How To Install And Run A Node.js App On Centos 6.4 64bit

Authored by: ***ASPHostServer Administrator*** *[asphostserver@gmail.com]*
Saved From: *http://faq.asphosthelpdesk.com/article.php?id=222*

Node.js installation

Now that we're ready to install Node.js from sources. First, we'll move to /usr/src directory - the usual place to hold software sources.

```
cd /usr/src
```

Now, we pick the latest compressed source archive from Node.js website at http://nodejs.org/download/.

```
wget http://nodejs.org/dist/v0.10.4/node-v0.10.4.tar.gz
```

We could and should replace the url and use the more recent version of node.js, if there is one. Next, we are uncompressing the source files and move into that directory.

```
tar zxf node-v0.10.4.tar.gz
cd node-v0.10.4
```

Now the source for Node.js is extracted and we're in the source directory. We can now prepare our compiler commands, by executing the configure script:

```
./configure
```

It will read the properties of our system to prepare compiler flags. Ie. it could be a system architecture (32/64bit, CPU specific flags etc). With it, we're ready to actually compile the source now. To do that, just type:

```
make
```

This is probably the most time-consuming task here: on my example server it took about 6 minutes and 34 seconds to complete. When we're done, we need to make this available system-wide:

```
make install
```

The latest command will place the compiled binaries in system path, so all users could use it without any further setup. By default, node binary should be installed in /usr/local/bin/node.

Install Express.js

We now have Node.js installed and complete, we can start developing right away, deploy an already done application or we can proceed to create our Express.js application. First, we'll use npm, nodes' module manager, to install express middleware and supervisor - a helpful module that keeps our app started, monitors for file changes (ie. when we're developing the app) and restarts the server when needed.
UPDATE: To be able to run an executable in /usr/local/bin through sudo, you have add /usr/local/bin to your

secure_path using visudo.

```
sudo visudo
```

Look for secure_path, and append the following to it: ":/usr/local/bin". Once you have done that, you're now ready to install the express and supervisor modules.

```
npm -g install express supervisor
```

npm -g install will install the express and supervisor modules from npm software repository and make it available to the whole system. The -g switch in this command means "global" - the express and supervisor commands will be available accross the whole system.

Add non-privileged user

You should now, for security reasons, create a regular system user and run node under non-privileged account. To do this, add the user first. You can replace "exampleuser" with whatever name your prefer.

```
useradd exampleuser
```

We have a new system user. Add a decent password for the new user:

```
passwd exampleuser
```

Log out, and log back in as the new user.

This changes our login shell from root (system user) to exampleuser (non-privileged user who can compromise the system with less damage).

Creating an express app

Express is powerfull framework, and to create our first application, all we have to do is type:

```
express hello
```

The command will create a "hello" directory and setup some basics for a new application. Now we should enter this directory and install express dependencies:

```
cd hello && npm install
```

npm install part of the command will read all the module dependencies from a generated package.json file and install it from npm software repository. We should start a new screen session so we can leave node app running:

```
screen
```

Finally, we can start our application with the help of supervisor that we installed earlier.

```
supervisor app
```

Now we're able to access our first express app at your server IP. For example http://123.456.78.90:3000/.