

BuildPulse Jenkins Plugin Development

Install groovy dsl plugin in Jenkins.....	2
Defining environment variables.....	3
Defining new job in Jenkins.....	4
Defining Optional Build Parameters.....	5
Groovy dsl script definition to execute run.sh file.	5
Github Credentials creation using personal access token:	6
Configure Source Code Management in Jenkins Job	8
Configuring specific directory to clone the GitHub code	8
Definition of Code Build Command in Jenkins.....	8
Build to view report submission status.....	9
Modified run.sh File:	11

Pre-requisite: (Groovy DSL script to execute run.sh file)

1. Linux machine with Jenkins automation server, with internet connection
2. Install groovy dsl plugin in Jenkins.
3. Defining environment variables configuration change.
4. Defining new job in Jenkins
5. Defining optional build parameters
6. Groovy dsl script definition to execute run.sh file.
7. Github Credentials creation using personal access token.
8. Configure Source Code Management in Jenkins Job
9. Configuring specific directory to clone the GitHub code.
10. Build to view report submission status.

Install groovy dsl plugin in Jenkins.

Manage Jenkins -> Plugins -> search the name “groovy” and select the checkbox “Groovy” and then click button “**Install without restart**”

The screenshot shows the Jenkins 'Plugins' page. On the left sidebar, 'Available plugins' is selected. The main area has a search bar with 'groovy' entered. Below the search bar, a table lists plugins. The 'Groovy' plugin (version 453.vc8b_a_c5c99890) is checked and marked as 'Installed'. Below it, a warning message states: 'This plugin is deprecated. In general, this means that it is either obsolete, no longer being developed, or may no longer work. Learn more.' Other plugins listed include 'Pipeline: Deprecate Groovy Libraries', 'Config File Provider', 'Groovy Postbuild', and 'Scriptler'. At the bottom, there are buttons for 'Install without restart' and 'Download now and install after restart'.

Jenkins -> Manage Jenkins -> Tools -> in **Groovy** title select “Groovy Installations” option.

Note: if the Groovy title is not available restart the jenkins instance.

“Groovy Installations” click the button “Add Groovy” it will display a container to define available groovy version select the version and click **save** button.

The screenshot shows the 'Groovy Installations' configuration page. At the top, there is a tab labeled 'Groovy Installations' and a button 'Add Groovy'. Below this, a form is displayed with the following fields: 'name' (set to 'Groovy'), 'Install automatically' (checked), and 'Version' (set to 'Groovy 4.0.9'). There is also an 'Add Installer' button. At the bottom of the form, there is a 'Save' button and an 'Apply' button.

Defining environment variables

Environment variables are common for all Jenkins jobs, to differentiate it we used the prefix BPTR_ we need to define following data in Jenkins environment variables.

S.no	In run.sh used variables	Allocated new name with prefix, same is used in run.sh file.	Value
1.	INPUT_ACCOUNT	BPTR_INPUT_ACCOUNT	
2.	INPUT_COMMIT	BPTR_INPUT_COMMIT	
3.	INPUT_KEY	BPTR_INPUT_KEY	
4.	INPUT_PATH	BPTR_INPUT_PATH	
5.	INPUT_REPOSITORY	BPTR_INPUT_REPOSITORY	
6.	INPUT_REPOSITORY_PATH	BPTR_INPUT_REPOSITORY_PATH	
7.	INPUT_SECRET	BPTR_INPUT_SECRET	
8.	RUNNER_OS	BPTR_RUNNER_OS	Linux
9.	BUILDPULSE_BUCKET	BPTR_BUILDPULSE_BUCKET	buildpulse-uploads-staging

Jenkins Dashboard -> Manage Jenkins -> System -> search “Global Properties” -> select the checkbox of “Environment variables.”

Click **the Add** button for more variable definition and finally click **Save** button.

Dashboard > Manage Jenkins > System >

Global properties

☐ Disable deferred wipeout on this node ?

☒ Environment variables

List of variables ?

Name

BPTR_INPUT_ACCOUNT

Value

9854675766

Name

BPTR_INPUT_COMMIT

Value

GHGDF65374GDH

Name

Save

Apply

Defining new job in Jenkins

In Jenkins, Dashboard -> New Item -> define your job name.

Dashboard >

+ New Item

People

Build History

Project Relationship

Check File Fingerprint

Manage Jenkins

My Views

Build Queue

No builds in the queue.

After entering the new **job name**, select **Freestyle project**, and click the **OK** button.

Enter an item name

>> This field cannot be empty, please enter a valid name

Freestyle project

This is the central feature of Jenkins. Jenkins will build your project, combining any SCM with any build system, and this can be even used for something other than software build.

Pipeline

Orchestrates long-running activities that can span multiple build agents. Suitable for building pipelines (formerly known as workflows) and/or organizing complex activities that do not easily fit in free-style job type.

Available jobs listed in Dashboard as follows.

Dashboard > All >

+ New Item

People

Build History

Project Relationship

Check File Fingerprint

Manage Jenkins

My Views

Build Queue

No builds in the queue.

Build Executor Status

Build-In Node

1 Idle

All +

S	W	Name ↓	Last Success	Last Failure	Last Duration	
		BuildPulse	14 days #3	N/A	96 ms	
		demo	2 hr 27 min #63	2 hr 29 min #62	0.2 sec	
		OracleForms	1 mo 5 days #79	1 mo 5 days #65	1 min 0 sec	

Icon: S M L

Icon legend Atom feed for all Atom feed for failures Atom feed for just latest builds

Add description

Defining Optional Build Parameters

Open your job -> configuration-> General -> select the check box “**This project is parameterized**” -> click the button “**Add Parameter**” and select the parameter type “**String parameter**” new container display in that define your job parameter variables with description, finally click **save** button.

Following 2 variables we need to define.

S.no	Allocated new name with prefix, same is used in run.sh file.	Value
1.	OPTIONAL_BPTR_INPUT_COMMIT	Each build time user may enter data, if not in the environment variable defined value to be used by run.sh file.
2.	OPTIONAL_BPTR_INPUT_REPOSITORY_PATH	

The screenshot shows the Jenkins Configuration page for a job named 'demo'. The 'General' tab is selected. Under the 'Description' field, there is a checkbox labeled 'This project is parameterised' which is checked. Below this, a 'String Parameter' is defined with the name 'OPTIONAL_BPTR_INPUT_REPOSITORY_PATH'. The 'Default Value' field is empty, and the 'Description' field contains the text 'Input repository path - optional parameter, when you give script uses it, otherwise default value read from environment.' At the bottom, there are 'Save' and 'Apply' buttons.

Groovy dsl script definition to execute run.sh file.

Note: BUILDPULSE_BUCKET key we need to use in run.sh file. Modified run.sh file available at end of the document.

Open your job -> configuration-> in **Build Steps** title below dropdown box “**Add build step**” is available in that select the value “**Execute Groovy script**” value, now new container will be added.

The screenshot shows the Jenkins Configuration page for a job named 'demo'. The 'Build Steps' tab is selected. A dropdown menu is open, showing various build step options. The 'Add build step' button is highlighted at the bottom of the dropdown. The options in the dropdown are: Execute Groovy script, Execute Windows batch command, Execute shell, Execute system Groovy script, Invoke Ant, Invoke Gradle script, Invoke top-level Maven targets, Run with timeout, and Set build status to "pending" on GitHub commit. At the bottom, there are 'Save' and 'Apply' buttons.

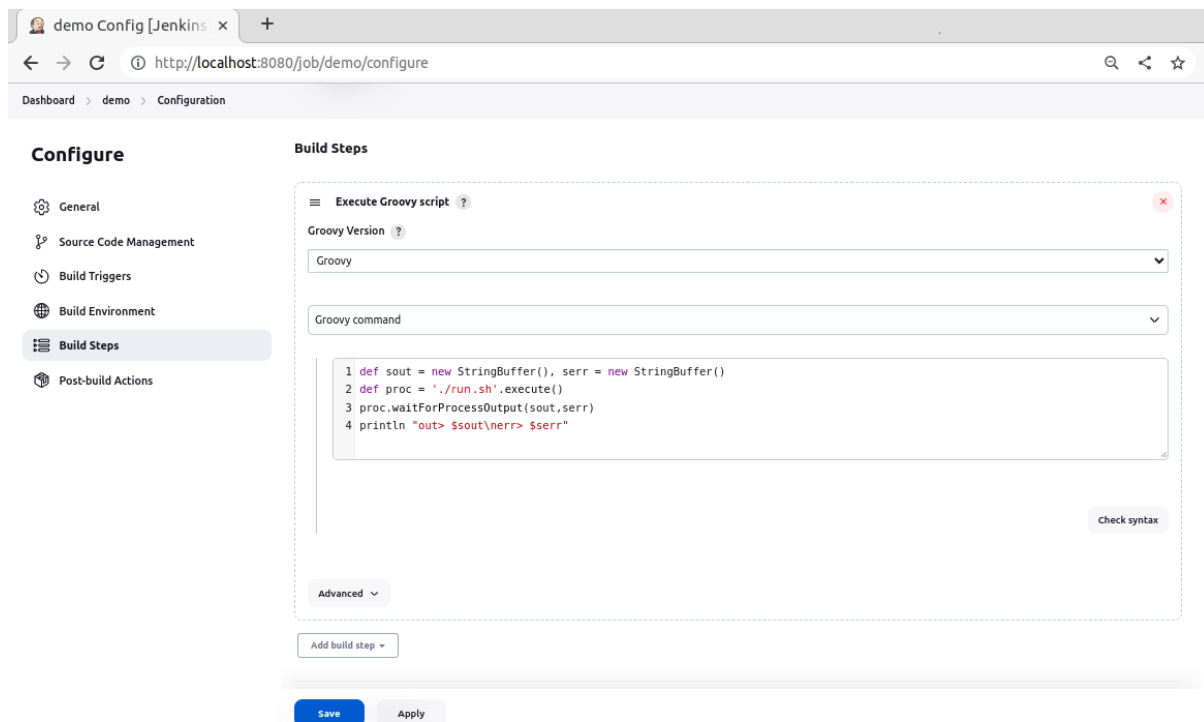
In the new container select the “**Groovy version**” as “**Groovy**”, and in the next drop down select the value “**Groovy Command**”.

Next command place holder places the groovy dsl script.

```
def sout = new StringBuffer(), serr = new StringBuffer()
def proc = './run.sh'.execute()
proc.waitForProcessOutput(sout,serr)
println "out> $sout\nerr> $serr"
```

Notes: place your **run.sh** file inside the workspace of job.
Example **/var/lib/jenkins/workspace/demo/**

finally click save button.



Github Credentials creation using personal access token:

Go to Dashboard -> Manage jenkins -> Configure System you will find the GitHub Servers. In that we need to provide necessary details.

Name

Reference name for our GitHub Credential

API URL

API endpoint of a GitHub server. To use public github.com, leave this field to the default value of <https://api.github.com>. Otherwise, if you use GitHub Enterprise, specify its API endpoint here (e.g., <https://ghe.acme.com/api/v3/>)

Credentials

You can create your own **personal access token** in your account GitHub settings. Token should be registered with scopes:

1. **admin:repo_hook** - for managing hooks (read, write, and delete old ones)
2. **repo** - to see private repos.
3. **repo: status** - to manipulate commit statuses.

In Jenkins, create credentials as «**Secret Text**», provided by Plain Credentials Plugin.

GitHub Server ?

Name ?

BuildPulse-Github

API URL ?

https://api.github.com

Credentials ?

- none -

+ Add

Test connection

☐ Manage hooks

Advanced...

Save Apply

Jenkins Credentials Provider: Jenkins

Add Credentials

Domain

Global credentials (unrestricted)

Kind

Secret text

Scope ?

Global (Jenkins, nodes, items, all child items, etc)

Secret

ID ?

Once you created the credentials, then click **Save**.

Configure Source Code Management in Jenkins Job

Go to Dashboard -> <Your_Job> -> Configurations -> Source Code Management.

Select **Git** option and click **Add Repositories**. In that provide the **Decentralized-Forums GitHub Path** and select the credentials you previously created in the GitHub Server section. Afterwards you need to specify the respective branch name in “**Branch Specifier**” section.

The screenshot shows the Jenkins configuration page for the 'Git' source code management system. The 'Repositories' section is expanded, showing a 'Repository URL' field with the value 'https://github.com/BuildPulseLLC/decentralized-forums.git'. Below it, the 'Credentials' dropdown is set to 'buildpulse-user/***** (To Clone the Source Code From BuildPulse Decentralized Forums)'. There are '+ Add' and 'Advanced...' buttons. An 'Add Repository' button is located below the repository list. The 'Branches to build' section is also expanded, showing a 'Branch Specifier (blank for \'any\')' field with the value '*/main'.

Configuring specific directory to clone the GitHub code

If you want to configure the Specific Directory for the GitHub Clone, you need to click the “**Additional Behaviours**” option. In those options, select **Checkout to Sub-Directory** to provide the specific path for the GitHub Code

The screenshot shows the 'Additional Behaviours' section of the Jenkins configuration. The 'Repository browser' dropdown is set to '(Auto)'. Under 'Additional Behaviours', the 'Check out to a sub-directory' option is selected. The 'Local subdirectory for repo' field is set to '/var/jenkins_home/workspace/BuildPulse/DecentralizedForumsCode/'. There is an 'Add' button at the bottom left of the behaviours section.

Definition of Code Build Command in Jenkins

For this POC Activity, we build the **decentralized-forums code manually in the server** and used the system generated report XML path in the Jenkins environment.

To make it automation, define the code build commands in the jenkins job build steps.

Build to view report submission status.

In your **job dashboard** click the link **Build with Parameter** now to execute the build, it will run the groovy dsl script which will execute the run.sh file to submit the junit test result to build pulse via buildpulse-test-reporter tool.

The screenshot shows the Jenkins job dashboard for a project named 'demo'. On the left, there is a sidebar with navigation links: Status, Changes, Workspace, Build with Parameters, Configure, Delete Project, and Rename. The 'Build History' section is active, showing a list of builds with their IDs and timestamps. The main area displays the 'Project demo' configuration, which includes two optional parameters: 'OPTIONAL_BPTR_INPUT_REPOSITORY_PATH' and 'OPTIONAL_BPTR_INPUT_COMMIT'. Below the parameters, there are 'build' and 'Cancel' buttons. The build history table lists builds #109 through #104, all of which are successful (indicated by green checkmarks).

Build ID	Timestamp	Status
#109	3 Jul 2023, 15:35	Success
#108	3 Jul 2023, 15:33	Success
#107	3 Jul 2023, 15:30	Success
#106	3 Jul 2023, 15:29	Success
#105	3 Jul 2023, 14:04	Success
#104	3 Jul 2023, 14:00	Success

Click the **last execution build number** and select the **console output** to view the report submission status.

Console Output

```
Started by user DCEP
Running as SYSTEM
Building on the built-in node in workspace /var/jenkins_home/workspace/BuildPulse
The recommended git tool is: NONE
using credential buildpulse-justin--github-personal-token
> git rev-parse --resolve-git-dir /var/jenkins_home/workspace/BuildPulse/DecentralizedForumsCode/.git # timeout=10
Fetching changes from the remote Git repository
> git config remote.origin.url https://github.com/BuildPulseLLC/decentralized-forums.git # timeout=10
Fetching upstream changes from https://github.com/BuildPulseLLC/decentralized-forums.git
> git --version # timeout=10
> git --version # 'git version 2.30.2'
using GIT_ASKPASS to set credentials To Clone the Source Code From BuildPulse Decentralized Forums
> git fetch --tags --force --progress -- https://github.com/BuildPulseLLC/decentralized-forums.git +refs/heads/*:refs/remotes/origin/* # timeout=10
> git rev-parse refs/remotes/origin/main^{commit} # timeout=10
Checking out Revision c7476079a82ac3b66d4e01fa352bbc45b4d40bb7 (refs/remotes/origin/main)
> git config core.sparsecheckout # timeout=10
> git checkout -f c7476079a82ac3b66d4e01fa352bbc45b4d40bb7 # timeout=10
Commit message: "reset back to pipeline"
> git rev-list --no-walk c7476079a82ac3b66d4e01fa352bbc45b4d40bb7 # timeout=10
[BuildPulse] $ /var/jenkins_home/tools/hudson.plugins.groovy.GroovyInstallation/Groovy/bin/groovy /var/jenkins_home/workspace/BuildPulse/hudson4576387500804592300.groovy
out> This bash shell version supports double globbing: '5.1.4(1)-release'.
1. account: 46661156
2. repository: 608133431
3. path: /var/jenkins_home/workspace/BuildPulse/DecentralizedForumsCode/spec/reports/
4. key: ***
5. secret: ***
6. repository path: /var/jenkins_home/workspace/BuildPulse/DecentralizedForumsCode/
7. commit sha: c7476079a82ac3b66d4e01fa352bbc45b4d40bb7
8. operating system: Linux
report tool download success....
<buildpulse> Current version: BuildPulse Test Reporter 0.27.2 (linux 7e01ff7 go1.19.12)
<buildpulse> Initiating `submit`
<buildpulse> Received args: /var/jenkins_home/workspace/BuildPulse/DecentralizedForumsCode/spec/reports/ --account-id 46661156 --repository-id 608133431 --repository-dir
/var/jenkins_home/workspace/BuildPulse/DecentralizedForumsCode/
<buildpulse> Using working directory: /var/jenkins_home/workspace/BuildPulse
<buildpulse> Looking for git repository at /var/jenkins_home/workspace/BuildPulse/DecentralizedForumsCode/
<buildpulse> Found git repository at /var/jenkins_home/workspace/BuildPulse/DecentralizedForumsCode/
<buildpulse> Gathering metadata to describe the build
<buildpulse> Detected build environment: jenkins
<buildpulse> Using $GIT_COMMIT environment variable as commit SHA: c7476079a82ac3b66d4e01fa352bbc45b4d40bb7
<buildpulse> Looking up info for commit `c7476079a82ac3b66d4e01fa352bbc45b4d40bb7` in git repository
<buildpulse> Found commit info
<buildpulse> Writing metadata to /tmp/buildpulse-1465516518.yml
<buildpulse> Preparing tarball of test results:
<buildpulse> - /var/jenkins_home/workspace/BuildPulse/DecentralizedForumsCode/spec/reports/rspec.xml
<buildpulse> - /var/jenkins_home/workspace/BuildPulse/DecentralizedForumsCode/coverage/lcov/DecentralizedForumsCode.lcov
<buildpulse> Adding buildpulse.yml to tarball
<buildpulse> Flushing log to /tmp/buildpulse-1352517512.log
<buildpulse> Adding buildpulse.log to tarball
<buildpulse> Gzipping tarball (/tmp/buildpulse-3421686409.tar)
<buildpulse> Sending /tmp/buildpulse-230073231.gz to BuildPulse
<buildpulse> Delivered test results to BuildPulse (46661156/608133431/buildpulse-448581da-c3fc-4007-a2cc-ae9d5448a43.gz)

err> + curl -fsSL --retry 3 --retry-connrefused --connect-timeout 5 https://get.buildpulse.io/test-reporter-linux-amd64
+ BUILDPULSE_ACCESS_KEY_ID=AKIA3SVVRUSE3N4K486P
+ BUILDPULSE_SECRET_ACCESS_KEY=edqTZHGwHCLY2V0rEKf/Ro2iBAq75ltxfbWFKT4A
+ GITHUB_SHA=c7476079a82ac3b66d4e01fa352bbc45b4d40bb7
+ BUILDPULSE_BUCKET=buildpulse-uploads-staging
+ ./buildpulse-test-reporter submit /var/jenkins_home/workspace/BuildPulse/DecentralizedForumsCode/spec/reports/ --account-id 46661156 --repository-id 608133431 --repository-dir
/var/jenkins_home/workspace/BuildPulse/DecentralizedForumsCode/

Finished: SUCCESS
```

Modified run.sh File:

Updated run.sh script file is listed below, in the echo message available images are displayed as question mark in this document.

```
#!/bin/bash

# Enable double globbing
shopt -s globstar

set -e

# Enable double globbing if supported by the shell on the base github runner
if shopt -s globstar; then
    echo "This bash shell version supports double globbing: '${BASH_VERSION}'."
else
    echo "This bash shell version does not support double globbing: '${BASH_VERSION}'. Please upgrade to bash 4+."
fi

if ! echo $BPTR_INPUT_ACCOUNT | egrep -q '^[0-9]+$'
then
    echo "?? The given value is not a valid account ID: ${BPTR_INPUT_ACCOUNT}"
    echo "?? To resolve this issue, set the 'account' parameter to your numeric BuildPulse Account ID."
    exit 1
fi
ACCOUNT_ID=$BPTR_INPUT_ACCOUNT

echo "1. account: $ACCOUNT_ID"

if ! echo $BPTR_INPUT_REPOSITORY | egrep -q '^[0-9]+$'
then
    echo "?? The given value is not a valid repository ID: ${BPTR_INPUT_REPOSITORY}"
    echo "?? To resolve this issue, set the 'repository' parameter to your numeric BuildPulse Repository ID."
    exit 1
fi
REPOSITORY_ID=$BPTR_INPUT_REPOSITORY

echo "2. repository: $REPOSITORY_ID"

for path in $BPTR_INPUT_PATH; do
    if [ ! -e "$path" ]
    then
        echo "?? The given path does not exist: $path"
        echo "?? To resolve this issue, set the 'path' parameter to the location of your XML test report(s)."
        exit 1
    fi
done
REPORT_PATH="${BPTR_INPUT_PATH}"

echo "3. path: $REPORT_PATH"

if [ ! -d "$BPTR_INPUT_REPOSITORY_PATH" ]
then
    echo "?? The given path is not a directory: ${BPTR_INPUT_REPOSITORY_PATH}"
    echo "?? To resolve this issue, set the 'repository-path' parameter to the directory that contains the local git clone of your repository."
    exit 1
fi

# >>>>> additional input replace START <<<<<<
REPOSITORY_PATH="${OPTIONAL_BPTR_INPUT_REPOSITORY_PATH}"
# from the user received INPUT_REPOSITORY_PATH value empty scenario read INPUT_REPOSITORY_PATH value from environment variable.
if [ -z "$OPTIONAL_BPTR_INPUT_REPOSITORY_PATH" ]
then
    REPOSITORY_PATH="${BPTR_INPUT_REPOSITORY_PATH}"
fi

COMMIT_SHA="${OPTIONAL_BPTR_INPUT_COMMIT}"
# from the user received INPUT_COMMIT value empty scenario read INPUT_COMMIT value from environment variable.
if [ -z "$OPTIONAL_BPTR_INPUT_COMMIT" ]
then
    COMMIT_SHA="${BPTR_INPUT_COMMIT:-$GITHUB_SHA}"
fi

# >>>>> additional input replace END <<<<<<

#echo "4. key: $BPTR_INPUT_KEY"
#echo "5. secret: $BPTR_INPUT_SECRET"
```

```

echo "4. key: ****"
echo "5. secret: ****"
echo "6. repository path: $REPOSITORY_PATH"
echo "7. commit sha: $COMMIT_SHA"

if test -z "$BPTR_INPUT_KEY" && test -z "$BPTR_INPUT_SECRET" && test "$GITHUB_ACTOR" = "dependabot[bot]"
then
    echo "::warning::No value available for the 'key' parameter or the 'secret' parameter. Skipping upload to BuildPulse."
    echo "?? ?? ?? As of March 1, 2021, Dependabot PRs cannot access secrets in GitHub Actions. See details on the GitHub blog at
https://bit.ly/3KAolBf"
    echo "?? ?? ?? Secrets are necessary in order to authenticate with external services like BuildPulse."
    echo "?? ?? ?? Since secrets aren't available in this build, the build cannot authenticate with BuildPulse to upload test results."
    exit 0
fi

echo "8. operating system: $BPTR_RUNNER_OS"

case "$BPTR_RUNNER_OS" in
    Linux)
        BUILDPULSE_TEST_REPORTER_BINARY=test-reporter-linux-amd64
        ;;
    macOS)
        BUILDPULSE_TEST_REPORTER_BINARY=test-reporter-darwin-amd64
        ;;
    Windows)
        BUILDPULSE_TEST_REPORTER_BINARY=test-reporter-windows-amd64.exe
        ;;
    *)
        echo "::error::Unrecognized operating system. Expected RUNNER_OS to be one of \"Linux\", \"macOS\", or \"Windows\",
but it was \"$RUNNER_OS\"."
        exit 1
esac

BUILDPULSE_TEST_REPORTER_HOSTS=(
    https://get.buildpulse.io
    https://github.com/buildpulse/test-reporter/releases/latest/download
)
[ -n "${INPUT_CLI_HOST}" ] && BUILDPULSE_TEST_REPORTER_HOSTS=("${INPUT_CLI_HOST}"
"${BUILDPULSE_TEST_REPORTER_HOSTS[@]}")

getcli() {
    local rval=-1
    for host in "${BUILDPULSE_TEST_REPORTER_HOSTS[@]"; do
        url="${host}/${BUILDPULSE_TEST_REPORTER_BINARY}"
        if (set -x; curl -fsSL --retry 3 --retry-connrefused --connect-timeout 5 "$url" > "$1"); then
            return 0
        else
            rval=$?
        fi
    done;

    return $rval
}

if getcli ./buildpulse-test-reporter; then
    : # Successfully fetched binary. Great!
    echo "report tool download success...."
else
    echo "report tool download issue...."
    msg=$(cat <<-eos
        ::warning::Unable to send test results to BuildPulse. See details below.

        Downloading the BuildPulse test-reporter failed with status $?.

        We never want BuildPulse to make your builds unstable. Since we're having
        trouble downloading the BuildPulse test-reporter, we're skipping the
        BuildPulse analysis for this build.

        If you continue seeing this problem, please get in touch at
        https://buildpulse.io/contact so we can look into this issue.

        eos
    )

    echo "${msg//$'\n'/%0A}" # Replace newlines with URL-encoded newlines for proper formatting in GitHub Actions annotations
    (https://github.com/actions/toolkit/issues/193#issuecomment-605394935)
    exit 0
fi

chmod +x ./buildpulse-test-reporter

set -x

```

```
BUILDULSE_ACCESS_KEY_ID="${BPTR_INPUT_KEY}" \
BUILDULSE_SECRET_ACCESS_KEY="${BPTR_INPUT_SECRET}" \
GITHUB_SHA="${COMMIT_SHA}" \
BUILDULSE_BUCKET="${BPTR_BUILDULSE_BUCKET}" \
./buildpulse-test-reporter submit $REPORT_PATH --account-id $ACCOUNT_ID --repository-id $REPOSITORY_ID --repository-dir
"${REPOSITORY_PATH}"
```