

Cloud -V

A developer's way to RISC-V
Architecture



License

Copyright © 10xEngineers



Section 1

On-boarding Developers



Overview

Cloud-V enables software developers to use online emulation platform to build, run and test their applications on RISC-V architecture.

Objective: Create a scalable and flexible cloud-based infrastructure for CI testing of applications running on an emulated RISC-V processor system.

Features:

- Multiple runners with x86 and QEMU runner instances
- Automated builds with version control (github and gitlab)
- Ease of use with shell and CI scripting
- Out of the box RISC-V-compiled binary execution with QEMU user mode
- Scalability with addition of extra CPUs and emulated instances

Developer's Onboarding

Important links:

- Cloud-V currently hosting live at <https://cloud-v.co>
- User Guide is available at 10xengineers.github.io

Getting a Cloud-V account:

- Through E-mail: ali.tariq@10xEngineers.ai
- Through Github issues: Create a github issue for account [here](#) and add details:
 - Name, E-mail, URL of github repository, Name of Organization

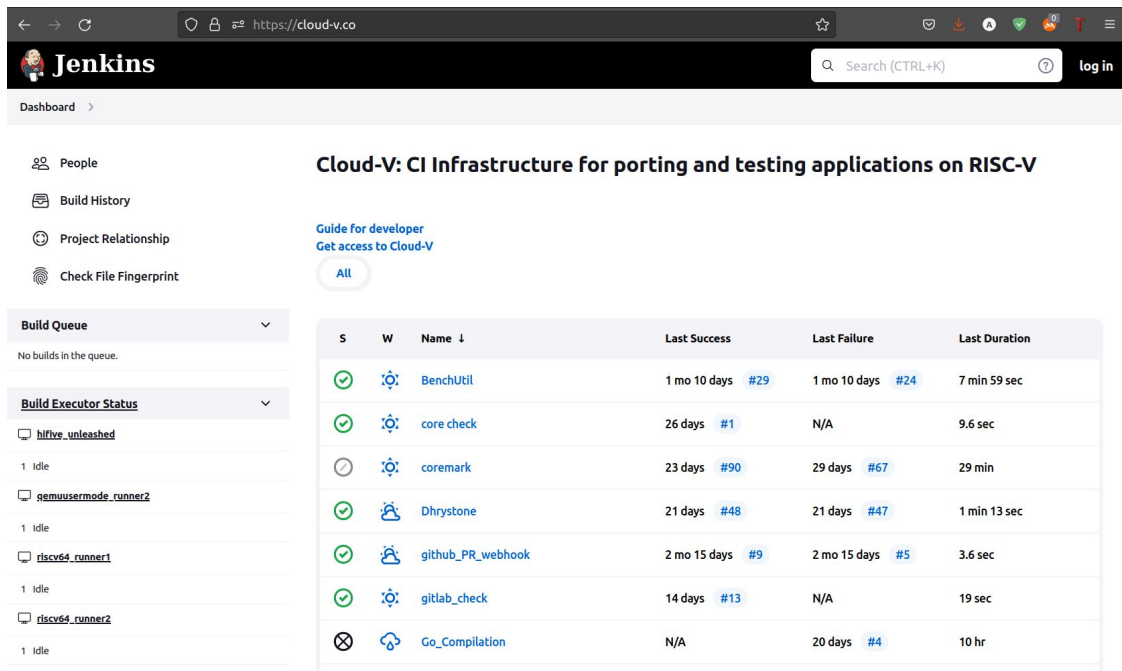
Developer's Onboarding

Getting started:

- Create a github repository and add credentials in jenkins
- Apply for job build creation using [github issue](#) with following details:
 - Name of job build
 - Triggers for job build execution (manual or version-control trigger)
- View job build progress and summary at Cloud-V [URL](#)

Developer's Onboarding

Jenkins Dashboard:



Jenkins Search (CTRL+K) log in

Dashboard >

People
Build History
Project Relationship
Check File Fingerprint

Cloud-V: CI Infrastructure for porting and testing applications on RISC-V

[Guide for developer](#)
[Get access to Cloud-V](#)

All

S	W	Name ↓	Last Success	Last Failure	Last Duration
✓	⚙️	BenchUtil	1 mo 10 days #29	1 mo 10 days #24	7 min 59 sec
✓	⚙️	core check	26 days #1	N/A	9.6 sec
⌚	⚙️	coremark	23 days #90	29 days #67	29 min
✓	🔗	Dhrystone	21 days #48	21 days #47	1 min 13 sec
✓	🔗	github_PR_webhook	2 mo 15 days #9	2 mo 15 days #5	3.6 sec
✓	⚙️	gitlab_check	14 days #13	N/A	19 sec
✗	🔗	Go_Compilation	N/A	20 days #4	10 hr

Build Queue ▼
No builds in the queue.

Build Executor Status ▼

- hifive_unleashed
1 Idle
- qemuusermode_runner2
1 Idle
- riscv64_runner1
1 Idle
- riscv64_runner2
1 Idle

Developer's Onboarding

Jenkins pipeline stage view:

The screenshot shows the Jenkins web interface for a pipeline named 'Python_on_riscv'. The left sidebar contains navigation links: Status, Changes, Full Stage View, and GitHub. Below these is the 'Build History' section, which lists recent builds with their IDs, timestamps, and commit messages. The main area displays the 'Stage View' for the selected build. It features a table with stages and their durations, along with a summary of average stage and full run times.

Build History

Build ID	Timestamp	Commit Message
#165	Dec 13, 2022, 6:42 PM	PR #19: Update setup.py
#164	Dec 7, 2022, 10:55 AM	PR #19: Update setup.py
#163	Dec 6, 2022, 2:19 PM	PR #19: Update setup.py
#162	Dec 6, 2022, 7:05 AM	PR #19: Update setup.py

Pipeline Python_on_riscv

Stage View

Average stage times: (Average full run time: ~45min 57s)

Stage	Duration
*** Cleanup***	2s
*** 60s Sleep ***	1min 0s
*** Checkout SCM Hifive Unleashed ***	4min 31s
*** PRE-BUILD SETTINGS Hifive Unleashed ***	11s
*** CONFIGURATUION PHASE Hifive Unleashed ***	3min 55s
*** make -j1 Hifive Unleashed ***	16min 46s
*** make -j1 Install Hifive Unleashed single core ***	4min 35s

Build #165 (Dec 13, 2022, 18:42) has 1 commit.

Build #164 (Dec 07, 2022, 10:55) has 1 commit.

Developer's Onboarding

Once the build is complete, the status will be shown on cloud-v as well as on commit

Section 2

Architecture



Overview

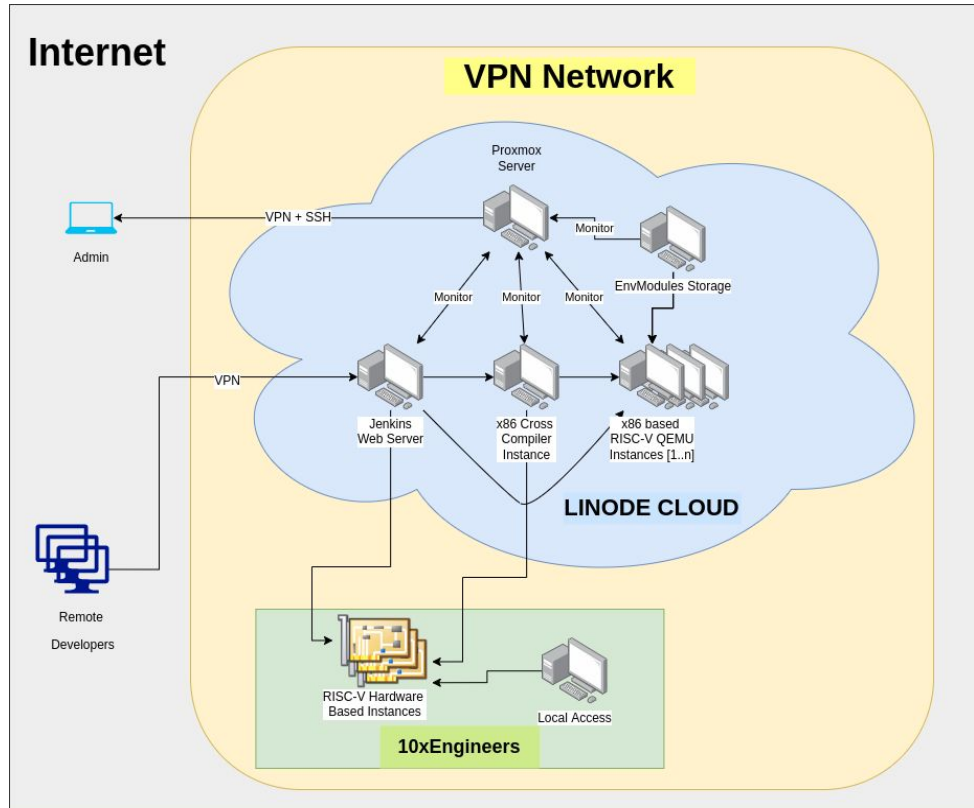
Cloud-V enables software developers to use online emulation platform to build, run and test their applications on RISC-V architecture.

Objective: Create a scalable and flexible cloud-based infrastructure for CI testing of applications running on an emulated RISC-V processor system.

Features:

- Multiple runners with x86 and QEMU runner instances
- Automated builds with version control (github and gitlab)
- Ease of use with shell and CI scripting
- Out of the box RISC-V-compiled binary execution with QEMU user mode
- Scalability with addition of extra CPUs and emulated instances

Architecture



Section 3

Jenkins Setup



Overview

Cloud-V enables software developers to use online emulation platform to build, run and test their applications on RISC-V architecture.

Objective: Create a scalable and flexible cloud-based infrastructure for CI testing of applications running on an emulated RISC-V processor system.

Features:

- Multiple runners with x86 and QEMU runner instances
- Automated builds with version control (github and gitlab)
- Ease of use with shell and CI scripting
- Out of the box RISC-V-compiled binary execution with QEMU user mode
- Scalability with addition of extra CPUs and emulated instances

Jenkins Setup

Jenkins Installation:

- Get source code from [github](#)
- Read “CONTRIBUTING.md” for building
- Expose a port or run on localhost:8080 (by default)
- Complete initial setup on browser

Jenkins Setup

Jenkins Nodes:

- They are instances/runners to run builds on
- Add them in jenkins configuration using SSH
- Refer to the [documentation](#) for creating jenkins nodes

Section 4

Creating a Pipeline



Overview

Cloud-V enables software developers to use online emulation platform to build, run and test their applications on RISC-V architecture.

Objective: Create a scalable and flexible cloud-based infrastructure for CI testing of applications running on an emulated RISC-V processor system.

Features:

- Multiple runners with x86 and QEMU runner instances
- Automated builds with version control (github and gitlab)
- Ease of use with shell and CI scripting
- Out of the box RISC-V-compiled binary execution with QEMU user mode
- Scalability with addition of extra CPUs and emulated instances

Creating a pipeline in jenkins

In Jenkins Dashboard:

- Click on “new item”
- Select “Pipeline” in job types
- Name the project
- Click “OK”

In Job configuration:

- Choose Triggers – How the build will start
- Choose the post triggers – What to do when job finishes

Section 5

Jenkins Pipeline Debugging



Overview

Cloud-V enables software developers to use online emulation platform to build, run and test their applications on RISC-V architecture.

Objective: Create a scalable and flexible cloud-based infrastructure for CI testing of applications running on an emulated RISC-V processor system.

Features:

- Multiple runners with x86 and QEMU runner instances
- Automated builds with version control (github and gitlab)
- Ease of use with shell and CI scripting
- Out of the box RISC-V-compiled binary execution with QEMU user mode
- Scalability with addition of extra CPUs and emulated instances

Jenkins Pipeline Debugging

Following steps can be used to debug Jenkins pipeline:

- Check if there is a syntax error in Jenkinsfile
- See Console output – In Jenkins Job, click on build number and then on “Console Output”
- Check if the webhook is properly configured – Only if you enabled version control triggers
- Check if the desired node is selected – Nodes are runners on which job builds

Jenkins Pipeline Debugging

Console output of job build:

Dashboard > BenchUtil > #29

Status

Changes

Console Output

View as plain text

Edit Build Information

Delete build '#29'


Git Build Data

Replay

Pipeline Steps

Workspaces

Previous Build

 Console Output

```
Started by user jenkins_user
Checking out git https://github.com/alitariq4589/BenchUtil.git into /home/jenkins_user/.jenkins/workspace/BenchUtil@script/0e5df252c0f957f1707f475be03168229d3fa389639d19fbf68fea07cf69eae1 to read jenkinsfile
The recommended git tool is: NONE
using credential 95985a22-8a1a-483d-9ae8-5090a4cf0b6c
> git rev-parse --resolve-git-dir /home/jenkins_user/.jenkins/workspace/BenchUtil@script/0e5df252c0f957f1707f475be03168229d3fa389639d19fbf68fea07cf69eae1/.git # timeout=10
Fetching changes from the remote Git repository
> git config remote.origin.url https://github.com/alitariq4589/BenchUtil.git # timeout=10
Fetching upstream changes from https://github.com/alitariq4589/BenchUtil.git
> git --version # timeout=10
> git --version # 'git version 2.34.1'
using GIT_ASKPASS to set credentials Github credentials for ali.tariq@10xengineers.ai
> git fetch --tags --force --progress -- https://github.com/alitariq4589/BenchUtil.git +refs/heads/*:refs/remotes/origin/* # timeout=10
> git rev-parse origin/master^{commit} # timeout=10
Checking out Revision 4d9256c49ce40242113e9480d9746822832eba63 (origin/master)
> git config core.sparsecheckout # timeout=10
> git checkout -f 4d9256c49ce40242113e9480d9746822832eba63 # timeout=10
Commit message: "Update jenkinsfile"
> git rev-list --no-walk 369af73f74b49a3b3584bf33de541437e448836b # timeout=10
[Pipeline] Start of Pipeline
[Pipeline] node
Running on x86_runner2 in /home/alitariq/runner2_dir/workspace/BenchUtil
```


Section 6

Github-Jenkins Integration



Overview

Cloud-V enables software developers to use online emulation platform to build, run and test their applications on RISC-V architecture.

Objective: Create a scalable and flexible cloud-based infrastructure for CI testing of applications running on an emulated RISC-V processor system.

Features:

- Multiple runners with x86 and QEMU runner instances
- Automated builds with version control (github and gitlab)
- Ease of use with shell and CI scripting
- Out of the box RISC-V-compiled binary execution with QEMU user mode
- Scalability with addition of extra CPUs and emulated instances

Github-Jenkins Integration

Github configuration:

- Add “Jenkinsfile” in github repository – see [this](#) link for creating Jenkinsfile
- Add jenkins webhook in repository settings
- Ping webhook to check the connection

Jenkins Configuration:

- Add github credentials in jenkins having owner permissions to repository
- Add triggers in jenkins job

Section 7

Grafana and Prometheus Setup



Overview

Cloud-V enables software developers to use online emulation platform to build, run and test their applications on RISC-V architecture.

Objective: Create a scalable and flexible cloud-based infrastructure for CI testing of applications running on an emulated RISC-V processor system.

Features:

- Multiple runners with x86 and QEMU runner instances
- Automated builds with version control (github and gitlab)
- Ease of use with shell and CI scripting
- Out of the box RISC-V-compiled binary execution with QEMU user mode
- Scalability with addition of extra CPUs and emulated instances

Grafana and Prometheus Setup

Important Links:

- Grafana is live at: <https://monitor.cloud-v.co>
- Prometheus is not hosted publicly

Installation:

- Get precompiled binaries of [Grafana](#) and [Prometheus](#)
- Get [node exporter](#) precompiled binary
- Extract the tarballs

Grafana and Prometheus Setup

Setup:

- Copy node exporter to the runners being monitored
- Add node exporter links to Prometheus
- Add Prometheus hosting link to Grafana
- Add suitable queries in panels for resource monitoring
- Add Dashboards in Grafana to start monitoring

Section 8

Tools available on Cloud-V



Section 7

Tools available on Cloud-V



Overview

Cloud-V enables software developers to use online emulation platform to build, run and test their applications on RISC-V architecture.

Objective: Create a scalable and flexible cloud-based infrastructure for CI testing of applications running on an emulated RISC-V processor system.

Features:

- Multiple runners with x86 and QEMU runner instances
- Automated builds with version control (github and gitlab)
- Ease of use with shell and CI scripting
- Out of the box RISC-V-compiled binary execution with QEMU user mode
- Scalability with addition of extra CPUs and emulated instances

Tools available on Cloud-V

Following are the platforms available on Cloud-V for software developers:

- One x86 runner instance
- One RISC-V QEMU user mode instance – For running RISC-V precompiled binaries
- One Hifive Unleashed board – For building and running applications on RISC-V linux
- One RISC-V QEMU linux – For building and running applications on RISC-V linux

Tools available on Cloud-V

Following are the softwares available on Cloud-V for software developers:

- RISC-V GNU Toolchain – Cross-compiler for producing RISC-V binaries on x86
- Java Development Kit – OpenJDK
- Ruby
- Python3
- Go
- Rust
- Openssl
- Git

Tools available on Cloud-V

Refer to [this](#) document for specific tooling information