Testing our datapath

Our journey into creating a framework to automate testing of our datapath
Nick Bouliane
Software Engineer at DO since 2017

Started hacking on iptables/netfilter early 2000
http://people.netfilter.org/acidfu

Working on SDN primitives
Open vSwitch
Exploring ebpf
Blue Thunder Somogyi
Software Engineer at DO since 2018

Hacked XConq 1.0
(with help from K&R)
Spent too many years at Cisco
Huge DTrace and ZFS Fan
DigitalOcean

cloud-hosting company
1.15 million droplets
Topics

- Landscape of the datapath
- How things are organized
- What complexifies the testing of our datapath
- What is our datapath composed of
Networking at Digital Ocean

- Initially used linux bridge, iptables, ebtables and a bunch of perl scripts
- Started using OVS in 2014
- Helps unify the logic of our datapath
- Easier to test, reason about and less moving parts
Open vSwitch at Digital Ocean

- More than 18,500 hypervisors
- 12 Data centers
  - NYC1, NYC2, NYC3: New York City, United States
  - AMS2, AMS3: Amsterdam, the Netherlands
  - SFO1, SFO2: San Francisco, United States
  - SGP1: Singapore
  - LON1: London, United Kingdom
  - FRA1: Frankfurt, Germany
  - TOR1: Toronto, Canada
  - BLR1: Bangalore, India
Data center complexity
Open vSwitch version

- Ubuntu Trusty → Ubuntu Bionic
- Open vSwitch 2.7.3 → 2.11.0 (our own package)
- Bionic provides 2.9.2
Some projects that use openflow

- Floating IP
- Firewall
- VPC (Virtual Private Chassis)
- LBaaS - Load Balancer as a Service
Some projects that use openflow

- Bandwidth billing
- L3/Gateway
  - underlay traffic is now routed instead of being switched
- Internal services
  - DHCP (behind addr0 interface)
  - Metadata (behind meta0 interface)
  - ...
Hvflowd

- No SDN controller
- We control MAC and IP
- Push flows as soon as possible
- gRPC calls
  - Droplet creation
  - Firewall applied
- Use go-openvswitch
  - ovs-vsctl and ovs-ofctl
{  
    Priority: 4010,  
    Protocol: ovs.ProtocolUDPv4,  
    Matches: []ovs.Match{  
        ovs.TransportSourcePort(dhcp4Client),  
        ovs.TransportDestinationPort(dhcp4Server),  
    },  
    Table: tableForwarding,  
    Actions: []ovs.Action{  
        ovs.Output(addr),  
    },  
}
Recap

- Many projects
- Flowset orchestration
- Multiple configurations
The Datapath Validation Framework
Datapath (DP) Validation Framework Topics

- DP Validation Design Goals
- DP Validation Implementation Choice
- DP Validation Modes of Operation
- Example Validation Test
- Challenges Encountered With DP Validation
- Next Steps for DP Validation
Design Goals

- Detect breaking changes
- CI/CD Integration
- Non-disruptive Production Flow Validation
- Decouple Tests from OVS
- Improve product team agility
Implementation Direction

Utilized `go test` tooling driven by Make targets

- Allows for easy integration with CI/CD infrastructure (Concourse)

```plaintext
Datapath-CI  APP  2:29 PM
Datapath Validation Passed
Job run-L3-backfill-validation-on-HV-v2-twoHV for digitalocean/cthu
master@5adceb570a7 (Details)

Datapath Validation Passed
Job run-L2-validation-on-HV-v2-twoHV for digitalocean/cthu
master@5adceb570a7 (Details)
```

- Local testing with validation framework seamless
- `go test -o` binary generation
package pkt

type Port struct {
    OfPort int
    DpPort int
}

type Packet struct {
    Dropped bool
    InPort, OutPort Port
    CtNext    CtState
    Metadata uint64
    Frame     Frame
}

// ConvertOVS returns the go-openvswitch matches corresponding to this packet.
func (p *Packet) ConvertOVS() []ovs.Match {

package actions

type DataPathAction interface {
    Apply(*pkt.Packet) error
}

type Output struct {
    pkt.Port
}

func (action Output) Apply(packet *pkt.Packet) error {
    packet.OutPort = action.Port
    return nil
}

type Drop struct{

func (action Drop) Apply(packet *pkt.Packet) error {
    packet.Dropped = true
}
Example Test

TestL2V4InternetEgressArpRequestForGateway

```go
func TestL2V4InternetEgressArpRequestForGateway(t *testing.T, publicPort *netparams.NetworkParamsVNIC) {
    packet := pkt.Packet{
        InPort: f.GetPortByName(publicPort.Name).Port,
        Frame: &pkt.EthernetFrame{
            Src: sourceMac,
            Dst: "ff:ff:ff:ff:ff:ff",
            Frame: &pkt.ArpFrame{
                Op:  f.ArpOpRequest,
                Sha: sourceMac,
                Spa: address,
                Tpa: gw,
            },
        },
    }
    port := f.GetHVPublicPort()
    expected := []actions.DataPathAction{
        actions.PushVlan{Vid: vlan},
        actions.Output{Port: port.Port},
    }
    if f.HvConf.L3State == hvflow.Layer3GatewayStateCompleteStr {
        port := f.GetPortByName(f.RespondPort)
        expected = []actions.DataPathAction{
            actions.Output{Port: port.Port},
        }
    }
    f.ValidateDataPathActions(t, packet, expected)
}
```
Modes of Operation

- **Local** `make test` or `make <test target>`
- **Local** `make sandbox`
- **Execution of `validate-dp` binary on staging or production hosts**

```
/opt/apps/hvflow/bin/validate-dp --hv /etc/dp-validation.yaml --vpc
/etc/dp-validation/testbed/vpcs/2.json --droplet
/etc/dp-validation/testbed/droplets/1194037.json --remote-droplet
TestDroplet2RemoteDroplet
```
Implementation Challenges

- HVFlowd Interface Expectations
- Static HV and Droplet configurations
- Test-to-Configuration Mapping & Test Coverage
- No-ops and OVS Action String Ordering
Bugs Found

- Removal of Legacy (pre-encapsulation) VLAN from private traffic causes v2/v3 Interop problem
- Incorrect Priority on Overlapping IP addresses (in fix for above issues)

cbaldwin approved these changes on Oct 21

cbaldwin left a comment

I think it is super cool that this was exposed using datapath validation tests.
What’s Next

- Dynamic Configuration
- Table-Driven Tests
- Test Coverage Tracking
- Connection Tracker Traversal
- HVFlowd Binary Testing
- Datapath Versioning
Conclusion

- Confidence Provided by Version 1 of Datapath Validation
- Instrumental in both L3 Public rollout and VPCv3 migrations
- Rapid Growth of Number of Tests and Configurations Supported Created Usability Challenges
- Existing Validation Framework a Solid Foundation for Next Generation of Validation Features
Thank You!

digitalOcean