster
node01    node01.cluster

uptime
head      12:12:51 up 24 min,  1 user,  load average: 1.32, 0.82, 0.42
node01    12:12:51 up 16 min,  1 user,  load average: 0.00, 0.07, 0.07
```

- **Boot/Power control** - Similar to **Execute command**, nodes can be selected individually by name and/or in bulk using group names. However, instead of executing commands on the nodes, this interface provides the ability to set a node's network boot image and/or control the power of the node through SSH or IPMI as well as view if the node is accessible through SSH or IPMI.



Nodes: node001, head

Groups: Select groups...

Node	Netboot	SSH Status	SSH Controls	IPMI Status	IPMI Controls
node001	cloner	Ok	Reboot Shutdown	Power is on	Hard-reset Soft-off Power-off Power-on
head	<undefined>	Ok	Reboot Shutdown	connection timeout	

> These controls can also be accessed from the **Execute command** page and **Boot / Power control** page in ClusterVisor.

For **Boot / Power control**, the distinction between using SSH or IPMI is that SSH requires the node to be online and can only power off or restart the node, while IPMI can power on, power off, and restart the node regardless of if the node is powered off or powered on. However, also keep in mind that the power controls through SSH are the safe means of powering down since it is being done through the OS, while the "Power Off" option from IPMI is a "hard" shutdown that is equivalent to pulling the power from the node (although this may be needed if the node becomes completely unresponsive). However, the "Soft Off" IPMI option from IPMI will deliver the equivalent of once pressing the power off button on the node to instruct it to begin shutting down, which would be useful to safely power off the node where it is not locked but also cannot be reached via SSH.

Do note that the IPMI power controls may take a few seconds to show up, since the test for IPMI connectivity is not immediate. However, the controls have not shown up after a minute has passed, this likely means that IPMI is either not setup on the node or is no longer working properly. If that is the case, see the **Troubleshooting** section below for how to handle that issue.

For changing the node's network boot image, select the desired image under the node's **Netboot** column and it will be used the next time the node is rebooted. To view or change the available network boot images navigate to the **Netboot** tab under the **Configuration** page (for more on this see section 4. Configuring nodes(see page 9) of this guide).

There are two other places in ClusterVisor that can be used to remotely manage the cluster, the **Log viewer** page and the **Terminal** page. The former can be used to view various log files of nodes, while the latter provides a terminal window to run commands on the node in real-time (this being different from **Execute command** because the commands can be run interactively and use tools like *vim* to edit files). Both pages provide a drop-down menu to select the desired node to interact with. As well, the **Log viewer** page contains the log files for ClusterVisor, so it can be used for debugging if something is not working properly.

3.1 Using the command line

The command line utilities responsible for executing commands and power control are cv-exec and cv-power.

The cv-exec command will execute any shell command provided to it onto the selected nodes and will display their outputs with each node name in brackets with the output of the command listed below. If any commands ran into an error, beneath all of the outputs will be a separate section with the node name and errors listed together.

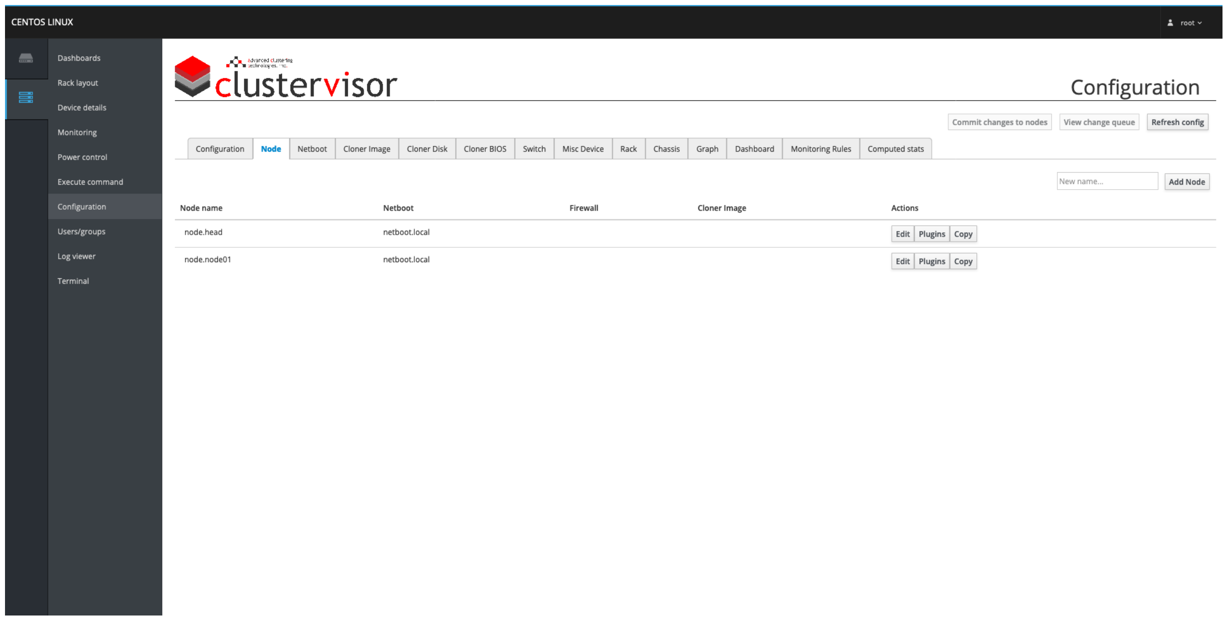
The cv-power command can access the current power status of the selected nodes, how long the nodes have been online, why the nodes were last restarted, and controls for remotely powering on/off the nodes. This is done through IPMI, so the same concerns to keep in mind for hard powering off the nodes still apply, but the "soft-off" option explained above is still available.

The cv-netboot command can view or change the network boot image of the selected nodes.

Since the cv-exec internally communicates to the nodes through SSH, it would also be the utility used to perform the same SSH power control actions as the web interface. This would be accomplished by passing the nodes either the "shutdown", "poweroff", or "reboot" command.

Lastly, the log files for ClusterVisor by default are stored under /var/log/clustervisor. Each component is listed as a separate ".log" file and can be viewed like any other log file from the command line using utilities like cat, tail, less, and so on.

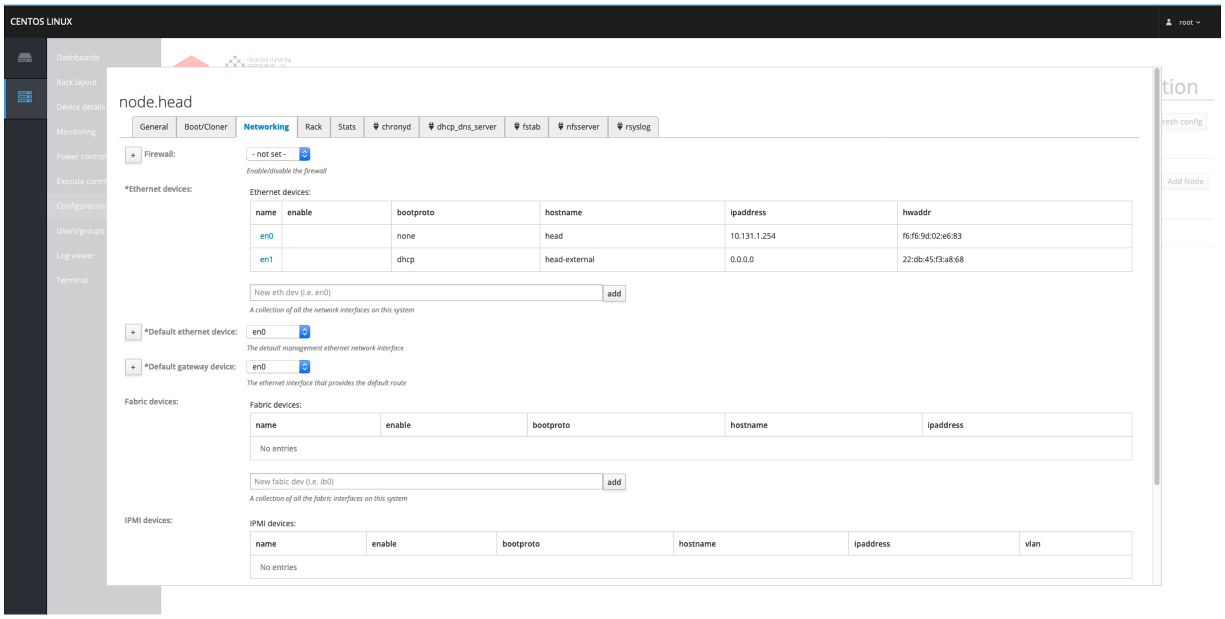
4 4. Configuring nodes



Node name	Netboot	Firewall	Cloner Image	Actions
node.head	netboot.local			Edit Plugins Copy
node.node01	netboot.local			Edit Plugins Copy

4.1 Configurations in ClusterVisor

ClusterVisor makes it easy to make changes in the cluster by consolidating all of the settings and configurations into one location, this being the **Configuration** page. This page organizes all of the settings that can be changed into collections, and each collection has a list of sections that belongs to it (e.g. the node sections belong to the **Node** collection, the cloner image sections belong to the **Cloner Image** collection, etc.). Throughout ClusterVisor, the sections are referred to in the syntax *collection.section*. This means that a node named "storage01" would be referred to as *node.storage01*, and a cloner image named "node" would be referred to as *cloner_image.node*.



node.head

Firewall: - not set -
Enable/disable the firewall

*Ethernet devices:

Ethernet devices:

name	enable	bootproto	hostname	ipaddress	hwaddr
en0		none	head	10.131.1.254	f6:f6:9d:02:e6:83
en1		dhcp	head-external	0.0.0.0	22:db:45:f3:a8:68

New eth dev (i.e. en0)
A collection of all the network interfaces on this system

*Default ethernet device: en0
The detault management ethernet network interface

*Default gateway device: en0
The ethernet interface that provides the default route

Fabric devices:

Fabric devices:

name	enable	bootproto	hostname	ipaddress
No entries				

New fabic dev (i.e. ib0)
A collection of all the fabric interfaces on this system

IPMI devices:

IPMI devices:

name	enable	bootproto	hostname	ipaddress	vlan
No entries					

Changes to the sections can be done by clicking on the **Edit** button in the **Actions** column of the section. This will display a pop-up window that will display all of that section's fields, and some sections with a large number of fields may have tabs in this window to help organize them. The fields will be listed with the name of the field on the left, the value on the right, and beneath the value will be a description of what the field is for. While editing the fields, any changes made will not actually be saved unless the **Save** button at the bottom of the window is pressed; otherwise they will be discarded once the window is closed. As well, to help make sure invalid options are not used, every time the changes are saved they will be checked to make sure they are valid before actually being saved. If any invalid changes have been made, an error message displaying what went wrong will be displayed both on the top and bottom of the window (in case either is not currently visible). From that point, the changes can either be corrected to validate properly or be discarded by exiting the window.

Some fields are references to other fields, so trying to delete a referenced field will fail validation until all references to that field are removed. Also, some fields will be prefixed with an asterisk (*) to indicate that the field is required. If a required field is left blank, it will fail validation.

If any of the values in the sections are changed and saved, those changes will be queued so that they can be committed to the affected nodes. Once ready to commit those changes, click on the **Commit changes to nodes** button at the top of the page (if the button is greyed out, this means that no actionable settings have been changed yet). This provides a list of the nodes that will be receiving modifications from ClusterVisor along with what types of changes will be made (e.g. networking, hosts, timezone, etc). If any of the committed changes fail upon reaching the node, they will be queued to try again later. Those queued changes, along with the message received from the node on why those changes could not be made, can be viewed by clicking the **View change queue** button. As well, if there is a need to purge any of the committed changes from being sent to the nodes, after clicking either the **Commit changes to nodes** or **View change queue** button, at the bottom of the pop-up window is a button to purge the entries.

Some of the changes (particularly the network changes) can cause the node to become unresponsive. The fields that can cause these issues are flagged with a hazard icon (i.e., an orange triangle with an exclamation mark) and should only be changed if one is prepared to handle those issues. Additionally, whenever a node is rebooted, by default it will reconfigure itself based on the current settings it has in ClusterVisor. This can be used to fix a node if it gets put into an odd state for whatever reason.

4.1.1 Using the command line

The configurations in ClusterVisor can be changed from the command line using the utility cv-conf. All of the collections can be listed using:

```
$ cv-conf --collections
```

All of the sections within a collection can be listed using:

```
$ cv-conf --sections COLLECTION
or
$ cv-conf -s COLLECTION
```

All of the collections and their respective sections can be listed using:

```
$ cv-conf --sections-all
or
$ cv-conf -sa
```

The way to add, edit, delete, and copy a section is by using the *collection.section* syntax used in the web interface, so the --sections-all flag will typically be the most useful for finding the needed parameter. When it comes to creating or editing a section, it will open up the entry in whatever text editor the $EDITOR environment variable is set to. The document will contain the existing fields at the top of the document along with the documentation at the bottom listing all the possible fields that can be added alongside their description. The format for the document is in YAML[2] by default, but can also be set to use the formats:

- Apache config[3]
- TOML[4]
- JSON[5]

Regardless of the format used, be sure any modifications are valid in that format or the changes will be rejected. Furthermore, much in the same way as the web interface, all changes will be checked for validation and upon failure will display the errors it found and prompt if it should a) return to the editor so the invalid entries can be corrected or b) to quit to discard the changes.

Assuming the changes passed validation, those changes can then be committed so that any reconfiguration to the nodes can be performed by using the cv-commit utility. This utility can display what type of changes will be committed, proceed with committing those changes, display the status of failed commits that will be re-queued,

2 https://yaml.org/
3 https://httpd.apache.org/docs/2.4/configuring.html
4 https://github.com/toml-lang/toml/blob/master/README.md
5 https://json.org/

and clear the commits altogether. Additionally, to ensure that the cv-commit utility is run after using cv-conf, if any changes need to be committed to the nodes the cv-conf utility will instruct that cv-commit needs to be run after the changes have been saved.

4.2 Modifying node configurations

The nodes can be modified on the **Configuration** page under the **Node** tab; this will list all of the sections of the node collection. Instead of each node being its own section, they are instead grouped together so that each "type" of node will have its own section. The reason for this is that clusters tend to have many nodes that are setup exactly the same as others of their same type, with the only difference being their name and networking information. As well, in most cases any changes made to one node will also need to be made to others of the same type. To handle this, the name of each section is a syntax showing the range of nodes that belong in the section (the same syntax used in Slurm's configuration file for configuring nodes). For example, given node01, node02, node03, and node04 they would be referred to as *node.node[01-04]* in ClusterVisor. Do note that sections do not *need* to handle multiple nodes; e.g., for single head node clusters there will be a section that will manage just the head node and nothing else.

4.2.1 Variant fields and field variables

For sections handling multiple nodes, rather than just one, when any setting is changed by default that setting is applied to **all** of the nodes in that section. However, to change a field for a specific node or sub-section of the nodes the "+" button to the left of the field name can be pressed to add a **variant** to the field. The variant field will present two input boxes, the left being for the desired value and the right being for the node(s) to apply the value to. In practice, the variant fields will primarily be used for things like the MAC address of the node since they will be different for every node, but there are other ways to handle fields that need a different value for each node. An example of this is the IP address for a node; while it can be set individually using variant fields, it can also use a **field variable** if the value follows a sequential pattern for each node.

Typically, the IP address of a node will tend to end with the index of the node (e.g., node01 will be 10.1.1.1, node02 will be 10.1.1.2, and so on). As well, the hostname of a node will usually use its own name (e.g., node01 will have the hostnames node01, node01-ib, and node01-ipmi). For this reason, ClusterVisor provides field variables to make use of these patterns by providing *_name* and *_index*. The field variables can be used by wrapping them with **${** and **}** at the beginning and end, so using them as a value will looks like *${_name}* and *${_index}* where anything else can be on either side of the field variables (e.g., *${_name}-ib* and *10.1.1.${_index}* would both be valid).

The *_name* field variable is the replaced with each node's name in the section; e.g., given the section node[01-04], when getting the values for node01 the *_name* field variable will become node01. For an example of how to use it, if one wanted to set the hostname of the InfiniBand interface to be the name of the node with "-ib" at the end (e.g., node01-ib), then setting the value of the hostname field to *${_name}-ib* would accomplish this.

The *_index* field variable would be the index of the node within its section; so in node[01-04] the value of *_index* for node01 would be 1 (because it is the first entry), for node02 would be 2 (because it is the second entry), and so on. However, if there were another section, node[05-08], then in that section the value of *_index* for node05 would also be 1, for node06 would be 2, and so on. To handle these kinds of offsets, basic math will work with the *_index* variable inside the field variable wrapper. For example, if one wanted to set the IP address for an interface in the section node[05-08] from 10.1.1.5 to 10.1.1.8, respectively, then using the value *10.1.1.${_index+4}* to add four to the value of *_index* (making the *_index* value for node05 become 5, node06 become 6, and so on) would accomplish this.

ⓘ Be aware that field variables are just a nicety and can be substituted with manually defining each unique value with variant fields if one does not prefer to use field variables. As well, variant fields and/or field variables are only needed in cases where the value of a field differs for each node; otherwise neither needs to be used.

Using the command line

The web interface provides the "+" button for adding variant fields, but to accomplish this in cv-conf the field name just needs to be appended with a colon followed by the node(s) used in the variant. For example, given the *hostname* field (in YAML format):

```
hostname: ${_name}
```

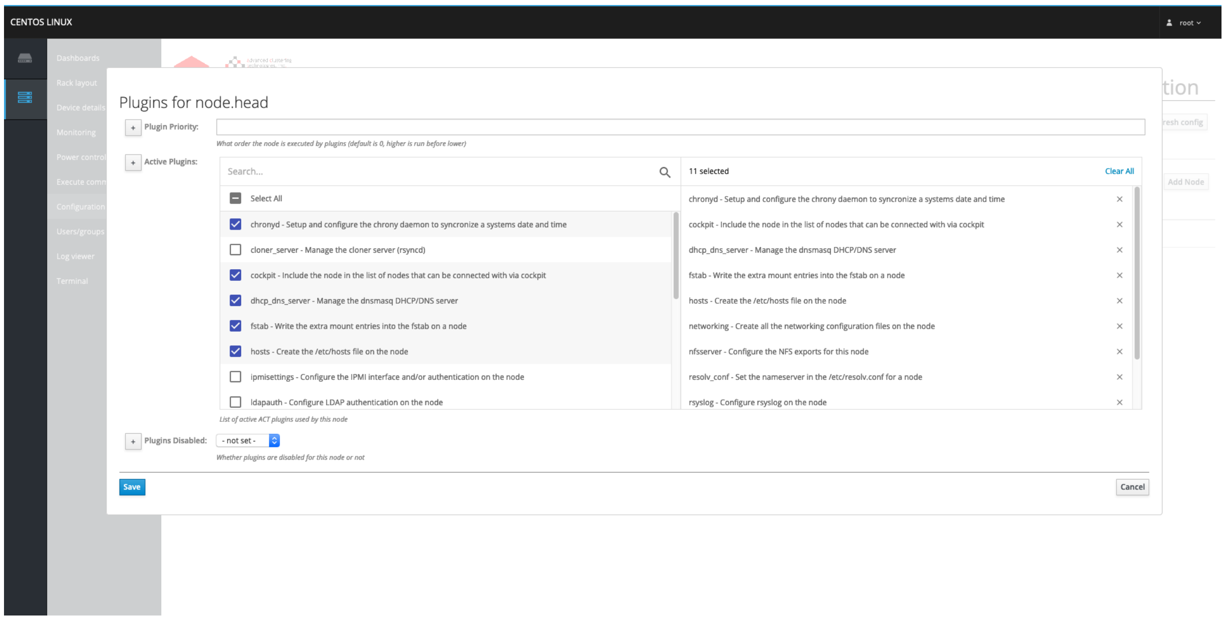
A variant for node01 and node[02-03] can be added like so:

```
hostname: ${_name}
hostname:node01: firstnode
hostname:node[02-03]: othernode${_index}
```

These fields can be added, modified, or removed by editing them with cv-conf using the --edit flag.

The plugins can be edited for the nodes by using the --plugins flag in cv-conf, which will list the same fields that are presented in the web interface and will display the list of all available plugins in the documentation at the bottom of the document. The additional plugin fields are all listed under the *_plugins* field of the node, but keep in mind that only the fields entered for the enabled plugins will actually be utilized by the plugins.

4.2.2 Node plugins



Plugins for node.head

Plugin Priority:

What order the node is executed by plugins (default is 0, higher is run before lower)

Active Plugins:

Search...

- [–] Select All
- [✓] chronyd - Setup and configure the chrony daemon to syncronize a systems date and time
- [ ] cloner_server - Manage the cloner server (rsyncd)
- [✓] cockpit - Include the node in the list of nodes that can be connected with via cockpit
- [✓] dhcp_dns_server - Manage the dnsmasq DHCP/DNS server
- [✓] fstab - Write the extra mount entries into the fstab on a node
- [✓] hosts - Create the /etc/hosts file on the node
- [ ] ipmisettings - Configure the IPMI interface and/or authentication on the node
- [ ] ldapauth - Configure LDAP authentication on the node

11 selected

- chronyd - Setup and configure the chrony daemon to syncronize a systems date and time
- cockpit - Include the node in the list of nodes that can be connected with via cockpit
- dhcp_dns_server - Manage the dnsmasq DHCP/DNS server
- fstab - Write the extra mount entries into the fstab on a node
- hosts - Create the /etc/hosts file on the node
- networking - Create all the networking configuration files on the node
- nfsserver - Configure the NFS exports for this node
- resolv_conf - Set the nameserver in the /etc/resolv.conf for a node
- rsyslog - Configure rsyslog on the node

List of active ACT plugins used by this node

Plugins Disabled: - not set -

Whether plugins are disabled for this node or not

When changes are made in ClusterVisor, what determines if those changes make any modifications to the node(s) are the plugins loaded on the node(s). The loaded plugins can be viewed from the **Plugins** button in the **Actions** column of the section (instead of from the pop-up window from the **Edit** button). The plugins window follows the same rules mentioned earlier as the edit window, the only different being that it is only listing the plugin properties of the node rather than the rest of the node's fields. The plugin properties available are the active plugins (which plugins the node(s) are using), the plugin priority (whether the actions from the plugins need to run before or after the other sections of nodes), and if the plugins on the section of nodes are disabled or not. Do note, regardless of what plugins are selected, if the **Plugins Disabled** field is set to *true* then none of the plugins will be active.

The plugins themselves will dictate what features that ClusterVisor will be managing on the node. For instance, with no plugins enabled there will be nothing changed on the nodes if any configurations are changed, but if the **networking** plugin is enabled then changing any networking field on the node(s) will cause ClusterVisor to reconfigure the node to match those settings (but only after the changes have been committed). The reason not all plugins are enabled is that some nodes will have different responsibilities than other nodes; e.g., one may want to enable the DHCP/DNS server plugin on the head node, but not the compute nodes as they would be clients to the DHCP/DNS server.

Along with enabling which features that ClusterVisor will manage for a section of nodes, some plugins can expose additional fields to configure for those nodes. These additional fields will be in their own tab in the pop-up window from the **Edit** button and will be delineated with a plugin icon in front of the tab name. Plugins fields are used for configuring the service(s) offered by the plugin; e.g., the DHCP/DNS server plugin will provide additional fields for configuring the DHCP and DNS servers.

> ⓘ Keep in mind that disabling a plugin that exposes additional fields will result in the values of those fields being cleared out, so be sure to copy any values that may be needed before disabling the plugin.

4.3 Adding new nodes

4.3.1 Export and import

The simple way to add new nodes to the cluster if they are being purchased from Advanced Clustering Technologies, Inc. (a.k.a. ACT) is to:

1. Use the **Export** button on the **Configuration** page to download the current state of ClusterVisor to a file.
2. Send the file ACT via email (support@advancedclustering.com[6]).
3. The ACT support team will then add the new node entries from the purchase order.
4. Once that is completed and the revised file has been received, it can be added to ClusterVisor by running the following on the node hosting the ClusterVisor server daemon:

   ```
   $ systemctl stop cv-serverd
   $ cv-db-image --auto-detect --import-from <path to file> --overwrite
   $ systemctl start cv-serverd
   ```

   Which will effectively stop the ClusterVisor server daemon (since changing the data underneath the service can cause unexpected behaviors), upload the revised file to ClusterVisor, and then start the server daemon back up.

4.3.2 Manually adding new node

If the export/import method is not being used, the new nodes will needed to be added in manually. At this point, either an existing set of nodes is being expanded (i.e., where the hardware matches the other nodes) or a completed new set is being added.

In the case of the former, all that would need to be done is:

1. Click on the **Edit** button from the **Actions** column of the node section being expanded.
2. In the pop-up window, go to the **General** tab and edit the range of nodes in the **Name** field to include the new nodes (e.g. if four new nodes are being added to node[01-04] it would become node[01-08] to accommodate them).
3. Add any variant fields for any unique information about the nodes.

The necessary variant fields may vary depending on the setup of the cluster, but at minimum this will require the serial number(s) (found under the **General** tab) and the MAC address(es) (found under the **Networking** tab under each interface) of each new node to be added.

In the case of the latter where nodes with different hardware are being added and/or servicing a different role than the existing set of nodes, the new nodes can be added either from scratch or by using the existing set of nodes as a template. To copy values from an existing section of nodes for the new nodes, click the **Copy** button in the **Actions** column of the node section and a pop-up window will be presented. At minimum, the **Name** field under the **General** tab needs to be changed to match the names of the new nodes, but otherwise only change the fields that differ from the copied values.

However, if starting from scratch, the new section can be created by filling out the input box at the top of the **Node** collection with the name of the new section followed by clicking the **Add Node** button to the right of the input box. This will display a pop-up window where all of the values for the new nodes can be entered. While not all the fields

6 mailto:support@advancedclustering.com

need to be filled out, at a minimum the serial number(s), MAC address(es), and all required fields (which will be prefixed with an asterisk) need to be completed.

4.3.3 Using the command line

The configurations can be exported from the command line using the utility cv-db-image. To export the ClusterVisor configuration to a file, the following command needs to be run:

```
$ cv-db-image --uri localhost:27017/clustervisor --export <path to file>
```

Where *<path to file>* would be replaced with the desired filename of the export file (note, this file does not need to exist already). This will create a JSON[7] file that can edited and imported back into ClusterVisor using:

```
$ systemctl stop cv-serverd
$ cv-db-image --uri localhost:27017/clustervisor --import-from <path to file> --overwrite
$ systemctl start cv-serverd
```

Where *<path to file>* would be replaced with the path to the export file from the previous step.

As for mimicking the expanding node sections and making a copy of a node section mentioned above, both can be done using the cv-conf utility. The former can be done by using the --edit flag and expanding the __*name*__ field to include the new nodes, and the latter can be done using the --copy flag and editing the __*name*__ field to the names of the new nodes. Lastly, a new section can be added from scratch by using the --add flag to start with a blank node section.

To avoid confusion, while cv-conf does have a --dump flag to dump the configuration file and looks similar to the contents generated by the export flag of cv-db-image, the output from --dump is missing fields needed to by cv-db-image to properly restore the data back into ClusterVisor. The purpose of the --dump flag from cv-conf is to view all the configurations from ClusterVisor in varying format, not for any export/ import purposes.

4.4 Replacing existing nodes

If at any point in time an existing node needs to be replaced, the process can be simplified using ClusterVisor. First, make any and all changes that need to made to the node's configuration to adhere to the replacement node (for more information on this, see section Modifying node configurations(see page 12) of this guide). At a minimum, the node's serial number (found under the **General** tab) and MAC address(es) (found under the **Networking** tab for each interface) need to be updated to match the replacement node. Once the changes are completed, to reconfigure the replaced node to match the configuration it needs to be set to boot into Clustervisor's **cloner** software. This can be done by doing the following:

1. Navigate to the **Boot/Power control** page.
2. From the drop-down menu select the desired nodes.
3. Find the **Netboot** column and select *cloner* from the drop-down menu.

Next, assuming that the boot order of the node has network boot set as its first boot option, all that is left is to power on the node (unfortunately, since the node is not yet handled by ClusterVisor this will need to be done

7 https://json.org

manually). Once the node is powered on, if at least one of the MAC addresses was set properly it will network boot from ClusterVisor's cloner image and will begin re-configuring the node to match what was set in ClusterVisor. Once the cloning process has been completed, it will automatically reboot from the OS drive and will be ready to be used.

In case there are any issues during the cloning process, the status can be monitored in ClusterVisor or by connecting a monitor to the node. To monitor from ClusterVisor:

1. Navigate to the **Log viewer** page.
2. Select the node from the drop-down menu.
3. Click on the **Cloner** tab.

This will update in real-time with latest log entries from the cloning process. If any errors are occurring, refer to the Troubleshooting(see page 0) section of this guide.

4.4.1 Using the command line

The same steps above for changing the network boot image can be done from the command line utility cv-netboot using the --set flag to set it to the *cloner* image. As for viewing the cloner log, by this can be done by viewing the contents of the file */var/log/clustervisor/cloner.log*, but the directory can be different if the *log_directory* field from *config.global* is not set to */var/log/clustervisor* (in which case, the *cloner.log* file will be stored under that directory instead).

4.5 Removing nodes

If nodes are being removed from ClusterVisor, they are either being removed from a section or the section itself is being removed. In the case of the latter, this can be done by clicking the **Edit** button from the **Actions** column of the node section and clicking the red **Delete** button at the bottom of the pop-up window. This will remove all entries of the node from ClusterVisor, and they will no longer be managed by ClusterVisor. For this reason, before removing the node it is advisable to remove all plugins from the node (or to set the **Plugins Disabled** field on the node to *true*) so that ClusterVisor can stop any services it was running on behalf of the node (otherwise this will need to be done manually).

If a node needs to be removed from a section, the section will need to be partitioned at the removed node (e.g. if the section is node[01-08] and node04 is being removed, the section will need to be partitioned to node[01-03] and node[05-08]). This can be accomplished by:

1. Using the **Copy** button from the **Actions** column of the node section to duplicate the section.
2. From the original node section, rename the section to the name of the first partition.
3. In the duplicate node section, rename the section to the name of the second partition.

This will effectively remove the node from ClusterVisor, but with the node sections partition, this will result in any changes needing to be made to the nodes to be done in both sections. A way to counter against this would be to instead reassign a node at the end of the node section to replace the removed node (e.g. if the section is node[01-08] and node04 is being removed, then node08 would be reassigned as node04). This can be done by:

1. Remove all variant fields of the removed node.
2. Reassign the variant fields of the reassigned node to the removed node (e.g., following the previous example, this would mean changing all node08 variant fields to be node04 instead).
3. Rename the section so that only the last node is removed (e.g., following the previous example, the section name would change from node[01-08] to node[01-07]).
4. Reboot the reassigned node so that it can be reconfigured (e.g., following the previous example, node08 would need to be rebooted so that it will then be configured as node04).

This method would not require any duplicate sections to be created, so there would be no partition to manage like the other method. However, this can potentially lead to confusion since the labels on the node will still have the old node names on them (e.g., following the previous example, although node08 has been reassigned as node04, its labels will still show it is node08).

There is one other edge case for removing a node, which is if the node that needs to be removed is the beginning or end of a node's section (e.g., if the section is node[01-08], the beginning and ending nodes would be node01 and node08, respectively). In this case, nothing special needs to be done like the previous two methods, and only the node section needs to be renamed with the beginning or end removed (e.g., using the previous example, the section name would become node[02-08] if node01 is removed or node[01-07] if node08 is removed).

ⓘ It is important to note that nodes do not **need** to be removed from ClusterVisor for the node to no longer be managed by it. Using a variant field to disable the plugins for a node will effectively make it so that ClusterVisor will no longer make modifications to the node. The method of using variant fields to isolate a node from its node section is a way to "softly" remove a node without needing to manage any of the hassles listed above.

4.5.1 Using the command line

Most of what has been described above has already been outlined in previous sections on how to modify the configuration values of nodes. However, the one new concept introduced was being able to delete a section. This can be done using the cv-conf utility using the --delete flag, but as mentioned above it is advisable to remove or disable the plugins for the nodes being removed first, which can be done using the --plugins flag.

5 5. Managing users

If ClusterVisor was installed with its full suite of features, it installs and sets up an LDAP server (specifically, a 389 directory server[8]) to use for managing users and groups across the cluster. The page in ClusterVisor to manage the users and groups can be found under **Users/groups** and behaves much like **Configuration** page. The **Users/groups** page has two tabs, **Users** and **Groups**, and in this section all actions for users and groups will be occurring in those two tabs.

5.1 View all users and groups

All of the users and groups managed by ClusterVisor will be listed in the **Users** and **Groups** tabs, respectively. It should be noted that this is not all of the users and groups available on the cluster since users and groups can still be created natively on a node (e.g. using useradd and groupadd).

5.1.1 Using the command line

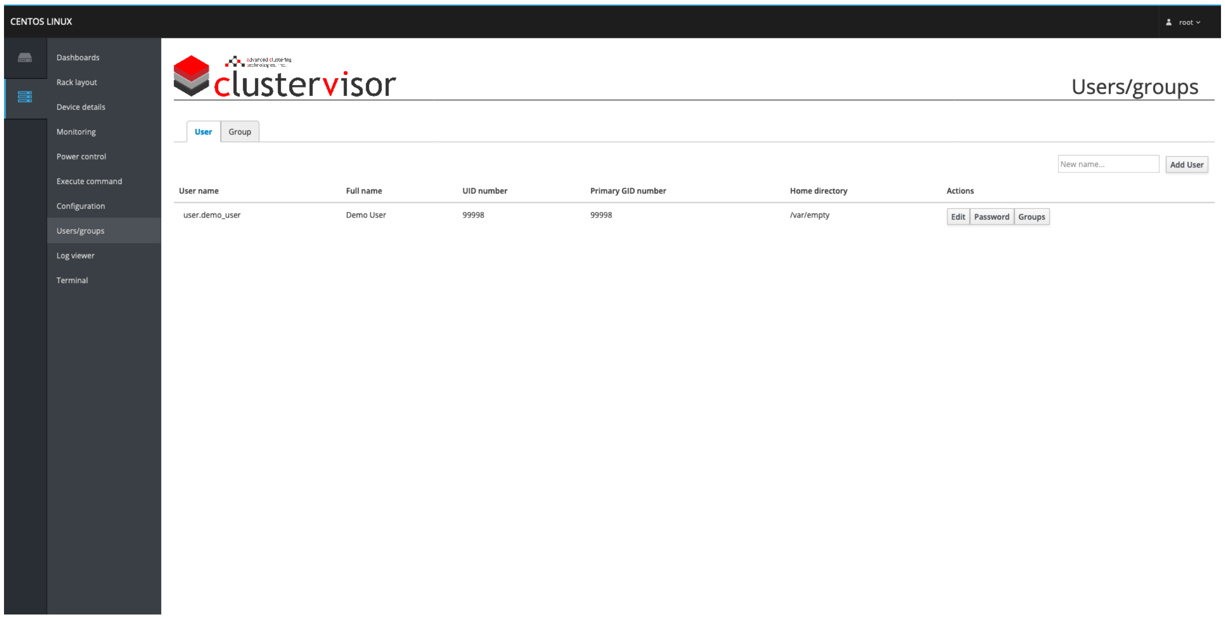
All of the users and groups managed by ClusterVisor can be viewed using the cv-useradm utility. Specifically, this can be done by running:

https://intranet.corp.advancedclustering.com/bitbucket/projects/CV/repos/clustervisor/pull-requests/45/overview

This will display all of the users and groups, with the users at the top and the groups listed at the bottom, and they are ordered by the ID of the user/group (which is listed in parentheses). The users added to the nodes natively can still be viewed from the /etc/passwd and /etc/group files, but are not managed by ClusterVisor. There is, however, a way to synchronize the natively added users across the cluster if using LDAP for user management is not desired, which is explained further in the Synchronizing native users(see page 27) section of this guide.

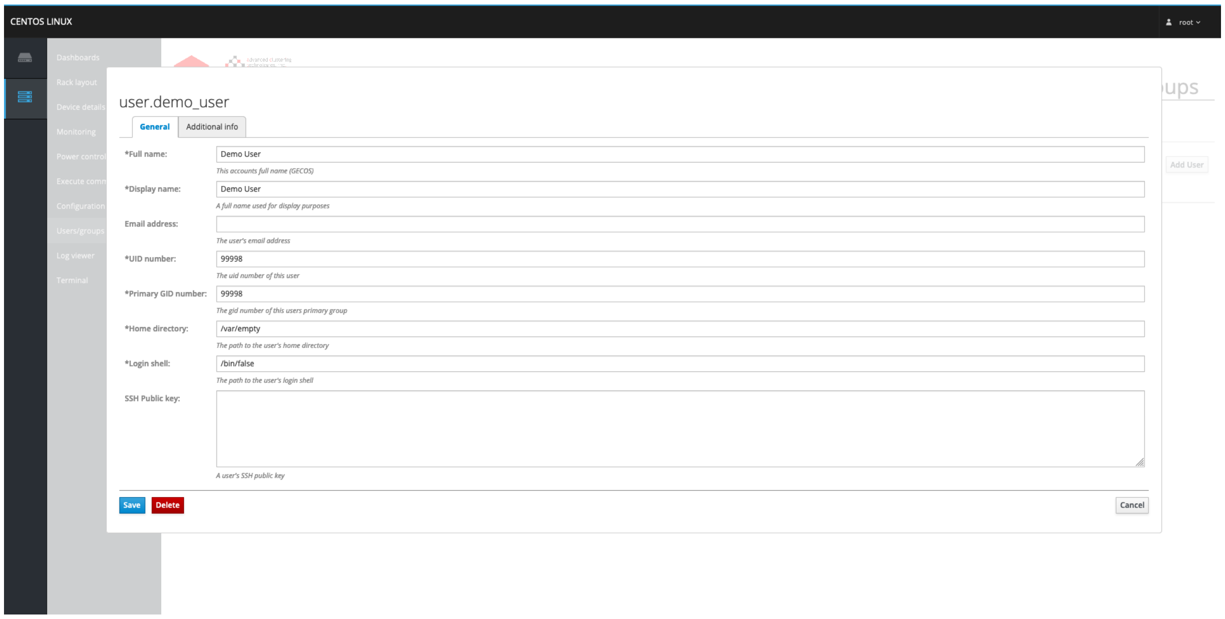
8 https://directory.fedoraproject.org/

5.2 Adding a user



User name	Full name	UID number	Primary GID number	Home directory	Actions
user.demo_user	Demo User	99998	99998	/var/empty	Edit Password Groups

From the **Users** tab, above the listed entries of users is an input box to the left of the **Add User** button. In the input box, put in the desired username for the new user and click on the **Add User** button. This will present a pop-up window with two tabs, **General** and **Additional Info**, and in both tabs all fields that are prefixed with an asterisk (*) are required fields that need to be given a value for the user to be added. Some of those fields will be automatically filled out with sane defaults, so unless there is a necessity to change them they can be left as-is.



user.demo_user

*Full name:	Demo User — This accounts full name (GECOS)
*Display name:	Demo User — A full name used for display purposes
Email address:	— The user's email address
*UID number:	99998 — The uid number of this user
*Primary GID number:	99998 — The gid number of this users primary group
*Home directory:	/var/empty — The path to the user's home directory
*Login shell:	/bin/false — The path to the user's login shell
SSH Public key:	— A user's SSH public key

The non-required fields do not need to be filled out and are presented only as a convenience. For this reason, the fields under the **Additional Info** tab can be ignored in most cases. However, it should be noted that the non-required field **SSH Public Key** under the **General** tab will also serve the purpose of allowing the user to login via SSH regardless of whether the /home directory is mounted on the system or not. Usually this is not the case since a user's SSH public key is typically stored under the .ssh directory in the user's home directory (so it is no longer present if the /home directory is not mounted).

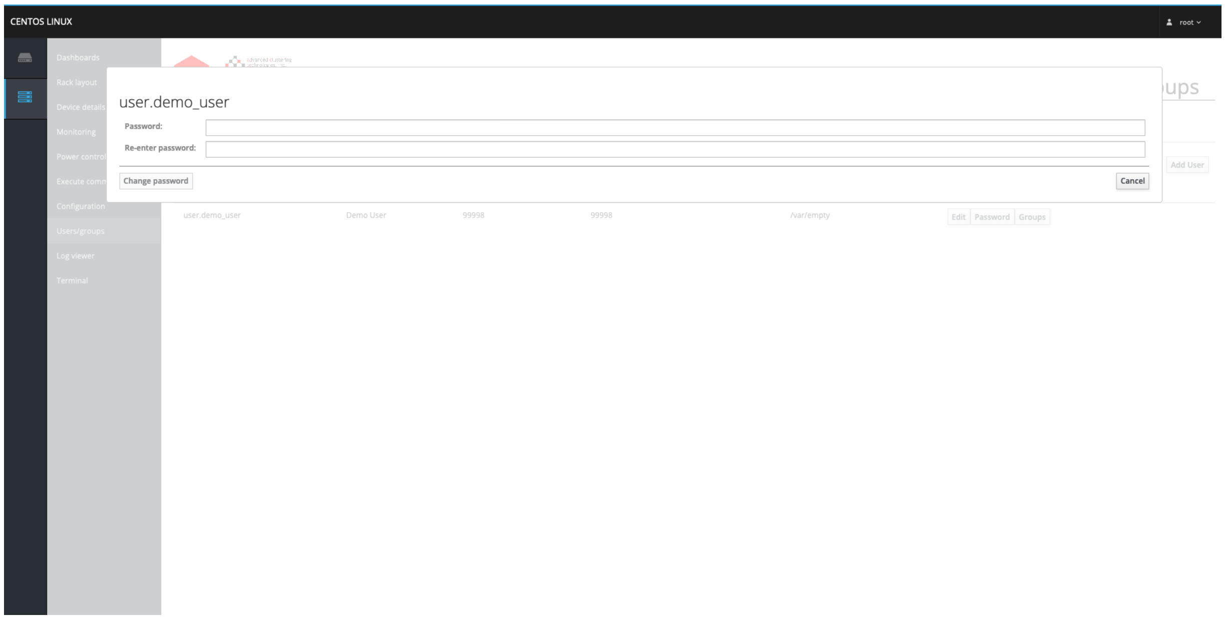
5.2.1 Using the command line

A user can be added in ClusterVisor from the command line using the cv-useradm utility. This can be done by running the following command:

```
$ cv-useradm --add user.<username>
or
$ cv-useradm -a user.<username>
```

Where *<username>* would be replaced with the desired username for the new user. Similar to using the --add flag with cv-conf, the interface is presented in YAML[9] by default with the editable fields at the top and the documentation of each possible field at the bottom. Like the web interface, most of the fields will be populated automatically with sane defaults, but the **cn** field will still need to be edited with the full name of the user along with **displayName** for the displayed name of the user. Upon saving and exiting the file, a prompt will be given to add a password to the new user. Once this is done, the user has been added and is ready for use.

5.3 Changing a user's password



user.demo_user

Password:

Re-enter password:

user.demo_user Demo User 99998 99998 /var/empty

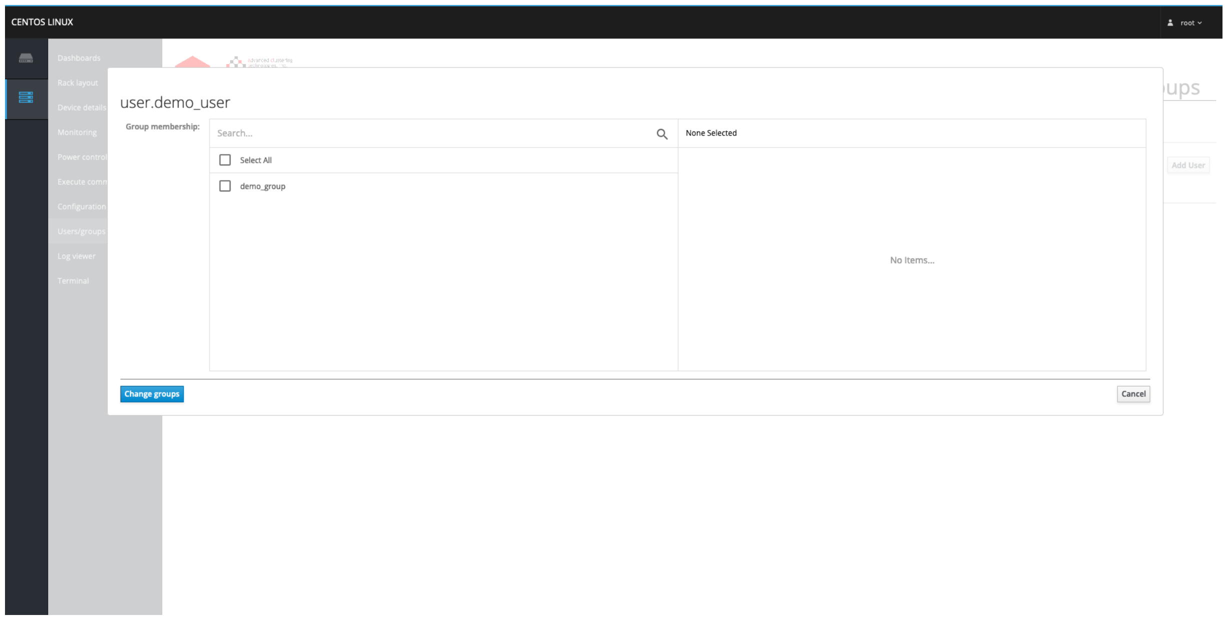
From the **Users** tab, in the **Actions** column for each user is a **Password** button. Clicking this brings up a pop-up window to put in a new password for the user, which can be applied by clicking the **Change password** button.

9 https://yaml.org/

5.3.1 Using the command line

The method for changing a password in ClusterVisor depends on whether a user wants to change their own password or if the root user wants to change another user's password. For the former, this can be done like any other account using the passwd utility. For the latter, the root user is not allowed to change a ClusterVisor user's password using passwd, so instead needs to use cv-useradmin with the --password flag to change a user's password. This will present a prompt to enter and then re-enter the new password, after which the password will be changed for the user.

5.4 Adding a group



From the **Groups** tab, above the listed entries of users is an input box to the left of the **Add Group** button. In the input box, put in the desired name for the new group and click on the **Add Group** button. Similar to adding a user, sane defaults will be filled out for some of the fields, so only the **Description** field needs to be filled out. Optionally, the group can also assign members to it under the **Members** field, but this can also be done later after the group is created.

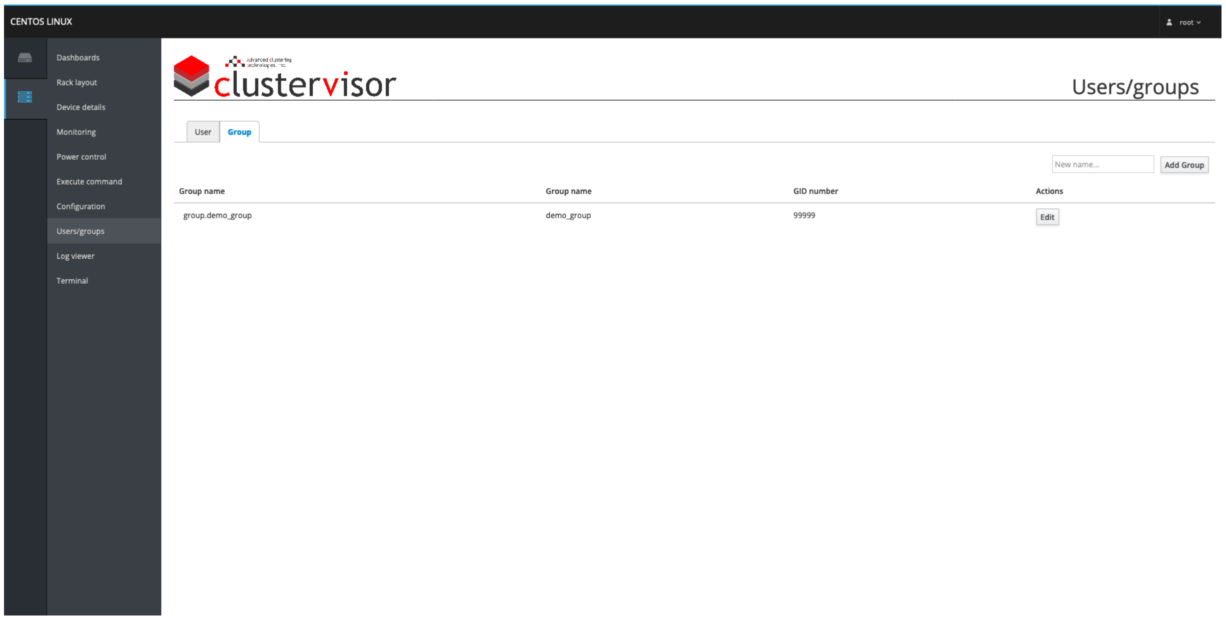
5.4.1 Using the command line

A group can be added in ClusterVisor from the command line using the cv-useradm utility. This can be done by running the following command:

```
$ cv-useradm --add group.<name>
or
$ cv-useradm -a group.<name>
```

Where <*name*> would be replaced with the desired name for the new group. Similar to using the --add flag with cv-conf, the interface is presented in YAML[10] by default with the editable fields at the top and the documentation of each possible field at the bottom. Like the web interface, most of the fields will be populated automatically with sane defaults, but the **description** field will still need to be edited with a description of the new group and optionally a list of the members of the group (which can also be added later). The group will be added once the file has been saved and exited.

5.5 Group management for users



Group name	Group name	GID number	Actions
group.demo_group	demo_group	99999	Edit

A user can be added/removed from a group in two different ways in ClusterVisor, either from the **Users** tab or the **Groups** tab. To do so from the **Users** tab, in the **Actions** column for each user is a **Groups** button, and clicking it will bring up a pop-up window to select which groups to add/remove from the user. To do so from the **Groups** tab, clicking the **Edit** button from a group's **Action** column will bring up a pop-up window to select which users to add/ remove from the group.

Each of these methods has its pros and cons, primarily depending upon whether a user needs to be added/removed from any groups (in which case, use the **Users** tab) or if multiple users need to be added/removed from a group (in which case, use the **Groups** tab).

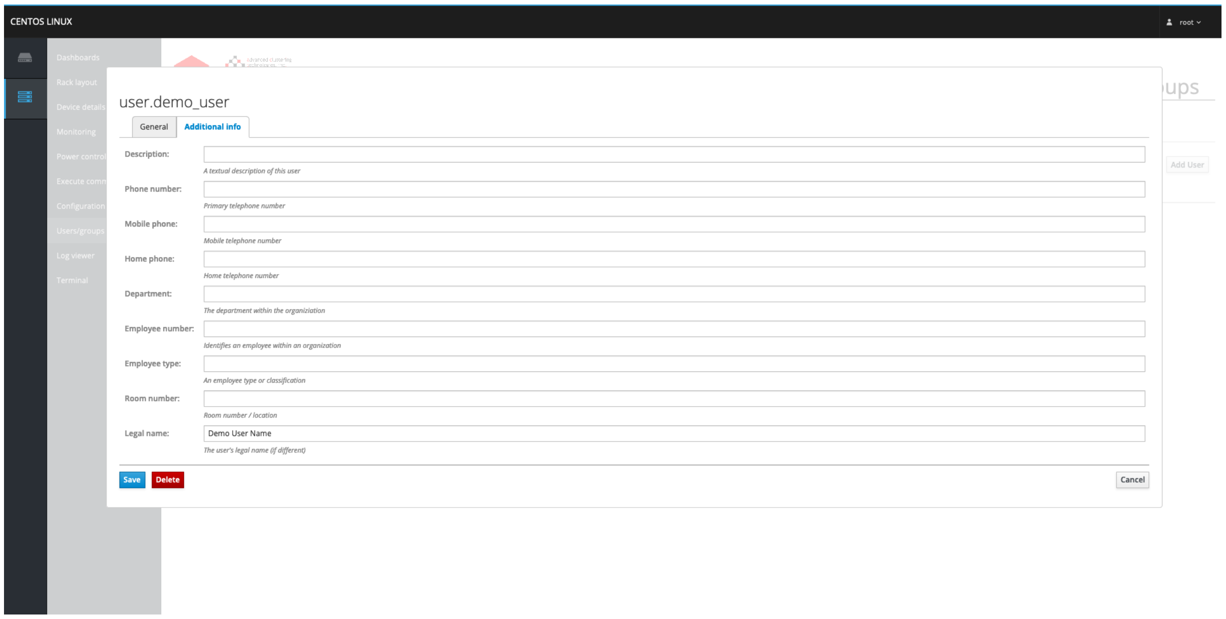
5.5.1 Using the command line

Similar to the web interface, a user can be added/removed from a group in two different ways from the command line, both of which are done using the cv-useradm utility. To add/remove multiple groups from a single user, the --group-membership flag should be used on the user and the groups can be added/removed by changing the value of the group between true/false (true meaning the user is added to the group, and false meaning the user is removed from the group). To do the inverse of adding/removing multiple users from a single group, the --edit flag should be used on the group to add/remove users from the **member** field. The available users will be listed below in the documentation portion, and be sure to format the entries as *user.<username>* where *<username>* is replaced with the username of the user.

10 https://yaml.org/

If all users are being removed from a group, the value should be set to *[]*, otherwise it will fail validation because the **member** field is not a list.

5.6 Modifying a user



user.demo_user

Description:	
	A textual description of this user
Phone number:	
	Primary telephone number
Mobile phone:	
	Mobile telephone number
Home phone:	
	Home telephone number
Department:	
	The department within the organiziation
Employee number:	
	Identifies an employee within an organization
Employee type:	
	An employee type or classification
Room number:	
	Room number / location
Legal name:	Demo User Name
	The user's legal name (if different)

From the **Users** tab, in the **Actions** column for each user is an **Edit** button and clicking this brings up a pop-up window showing all of the configurable fields of the user. Any of these fields can be changed, but if any of the required fields (those prefixed with an asterisk) are left blank when the changes are saved it will not commit the changes until either the required fields are given a value or the pop-up window is closed (which will discard all of the changes).

5.6.1 Using the command line

A user can be modified in ClusterVisor from the command line using the cv-useradm utility. This can be done by running the following command:

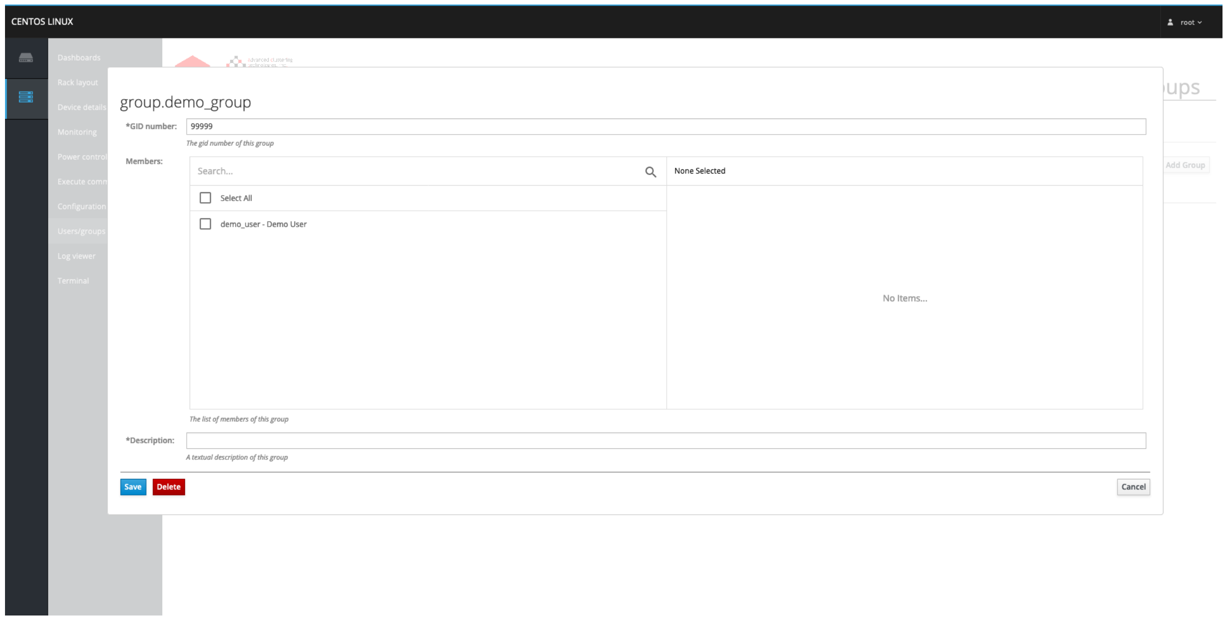
```
$ cv-useradm --edit user.<username>
or
$ cv-useradm -e user.<username>
```

Where <*username*> would be replaced with the username for the user being modified. Similar to using the --edit flag with cv-conf, the interface is presented in YAML[11] by default with the editable fields at the top and the documentation of each possible field at the bottom. Like with the web interface, there are required fields (designated with the term *required* next to the field's name in the documentation below) that need to have a value

11 https://yaml.org/

to pass validation before any saved changes are committed. Assuming validation passed, upon saving and exiting the file the changes will be applied.

5.7 Modifying a group


group.demo_group
*GID number: 99999
The gid number of this group
Members:
demo_user - Demo User
None Selected
The list of members of this group
*Description:
A textual description of this group

From the **Groups** tab, in the **Actions** column for each group is an **Edit** button and clicking this brings up a pop-up window showing all of the configurable fields of the group. Any of these fields can be changed, but if any of the required fields (those prefixed with an asterisk) are left blank when the changes are saved it will not commit the changes until either the required fields are given a value or the pop-up window is closed (which will discard all of the changes).

5.7.1 Using the command line

A group can be modified in ClusterVisor from the command line using the cv-useradm utility. This can be done by running the following command:

```
$ cv-useradm --edit group.<name>
or
$ cv-useradm -e group.<name>
```

Where <*name*> would be replaced with the name for the group being modified. Similar to using the --edit flag with cv-conf, the interface is presented in YAML[12] by default with the editable fields at the top and the documentation of each possible field at the bottom. Like with the web interface, there are required fields (designated with the term *required* next to the field's name in the documentation below) that need to have a value to pass validation before any saved changes are committed. Assuming validation passed, upon saving and exiting the file the changes will be applied.

12 https://yaml.org/

5.8 Removing a user / group

The process for removing a user and group is the same. From the **Actions** column for each user /group is an **Edit** button and clicking this brings up a pop-up window. At the bottom of this window is a red **Delete** button, and when clicked this will provide a confirmation box to verify that the user / group should be deleted (in case clicked by accident). In the case of the user, while this process will remove the user its home directory will remain until manually deleted.

5.8.1 Using the command line

A user / group can be removed in ClusterVisor from the command line using the cv-useradm utility. This can be done by running the following command:

```
$ cv-useradm --delete user.<username>
$ cv-useradm --delete group.<name>
or
$ cv-useradm -d user.<username>
$ cv-useradm -d group.<name>
```

Where <*username*> would be replaced with the username for the user being removed and <*name*> would be replaced with the name of the group being removed. This will display a prompt asking if the user / group should truly be removed or giving the option to quit the operation in case it was done by mistake. This dialog can be skipped by also passing the --yes flag, which will agree to removing the user /group without confirmation. Same as the web interface, a user's home directory will remain until manually removed.

5.9 Configuring LDAP

While ClusterVisor provides its own 389 directory server[13] with sane default configurations, these settings can be modified along with replacing the LDAP server being used altogether. These configurations can be accessed from the **Configuration** page under the **Configuration** collection, specifically the *config.ldap* section. The usage of some of the less obvious fields (not already explained by their descriptions) is as follows:

- **Manage users / group** - Determines whether or not ClusterVisor is managing an LDAP instance. This will effectively disable the **Users / Groups** page and all command line utilities that manage the LDAP users (e.g., cv-useradm).
- **LDAP URI** - This is what points to the actual LDAP server being used. Keep in mind that if a non-389 directory server LDAP instance is used, the **Manage users / groups** field should most likely to be set to *true* to prevent unanticipated behavior.
- **LDAP CA Cert** - This stores the contents of the CA SSL certificate for LDAP.
- **Min UID** - There can be issues if a UID in LDAP clashes with the UIDs from the native users on the system. This field determines at what point the automatically generated UID will be given for the ClusterVisor users (which should at the very least start at 2000, but it is recommended to use something higher like 10000).
- **Min GID** - Same as **Min UID** but for the GID; the same principles apply.
- **Default user GID** - As opposed to automatically assigning users to their own groups, ClusterVisor will by default assign users to a single default group to make managing group permissions easier. This is the GID number that users will be assigned to be default.

13 https://directory.fedoraproject.org/

- **Triggers** - This provides a way to have scripts automatically triggered when actions are performed on ClusterVisor's users or groups. By default all types of triggers are already assigned a script, but they are intentionally left blank so that they can be edited rather than needing new scripts assigned (although there would be no issue if multiple scripts were assigned to the same type of trigger). An example use case for a trigger would be to clean up the home directory of the user being deleted.

5.9.1 Using the command line

The *config.ldap* section is modified just like any other section. For documentation on modifying configurations in ClusterVisor, see section Configurations in ClusterVisor(see page 0) of this guide.

5.10 Synchronizing native users

On the cluster there are different types: users that are managed by ClusterVisor and users that are not managed by ClusterVisor. The latter can be broken down into two further categories: users that are managed by some other service (e.g., Active Directory) and users that were natively added onto the system (e.g. using useradd), the latter being the focus of this section.

If users are not being managed by ClusterVisor, but there is still a desire to synchronize users across the cluster, ClusterVisor provides a system that can facilitate this through **authsync**. This feature is not enabled by default since the preferred method of user management in ClusterVisor is through its LDAP server, but it can be enabled by doing the following:

1. From the **Configuration** page under the **Configuration** collection, click the **Edit** button under the **Actions** column of the *config.global* section.
2. In the pop-up window, navigate to the **Authsync** tab.
3. For the **Source** field, from the drop-down menu select the node that will be the "source of truth" for the users/groups that will be synchronized to the other nodes (this will typically be the head node).
4. For the **Minimum UID** field, setting this to 1000 will only sync user accounts (since their UIDs start at 1000) while setting it to 0 will sync all accounts (however, using 0 can cause issues if the system users on the nodes do not match the UID of the source node).
5. For the **Minimum GID** field, this will typically match the **Minimum UID** field but can differ if a different set of groups needs to be synchronized.
6. For the **Lookup method** field, using **getent** will synchronize all users of the system (both native and non-native users) and using **files** will only synchronize native users (if LDAP is still being used, use **files** to avoid creating duplicate users).
7. Save the changes made.
8. Navigate to the **Node** collection and edit the **Authsync** field (found under the **General** tab) for each node section desired to be synchronized with the source node.
9. Save the changes made for each edited node section.

From this point, the cv-authsync utility can be used to synchronize the users/groups from the source node to the selected nodes (using the node selector flags, to sync all node use the --all flag). As well, to help automate the synchronization whenever changes are made to the native nodes, the auth watch daemon can be enabled to do so by running the following on the source node selected earlier:

```
$ systemctl enable --now cv-auth-watchd
```

This will simultaneously start the daemon and enable it to start up automatically on all following reboots. This daemon will check to see if any native users have been changed, and if so will automatically run cv-authsync --all to sync the changes to the other nodes.

As stated earlier, none of this needs to be done if users are being managed through ClusterVisor or some other service. This is only if native users are being used and need to be synchronized across the other nodes in the cluster.

6 6. Managing ClusterVisor permissions

The root user has ful privileges of ClusterVisor and can both view and edit any part of the web interface and use any of the command line utilities. All non-root users will be presented with a read-only version of the pages where no actions can be performed on either the nodes or the data in the **Configuration** page. Additionally, all fields in the **Configuration** page with sensitive data (e.g. any usernames and passwords) are redacted on the non-root user's version to prevent any security leaks. However, ClusterVisor also has settings that can limit what pages are available to non-root users as well as elevate the privileges of other non-root users.

6.1 Hiding ClusterVisor pages

While non-root users cannot do anything actionable to any of the pages in ClusterVisor, one may want to hide some of the pages anyways. The settings to make these changes can be found under the **Configuration** page under the **Configuration** tab followed by clicking the **Add** / **Edit** button for **config.permissions**. From here any page can be hidden using the appropriate field. For instance, to hide the **Configuration** page from non-root users just select "true" from the drop-down menu for the field **Hide Configuration Page**.

6.2 Managing ClusterVisor admin users

To allow a non-root user to use ClusterVisor without any restrictions they can be made an admin user. The settings to make these change can be found under the **Configuration** page under the **Configuration** tab followed by clicking the **Add** / **Edit** button for **config.permissions**.

To make a non-root user an admin click on the **Add Admin UIDs** button next to the **Admin UIDs** field and put in the user's UID and optionally a note to help identify which user this is (e.g. their full name or user name). The user will then immediately be able to use all of the ClusterVisor command line utilities without any restrictions and the next time they log into the web interface will have the unrestricted privileges.

To remove a user's admin privileges click on their **Index** in the table to the right of the **Admin UIDs** field and click on the **Delete** button. The user will then immediately have the same privileges as a non-root user for all of the ClusterVisor command line utilities as well as all further actions in the web interface will be treated as being from a non-root user. However they will still be able to view (but not make any changes to) the hidden pages until their next user session.

ⓘ The reason that a UID is needed here rather than a username is due to the limitations of how munge[14] works. Munge is what ClusterVisor uses to handle its security, but it only sends the UID of the user making the request along with the GID of their default group (not every group the user is assigned to).

6.2.1 Managing sudoer users

While an admin user can make changes to the configuration of the cluster and use the power controls through ClusterVisor, any commands they run will still be treated as a non-root user. This is because an admin user is not a sudoer user on the nodes. However, the sudoers of the cluster can also be managed using ClusterVisor by enabling the **sudoers** plugin on the nodes (for more information on enabling or disabling plugins see section Node plugins(see page 0) of this guide). When editing the plugin fields for the sudoers plugin on a node, to have all of admin users also be sudoer uses on the node select "true" from the drop-down menu of the **Use Admin UIDs** field under

14 https://github.com/dun/munge/wiki/Man-7-munge

the **Full Admins** tab. To manage all other sudoer users that are not also admin users, they can be managed depending on what type of sudoer they will be.

In Linux, a sudoer can either have full unrestricted access to any command and can run them as any user or group they want, or they can run only specific commands as specific user and/or group. In ClusterVisor the former are called a **Full Admins** and the latter are called **Restricted Admins**, each with their own respective tab. Since the **Full Admins** have no restrictions they can easily be added in or removed from the list of usernames and groups in the respective **Full Sudoer Users** and **Full Sudoer Groups** fields. However, for **Restricted Admins** each user or group needs to be added individually since they each can have their own limitations on what commands they can run as a given user and/or group.

Unlike other plugins, the **sudoers** plugin will only commit the changes made if the data generates valid syntax for the node's /etc/sudoers.d/clustervisor file, otherwise the previous version of the file will be kept. This is to ensure that the system does not end up in a state where it has a corrupt sudoers file since it could prevent all users from using sudo (which should be avoided if at all possible). So if the file is not being updated this is probably due to the fields having invalid data rather than ClusterVisor not working.

7 7. Using cloner

Currently, there is no way to use the web interface to use ClusterVisor's cloner utility, so this section will only be covering its command line utility cv-cloner. This utility provides a way to clone nodes by creating images from already configured nodes and applying those images to other nodes. The process is done in such a way that configurations unique to each node (e.g. MAC addresses and IP addresses) are still applied to each node from a single image. This means that installing new software to all of the nodes can be accomplished by setting it up on one node and deploying the new cloner images to the other nodes, or using an existing image to set nodes back into a "sane" state if their behavior begins to deviate from other nodes of the same type.

Do note that cv-cloner should not be used as a means for backing up data since it is focused towards copying the configurations of the node rather than everything stored on it. Any crucial data should be independently backed up through other means.

7.1 Basic cloner operations

7.1.1 Creating a new cloner image

Assuming the image is being created for a node with a newer kernel / initrd than the nodes the image is being created for, the first step would be to update the cloner installer by running on the node an image is being made from:

```
$ cv-cloner make-installer --overwrite
```

A cloner image is then created by running the following command on the same node:

```
$ cv-cloner new-image --server=head --image=node --disklayout=node
```

Depending on how much data is being copied off the node, this may take awhile so be sure to give it time to complete. Also, the --server flag will point to the hostname that the ClusterVisor server daemon, cv-serverd, is running on and the --image / --disklayout names can be whatever makes the most sense for the end-user of cloner. For instance, if two sets of nodes have different sets of drives (e.g. 2 disks and 3 disks), it may make more sense to use "2disk" and "3disk", respectively, for --disklayout rather than something more ambiguous like "node1" and "node2".

Also, because cloner splits the disk layout from the image holding the data on the node a new cloner image can use an existing disk layout without creating a new one. This is done the same way as above, but specifying the name of the existing disk layout rather than using a new name. This should only be done in situations where the hardware between drive setup between the two images are already the same.

7.1.2 Updating a cloner image

If a cloner image already exists, but needs to be updated with new data (e.g. new software was installed), run the following on the node being used as the image source:

```
$ cv-cloner update-filesystem --server=head --image=node
```

The --server flag will point to the hostname that the ClusterVisor server daemon, cv-serverd, is running on and the --image flag will be the name of the image being updated.

However, if the disk layout is being updated (i.e. adding / removing a drive or replacing with a different sized drive) then instead the following should be run on the node:

```
$ cv-cloner update-disklayout --server=head --disklayout=node
```

The same from before still applies to the --server flag and the --disklayout flag will be the name of the disk layout being updated.

7.1.3 Deploying a cloner image

The available cloner images can be listed using:

```
$ cv-cloner image-list
```

Once the image needed is found (for this example "node" will be used), it can be applied to the nodes by using either the **Configuration** page in the web interface to change the **Cloner Image** field of the nodes or the cv-conf command-line utility to change the **cloner_image** field of the nodes to point to the desired cloner image. Once that is set, either from the **Configuration** page in the web interface change the **Netboot** field to point to "cloner" or using the cv-netboot command-line utility to set the node(s) to "cloner" by running:

```
$ cv-netboot --nodes node01,node02 --set cloner
```

Where the --nodes would be selecting the nodes to be cloned (or using a different node selector if more appropriate). Once the cloner image has been set and the netboot of the nodes has been set to cloner, either use the **Power control** page in the web interface or the cv-power command-line utility to reboot the nodes by running:

```
$ cv-power --nodes node01,node02 reset
```

This will reboot the nodes into the "cloner" boot image which will begin applying the selected cloner image to the node(s). The progress of the nodes can be monitored by either connecting a monitor to the node directly, using the cv-console command-line utility, or by logging into the node using the "ssh" user:

```
$ ssh ssh@node01
```

The password in this mode is "breakin" (without the quotes) and will present the output from the cloning process. However, monitoring the process is not required since the node will automatically change its netboot back to "local" and reboot once the process is completed. At this point, the cloning process has been completed for the node.

7.2 Cloner sub-commands

The cv-cloner utility is unique compared to the other ClusterVisor command line utilities due to it being a group of sub-commands that all operate under a single command, and each sub-command has its own set of flags. For this reason, in this section each sub-section will be explaining one of the sub-commands.

7.2.1 new-image

Once a node has been setup, to clone it across other nodes an image needs to be taken of the node's current state. This is done using the new-image sub-command. To create the image the sub-command needs two bits of information, the disk layout name and the image name, which are specified using the --disklayout and --image flags (respectively).

To help explain the purpose behind the image and disk layout, it helps to understand that whenever a cloner image is being applied to a node there are two primary operations occurring: partitioning the drives and copying over data. The disk layout in cloner will hold the instructions for how to partition the drives and the image will hold the instructions for how to migrate over the data. The reason for the separation is that a cluster will usually have many nodes with the same drives and partition setup, but may only differ in the software on each section of nodes.

Once the flags are set and the command is run, it will list the drive configurations it detected along with the directories that will be cloned alongside a prompt asking if everything looks correct. Assuming it is told to proceed, the utility will begin sending data to the cloner server to create the new image. Depending on how much data needs to be cloned, this may take some time.

> ⓘ If the compute nodes are storing any temporary or large amounts of data, be sure to check if they exceed the available space left on the cloner server. If it does, those directories can be excluded from the image using the flags below.

While only the --image and --disklayout flags are required for using the new-image sub-command, there are many other flags available for configuring the cloner image. Due to how many flags are available, only the flags whose names do not describe the entirety of their behavior or whose behavior is less obvious will be listed:

- --server - Can be used to point to a different cloner server by its IP address, primarily used in clusters with multiple cloner servers (otherwise the flag can be left out)
- --exclude-devices - Used to specify a list of devices (e.g. /dev/sda) to be excluded during the image creation process, which is primary used in situations where either a node has a drive not included in other nodes or if an image should ignore the specified drives on the other nodes
- --exclude-paths - Like --exclude-devices, but can specify a list of paths to not include in the image, which can be helpful to exclude directories with large amounts of temporary data
- --dry-run - Will perform a dry run of creating the new cloner image, i.e. it will perform all the actions that would collect the data needed for the image without actually creating it and sending it to the cloner server (primarily used to verify if all chosen settings are correct)

7.2.2 update-filesystem

If an image already exists and the cloned data needs to be updated (without also updating the disk layout), this can be done using the update-filesystem sub-command. The only flag needed to run this sub-command is the --image flag, which specifies the image being updated with the data on the node. Similar to creating an image, this may take some time to complete. Aside from the disk layout flags, all of the other flags from new-image are also available to update-filesystem. However, it also has its own unique flag, --update-conflict, which is used to determine how conflicts should be handled. The --update-conflict options are as follows:

- prompt - This is the default option if none is specified and will provide a user prompt to handle each conflict manually
- server - Any conflicts will default to what is on the server (regardless of what other command line arguments are passed)
- commandline - Any conflicts will default to what was specified through the other command line arguments (and may overwrite what already exists on the cloner server)

7.2.3 update-disklayout

Similar to the update-filesystem, the update-disklayout sub-command will only update the disk layout settings already saved on the cloner server with what is currently on the node, but leave the already cloned data as-is. Do note, unless if the partitioning on the nodes has changed and/or if a drive is being added/removed/replaced on each node, then update-disklayout is not needed and update-filesystem is instead what should be used for updating an image with changes. To utilitize the sub-command, both the --image and --disklayout flags are needed to specify both the image name and disk layout name, respectively, that is being updated. Lastly, aside from any filesystem related flags, all other flags from new-image are available to update-disklayout along with --update-conflict (which works the same way as it does for update-filesystem).

7.2.4 update-options

When a cloner image is created, the server will store the options that were used to create the image. This is used during subsequent updates to determine if the updates have made any changes to the options and whether or not a conflict has occurred. The update-options sub-command is used to update the options of the cloner image stored on server without updating the cloned data. It operates using all of the same flags used by update-filesystem (but without --update-conflict).

7.2.5 multicast-status

When one or more nodes have booted into the "cloner-multicast" netboot image they will check every 30 seconds to see if the multicast server is ready or not. To check to see which nodes are queued, the multicast-status sub-command can be used to list all of the nodes waiting. Once they are ready, the multicast-image sub-command can be used to start the multicast server for the selected nodes.

7.2.6 multicast-image

Trying to apply a cloner image to a large number of nodes simultaneously may result in lots of latency or overwhelming the cloner server. To get around this issue, a cloner image can be deployed using multicast (as opposed to unicast) through the multicast-image sub-command. Assuming that the nodes are already booted to the "cloner-multicast" (which can be checked using the multicast-status sub-command), running the multicast-image sub-command using the --image flag to point to the desired cloner image and the selected nodes to expect will start the multicast server for those nodes. Similar to a unicast clone, once a node is finished it will automatically set its netboot back to "local" and reboot itself.

7.2.7 identify

This sub-command will obtain the cloner image data for the image assigned to that node. Optionally, it can be used to query other nodes by using the --method flag to specify whether a search using MAC addresses or nodes will be used, and respectively use the --macaddr or --node flag to provide the comma-separated list of MAC addresses / node names to query.

7.2.8 config

This sub-command provides the configuration details for the image and disk layout provided using the respective --image and --disklayout flags.

7.2.9 script

For unique setups that may require certain changes to be made before or after the cloning process is done, cloner has the ability to provide pre-scripts and post-scripts that it will run automatically. The script sub-command is used to manage these scripts and will always need the --image flag to specify which image the script applies to and --type to indicate whether a "pre" or "post" script is being used. The other flags it provides are:

- --set - To specify the file name of the script which will be uploaded to the server for the image to use
- --get - To specify an already uploaded script to download from the server

7.2.10 image-list

A single purpose sub-command that will list all of the existing cloner images already on the server.

7.2.11 image-details

Provides the details of the cloner image specified using the --image flag.

7.2.12 disklayout

The details for a disk layout can be obtained using the disklayout sub-command by providing the cloner image and its disk layout using the respective --image and --disklayout flags. If this is being done to change those settings, use the --output flag to specify the file name to store the settings in for using later with the disklayout-upload sub-command.

7.2.13 disklayout-upload

In case any manual adjustments need to be made to a disk layout, the disklayout sub-command can be used to export the settings and then re-uploaded using the disklayout-upload sub-command. This is done by using the --image and --disklayout flags to respectively specify the image and disk layout being changed and the --file flag to specify the file name of the revised disk layout settings.

7.2.14 biossettings

The biossettings sub-command is used to pull the BIOS settings from an image that has already been uploaded using the biossettings-upload sub-command. The flags needed to obtain these settings is --image for the name of the cloner image and --biossettings for the name given to the BIOS settings during the upload. Additionally, the settings can be saved to a file rather than just bring displayed by using the --output flag to specify the name of the file to create.

7.2.15 biossettings-upload

Nodes with Intel motherboards provide a tool called syscfg to query and change the BIOS settings of the node from the OS, although, a reboot will still be required for them to take effect (see here[15] for more information on this). Rather than using this tool to manually change the BIOS settings of every Intel node the BIOS settings can be pulled from a node, changed accordingly, and then uploaded to a cloner image. This will instruct cloner to apply those BIOS changes to the node during the cloning process. All that is needed is the name of the cloner image using the --image flag, the desired name for the BIOS settings being uploaded using the --biossettings flag, and the name of the exported BIOS settings file using the --file flag.

7.2.16 make-installer

When cloner is applying a cloner image to a node, it helps to have the kernel and initrd of the cloner boot image match what is on the nodes. The make-installer sub-command is used to do exactly that, so it only needs to be run whenever the kernel and/or initrd on a node has changed. If the installer is initially being created no flags need to be passed, but if the kernel / initrd is being updated then the --overwrite flag needs to be passed.

15 https://www.intel.com/content/dam/support/us/en/documents/motherboards/server/sb/g88217001_syscfg_userguide_rev04.pdf

8 8. All command line utilities

The following is an overview of each of the ClusterVisor command line utilities outlining their purpose and usage (listed in alphabetical order).

8.1 cv-actutil-import

This is used to take the *act_nodes.conf* file from actutils2 and migrate those nodes into ClusterVisor. To use the tool, simply point to the configuration file and optionally use the --convert-eth flag to convert any *ethX* interfaces to be called *enX* instead.

8.2 cv-authsync

This is used to synchronize native users if ClusterVisor's LDAP server is not being used to manage users. For setup and additional information on how to use the utility, see section Synchronizing native users(see page 0) of this guide.

8.3 cv-auth-watchd

This is the daemon used with cv-authsync to automate running the utility whenever changes are made to the native users on the source node (which is explained further in section Synchronizing native users(see page 0) of this guide). The following flags also be can be used to adjust the behavior of the daemon:

- --debug - This can add additional debugging information.
- --authsync - This takes a path as an argument and is used to point to the cv-authsync executable if it exists in an unconventional location.
- --delay - This takes a number of seconds as an argument and can be used to have the daemon wait longer in between calling cv-authsync. This is used for when the watched files are changed rapidly and will throttle how often cv-authsync is called after each of those changes.

Do note, this command will usually not be called directly and one should instead just use the systemd **cv-auth-watchd** daemon.

8.4 cv-clientd

This is ClusterVisor's client daemon which sends and receives data from the ClusterVisor server daemon. By default, this will operate over the TCP port 13202, but it can be changed using cv-conf under *config.global.cv-clientd.listen_port*. The only configuration option the daemon provides is from the --no-reconfigure flag which will prevent it from re-configuring the node immediately after the daemon is up and running (which can be useful when troubleshooting ClusterVisor).

Do note, this command will usually not be called directly and one should instead just use the systemd **cv-clientd** daemon.

8.5 cv-cloner

Due to the complexity of ClusterVisor's cloning utility, it has its own section dedicated to explaining it and all of its sub-commands / flags. For more information, see the section Using cloner(see page 0) of this guide.

8.6 cv-cockpit-helper

The web interface for ClusterVisor is built atop Red Hat's Cockpit project[16] and uses the cv-cockpit-helper utility to help facilitate interactions between the ClusterVisor server and Cockpit. Due to the nature of the utility, it will only pass back all data in JSON[17]. Everything this utility does can be accomplished more easily by the other ClusterVisor command line utilities, so this utility should not be used directly.

8.7 cv-commit

This is used to commit saved changes in ClusterVisor's configuration so that the nodes can be reconfigured against the changes. The following controls are available through the flags:

- --show - Will display the queued changes that have not yet been committed
- --commit - Triggers the queued changes to be committed
- --status - Will display the status of any changes that failed to be applied to the nodes
- --clear - Will clear the failed changes from the queue

8.8 cv-conf

Any changes made to ClusterVisor's configurations can be made using cv-conf. The details of how to use the bulk of the utility can be found in the section Configuring nodes(see page 0) of this guide under the "Using the command line" sub-sections. The only feature not covered in those sections is the --ignore-lastupdate flag, which can be used to force the utility to save the changes made (even if someone else may have saved their own changes in the meantime). For this reason, that flag should be used only if absolutely necessary and otherwise avoided.

8.9 cv-console

Sometimes an SSH connection to a node is not enough and it would be help to actually see what would actually be displayed on a monitor for a node. The cv-console utility provides this feature by using the serial console provided through IPMI and allows for both observing and interacting with the node remotely (e.g. changing boot options in GRUB). While this command requires no arguments outside of the node selector(s) the other flags available are the following:

- --timeout - Determines how many seconds the utility will wait before giving up on making a connection to a node
- --escape-char - Which escape character will close the console session
- --serial - Instead of copying the files in parallel (the default), it will do one at a time for each node

8.10 cv-cp

The cv-cp utility can be used to copy anything stored on the current node to other nodes. The other nodes can be selected using the node selector flags (explained further in the "Using the command line" sub-section of the Accessing ClusterVisor(see page 0) section of this guide); and the source of the location and the desired destination location on the other nodes are specified, respectively, using the positional arguments of the utility. The other controls for the utility can accessed through the following flags:

16 https://cockpit-project.org/
17 https://json.org

- --timeout - Determines how many seconds the utility will wait before giving up on making a connection to a node
- --recursive - Used when copying directories, will recursively copy everything contained within the directory
- --preserve - Will apply both the ownership and permissions of the source file/directory to the copy
- --verbose - Displays all of the files/directories being copied to the nodes
- --serial - Instead of copying the files in parallel (the default), it will do one at a time for each node

8.11 cv-db-image

This provides a means to import to / export from the database. Regardless of whether importing or exporting, one of the requiring flags is needed to specify the connection info:

- --database-uri - Explicitly provides the connection URI of the MongoDB database
- --config-file - Points to the cv-serverd.conf file to extract the MongoDB connection information and database used
- --auto-detect - Will automatically detect the ClusterVisor database installed

Unless if the non-default database is being used or the config file has been moved from its default location, the --auto-detect flag is the correct choice. The --import-from and --export flags can then be used to specify the path to the file to import from or desired file name and path to export the database as a file to, respectively. Lastly, if importing over an existing database the --overwrite flag needs to be used to signal that the database should be overwritten.

8.12 cv-dumpcfg

The cv-conf utility has --dump to dump the entire contents of all configurations, cv-db-image can export the database as an image that can be used to import from at a later point, and cv-dumpcfg can be used to dump specific portions of the configurations in different formats. The scope of how much the utility will dump is determined by using one of the following:

- --collection - Will dump an entire collection
- --section - Will dump a specific section from a collection
- --all - Will dump all configurations

The --format flag is used to specify what format to dump the configurations as (YAML[18], TOML[19], JSON[20], or Apache config[21]), and --type is used to determine if the *config* (the raw contents from the database) or *expanded* (the more verbose, post-processed form) type should be used. Using the *expanded* type can be particularly helpful for debugging the value of each node after all of the values are expanded and variables are replaced.

8.13 cv-exec

This command can be remotely executed across the nodes in the cluster using cv-exec. The positional arguments are used to specify the command and its arguments to run on the nodes, although quotes should be wrapped around the command if Unix pipes should be included with the command and not process the response from the cv-exec command (e.g. cv-exec -g nodes dmesg | grep error is not the same as cv-exec -g nodes "dmesg | grep error"). The following flags can be used to further control the cv-exec command:

- --format - Changes the format for how the results from the command is displayed

18 https://yaml.org
19 https://github.com/toml-lang/toml/blob/master/README.md
20 https://json.org
21 https://httpd.apache.org/docs/2.4/configuring.html

    - inline - Displays the node name and the output on the same line
    - header - Places the node name and the output on a different line
    - consolidate - Presents nodes with the same output as a single consolidated name to remove duplicate entries
    - realtime - Displays the output from nodes as they complete rather than all at once
    - json - Will present the response back as JSON so it can be parsed by another program
- --timeout - Determines how many seconds the utility should wait before giving up on making a connection to a node
- --serial - Instead of executing the commands in parallel (the default), it will do one at a time for each node
- --serialdelay - How many seconds to wait between running the command on each node in serial mode

8.14 cv-external-ipmi-statsd

By default, the cv-clientd daemon will collect the IPMI statistics from the node it is running on and dispatch it to the node running the ClusterVisor server. However, some other devices (e.g. node chassis) may also have IPMI data and to aggregate that data the cv-external-ipmi-statsd utility needs to be running on a node (preferrably, but not necessarily, the same one running cv-serverd). It will gather all of the known IPMI connection details for all of the non-node devices in the cluster and periodically collect and dispatch those statistics to the ClusterVisor server. The only flag it has is the --daemon that will run the process in background as a daemon.

8.15 cv-identify

To help make it easier to locate a single node from its peers, almost all chassis / sleds come with an ID LED on both the front and back of the node. This LED that can be enabled / disabled by either physically pressing the LED or issuing an IPMI command. The cv-identify utility allows for one or many nodes to have their ID LED enabled / disabled by using either the "on" or "off" actions, respectively. There is, unfortunately, no easy way to query the current status of the ID LED, but issuing the "off" action on all of the nodes using the --all node selector flag will disable any lingering ID LEDs.

8.16 cv-installer

The cv-installer is used to install and setup ClusterVisor. If launched without any arguments, it will enter an interactive prompt to guide the user through the install/setup options. However, to automatically select all of the default options in the installer the --auto-install flag can be passed to automatically fill out those questions. The following flags can also be used to control the installation process:

- --config-file - Used to optionally point to the cv-serverd.conf file to automatically obtain the MongoDB connection info being used for the install / setup (if left blank it will be asked in a prompt later or automatically populated if using --auto-install)
- --database-uri - Can optionally specify the MongoDB connection URI desired for ClusterVisor's database install target (if left blank it will be asked in a prompt later or automatically populated if using --auto-install)
- --skip - Will skip over sections of the installer (particularly if running the installer a second time or if only part of the installation is needed); the sections being *munge*, *389ds*, *influx*, and *mongo*

The sections that the installer can install and setup, along with what is being done, are outlined below:

- munge - Because munge is a dependency of ClusterVisor it is already installed; so the installer will only setup munge by creating the munge key, setting the correct permissions to the key, and starting the munge daemon

- 389ds - This will install the 389 directory server[22] for ClusterVisor, along with setting up the ClusterVisor domain controller and the directory manager user
- influx - InfluxDB is needed for storing the stats data from the nodes; so this section will add the InfluxDB repository, install the InfluxDB package, and setup the user credentials and security settings for the Influx database
- mongo - MongoDB is needed as the primary database for ClusterVisor; so this section will add the MongoDB repository, install the MongoDB package, and setup the user credentials and security settings for the Mongo database

8.17 cv-monitoring

The monitored stats from the nodes in the cluster can be viewed via the cv-monitoring utility. By default, they will be displayed with the monitored stat as the header and the value of that stat for each node listed below the stat. This can be inverted to see each node as the header and its respective monitored stat values listed beneath it by using the flag --format=bynode.

8.18 cv-netboot

While the **netboot** field of a node can be changed manually using cv-conf, it can also be quickly observed and changed using the cv-netboot utility. The current netboot value of the node(s) can be queried using the --get flag while it can be set to a new value using the --set value, whose argument will be used to replace the existing **netboot** value of the node(s).

8.19 cv-nodenames

This simple utility is used to display all of the node names within the given node selection. For individual node selections this holds little value, but for selections using groups or chassis this can be helpful to see exactly what nodes are included in the selection. By default, each node name will be listed on its own line, but the behavior changed using the following flags:

- --delimiter - Specify what will separate each node name entry
- --format - This can be used to get the output in JSON[23] (by default it is in plain-text)

8.20 cv-power

The power for the node(s) can be remotely managed using the cv-power utility. Internally, this uses IPMI so it will work on the node as long IPMI is setup on the nodes and power is connected to the nodes. The following power operations can be performed by the utility:

- status - Query the current power status of the node(s)
- on-hours - How many hours the node(s) have been powered on
- restart-cause - The cause of the last restart performed by the node(s)
- on - Attempts to remotely power on the node(s)
- soft-off - Attempts to softly power off the node(s), is effectively the equivalent of pressing the power button once on a node
- off - Attempts to remotely power off the node(s), is effectively the equivalent of pressing and holding the power button on a node (only use if the node is locked without any other way to power off)
- reset - Attempts to remotely restart the node(s)

22 https://directory.fedoraproject.org/
23 https://json.org

- cycle - Attempts to remotely power cycle the node(s)

The behavior of the utility can be adjusted using the following flags:

- --debug - Will display the debug output from the utility
- --timeout - How many seconds the utility will wait for a response from the node before giving up (default is 5 seconds, but any duration between 2-60 seconds can be used)
- --format - Can change the format the output is presented in; "inline" to put the node name and output on the same line, "consolidate" to merge node names with identical outputs into a single entry, and "json" to present the output in JSON[24]
- --format - Changes the format for how the results from the command is displayed
    - inline - Displays the node name and the output on the same line
    - consolidate - Presents nodes with the same output as a single consolidated name to remove duplicate entries
    - json - Will present the response back as JSON[25] so it can be parsed by another program

8.21 cv-python

This is just the Python binary from the virtual environment[26] ClusterVisor uses to load some of its dependencies. Unless debugging ClusterVisor, there is no reason to use this.

8.22 cv-reconfigure

The nodes can be manually reconfigured using the current configurations in ClusterVisor using the cv-reconfigure utility. The only flag that changes the behavior of the utility is the --yes flag, which will automatically agree to any prompts from the utility rather than needing to manually fill them out.

8.23 cv-sel

The IPMI interface of a node will over time accumulate hardware events that have occurred on the node that can later be use for troubleshooting / diagnosing. IPMI calls this list of events the "System Event Log" (a.k.a. SEL) and they can be easily viewed from one or many nodes using the cv-sel utility. Running the utility only specifying the desired nodes to pull the entries from will list all of the events logged on each node, but this can be refined down by using the following flags:

- --level - By default all event levels are shown but can be filtered to display only "critical", "nominal", or "warning" level events
- --search - Will filter out all but entries that match the provided regular expression[27]
- --head - Will display the number of events specified starting from the top
- --tail - Will display the number of events specified starting from the bottom

The other flags available are:

- --timeout - How many seconds the utility will wait for a response from the node before giving up (default is 5 seconds, but any duration between 2-60 seconds can be used)
- --format - Changes the format for how the results from the command is displayed
    - inline - Displays the node name and the output on the same line
    - header - Places the node name and the output on a different line

24 https://json.org
25 https://json.org
26 https://docs.python.org/3/tutorial/venv.html
27 https://en.wikipedia.org/wiki/Regular_expression

    - json - Will present the response back as JSON[28] so it can be parsed by another program
- --clear - The SEL can only hold a limited number of lines and can be manually cleared if it is filled using this flag

8.24 cv-sensor

The IPMI interface of a node is always collecting data from its various sensors (e.g. temperatures and fan speeds) and the current values of those sensors can be queried using the cv-sensor utility for one or many nodes. Running the utility only specifying the desired nodes to pull the sensor data from will list all of the sensors on each node, but this can be refined down by using the following flags:

- --type - Will filter out all but sensors whose type matches the provided regular expression[29](e.g. temperatures and fans)
- --level - By default all sensor levels are shown but can be filtered to display only "critical", "nominal", or "warning" level sensors
- --name - Will filter out all but sensors whose name matches the provided regular expression[30]
- --value-gt - Will filter out all but sensors whose value is greater than the provided value
- --value-lt - Will filter out all but sensors whose value is less than the provided value

All other flags available are:

- --timeout - How many seconds the utility will wait for a response from the node before giving up (default is 5 seconds, but any duration between 2-60 seconds can be used)
- --format - This can be used to get the output in JSON[31] (by default it is in plain-text)
- --thresholds - Will also display the threshold for each sensor before they are in a critical state
- --value-mode - Can be used to only show the values rather than highlighting them

8.25 cv-serveradm

Changes can be made to the internal settings of the ClusterVisor server daemon, cv-serverd, using the cv-serveradm utility. This tool is primarily used for debugging problems with cv-serverd without needing to restart it, which can be done using the following flags:

- --dump-cache - Can present the plugin data on the server for either the cache of fields that will trigger plugins or the cache of init instructions that gets sent to each node on startup (or both if "all" is passed)
- --flush-cache - Will instruct the server to flush out the cache of either the trigger or init cache and will then re-generate it (do note, this task will take a few seconds to complete on the server end and will be CPU intensive during that time)
- --field-triggers - Will present the fields that trigger plugins from a given collection
- --plugin-cache - Will present the plugin instructions cached for a given node
- --debug-level - Can query or change the debug level (by default it should be 0, anything above 0 will enable verbose logging of exceptions)
- --mem-dump - Can query, enable, or disable the memory dump mode of the server's logging (by default it should be disabled)
- --function-timer - Can query, enable, or disable the function timer mode of the server's logging (by default it should be disabled)

This tool can also be used to manually send HTTP GET or POST requests to the server using the following flags:

- --get-api - Will return the response from the server given the URL for an HTTP GET route from the server's API

28 https://json.org
29 https://en.wikipedia.org/wiki/Regular_expression
30 https://en.wikipedia.org/wiki/Regular_expression
31 https://json.org

- --post-api - Will return the response from the server given the URL for an HTTP POST route from the server's API

When using the API flags, the following flags can optionally be used:

- --output - Determines whether the response from the server will be returned as raw JSON, prettified JSON, or YAML (will use prettified JSON by default)
- --server - Manually specify what server address to connect to
- --port - Manually specify what server port to connect to
- --post-input - Point to a file with the HTTP POST data to send with the request
- --parallel - How many parallel requests should be made to the API
- --hide-time - Hides the timing output of the threads from the response

8.26 cv-serverd

This is the ClusterVisor server daemon, which will communicate with the other components that make up ClusterVisor (e.g. the client daemon, web interface, etc). By default, this will operate over the TCP port 13201, but can be changed in the configuration file under /etc/clustervisor/cv-serverd.conf by altering the value of *port* in the file. The daemon can provide further logging information by using the --debug flag to raise the debug level of the daemon (by default this is 0, but can be set from 1-4).

Do note, this command will usually not be called directly, and should instead just use the systemd **cv-serverd** daemon.

8.27 cv-stats

The stats from the nodes can be viewed using the cv-stats utility. It will display all of the stats collected by ClusterVisor for all of the selected nodes. To change the behavior of the utility, use the following flags:

- --regex - Use a regular expression[32] to filter the stats that are presented
- --format - Can change the format the results are presented in to be consolidate (where nodes with the same stat value will be consolidated into a single entry) or JSON[33]

8.28 cv-useradm

The users managed by ClusterVisor can be handled using the cv-useradm utility. The usage of this utility is explained in the "Using the command line" sub-sections of the Managing users(see page 0) section of this guide.

32 https://en.wikipedia.org/wiki/Regular_expression
33 https://json.org

9 9. Troubleshooting

A support ticket can be submitted to Advanced Clustering Technologies, Inc. by sending an email to support@advancedclustering.com[34]. It will help to provide the six-digit ACT serial number of the affected node, which can be located from any of the following methods:

- From the web interface: Navigate to the **Configuration** page, the nodes are listed under the **Node** tab. The serial number can be found by clicking **Edit** on the entry containing the affected node and locating its **Serial** entry under the **General** tab in the pop-up window.
- Using the command line utility cv-conf, the node's **serial** field can be viewed by running:

  ```
  $ cv-conf -e node.<nodename>
  ```

  Where <*nodename*> is the hostname of the affected node. Then search for the **serial:<nodename>** field, again where <*nodename*> is the hostname of the node.
- Located physically on the node, there will be a sticker on the front, top, and back of the node that displays the ACT serial number of the node.

34 mailto:support@advancedclustering.com

10 10. Release Notes

- In version 0.3.7, the command line utility cv-serveradm was added to make changes to the internal settings of the cv-serverd daemon without needing to restart it.
- In version 0.4.0, the **Netboot** field (found under the **Node** tab in the **Configuration** page) has been deprecated and moved to the **Boot/Power control** page (originally called the **Power control** page when it only handled power options). The network boot image of a node can also still be changed using the cv-netboot command line utility.
- In version 0.5.15, non-root users can use ClusterVisor in a read-only mode and specific users can be flagged as admin users in ClusterVisor to allow them to use ClusterVisor as if they were the root user (akin to how sudo works from the command line).