VLANd in LAVA

Submitting jobs to LAVA with VLAN support
Overview

- Device selection
- Information for test authors
- Designing your jobs with VLANs
- Considerations within a test
- Worked example
- Upcoming features
The LAVA Architecture

LAVA Scheduler

LAVA Dispatcher

Test Device
Test Device
Test Device
Test Device
Test Device
Test Device
Test Device
Test Device
Test Device

How the Scheduler picks devices

- Simple single-node tests
- Multinode tests with role(s)
- Multinode tests with role(s) and VLANs
  - More complicated, some gotchas
- Scheduler sees
  - Device type
  - Device tags
  - Interface tags
What are tags?

Device tags
- Arbitrary labels
- Typically hardware based
- **Whatever** properties need to be described
- Often describe optional peripherals
- Examples:
  - sata, usb-flash, hdmi, WiFi

Interface tags
- Arbitrary labels
- Typically hardware based
- **Whatever** properties need to be described
- Typically describe connectors and supported speeds
- Examples:
  - RJ45, SFP+, 10M, 1G
Information available to test authors

- Interfaces data exposed in server-side UI
  - Device Dictionary
- Includes lots of details
  - MAC
  - sysfs path
  - interface tags
- Test writers select via tags
  - Some information passed on to the tests
Designing your VLANd job

- Multinode job
  - Need to define a role for each device
- VLANs created before test, and destroyed after test; no control of VLANs within test
  - For now… :-)
- Multiple network interfaces needed
  - One must be primary - dispatcher connection
    - Don’t assign tags to this interface…!
  - All other interfaces are valid for VLANd use
Within your test - Multinode

- Get information via the Multinode API
  - Synchronise your various sub-jobs
  - `lava-vland-self` and `lava-vland-tags`

- Multinode API calls are handled over serial
  - Consider `dmesg -n`
  - Use secondary connections

- Use inline definitions for synchronisation

- [http://yaml-online-parser.appspot.com/](http://yaml-online-parser.appspot.com/)
Within your test - Networking

- Control your own network interfaces
  - Except the primary!
- Interface naming is **HARD**
  - eth0 might not be eth0 next boot
  - eth0 and eth1 might look identical
  - your test **MUST** work this out
- Assign IP addresses etc. as needed
  - IPv4 or IPv6 ?
  - On a separate VLAN, so don’t worry
(Brief) Worked example

- Preparation
- Pipeline design
- Roles
- Protocol
- References
- Questions
Worked example - preparation

- Start with a simple single node job for each device type
  - make sure the basics work first!
  - supplied kernel, initramfs, NFS actually boot etc.
- Change one element at a time
- Create a simple multinode job with all devices but with no VLANs.
- Use comments.
VLANd in LAVA uses the new design

- Called *pipeline* due to its design
- YAML job submissions
  - `# comments` are supported, use liberally
  - `dictionary_key: value`
  - `- list_item`
  - `# indenting is important!`
- YAML test shell definitions
- Inline YAML test shell definitions
- Ongoing development
Documentation

https://lng.validation.linaro.org/static/docs/vland.html

Always room for improvement - please file bugs!
Adding VLANs to a multinode job: Roles

lava-vland:
  client:
    # friendly name
    vlan_one:
      tags:
        - 100M
        - RJ45

lava-vland:
  server:
    # role label
    vlan_two:
      tags:
        - 1G
        - RJ45
Adding VLANs to a multinode job: Protocol

protocols:

# part of the deploy action
lava-vland:

# action name from pipeline
- action: lava-vland-overlay

# API call to the protocol
request: deploy_vlans
Example job references

For reference when you get home

https://lng.validation.linaro.org/scheduler/job/5600/multinode_definition

https://git.linaro.org/lava-team/refactoring.git/blob_plain/HEAD:/bbb-multinode-vland.yaml
Questions

So far ...
Upcoming features (pipeline)

- **Wider device support**
  - Juno, D02, Seattle, Versatile Express, ...
  - LXC containers, AOSP, Fast Models, ...

- **Device Dictionary visualisation**
  - More friendly display for test authors

- **More documentation :-)**
  - With detailed worked examples

- **Standard build images**
  - With instructions
Upcoming features (VLANd)

- Separating the VLAN deployment from the device boot
  - Control VLAN setup from within the job
  - Allows more complex test setups
  - Allows assignment of primary interface to a VLAN
    - At your own risk!
- Use more features of the VLANd API
- Live visualisation of VLANs
More questions