SparkView – Deploy Applications in the cloud

- Deploy, run, and test applications in the cloud with Deploy Agent and SparkView

Deploy, run, and test applications in the cloud with Deploy Agent and SparkView

Deploy Agent is an automatic deployment / testing tool. It deploys scripts, applications to a remote server, which has a SSH server running on, through the SparkView. So the client only knows the remote server name but nothing else.

To use the Deploy Agent:

Preparation:

1. This feature is disabled by default. Please set deployment = true in gateway.conf to enable it.
2. Generating SSH key pair: the public key and private key (if need, refer to: https://confluence.atlassian.com/bitbucketserver/creating-ssh-keys-776639788.html), and deploying the public key on the remote server as described in above link.
3. Send the private key, together with the user name used to create the key pair to the client.
4. On the client side, download executable Deploy Agent from RemoteSpark website (e.g.: on windows platform, it is deploy.exe).

Configure the remote server on the gateway:

On the gateway, add the remote server in the servers.json file. As a sample, the following lines can be added in the list of “connections” of the file servers.json to set up the remote server on the gateway:

```
"connections": [
  {
    "id": "xUbuntu_SSH", //the remote server name, which is the only information the client will know
    "displayName": "xUbuntu_SSH",
    "server": "10.0.0.42", //the IP of the remote server behind of the firewall
    "shadowing": false,
    "protocols": "rdp",
    "ssh": {
      "port": 3389,
```

```
      "username": "vmuser",
      "password": "password",
      "fontSize": 13,
      "mapClipboard": true,
      "sessionRecord": 0
    },
    "userUsages": []
  },
```

In above sample, we setup a remote server which supports RDP connection and shadowing. Make sure, for Deploy Agent tool, the remote server MUST have SSH server running on the port 22.

Again, the only information of the remote server a client can see is the server name. So it perfectly prevent exposing too much information behind the firewall to the external users.

Run Deploy Agent on the client side:

The client keeps the downloaded Deploy Agent executable (e.g.: deploy.exe) and private key file (usually the id_rsa file) in local machine.

There are two typical ways to run the Deploy Agent tool:

1. Command line with arguments:
   - Run help to check all available arguments:
     `> deploy.exe –h`
   - Sample of the comamnd line:
     `> deploy -f C:\\workspace\\lab\\ScriptGauge\\sourceFiles -e ". ./newrunme.sh" -g my_gateway -k c:\\users\\user1\\.ssh\\id_rsa -s xUbuntu_SSH -u vmuser -l false`
   - Note: the client must provide either the config file by using argument `–c` or all the rest seven arguments `-e –g –k –s –f –l –u`.
2. Run with config file:
   - The config file follows standard INI format. The following sample, seven attributes are setup, and all are mandatory:

     ```
     # Gateway information
     [Gateway]
     gatewayName=my_gateway
     sslEnabled=false

     # SSH user information
     [User]
     user=vmuser
     privateKeyPath=c:\\users\\user1\\.ssh\\id_rsa
     ```

```
# Remote server information
[Server]
serverName=xUbuntu_SSH

# Deployment information
[Deployment]
sourcePath=C:\\workspace\\lab\\ScriptGauge\\ sourceFiles
execute=. ./runme.sh
```

3. In case the client runs the Deploy Agent tool by specifying config file with `–c` argument, as well as some or all of the other seven arguments, the arguments in command line will override the values set in the config file.

After running the tool, on the client side, a new INI file “deploy.ini” will be generated within the spark_data directory under the user’s home directory. It keeps a random client ID assigned to this user. For example, a deploy.ini file can look like:

```
[Client]
clientId=spark_cgwtIzp //This is a random string assigned to the user as client ID
```

> Note: if the deploy.ini has already existed, the Deploy Agent will use the ID in it without generating a new one.

All files under the “source path” in the client machine will be uploaded to the remote server through the gateway. The files will be saved within a new folder with the name of the client ID under the user’s home directory. In above example, on the remote server, a new subdirectory “spark_cgwtIzp” is created under the home directory of the user “vmuser”, and all files under local “sourceFiles” folder will be uploaded there.

> Note: each time, before uploading new source files, all pre-existing files under that folder will be removed automatically.

After all files get uploaded, an executable, which is specified by `-e` argument or “execute” attribute in the config file above, will be performed. In above example, the command `. ./runme.sh` will run. The standard output, as well as error output, will be sent back to the client console or terminal.