

System Shell Tools

In this article

- [Requirements Checker](#)
- [Services Manager](#)
- [Cluster Manager](#)
 - [Init Master](#)
 - [Master: Add new Slave](#)
 - [Slave: Promote to Master](#)
 - [Slave: Sync Files](#)
 - [Cluster Status](#)
- [Requirements Checker](#)
- [Services Manager](#)
- [Cluster Manager](#)
 - [Init Master](#)
 - [Master: Add new Slave](#)
 - [Slave: Promote to Master](#)
 - [Slave: Sync Files](#)
 - [Cluster Status](#)

This section describes **shell tools** required for management of the system at the very system level. All these tools are shipped with the VCS distributive. You need to use SSH or direct access to the server in order to run any of these tools. Please notice that most of given tools require **root** access and show inline help when run without arguments.

Important!

Please use these tools **only if you have a clear understanding** of what you are doing. Misuse of the tools may cause improper functioning of the system.

Attention

To free our clients from confusion while trying to indicate a path to VCS system, we introduce the **<VCS_PATH> variable** that differs depending on VCS version:

- **<VCS_PATH>** before version **3.13.0** is **“/usr/local/vcs”**
- **<VCS_PATH>** after version **3.13.0** is **“/opt/jerasoft/vcs”**

The example is as follows:
To restart RADIUS Server, you need to execute **<VCS_PATH>/bin/service bbradiusd restart**, which will look like **/usr/local/vcs/bin/service bbradiusd restart** for clients with VCS version **3.12.0 and lower** or **/opt/jerasoft/vcs/bin/service bbradiusd restart** for VCS version **3.13.0 and higher**.

Requirements Checker

The tool is aimed to check **minimal requirements** and **security recommendations** at your server before and after installation.

Usage

`<VCS_PATH>/bin/setup-checker`

The tool should be run under **root** permissions without any arguments. When the tool is run, it will perform a list of checks and show results for each of them. Use it:

1. Before installation of the VCS to check **minimal hardware and software requirements**. At this stage you may download the tool separately from the VCS.
2. After installation of the VCS to check **security recommendations**. Usually you need to tune your firewall settings according to the [VCS First Steps](#).

Attention

The tool checks only minimal requirements. **Real hardware requirements** highly depend on your traffic and deployment model.

Services Manager

The tool is used for **managing the VCS Services**. It allows to correctly start, stop, get status of the VCS Services as RADIUS Server, SIP Server, Calculator, etc.

Usage

```
<VCS_PATH>/bin/service <service-name> <action> [<options>]
```

The tool should be run under **root** or **vcs** user. Service manager takes 2 arguments:

Argument	Description
<service-name>	Short name of the service. Run the tool without arguments to get list of the services at your system.
<action>	Action to perform, one of the following: • <i>start</i> - start the service • <i>stop</i> - stop the service • <i>restart</i> - stop and then start the service • <i>reload</i> - force config re-read without restart • <i>status</i> - return current status of the tool

Additionally you may pass next options:

Option	Description
--path=<path>	Specify the VCS location. Only if different from <VCS_PATH>
--user=<user>	Specify user the VCS runs under. Only if different from "vcs".

For your convenience there are some examples below:

Restart RADIUS Server

```
<VCS_PATH>/bin/service bbradiusd restart
```

Start Files Downloader

```
<VCS_PATH>/bin/service files_downloader start
```

Get status of SIP Server

```
<VCS_PATH>/bin/service bbsipd status
```

Attention

The tool also complies with [LSB Init Scripts](#) standard. It allows to analyse exit code of each action in case of automated usage.

Cluster Manager

The tool is used to **manage clusters**. It allows to init the cluster, add and promote slave to master, etc.

Usage

```
<VCS_PATH>/bin/cluster <command> [<options>]
```

The tool requires "root" permissions. Cluster Manager takes command as mandatory argument and additional options. Some of commands may be run only on Master and others only on Slave. There are two available types of Slave: redundancy and reporting. All extra options for the commands are explained below in the corresponding subsection.

Init Master

The command is **run on Master** and used for Master to configure its parameters.

Bash

```
<VCS_PATH>/bin/cluster init-master --ip=<IP-of-master> [<extra-options>]
```

This command may require additional options:

Option	Description	Default
--ip=<ip>	IP Address of the Master server	none
--ssh-port=<port>	SSH Port at the master server	22
--data=<path>	PostgreSQL data path on master system	autodetect
--vcs-path=<path>	Path to the VCS on Slave	<VCS_PATH>

Master: Add new Slave

The command is **run on the Master** and used for:

- **first time initialization** of the cluster;
- **addition of the new slave** node to the cluster;
- addition of the **old master to work as slave** after failover.

Bash

```
<VCS_PATH>/bin/cluster init-slave --ip=<IP-of-slave> [<extra-options>]
```

This command encapsulates 4 sub-commands:

- configuration of the *Master* to ship replication logs (`init-master`)
- configuration of the *Master* to accept connections from the *Slave* (`init-master-access`)
- configuration of the *Slave* to receive replication logs (`init-slave`)
- show status of the cluster (`status`).

Each of these commands may be performed separately if you know exactly what you need.

This command may require additional options:

Option	Description	Default
--ip=<ip>	IP Address of the Slave server	none

--ssh-port=<port>	SSH Port at the slave server	22
--username=<login>	Login to the slave server	jerasupport
--data=<path>	PostgreSQL data path on slave system	same as master
--vcs-path=<path>	Path to the VCS on Slave	<VCS_PATH>
--role=<role>	Role of the secondary system. Could be [redundancy or reporting]	redundancy

Slave: Promote to Master

The command is **run on the Slave** and used to promote current *Slave* to *Master*.

Bash

```
<VCS_PATH>/bin/cluster promote
```

The command is used for failover. After you fix old *Master*, you may add it as new *Slave* using "*init*" command.



Attention

In case when you have more than 2 nodes in cluster, you need to re-init all other slaves from this new *Master*.

Slave: Sync Files

The command is **run on the Slave** and used to synchronize files from the *Master*. The command is mostly used in crontab for auto-sync.

Bash

```
<VCS_PATH>/bin/cluster sync-files
```

During initialization of the cluster tool adds this command to the crontab (*/etc/cron.d/vcs-cluster*) at the *Slave* node.

Cluster Status

The command may be run **on any node** and used to get status of the cluster.

Bash

```
<VCS_PATH>/bin/cluster status
```

When command is run on the *Master*, it shows type of the node and attached Slaves that are up to date:

```
[ NOTICE ] Cluster Status
ifconfig | grep 'inet addr:' | grep -v '127.0.0.1' | cut -d: -f2 | awk '{ print $1}'

Master (master ip): ACTIVE
Redundancy (slave ip): ACTIVE
```

When command is run on the *Slave*, it shows type of the node and synchronization delay.

```
[ NOTICE ] Cluster Status
ifconfig | grep 'inet addr:' | grep -v '127.0.0.1' | cut -d: -f2 | awk '{ print $1}'

Master (master ip): NOT AVAILABLE | Activity delay: -00:00:58.393423
Redundancy (slave ip): ACTIVE
```

Please don't worry if you see the following:

```
the row: Master (master ip): NOT AVAILABLE
```

It's a normal behavior. Instead, please pay attention to **Activity delay** parameter it shows large values if replication is broken and servers are not in sync.

In case when the *Slave* is not in synchronization (NOT-in-sync) with the *Master*, you need to re-initialize.

In this article

- [Requirements Checker](#)
- [Services Manager](#)
- [Cluster Manager](#)
 - [Init Master](#)
 - [Master: Add new Slave](#)
 - [Slave: Promote to Master](#)
 - [Slave: Sync Files](#)
 - [Cluster Status](#)
- [Requirements Checker](#)
- [Services Manager](#)
- [Cluster Manager](#)
 - [Init Master](#)
 - [Master: Add new Slave](#)
 - [Slave: Promote to Master](#)
 - [Slave: Sync Files](#)
 - [Cluster Status](#)

This section describes **shell tools** required for management of the system at the very system level. All these tools are shipped with the VCS distributive. You need to use SSH or direct access to the server in order to run any of these tools. Please notice that most of given tools require **root** access and show inline help when run without arguments.

Important!

Please use these tools **only if you have a clear understanding** of what you are doing. Misuse of the tools may cause improper functioning of the system.

Attention

To free our clients from confusion while trying to indicate a path to VCS system, we introduce the **<VCS_PATH> variable** that differs depending on VCS version:

- **<VCS_PATH>** before version **3.13.0** is **"/usr/local/vcs"**
- **<VCS_PATH>** after version **3.13.0** is **"/opt/jerasoft/vcs"**

The example is as follows:

To restart RADIUS Server, you need to execute **<VCS_PATH>/bin/service bbradiusd restart**, which will look like **/usr/local/vcs/bin/service bbradiusd restart** for clients with VCS version **3.12.0 and lower** or **/opt/jerasoft/vcs/bin/service bbradiusd restart** for VCS version **3.13.0 and higher**.

Requirements Checker

The tool is aimed to check **minimal requirements** and **security recommendations** at your server before and after installation.

Usage

```
<VCS_PATH>/bin/setup-checker
```

The tool should be run under **root** permissions without any arguments. When the tool is run, it will perform a list of checks and show results for each of them. Use it:

1. Before installation of the VCS to check **minimal hardware and software requirements**. At this stage you may download the tool separately from the VCS.
2. After installation of the VCS to check **security recommendations**. Usually you need to tune your firewall settings according to the [VCS First Steps](#).

Attention

The tool checks only minimal requirements. **Real hardware requirements** highly depend on your traffic and deployment model.

Services Manager

The tool is used for **managing the VCS Services**. It allows to correctly start, stop, get status of the VCS Services as RADIUS Server, SIP Server, Calculator, etc.

Usage

```
<VCS_PATH>/bin/service <service-name> <action> [<options>]
```

The tool should be run under **root** or **vcs** user. Service manager takes 2 arguments:

Argument	Description
<service-name>	Short name of the service. Run the tool without arguments to get list of the services at your system.
<action>	Action to perform, one of the following: • <i>start</i> - start the service • <i>stop</i> - stop the service • <i>restart</i> - stop and then start the service • <i>reload</i> - force config re-read without restart • <i>status</i> - return current status of the tool

Additionally you may pass next options:

Option	Description
--path=<path>	Specify the VCS location. Only if different from <VCS_PATH>
--user=<user>	Specify user the VCS runs under. Only if different from "vcs".

For your convenience there are some examples below:

Restart RADIUS Server

```
<VCS_PATH>/bin/service bbradiusd restart
```

Start Files Downloader

```
<VCS_PATH>/bin/service files_downloader start
```

Get status of SIP Server

```
<VCS_PATH>/bin/service bbsipd status
```

Attention

The tool also complies with [LSB Init Scripts](#) standard. It allows to analyse exit code of each action in case of automated usage.

Cluster Manager

The tool is used to **manage clusters**. It allows to init the cluster, add and promote slave to master, etc.

Usage

```
<VCS_PATH>/bin/cluster <command> [<options>]
```

The tool requires "root" permissions. Cluster Manager takes command as mandatory argument and additional options. Some of commands may be run only on Master and others only on Slave. There are two available types of Slave: redundancy and reporting. All extra options for the commands are explained below in the corresponding subsection.

Init Master

The command is **run on Master** and used for Master to configure its parameters.

Bash

```
<VCS_PATH>/bin/cluster init-master --ip=<IP-of-master> [<extra-options>]
```

This command may require additional options:

Option	Description	Default
--ip=<ip>	IP Address of the Master server	none
--ssh-port=<port>	SSH Port at the master server	22
--data=<path>	PostgreSQL data path on master system	autodetect
--vcs-path=<path>	Path to the VCS on Slave	<VCS_PATH>

Master: Add new Slave

The command is **run on the Master** and used for:

- **first time initialization** of the cluster;
- **addition of the new slave** node to the cluster;
- addition of the **old master to work as slave** after failover.

Bash

```
<VCS_PATH>/bin/cluster init-slave --ip=<IP-of-slave> [<extra-options>]
```

This command encapsulates 4 sub-commands:

- configuration of the *Master* to ship replication logs (`init-master`)

- configuration of the *Master* to accept connections from the *Slave* (`init-master-access`)
- configuration of the *Slave* to receive replication logs (`init-slave`)
- show status of the cluster (`status`).

Each of these commands may be performed separately if you know exactly what you need.

This command may require additional options:

Option	Description	Default
<code>--ip=<ip></code>	IP Address of the Slave server	none
<code>--ssh-port=<port></code>	SSH Port at the slave server	22
<code>--username=<login></code>	Login to the slave server	jerasupport
<code>--data=<path></code>	PostgreSQL data path on slave system	same as master
<code>--vcs-path=<path></code>	Path to the VCS on Slave	<VCS_PATH>
<code>--role=<role></code>	Role of the secondary system. Could be [redundancy or reporting]	redundancy

Slave: Promote to Master

The command is **run on the Slave** and used to promote current *Slave* to *Master*.

Bash
<code><VCS_PATH>/bin/cluster promote</code>

The command is used for failover. After you fix old *Master*, you may add it as new *Slave* using "*init*" command.

Attention

In case when you have more than 2 nodes in cluster, you need to re-init all other slaves from this new *Master*.

Slave: Sync Files

The command is **run on the Slave** and used to synchronize files from the *Master*. The command is mostly used in crontab for auto-sync.

Bash
<code><VCS_PATH>/bin/cluster sync-files</code>

During initialization of the cluster tool adds this command to the crontab (*/etc/cron.d/vcs-cluster*) at the *Slave* node.

Cluster Status

The command may be run **on any node** and used to get status of the cluster.

Bash
<code><VCS_PATH>/bin/cluster status</code>

When command is run on the *Master*, it shows type of the node and attached Slaves that are up to date:

```
[ NOTICE ] Cluster Status
ifconfig | grep 'inet addr:' | grep -v '127.0.0.1' | cut -d: -f2 | awk '{ print $1}'

Master (master ip): ACTIVE
Redundancy (slave ip): ACTIVE
```

When command is run on the *Slave*, it shows type of the node and synchronization delay.

```
[ NOTICE ] Cluster Status
ifconfig | grep 'inet addr:' | grep -v '127.0.0.1' | cut -d: -f2 | awk '{ print $1}'

Master (master ip): NOT AVAILABLE | Activity delay: -00:00:58.393423
Redundancy (slave ip): ACTIVE
```

Please don't worry if you see the following:

```
the row: Master (master ip): NOT AVAILABLE
```

It's a normal behavior. Instead, please pay attention to **Activity delay** parameter it shows large values if replication is broken and servers are not in sync.

In case when the *Slave* is not in synchronization (NOT-in-sync) with the *Master*, you need to re-initialize.