How to write your own tests

( not only for AGL )
Prerequisites

Laptop
- with a Linux distro of your choice
- A text editor
- Python version >= 2.7.1
  - apt-get install python
  - apt-get install python-pip
- python-jinja >= 2.9
  - pip install --upgrade jinja2

Internet access
- github (or any publicly accessible git) account
- that you can use to create a publicly visible project and upload to

Local LAN
- WiFi: baylibre  pwd: lavabaylibre
- ping lavabox

Local LAVA
- LAVA instance: http://lavabox:10080/
- LAVA user: demo
- LAVA auth token: tokendemo

Local kernel CI frontend
- http://lavabox:8080/
AGL CIAT overview
AGL CIAT loop (for training)
Submitting jobs to LAVA
LAVA: test “jobs”

YAML format with multiple sections
- timeouts
- actions (deploy, boot, test)
- protocols
- device-type
- notify

NOTE: we won’t be writing these from scratch, we’ll be using tools based on templates.
Generating LAVA test jobs

Get the tools:

$ git clone https://git.automotivelinux.org/AGL/releng-scripts/
$ cd releng-scripts
$ git checkout -b lavabox origin/lavabox

Generate example jobs:

$ ./utils/create-jobs.py --machine m3ulcb --url http://lavabox/ -o myjob.yaml

This time with some tests:

$ ./utils/create-jobs.py --machine m3ulcb --url http://lavabox/ -o myjob.yaml --test all

Take a look!

Take another look!
Submit jobs to LAVA

Authenticate with LAVA server (one-time setup)

$ apt-get install lava-tool

$ lava-tool auth-add http://demo@lavabox:10080/RPC2/
Enter token for http://demo@lavabox:10080/RPC2/:
Token added successfully for user demo.

Enter this token: tokendemo

Submit

$ lava-tool submit-job http://demo@lavabox:10080/RPC2/ myjob.yaml
submitted as job: http://lavabox:10080/scheduler/job/16

Keyring problems?

cat ~/.local/share/python_keyring/keyringrc.cfg
[backend]
default-keyring=keyring.backends.file.PlaintextKeyring
Test plans

Written as YAML templates, included into LAVA job.

A collection of “test suites”

Basic structure of a test plan (cf. releng-scripts/templates/tests)

- test:
  definitions:
    # The test suite definition that will be parsed and executed goes here
- test:
  definitions:
    # A second test suite
- test:
  definitions:
    # A third test suite, all part of the same test plan
Example test plan: remote git repo

Inside the test plan template
( local: releng-scripts/tests/templates)

- test:
  - timeout:
    - minutes: 2
  - definitions:
    - repository: git://github.com/baylibre/agl-test-definitions.git
      from: git
      path: examples/basic-inline.yaml
      name: test-example-basic-inline

Inside the remote git repo:

metadata:
  - name: metadata-name-basic-test
    format: "Lava-Test-Shell Test Definition 1.0"
    description: "A basic test definition."

run:
  - steps:
    - lava-test-set start constant
    - echo "Hello"
    - lava-test-case always-pass --result pass
    - lava-test-set stop constant
Example test plan: simple, inline commands

- test:
  timeout:
    minutes: 2
  definitions:
    - repository:
        metadata:
          name: metadataname-basictest
          format: "Lava-Test-Shell Test Definition 1.0"
          description: "A basic test definition."
        run:
          steps:
            - lava-test-set start basic-test-set
            - echo "Hello"
            - lava-test-case always-pass --result pass
            - lava-test-set stop basic-test-set
  from: inline
  name: test-example-basictest
  path: inline/basictest.yaml
Viewing test results
LAVA callbacks

LAVA v2 allows any job to send results via HTTP GET/PUT

Allows publishing / pushing results as to external tools/service as soon as job is finished.

External services can require authentication

```json
notify:
  criteria:
    status: finished
  callback:
    url: <URL>
    method: POST
    dataset: all
token: <auth token>
content-type: json
```
Example job with callbacks

This time with callbacks:

```
$ ./utils/create-jobs.py m3ulcb --urlbase demo -o myjob.yaml --test all
   --callback lab-baylibre-lavabox
```

Look for ‘notify’ section
• Jobs w/callbacks
• kernelCI backend
• kerneCI front-end
Next steps

- Updated, modern web UI
  - Flexible access to data results
- Better access to logs
  - e.g. jump directly to the relevant part from a test case
- Searchable logs
- Add yocto “ptests”

TBD: Add some nice captures
When LAVAbox is gone...
Moving from LAVAbox to AGL core lab

AGL LAVA server: [http://lava.automotivelinus.org/](http://lava.automotivelinus.org/)

- Need account (and auth-token) on LAVA server for today
  - User: AMMDEMO
  - Token: <ask instructor>
Detailed instructions for hands-on part:

https://goo.gl/R8ZUVJ