Git bare server - Automatic deployment

> *Requirements*

- Git
- SSH

In this tutorial we're going to make an automated deployment process between our local machine and server code. For example, you're having local project that you want to version with `git` and having always fresh code on server when you do push, without ssh-ing to server always.

So there is basically two scenarios:

1. Creating local repository
2. Using existing Git repository (next time)

In the `first case`, when we're already having git server - what we need to do is to clone repository on our local machine for example, and on sever. After that, we need to add git webhook(s) whereas when we push some code, we want to build it on server. (Note, there is various softwares that're aready doing this process, like Jetkins but with more advanced options).

In `second case`, we're not having an existing git repo, so everything is on our local machine. What is needed here is, we have to add git to our

project by simple typing: `git init` and setup your git settings[1]. Then we need to create an "custom" git server - bare repository on the server and configure it.

In both cases, what we need done is, when we do `git push origin master` for example, we should obtain new code changes on server automatically. In tutorial section both cases will be explained detailed.

Tutorial

–

> *Creating local repository*

1. Setup password-less access to our server if not already done

2. SSH to your server

3. Choose a location for our newby git server :3

   - I would prefer placing it at `/opt/git/bare-repositories/` or on other safe place.

4. Create folder called by your project name and add `.git` extension to it, due consistency, so for example: `project1.git`

5. Cd to newby created folder and init git bare repo by running: `git init --bare project1.git`

6. On local machine, `cd` to your project folder and switch remote server to our custom git server by running:

```
git remote set-url origin ssh://username@server_ip/opt/git/bare-repositories/project1.git
```

if this one fails with `fatal: No such remote 'origin'` try with:

```
git remote add origin ssh://username@server_ip/opt/git/bare-repositories/project1.git
```

7. Verify origin by running: `git remote -v` it should be remote URL we've just added.

8. Now, let's upload our code to the server by pushing it:

```
git push origin master --force-with-lease
```

9. SSH back to server and clone our project to place where it will run, from our custom repo:

```
cd /var/www/html/
git clone /opt/git/bare-repositories/project1.git .
```

10. Security tip: *VERY IMPORTANT*

When we're done with cloning, and when we have our project runing at `/var/www/html/project1/` we're having `.git` folder within, which is still unprotected, and that means everyone inquisitive can read source code and much more! To avoid this, run:

```
chmod -R og-rx /var/www/html/project1/.git
```

Also, if you're running Apache2 web server you can add `.htaccess` file to the `.git` with `Deny from all` clause. But if you're running `nginx` , and these lines to your vhost.

11. Add automatic deployment webhook
    After we're done with cloning, it's time for fun :)

- SSH to server and go to our bare repository folder (`/opt/git/bare-repositories/`) and go to `hooks/` folder, then create file named `post-receive` by running:
  ```
  touch post-receive
  ```

- Paste this script but before that, update folder paths:

```
```
#!/bin/bash
#CONFIG
LIVE="/var/www/html/project1/"

read oldrev newrev refname
```

```
if [$refname = "refs/heads/master"]; then   # if commit
s are comming to to `master` branch
    echo "===== DEPLOYING TO LIVE SITE =====" # this is m
essage you will see when you do `git push`
    unset GIT_DIR
    cd $LIVE
    git pull origin master
    echo "===== DONE ====="
fi
```
```

- Make script executable by runnng:

  ```
  chmod u+x ./post-receive
  ```

12. Bon apettit! :bowtie:

    We are done with all setup. Now you can try adding some changes on your local project and commit & push it, and check changes live!

References

—

[1] Install Git

[2] Setup Git User

[3] Setting SSH keys

[4] http://www.saintsjd.com/2011/03/automated-deployment-of-

wordpress-using-git/