ifupdown2: The ultimate Network Interface Manager
Progress and Status

Julien Fortin - Cumulus Networks

July 5th, DebConf2016 - Cape Town, South Africa
Outline

- Background & context
- What’s wrong with ifupdown?
- Ifupdown2 overview
- What’s new since October 2015?
- What’s next?
- How to get started
- Questions
From intern to Member of Technical Staff

- Background as a Software Engineer
- EPITECH, class of 2016.
- Undergrad Internship at Cumulus in 2014
- Hired in March 2016, joining the ifupdown2 team
The Mother of Dragons

- Roopa Prabhu
- First commit Nov 2013, ~2k lines of Python
The Mother of Dragons

- Roopa Prabhu
- First commit Nov 2013, ~2k lines of Python
  - 618 commits
  - 18243 lines of python
Cumulus Linux

- Debian (jessie) based distribution for Network Switches

Our Philosophy

- Manage your network switch as a server
- Use existing Linux tools to configure your network switches

Contributing back to the community

- Cumulus is upstreaming a lot of networking kernel patches (e.g. netlink, stp etc..)
Debian’s ifupdown

- Default Network Interface Manager for Debian
- Configuration file: /etc/network/interfaces
- Network interfaces represented by “iface” sections

```plaintext
iface br100
  bridge-ports swp1 swp2
  bridge-stp on
```
What’s wrong with ifupdown?

- Code written in C (hard to maintain/extend)
- Does not understand interface dependency
- Problems with provisioning interfaces at scale
- No support for incremental changes to interface config
- No support to query/validate running configuration
Our solution: ifupdown2

- New implementation in Python
- Backward compatibility with debian ifupdown
- Pluggable architecture with python add-on modules
- Ordered Network Interface dependency relationship handling
- Incremental changes and query live configuration
ifupdown2 architecture

Life of an iface object

# /etc/network/interfaces
auto br0
iface br0 inet static
   address 10.0.14.2/24
   bridge-ports swp1.100 swp2.100
   bridge-stp on

bridge.py
address.py

ip link add dev br0 type bridge
brctl br0 stp on

ip addr add 10.0.14.2/24 dev br0
ifupdown2 interface configuration graph dependency

- Dependency graph of all interfaces
- Uses topological sort to order network interface configurations
- Execute interface configuration in dependency order
ifupdown2 interface configuration graph dependency

- Dependency graph of all interfaces
- Uses topological sort to order network interface configurations
- Execute interface configuration in dependency order

Sorted interface list:
swp1, swp1.100, swp2, swp2.100, swp3, swp4, bond, bond.100, bridge100
ifquery: Troubleshooting network interface dependencies

$ ifquery br0 --print-dependency=list
br0 : ['bond1', 'bond2']
bond1 : ['swp32', 'swp33']
bond2 : ['swp30', 'swp31']
swp32 : None
swp33 : None
swp30 : None
swp31 : None

$ ifquery br0 --print-dependency=dot
/* Generated by GvGen v.0.9 (http://software.inl.fr/trac/wiki/GvGen) */
digraph G {
compound=true;
  node1 [label="br0"];  
  node2 [label="bond1"];  
  node3 [label="bond2"];  
  node4 [label="swp32"];  
  node5 [label="swp33"];  
  node6 [label="swp30"];  
  node7 [label="swp31"];  
  node1->node2; node1->node3; node2->node4; node2->node5; node3->node6; node3->node7; }

digraph G {node1 [label="br0"]; node2 [label="bond1"]; node3 [label="bond2"]; node4 [label="swp32"]; node5 [label="swp33"]; node6 [label="swp30"]; node7 [label="swp31"]; node1->node2; node1->node3; node2->node4; node2->node5; node3->node6; node3->node7; }
ifquery - - check

- We extended ifquery to query and validate applied state of a network interface
ifquery to persist running configuration

```bash
root@cel-redxp-06:~# ip link add name bond42 type bond
root@cel-redxp-06:~# ip link set swp42 down
root@cel-redxp-06:~# ip link set swp42 master bond42
root@cel-redxp-06:~# ip link set swp42 up
root@cel-redxp-06:~# ip link add name bridge21 type bridge
root@cel-redxp-06:~# ip link set swp21 master bridge21
root@cel-redxp-06:~# ip link set swp12 master bridge21
root@cel-redxp-06:~#
root@cel-redxp-06:~# ifquery --running bond42 bridge21
auto bond42
iface bond42
  bond-lacp-bypass-allow 0
  bond-slaves swp42
  bond-mode 802.3ad
  bond-use-carrier 1
  bond-lacp-rate 1
  bond-min-links 1
  bond-mimon 100
  bond-xmit-hash-policy layer3+4
auto bridge21
iface bridge21
  bridge-mcriouter 1
  bridge-ports swp21 swp12
  bridge-mcrifaddr 0
  bridge-mcriquerier 0
  bridge-mcsnoop 1
root@cel-redxp-06:~# ifquery --running bond42 bridge21 >> /etc/network/interfaces
root@cel-redxp-06:~#
```
Incremental changes on live configuration

- ifup/down work on already UP or DOWN interfaces
- Only applies the new changes without disturbing the existing config on interface
Example

```
root@cel-redxp-06:~# ifquery br42
auto br42
iface br42
    bridge-ports swp42 swp42.42
    bridge-stp yes

root@cel-redxp-06:~# ifquery br42 --check
auto br42
iface br42
    bridge-ports swp42 swp42.42
    bridge-stp yes

root@cel-redxp-06:~# ip link set swp42.42 nomaster
root@cel-redxp-06:~# ifquery br42 --check
auto br42
iface br42
    bridge-ports swp42
    bridge-stp yes

root@cel-redxp-06:~# ip link set swp42.42 nomaster
root@cel-redxp-06:~# ifquery br42 --check
auto br42
iface br42
    bridge-ports swp42
    bridge-stp yes
```

Julien Fortin - ifupdown2 - DebConf2016

cumulusnetworks.com
ifreload

Reloads the network interface configuration

- By running `ifdown` on interfaces that were removed from /etc/network/interfaces (or the provided file)
- And running `ifup` on all interfaces currently present in the config file.
- Pretty convenient rather than using `service networking restart`
JSON API

- Supports interfaces imports and queries in JSON format
- Enables easier integration with automation/orchestration tools

```
$ ifquery br0 —format=json
{
  "auto": true,
  "config": {
    "bridge-ports": "swp1.42 swp2.42",
    "bridge-stp": "on"
  },
  "addr_method": "static",
  "name": "br0",
  "addr_family": "inet"
}
```

```
$ ifup -a -t json -i /etc/network/interfaces.json
```
Template support

- Simplify cookie-cutter interface configurations with mako style templates
  
  http://www.makotemplates.org/

- Provides option to plug-in alternate python template engines

Example: interfaces template to create bridges for vlan 1000 to 1100:

```
%for v in range(1000,1100):
    auto br-$v$
    iface br-$v$ inet static
    bridge-ports glob swp1-6.$v$
    bridge-stp on
    bridge-ageing 200
    bridge-maxage 10
    bridge-fd 10
%endfor
```
Python add-on modules for interface configuration

- address
- address virtual
- bond
- bridge
- bridgevlan
- dhcp
- ethtool
- link
- mstpctl
- usercmd
- vlan
- vrrpd
What’s new since October 2015?

Support for vlan filtering mode in the bridge driver

New add-ons

- VXLAN
- VRF

Performances

- Netlink

Policy manager

- To set or override system defaults
New bridge driver

VLAN-aware Bridges

- New 802.1Q Linux kernel bridge driver enable vlan aware mode

Helps with scale

- Avoids creating 802.1Q vlan netdevs. Bypass netdevs and assign vlan ids to bridge ports

ifupdown2

- Now supports vlan-aware or vlan-filtering bridges
- Continues to support traditional bridge configuration
### Two Trunks Containing 200 VLANs for swp1 and swp2

<table>
<thead>
<tr>
<th>Traditional</th>
<th>VLAN-Aware</th>
</tr>
</thead>
<tbody>
<tr>
<td>auto br-vlan1</td>
<td>auto bridge</td>
</tr>
<tr>
<td>iface br-vlan1</td>
<td>iface bridge</td>
</tr>
<tr>
<td>bridge-ports swp1 swp2</td>
<td>bridge-vlan-aware yes</td>
</tr>
<tr>
<td>auto br-vlan2</td>
<td>bridge-ports swp1 swp2</td>
</tr>
<tr>
<td>iface br-vlan2</td>
<td>bridge-vids 1-200</td>
</tr>
<tr>
<td>bridge-ports swp1.2 swp2.2</td>
<td>bridge-pvid 1</td>
</tr>
<tr>
<td>auto br-vlan3</td>
<td></td>
</tr>
<tr>
<td>iface br-vlan3</td>
<td></td>
</tr>
<tr>
<td>bridge-ports swp1.3 swp2.3</td>
<td></td>
</tr>
<tr>
<td>.</td>
<td></td>
</tr>
<tr>
<td>auto br-vlan200</td>
<td></td>
</tr>
<tr>
<td>iface br-vlan200</td>
<td></td>
</tr>
<tr>
<td>bridge-ports swp1.200 swp2.200</td>
<td></td>
</tr>
</tbody>
</table>
The VXLAN protocol is a tunneling protocol designed to solve the problem of limited VLAN IDs (4096)

- 24 bits VXLAN ID max value: 16,777,216

Network overlay

- Encapsulation of layer 2 frames using layer 3 UDP packets.
VXLAN example

auto vtep1000
iface vtep1000
  vxlan-id 1000
  vxlan-local-tunnelip 172.10.1.1

auto br-100
iface br-100
  bridge-ports swp1.100 swp2.100 vtep1000

auto vtep1000
iface vtep1000
  vxlan-id 1000
  vxlan-local-tunnelip 172.20.1.1

auto br-100
iface br-100
  bridge-ports swp1.100 swp2.100 vtep1000
IP technology that provides traffic isolation at layer 3 for routing

- Similar to how VLANs provide isolation at layer 2

Linux Implementation

- Kernel v4.3 and forward

Developed by Cumulus Networks for Linux

- Consistent API across all Linux Devices - switches and hosts
VRF example

```
auto all
iface red
d    address 127.0.0.1/8
vrf-table auto
iface swp1
    vrf red
iface swp2
    vrf red
```

OR

```
auto all
iface red
    address 127.0.0.1/8
    vrf-table auto
%for i in range(1, 4):
    iface swp${i}
        vrf red
%endfor
```
New policy infrastructure

Defining default value for add-ons

- There is the expectation that removing configuration parameters from the networking config file (and doing an `ifreload -a`) will restore a default parameter (if one exists)
- The JSON configuration will dictate default setting and behavior for each modules implementing this policy configuration

Default location

- `/var/lib/ifupdown2/policy.d/*.json`

User location

- `/etc/network/ifupdown2/policy.d/*.json`

By Sam Tannous from Cumulus
Policy Manager example

```
root@cel-redxp-06:/var/lib/ifupdown2/policy.d# cat address.json
{
    "address": {
        "defaults": { "mtu": "1500" }
    }
}
root@cel-redxp-06:/var/lib/ifupdown2/policy.d# cat bridge.json
{
    "bridge": {
        "module_globals": { "warn_on_untagged_bridge_absence": "yes" },
        "defaults": { "bridge-stp": "on" }
    }
}
root@cel-redxp-06:/var/lib/ifupdown2/policy.d# cat defaults_policy.json | head
{
    "README": "This file is automatically generated. Do not edit this file.",
    "ethtool": {
        "defaults": {},
        "iface_defaults": {
            "swpl": {
                "link-autoneg": "off",
                "link-duplex": "Full",
                "link-speed": "10000"
            }
        },
```
The Netlink transition

Ifupdown2 scale but we can do better

- We are using standard Linux tools, such as iproute2, for many operations
- Execution of external commands slow things down

Netlink is the answer

- Netlink is a IPC mechanism primarily between the kernel and user space processes.
- Moving from iproute2 to direct netlink drastically improve performances.

Next : pure Netlink backend

- We plan on having a pure netlink backend with possibility to fall back on iproute2 when netlink capabilities are not available.
In-house python netlink library for networking ops

- By Daniel Walton from Cumulus.

Soon available on Debian

- Right now the code is shipped within ifupdown2 package
### ifupdown

<table>
<thead>
<tr>
<th>Config size</th>
</tr>
</thead>
<tbody>
<tr>
<td>auto swp3</td>
</tr>
<tr>
<td>iface swp3 inet manual</td>
</tr>
<tr>
<td>up link set $IPACE up</td>
</tr>
<tr>
<td>down link set $IPACE down</td>
</tr>
<tr>
<td>auto swp4</td>
</tr>
<tr>
<td>iface swp4 inet manual</td>
</tr>
<tr>
<td>up link set $IPACE up</td>
</tr>
<tr>
<td>down link set $IPACE down</td>
</tr>
<tr>
<td>auto bond0</td>
</tr>
<tr>
<td>iface bond0 inet manual</td>
</tr>
<tr>
<td>up link set $IPACE up</td>
</tr>
<tr>
<td>down link set $IPACE down</td>
</tr>
<tr>
<td>bond-slaves swp3 swp4</td>
</tr>
<tr>
<td>bond-mode 802.3ad</td>
</tr>
<tr>
<td>bond-xmit-hash-policy layer3+4</td>
</tr>
<tr>
<td>bond-lag-up</td>
</tr>
<tr>
<td>auto bond0.100</td>
</tr>
<tr>
<td>iface bond0.100 inet manual</td>
</tr>
<tr>
<td>up link set $IPACE up</td>
</tr>
<tr>
<td>down link set $IPACE down</td>
</tr>
<tr>
<td>auto bond0.200</td>
</tr>
<tr>
<td>iface bond0.200 inet manual</td>
</tr>
<tr>
<td>up link set $IPACE up</td>
</tr>
<tr>
<td>down link set $IPACE down</td>
</tr>
<tr>
<td>auto vlan100</td>
</tr>
<tr>
<td>iface vlan100 inet manual</td>
</tr>
<tr>
<td>up link set $IPACE up</td>
</tr>
<tr>
<td>down link set $IPACE down</td>
</tr>
</tbody>
</table>

### ifupdown2

<table>
<thead>
<tr>
<th>Config size</th>
</tr>
</thead>
<tbody>
<tr>
<td>auto bond0</td>
</tr>
<tr>
<td>iface bond0</td>
</tr>
<tr>
<td>bond-slaves swp3 swp4</td>
</tr>
<tr>
<td>bond-mode 802.3ad</td>
</tr>
<tr>
<td>bond-xmit-hash-policy layer3+4</td>
</tr>
<tr>
<td>bond-lag-rate</td>
</tr>
<tr>
<td>auto vlan100</td>
</tr>
<tr>
<td>iface vlan100</td>
</tr>
<tr>
<td>bridge-ports bond0.100</td>
</tr>
<tr>
<td>bridge-stp on</td>
</tr>
<tr>
<td>auto vlan200</td>
</tr>
<tr>
<td>iface vlan200</td>
</tr>
<tr>
<td>bridge-ports bond0.200</td>
</tr>
<tr>
<td>bridge-stp on</td>
</tr>
</tbody>
</table>
Ifupdown2 on debian sid and jessie-backports

- Since October 2015 thanks to our sponsor (Piotr):
  https://packages.debian.org/sid/ifupdown2

- Available on apt or download the deb file
  echo "deb http://ftp.de.debian.org/debian sid main" >> /etc/apt/sources.list
What’s new?

Next

- Ifupdown2 version 2.0 before code freeze in December
- Increase performances (netlink and other stuff)
- Fix bugs reported by the community and Cumulus clients.
Getting started

- Ifupdown2 on github: https://github.com/CumulusNetworks/ifupdown2
- Ifupdown2 documentation: https://cumulusnetworks.github.io/ifupdown2
- Cumulus Linux ships with ifupdown2 in releases 2.1 or greater: https://cumulusnetworks.com
- Get in touch with us:
  - julien@cumulusnetworks.com
  - roopa@cumulusnetworks.com
Add addon module for B.A.T.M.A.N. advanced interface configuration. #12

BarbarossaTM wants to merge 1 commit into CumulusNetworks:master from BarbarossaTM:batman-adv

BarbarossaTM commented 18 days ago

This commit adds support for configuring B.A.T.M.A.N. advanced interfaces with ifupdown2. B.A.T.M.A.N. advanced is a protocol to build Layer2 based mesh networks with. It's supported in the Linux kernel and thus available in many Linux environments.

A configuration could look like this

    auto bat0
    iface bat0
    batman-interfaces eth1 eth2.23
    batman-interfaces-ignore-regex .*_nodes
    batman-hop-penalty 23
    #
    address 192.0.2.42/24
ifupdown2 user base

>1.5M
Ports powered by Cumulus Networks technology

465+
companies across for main industry verticals

145
Telecom / Service Provider

139
Public / Private Cloud

101
Government / Education / R & D

81
SaaS / Web 2.0
Bringing the Linux Revolution to Networking

Thank You!

julien@cumulusnetworks.com

© 2014 Cumulus Networks. Cumulus Networks, the Cumulus Networks Logo, and Cumulus Linux are trademarks or registered trademarks of Cumulus Networks, Inc. or its affiliates in the U.S. and other countries. Other names may be trademarks of their respective owners. The registered trademark Linux® is used pursuant to a sublicense from LMI, the exclusive licensee of Linus Torvalds, owner of the mark on a world-wide basis.