Demand a Vagrant plugin within the Vagrantfile?

Asked 6 years, 6 months ago Active 3 months ago Viewed 21k times

76

20

Supposed execution of a `Vagrantfile` requires a specific Vagrant plugin to be installed. So, basically what you need to do is

```
$ vagrant plugin install foobar-plugin
$ vagrant up
```

If you skip the first step, `vagrant up` fails.

Is there an option in Vagrant to make it install the plugin automatically? In other words: Is it possible to specify within a `Vagrantfile` which plugins to install automatically before creating and booting up the machine?

vagrant

asked Oct 21 '13 at 11:01
Golo Roden
95.6k 68 232 347

12 Answers

Active Oldest Votes

16

2019 Update: Vagrant now has functionality to require plugins through the `Vagrantfile` via:

```ruby
Vagrant.configure("2") do |config|
  config.vagrant.plugins = "vagrant-some-plugin"

  # or as array:
  config.vagrant.plugins = ["vagrant-some-plugin", "vagrant-some-other-plugin"]

  # or as hash
  config.vagrant.plugins = {"vagrant-some-plugin" => {"version" => "1.0.0"}}
end
```

See https://www.vagrantup.com/docs/vagrantfile/vagrant_settings.html

answered Aug 29 '19 at 10:27
Moritz Gunz
502 4 14

UPDATE Aug 31, 2018: See @Starx's fix below for later versions of Vagrant (1.8 and above)

67

Here is version based on Louis St. Amour's solution together with Rob Kinyon's comment about re-exec if a new plugin was installeed, I use it successfully in my own setup:

```
required_plugins = %w(vagrant-share vagrant-vbguest...)

plugins_to_install = required_plugins.select { |plugin| not Vagrant.has_plugin? plugin }
if not plugins_to_install.empty?
  puts "Installing plugins: #{plugins_to_install.join(' ')}"
  if system "vagrant plugin install #{plugins_to_install.join(' ')}"
    exec "vagrant #{ARGV.join(' ')}"
  else
    abort "Installation of one or more plugins has failed. Aborting."
  end
end
```

edited Jan 7 at 3:31

answered Mar 2 '15 at 1:10
Amos Shapira
3,020 4 25 32

5 Definitely the best solution at the moment – Anis Mar 8 '15 at 15:54

1 I've updated the solution on April 20th to use only one "system" call to install all missing plugins. – Amos Shapira Apr 26 '15 at 4:51

4 This did not work. It went into an infinite loop of installing the plugins. Somehow the new vagrant process can not pick up the newly installed plugins in the parent vagrant process – Arif Akram Khan Jul 2 '15 at 22:58

1 Thanks for the update. I actually ended up following the solution suggested by mkuzmin here: github.com/mitchellh/vagrant/issues/4347. It involves using a plugin named vagrant plugins and then use some code (very similar to you) in Vagrantfile. – Arif Akram Khan Jul 9 '15 at 22:16

1 @SteveHenty I sympathise with your comment but moving this into a plugin will sort of defeat the purpose of the code (of just being able to `git clone...; cd ...; vagrant up` first time around). – Amos Shapira Apr 6 '16 at 21:56

51

Since I'm a Ruby dev, and Bindler is no longer being maintained, I found it most natural to just write some code at the top of my Vagrantfile to install required plugins if missing (e.g. before `Vagrant.configure` line)

The following works for me:

```
required_plugins = %w( vagrant-hostmanager vagrant-someotherplugin )
required_plugins.each do |plugin|
  system "vagrant plugin install #{plugin}" unless Vagrant.has_plugin? plugin
end
```

`system` , unlike using backticks, will echo the command to stdout, just as running the command yourself would. And this way I don't need yet another strangely named plugin or system to keep track of required plugins which can be updated by Vagrant anyway.

edited Jun 22 '16 at 23:55

answered Sep 18 '14 at 16:44

Louis St-Amour
3,697 1 24 25

8 You'll need to 'exec "vagrant #{ARGV.join' '}"' in order to restart the vagrant process with the plugin installed. – Rob Kinyon Nov 4 '14 at 21:20

1 Good point. In my case I'd hit an error, but see the installing plugin's output and know to re-run the command. Having it re-run automatically would be an even better enhancement. If re-writing it, perhaps I would check to see if any plugin was not installed, and if so, install the plugins first then re-run the script rather than check and install each plugin one at a time... – Louis St-Amour Nov 21 '14 at 11:03

2 It would be awesome to have `Vagrant.has_plugin?` command on the shell level or just have `vagrant plugin install` check if plugin is already there. – sakovias Sep 3 '15 at 14:49

1 Well you could always do something like `if [ $(vagrant plugin list | egrep 'vagrant-hostsupdater|vagrant-share' -c) == 2 ] ; then echo "All plugins installed." ; else echo "Missing plugin"; fi` but there's a reason nobody writes bash scripts if they can help it ;-) Maybe experiment with the vagrant plugin command further? – Louis St-Amour Sep 3 '15 at 17:43

48

As I pointed out on my answer to your other question, you can use bindler for installing a set of plugins specific to a project using a single command.

If bindler is installed and the required plugin is not, bindler will error out and will abort the process. There is also an open issue related to automatically installing plugins on `vagrant up` s but so far no one signed up for it yet.

If you don't want to use bindler, you can make use of `Vagrant.has_plugin?` (available on 1.3.0+) at the top of your `Vagrantfile` and error out if the required plugin is not installed.

Something like:

```
unless Vagrant.has_plugin?("vagrant-some-plugin")
  raise 'some-plugin is not installed!'
end

Vagrant.configure("2") do |config|
  config.vm.box = "box-name"
end
```

UPDATE: Bindler is no longer supported and no equivalent funcionality has been provided by Vagrant core as of May 11th, 2015

edited May 23 '17 at 12:34
Community ♦
1 1

answered Oct 22 '13 at 1:30
fgrehm
775 6 5

14 For future Googlers, please note that this answer is a little outdated. You can now specify your dependencies in your `Gemfile` under a `:plugins` group that Vagrant will check for. See my answer below for more details. – Jonathan Bender Dec 11 '14 at 21:20

2 `Gemfile` is intended for Vagrant plugin development. See github.com/mitchellh/vagrant/issues/8370 – mixel Mar 15 '17 at 17:18

10

Please note that as of Vagrant 1.5, you can specify your dependencies in your `Gemfile`. Per the [blog post on the update](#):

> Now, Vagrant 1.5 will automatically load any gems in the "plugins" group in your Gemfile. As an example, here is the Gemfile for a "vagrant-bar" plugin:
>
> ```
> source "https://rubygems.org"
>
> group :development do
>   gem "vagrant",
>     git: "https://github.com/mitchellh/vagrant.git"
> end
>
> group :plugins do
>   gem "vagrant-foo",
>   gem "vagrant-bar", path: "."
> end
> ```

edited Jan 15 '15 at 22:20

answered Dec 11 '14 at 21:18
Jonathan Bender
1,771 3 18 37

8 Isn't that blog post referring to Vagrant *plugin* development? Vagrant *boxes* do not generally have Gemfiles, they use Vagrantfiles. – Don McCurdy Jan 15 '15 at 18:47

1 You are correct that Vagrant boxes do not have Gemfiles per se (you could have a Vagrantfile that did not require any plugins), but if you *are* using plugins (the context of the blog was as dependencies for your own plugin, but the same holds true for Vagrantfiles), you should be using a Gemfile to specify those requirements. – Jonathan Bender Jan 15 '15 at 19:34

7 Thanks, that's helpful. Since many Vagrant users are not ruby devs and will not already have a `Gemfile`, would you mind explaining how you set that up? I've created one similar to your example, but `vagrant up` isn't automatically loading anything. Tried `bundle install` but that's giving a prompt about system Rubygems permissions, which doesn't sound right. – Don McCurdy Jan 15 '15 at 21:26

1 I'm guessing that you're working in OSX using the stock ruby install, and running into this problem: [stackoverflow.com/questions/14607193](#)? – Jonathan Bender Jan 15 '15 at 22:18

1 In the world of chef cookbooks, very few will have a gemfile – mr.buttons Apr 24 '15 at 21:55

6

Couldn't add a comment to Louis St-Amour's answer, but I wanted to post this just in case anyone needed help extending his solution.

```
# Check for missing plugins
required_plugins = %w(vagrant-list)
plugin_installed = false
required_plugins.each do |plugin|
  unless Vagrant.has_plugin?(plugin)
    system "vagrant plugin install #{plugin}"
    plugin_installed = true
  end
end

# If new plugins installed, restart Vagrant process
```

```
if plugin_installed === true
  exec "vagrant #{ARGV.join' '}"
end
```

answered Mar 18 '15 at 15:16

James Reimer
186 1 6

// , Note that `vagrant-list` in the above is an example of a plugin, not a necessary part of the code. You can see other Vagrant plugins here: vagrant-lists.github.io – Nathan Basanese Mar 9 '18 at 20:31

3

On the new version of Vagrant, answer by @Amos Shapira gets stuck in an infinite loop. The reason for that is each call to `vagrant plugin install` also processes the `Vagrantfile` and when processed executes the code relating to installing the plugin again and again and so on.

Here is my solution which avoids the loop.

```
# Plugins
#
# Check if the first argument to the vagrant
# command is plugin or not to avoid the loop
if ARGV[0] != 'plugin'

  # Define the plugins in an array format
  required_plugins = [
    'vagrant-vbguest', 'vagrant-hostmanager',
    'vagrant-disksize'
  ]
  plugins_to_install = required_plugins.select { |plugin| not Vagrant.has_plugin? plugin
}
  if not plugins_to_install.empty?

    puts "Installing plugins: #{plugins_to_install.join(' ')}"
    if system "vagrant plugin install #{plugins_to_install.join(' ')}"
      exec "vagrant #{ARGV.join(' ')}"
    else
      abort "Installation of one or more plugins has failed. Aborting."
    end

  end
end
```

answered Aug 20 '18 at 6:17

Starx
69.1k 40 166 247

1

I just noticed here http://docs.vagrantup.com/v2/plugins/packaging.html an instruction

```
Vagrant.require_plugin "vagrant-aws"
```

which does exactly the same thing as what descibed fgrehm: raising quickly an error if the plugin is not installed.

As far as I know, there are stil no way to auto-install plugins

answered Feb 13 '14 at 13:57
Julien
27 1

5 Vagrant.require_plugin has been deprecated on 1.5+ – eis May 7 '14 at 20:41

1 Here's the official blog post about the deprecation: vagrantup.com/blog/vagrant-1-5-plugin-improvements.html From my limited understanding, it was deprecated because it's no longer necessary, Vagrant will now automatically check plugin dependancies. Not fully confident that I understand the blog post though... – Jeff Widman Jul 2 '14 at 0:29

The post talks automatically checking *plugin* dependencies, but I haven't seen anything about dependencies of the Vagrant box itself. – Don McCurdy Jan 15 '15 at 19:06

`Vagrant.require_plugin` is deprecated. Checked with Vagrant `1.7.4`. – czerasz Aug 17 '15 at 14:31

1

My answer is very close to Louis St-Amour's answer, but instead of installing plugins automatically, it just raises an error message, so that the user has to install the plugin manually.

I would rather users be aware of any plugins that get installed, because they apply globally to all Vagrant instances, not just to the current Vagrantfile.

Put at the top of `Vagrantfile` one line like this for each plugin, in this example, `vagrant-vbguest`:

```
raise "vagrant-vbguest plugin must be installed" unless Vagrant.has_plugin? "vagrant-vbguest"
```

edited May 23 '17 at 12:18
Community ♦
1 1

answered Apr 29 '15 at 9:29
Flimm
78.4k 28 169 182

1

You could use this project (I am the author): https://github.com/DevNIX/Vagrant-dependency-manager

It's based on similar answers but trying to be more complete and stable. The big advantage of this idea is, you can distribute your Vagrantfile and it will run on every computer regardless of the installed plugins on that environment.

It is easy to use:

1. Copy dependency_manager.rb next to your Vagrantfile
2. Include it and call `check_plugins` passing your dependencies as an array

   Example:

```
# -*- mode: ruby -*-
# vi: set ft=ruby :

require File.dirname(__FILE__)+"./dependency_manager"
```

```
check_plugins ["vagrant-exec", "vagrant-hostsupdater", "vagrant-cachier", "vagrant-
triggers"]

Vagrant.configure(2) do |config|

  config.vm.box = "base"

end
```

3. ???
4. Profit!

I would love to merge pull requests, fix any issue you could have, and to get ideas of new features. Currently I'm thinking about updating the dependency manager itself, and requiring specific plugin versions :D

Regards!

answered Aug 26 '16 at 9:09
Dev_NIX
97 14

0

I got a problem with new install of Vagrant, where .vagrant.d directory is not created yet. I was able to make the snippet from Luis St. Amour working by catching the exception.

```
required_plugins = %w(vagrant-share vagrant-vbguest)

begin
    plugins_to_install = required_plugins.select { |plugin| not Vagrant.has_plugin?
plugin }
    if not plugins_to_install.empty?
        puts "Installing plugins: #{plugins_to_install.join(' ')}"
        if system "vagrant plugin install #{plugins_to_install.join(' ')}"
            exec "vagrant #{ARGV.join(' ')}"
        else
            abort "Installation of one or more plugins has failed. Aborting."
        end
    end
rescue
    exec "vagrant #{ARGV.join(' ')}"
end
```

answered Mar 16 '16 at 17:15
Guillaume Hardy
1 2

On Windows this gives me the error "Bringing machine 'default' up with 'virtualbox' provider... but another process is already executing an action on the machine" because `up` is being run by `up`. Is there any way to avoid this without running `vagrant up` twice? – Tom B Sep 16 '16 at 11:56

Here's what I am using on Vagrant 1.8 and it is working fine. Put this somewhere before the configure block in your Vagrantfile.

0

```
# install required plugins if necessary
if ARGV[0] == 'up'
    # add required plugins here
    required_plugins = %w( plugin1 plugin2 plugin3 )
    missing_plugins = []
    required_plugins.each do |plugin|
        missing_plugins.push(plugin) unless Vagrant.has_plugin? plugin
    end

    if ! missing_plugins.empty?
        install_these = missing_plugins.join(' ')
        puts "Found missing plugins: #{install_these}.  Installing using sudo..."
        exec "sudo vagrant plugin install #{install_these}; vagrant up"
    end
end
```

answered Jun 15 '16 at 20:29
lps
1,025 8 20

It appears that's not cross-platform. – thorr18 Nov 26 '17 at 7:05

Highly active question. Earn 10 reputation in order to answer this question. The reputation requirement helps protect this question from spam and non-answer activity.