edX Load Tests

Introduction And Current Status

Troy Sankey

August 22, 2017

Introduction

- Tests live in the edx/edx-load-tests repo.

Introduction

- Tests live in the edx/edx-load-tests repo.
- Uses locust.io load testing framework.

Introduction

- Tests live in the edx/edx-load-tests repo.
- Uses locust.io load testing framework.
- Tasks are written as python functions.

Introduction

- Tests live in the edx/edx-load-tests repo.
- Uses locust.io load testing framework.
- Tasks are written as python functions.
- Tasks grouped into TaskSets.

Introduction

- Tests live in the edx/edx-load-tests repo.
- Uses locust.io load testing framework.
- Tasks are written as python functions.
- Tasks grouped into TaskSets.
- TaskSet nesting.

User Experience

- No browser, no javascript, no good measure of user experience.

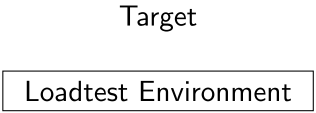
User Experience

- No browser, no javascript, no good measure of user experience.
- Focus on the system response, not the users.

User Experience

- No browser, no javascript, no good measure of user experience.
- Focus on the system response, not the users.
- If you want, during a load test you could login from a browser.

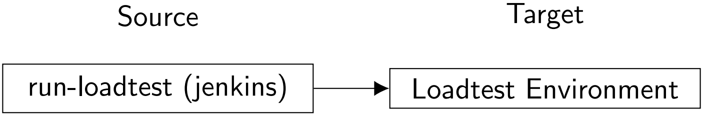
Running Load Tests Overview



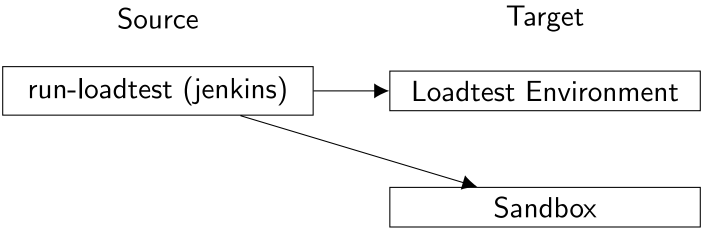
Target

Loadtest Environment

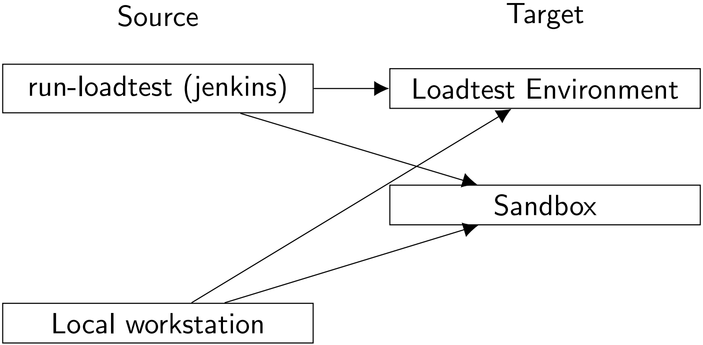
Running Load Tests Overview



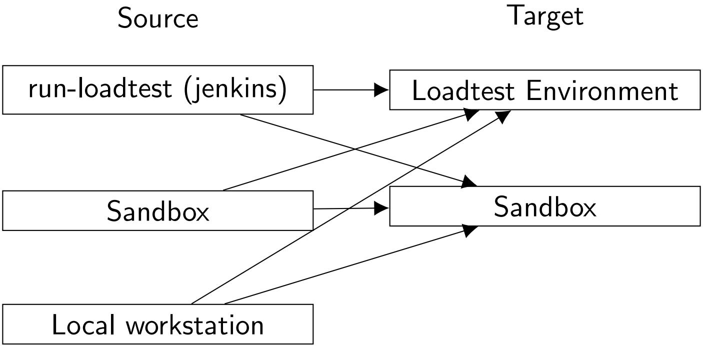
Running Load Tests Overview



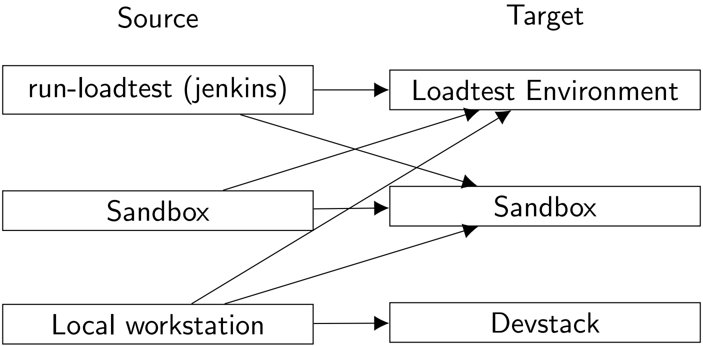
Running Load Tests Overview



Running Load Tests Overview



Running Load Tests Overview



Source

Target

run-loadtest (jenkins)

Loadtest Environment

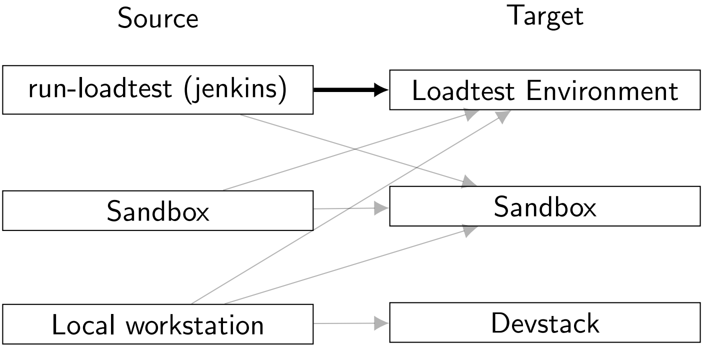
Sandbox

Sandbox

Local workstation

Devstack

Running Load Tests Overview



Settings

- Each load test is configurable via a YAML file.

Settings

- Each load test is configurable via a YAML file.
- Use `helpers.settings`.

Settings

- Each load test is configurable via a YAML file.
- Use `helpers.settings`.
- The Jenkins job (run-loadtest) will use settings specified by a job parameter.

Evaluating Load Test Runs

- Locust provides response time breakdown.

Evaluating Load Test Runs

- Locust provides response time breakdown.
- Click on the generated NewRelic links.

Evaluating Load Test Runs

- Locust provides response time breakdown.
- Click on the generated NewRelic links.
- NewRelic provides deeper application insight.

Evaluating Load Test Runs

- Locust provides response time breakdown.
- Click on the generated NewRelic links.
- NewRelic provides deeper application insight.
- Custom metrics in edx-platform: make use of the custom metrics middleware for peering into application behavior.

Record Your Findings

- Create a wiki page before testing.

Record Your Findings

- Create a wiki page before testing.
- Caution: Beware of NewRelic data atrophie. NR gets hungry and eats your old dots.

Record Your Findings

- Create a wiki page before testing.
- Caution: Beware of NewRelic data atrophie. NR gets hungry and eats your old dots.
- Continuous deployment ⇒ continuous deletion of system metrics!

Distributed load testing

- Just run multiple load tests simultaneously in Jenkins.

Distributed load testing

- Just run multiple load tests simultaneously in Jenkins.
- You may need to prime N workers first.

Distributed load testing

- Just run multiple load tests simultaneously in Jenkins.
- You may need to prime N workers first.
- If you want/need real distributed load testing, demand it!

How to maintain load tests

- Task ratios will rot.

How to maintain load tests

- Task ratios will rot.
- Tasks themselves will rot.

How to maintain load tests

- Task ratios will rot.
- Tasks themselves will rot.
- New endpoints will need to be added as new tasks.

For fun: Real browser load testing

- `pip install realbrowserlocusts`, then subclass PhantomJSLocust in your locustfile.
- Could make certain tests more realistic (AJAX calls actually happen).
- Potentially far more resource intensive on the client side