CI-CD Jenkins JBoss/WildFly setup

- **Create an Instance and install Jenkins**
- **Create an AWS linux instance and install JBoss/Wildfly.**

  To configure Jboss/WildFly, login to your linux instance

  ec2-user

  sudo su -

Step 1. Install Java 8, then just run the commands below to install it.

```
sudo yum install java-1.8.0-openjdk-devel
```

Step 2: Download WildFly

```
sudo yum -y install wget
export WILDFLY_RELEASE="16.0.0"
wget https://download.jboss.org/wildfly/$WILDFLY_RELEASE.Final/wildfly-
$WILDFLY_RELEASE.Final.tar.gz
```

Once the file is downloaded, extract it.

```
tar xvf wildfly-$WILDFLY_RELEASE.Final.tar.gz
```

Move resulting folder to /opt/wildfly

```
sudo mv wildfly-$WILDFLY_RELEASE.Final/ /opt/wildfly
```

Step 3: Configure Systemd for WildFly

Let’s now create a system user and group that will run WildFly service.

```
sudo groupadd --system wildfly
sudo useradd -s /sbin/nologin --system -d /opt/wildfly  -g
wildfly wildfly
```

Create WildFly configurations directory.

```
sudo mkdir /etc/wildfly
```

Copy WildFly systemd service, configuration file and start scripts templates from the /opt/wildfly/docs/contrib/scripts/systemd/ directory.

```
sudo cp /opt/wildfly/docs/contrib/scripts/systemd/wildfly.conf /etc/wildfly/

sudo cp /opt/wildfly/docs/contrib/scripts/systemd/wildfly.service /etc/systemd/system/

sudo cp /opt/wildfly/docs/contrib/scripts/systemd/launch.sh /opt/wildfly/bin/

sudo chmod +x /opt/wildfly/bin/launch.sh
```

Set `/opt/wildfly` permissions.

```
sudo chown -R wildfly:wildfly /opt/wildfly
```

Reload systemd service.

```
sudo systemctl daemon-reload
```

Configure SELinux:

```
sudo restorecon -Rv /opt/wildfly/bin/
```

Start and enable WildFly service:

```
sudo systemctl start wildfly
sudo systemctl enable wildfly
```

Confirm WildFly Application Server status, it should be running

```
$ systemctl status wildfly
 ● wildfly.service - The WildFly Application Server
    Loaded: loaded (/etc/systemd/system/wildfly.service;
enabled; vendor preset: disabled)
    Active: active (running) since Wed 2019-04-03 16:22:58
```

Service should bind to port `8080`.

```
# ss -tunelp | grep 8080
```

Step 4: Add WildFly Users

WildFly 16 is now distributed with security enabled for the management interfaces. We need to create a user who can access WildFly administration console or remotely use the CLI. A script is provided for managing users.

Run it by executing the command:

```
     sudo /opt/wildfly/bin/add-user.sh
```

You will be asked to choose type of user to add. Since this the first user, we want to make it admin. So, choose a.

```
What type of user do you wish to add?
  a) Management User (mgmt-users.properties)
  b) Application User (application-users.properties)
 (a):
```

Provide desired username for the user. **Type a**

```
Enter the details of the new user to add.
Using realm 'ManagementRealm' as discovered from the
existing property files.
Username : devopprakashsuser
```

Set password for the user:

```
Password recommendations are listed below. To modify these
restrictions edit the add-user.properties configuration file.
The password should be different from the username
The password should not be one of the following restricted values
{root, admin, administrator}
```

```
The password should contain at least 8 characters, 1 alphabetic
character(s), 1 digit(s), 1 non-alphanumeric symbol(s)
Password : <Enter Password>
Re-enter Password :  <Confirm Password>
```

Press enter and agree to subsequent prompts to finish user creation.

```
What groups do you want this user to belong to? (Please enter a comma
separated list, or leave blank for none)[  ]: <Enter>
 About to add user 'devopsuser' for realm 'ManagementRealm'
 Is this correct yes/no? yes
 Added user 'devopsuser' to file '/opt/wildfly/standalone/configuration/mgmt-
users.properties'
 Added user 'devopsuser' to file '/opt/wildfly/domain/configuration/mgmt-
users.properties'
 Added user 'devopsuser' with groups  to file
'/opt/wildfly/standalone/configuration/mgmt-groups.properties'
 Added user 'devopsuser' with groups  to file
'/opt/wildfly/domain/configuration/mgmt-groups.properties'
 Is this new user going to be used for one AS process to connect to another
AS process?
 e.g. for a slave host controller connecting to the master or for a Remoting
connection for server to server EJB calls.
 yes/no? yes
 To represent the user add the following to the server-identities definition
```

Notice that:

User information is kept on: /opt/wildfly/domain/configuration/mgmt-users.properties

Group information is kept on: /opt/wildfly/standalone/configuration/mgmt-groups.properties

Step 5: Accessing WildFly Admin Console

To be able to run WildFly scripts from your current shell session, add `/opt/wildfly/bin/` to your $PATH.

```
cd /opt/wildfly/bin/

cat >> ~/.bashrc <<EOF
export WildFly_BIN="/opt/wildfly/bin/"
export PATH=\$PATH:\$WildFly_BIN
EOF
```

Source the bashrc file.

```
source ~/.bashrc
```

Now test by connecting to WildFly Admin Console from CLI with `jboss-cli.sh` command.

```
# jboss-cli.sh --connect
[standalone@localhost:9990 /] version
JBoss Admin Command-line Interface
JBOSS_HOME: /opt/wildfly
Release: 8.0.0.Final
Product: WildFly Full 16.0.0.Final
JAVA_HOME: /usr/lib/jvm/java-11-openjdk-11.0.ea.28-
8.el8.x86_64
java.version: 11-ea
java.vm.vendor: Oracle Corporation
java.vm.version: 11-ea+28
os.name: Linux
os.version: 4.18.0-32.el8.x86_64
```

Now exit (ctrl + d)

Accessing WildFly Admin Console from Web Interface

By default, the console is accessible on localhost IP on port 9990.

```
# ss -tunelp | grep 9990

tcp    LISTEN    0     50     127.0.0.1:9990  0.0.0.0:*
users:(("java",pid=6769,fd=404)) uid:999 ino:30407 sk:3 <->
```

We can start it on a different IP address accessible from outside the local server.
Edit `/opt/wildfly/bin/launch.sh` to look like this:

```
vi /opt/wildfly/bin/launch.sh
```

```
#!/bin/bash
if [ "x$WILDFLY_HOME" = "x" ]; then
    WILDFLY_HOME="/opt/wildfly"
fi
if [[ "$1" == "domain" ]]; then
    $WILDFLY_HOME/bin/domain.sh -c $2 -b $3
else
    $WILDFLY_HOME/bin/standalone.sh -c $2 -b $3 -bmanagement=0.0.0.0
fi
```

We added `-bmanagement=0.0.0.0` to start script line. This binds "management" interface to all available IP addresses. Restart wildfly service

```
sudo systemctl restart wildfly
```

Confirm

```
$ ss -tunelp | grep 9990
tcp     LISTEN    0   50      0.0.0.0:9990   0.0.0.0:*
users:(("java",pid=9496,fd=320)) uid:999 ino:73367 sk:c <->
```

Open ports on firewall

```
yum install firewalld -y
systemctl start firewalld
systemctl enable firewalld
sudo firewall-cmd --permanent --add-port={8080,9990}/tcp
sudo firewall-cmd --reload
```

Open your browser and URL http://<pulic IP of JBoss>:9990 to access WildFly Web console.

It will ask you for ID and password. Please provide ID and Password which you have created in previous steps. Services can be access from below url

http://<Public IP of JBoss>:8080

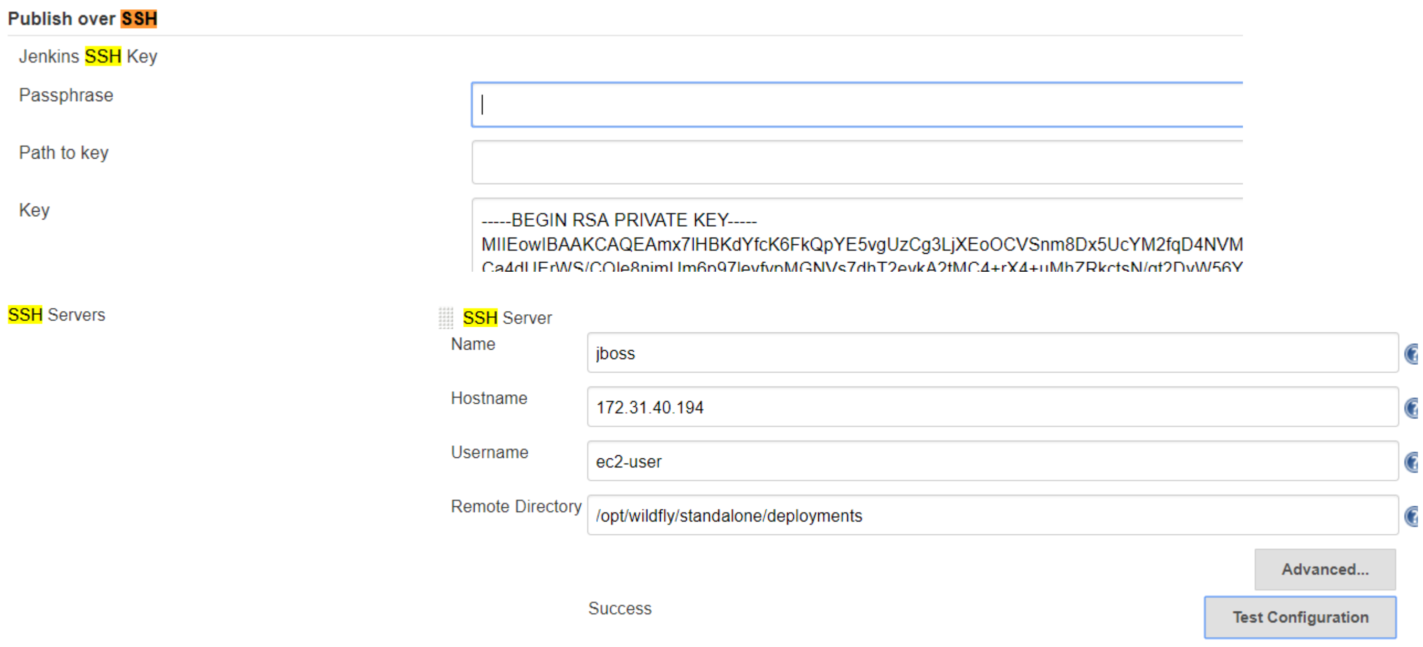
- **CI-CD Pipeline Setup.**

Step 1: Login to Jenkins console, and install the following plugins.

- WildFly Deployer
- Maven Integration
- Publish over ssh
- WildFly Deployer Plugin

Step2. Add SSH id for WildFly

Jenkins Dashboard->Manage Jenkins->Configure System->Publish Over SSH and add JBoss instance's pem key, private ip , remote directory and user name, then click on Test Configuration.



Publish over SSH

Jenkins SSH Key	
Passphrase	
Path to key	
Key	-----BEGIN RSA PRIVATE KEY----- MIIEowIBAAKCAQEAmx7lHBKdYfcK6FkQpYE5vgUzCg3LjXEoOCVSnm8Dx5UcYM2fqD4NVM Ca4dUErWS/COle8nimUm6n97levfvnMGNVs7dhT2evkA2tMC4+rX4+uMhZRkctsN/qt2DvW56Y

SSH Servers

SSH Server

Name	jboss
Hostname	172.31.40.194
Username	ec2-user
Remote Directory	/opt/wildfly/standalone/deployments

Success

Name: jboss

Hostname: <JBoss instance private IP>

Username: ec2-user

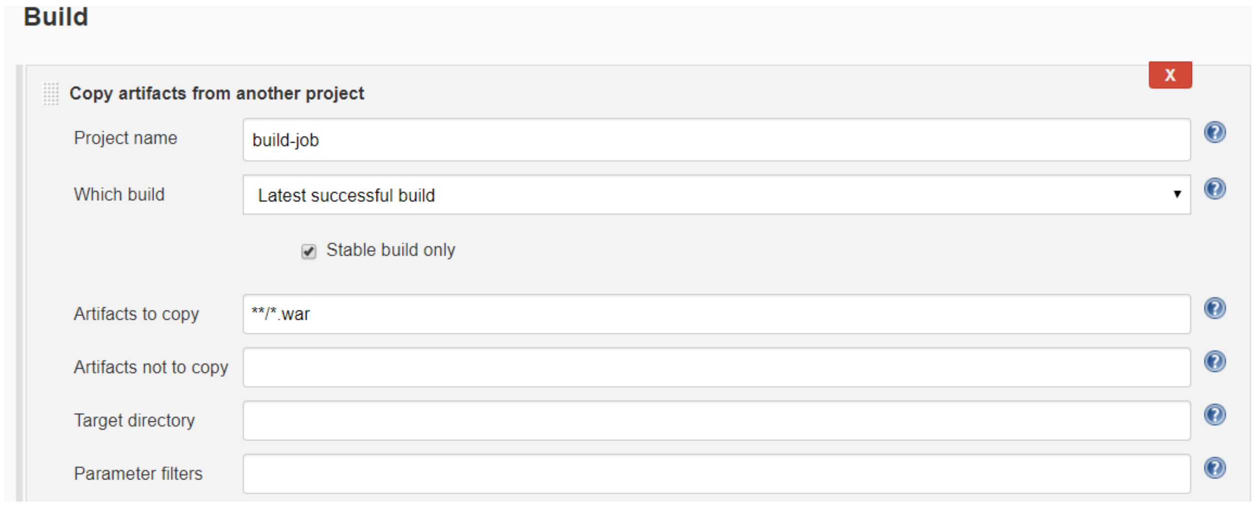
Remote Directory: /opt/wildfly/standalone/deployments

Step 3: Create a CI-Build Job using maven (package goal). You can use exiting repo.

https://github.com/prakashk0301/maven-project

Step 4: Create a CD-Deploy job.

Select copy artifact from another job from drop down under build section.

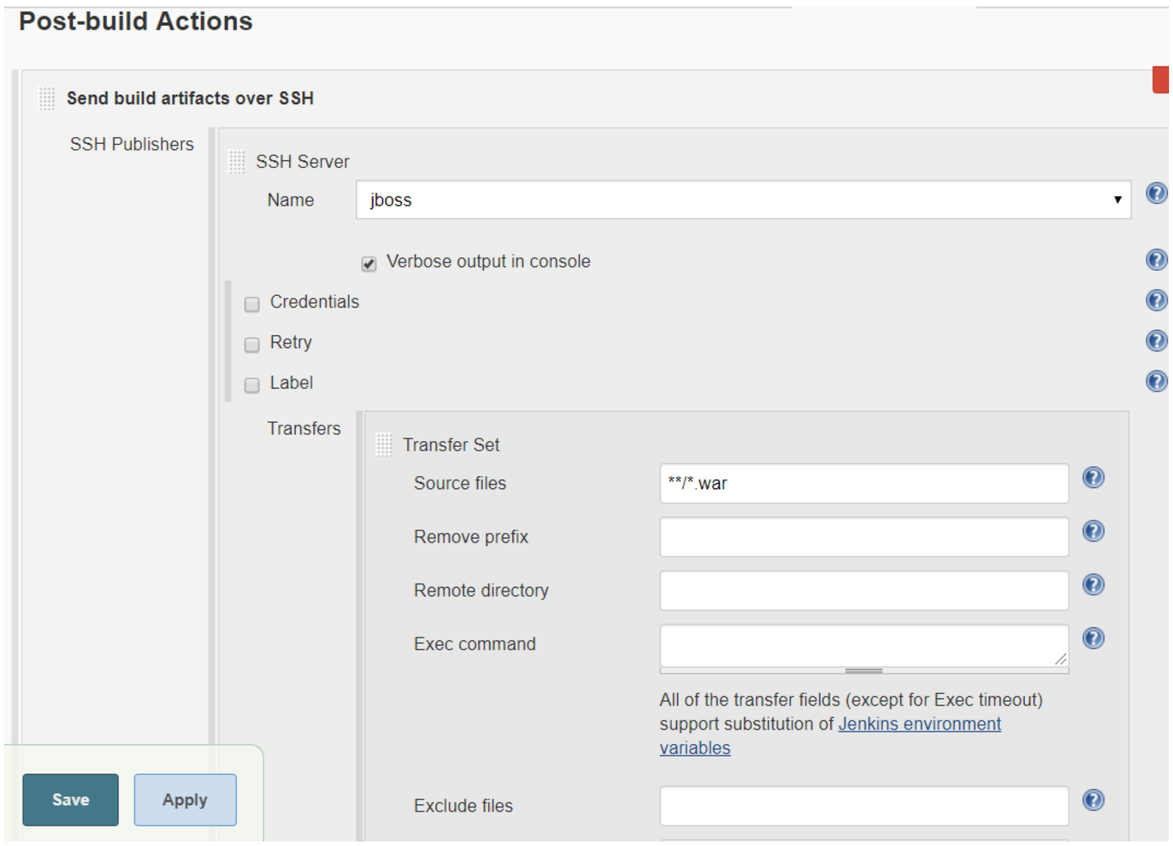


Build

Copy artifacts from another project

Project name	build-job
Which build	Latest successful build
	✓ Stable build only
Artifacts to copy	**/*.war
Artifacts not to copy	
Target directory	
Parameter filters	

Step 5. Select send build artifact over ssh from drop down under Post-Build Action. (If you face any problem during the deployment, its good practice to select verbose, click Advanced and check Verbose. It provides you detailed console output)



Post-build Actions

Send build artifacts over SSH

SSH Publishers

SSH Server

Name: jboss

☑ Verbose output in console

☐ Credentials

☐ Retry

☐ Label

Transfers

Transfer Set

Source files: **/*.war

Remove prefix:

Remote directory:

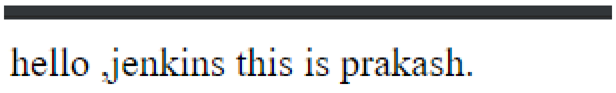
Exec command:

All of the transfer fields (except for Exec timeout) support substitution of Jenkins environment variables

Exclude files:

Step 6. Save deploy job and trigger your CI-Build job.

Step 7. http://<Public ip of your JBoss>:8080/webapp/



```
hello ,jenkins this is prakash.
```

Done. ☺