SDN with Link-State Routing Protocols

Olivier Tilmans
UCLouvain

PIRL
8th March 2016

Joint work with
S. Vissicchio (UCLouvain), L. Vanbever (ETH Zurich) and J. Rexford (Princeton)
IPv6 comes with a unique opportunity
IPv6 comes with a unique opportunity to improve configuration automation.
IPv6 comes with a unique opportunity to simplify the protocol stack.
IPv6 comes with a unique opportunity to change network designs
The state of the art includes two networking models based on opposite principles.

- **Traditional** (e.g., IGP, distributed MPLS)
- **SDN** (e.g., OpenFlow, Segment Routing)
SDN simplifies control-plane and management, but *sacrifices* robustness of distributed protocols.

<table>
<thead>
<tr>
<th>Characteristic</th>
<th>Traditional</th>
<th>SDN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Manageability</td>
<td>low</td>
<td>high</td>
</tr>
<tr>
<td>Flexibility</td>
<td>low</td>
<td>highest</td>
</tr>
<tr>
<td>Scalability</td>
<td>by design</td>
<td>ad hoc</td>
</tr>
<tr>
<td>Robustness</td>
<td>high</td>
<td>low</td>
</tr>
</tbody>
</table>
SDN simplifies control-plane and management, but *sacrifices* robustness of distributed protocols.

<table>
<thead>
<tr>
<th>Manageability</th>
<th>low</th>
<th>high</th>
</tr>
</thead>
<tbody>
<tr>
<td>Flexibility</td>
<td>low</td>
<td>highest</td>
</tr>
<tr>
<td>Scalability</td>
<td>by design</td>
<td>ad hoc</td>
</tr>
<tr>
<td>Robustness</td>
<td>high</td>
<td>low</td>
</tr>
</tbody>
</table>
We propose Fibbing, a network architecture which combines advantages of SDN and traditional networking

Fibbing

central control over a single link-state IGP
SDN with Link-State Routing Protocols

1. Manageability
2. Flexibility
3. Scalability & Robustness
SDN with Link-State Routing Protocols

1. Manageability
   achieving central control

2. Flexibility

3. Scalability & Robustness
Consider this simple network (implemented with Cisco routers)
An IGP control-plane computes shortest paths on a shared weighted topology.
IGP shortest paths are translated into forwarding paths on the data-plane.
In Fibbing, operators can ask the controller to modify forwarding paths.
The Fibbing controller injects information on *fake nodes and links* into the IGP control-plane.
Informations are flooded to all IGP routers in the network.
Fibbing messages *augment* the topology seen by all IGP routers.
Augmented topologies translate into new control-plane paths

requirement $(C,A,B,X)$

node V1 link $(V1,C)$ map $(V1,C)$ to $(C,A)$
Augmented topologies translate into new *data-plane* paths.

**Diagram:**
- Node V1
- Link (V1, C)
- Map (V1, C) to (C, A)

**Requirement:**
(C, A, B, X)
SDN with Link-State Routing Protocols

1. Manageability
2. Flexibility
3. Scalability & Robustness
Fibbing can enforce any set of forwarding DAGs
Fibbing can enforce any set of forwarding DAGs paths for the same destination not creating loops
SDN with Link-State Routing Protocols

1. Manageability

2. Flexibility
   - fine-grained control

3. Scalability & Robustness
In the following network, the blue destination is subject to a DoS attack.
Fibbing can steer away traffic on a per-destination basis

requirement $(C,D,X)$ and $(B,C,D,X)$
SDN with Link-State Routing Protocols

1. Manageability

2. Flexibility
   per-destination load-balancing

3. Scalability & Robustness
Leveraging multiple paths is hard when links/flows have different capacities/demands.

![Diagram showing network with different link bandwidths and traffic demands.](image-url)
Fibbing has fine-grained control over ECMP routing

Adding new equal-cost path

requirement
(A, C, X) and
(A, B, X) and
(A, D, X)
Fibbing has fine-grained control over ECMP routing
Introducing uneven load-balancing

requirements
splittingRatios
{
  (A,B): 2
  (A,C): 1
}
SDN with Link-State Routing Protocols

1. Manageability
2. **Flexibility**
   backup paths provisioning
3. Scalability & Robustness
Link failures may induce congestion or increased delays
Fibbing can provision backup paths

requirement
\((A, D, X)\) asBackupOf \((A, B, X)\)
SDN with Link-State Routing Protocols

1 Manageability

2 Flexibility

3 Scalability & Robustness
   IGP on steroids
We implemented a Fibbing controller

- Supports all presented use-cases
  Source-code on Github

- Works with vanilla OSPF and off-the-shelf routers
  IS-IS requires a protocol extension

- Induces very little overhead on the routers
  No impact on SPF computation
By building upon the underlying IGP, Fibbing is robust and reactive to failures

- Fibbing easily deals with network failures
  1. IGP is sufficient for some failures [Filsfils07]
  2. IGP provides a default for partitions

- Fibbing naturally supports replicated controllers
  1. IGP provides sync primitives
  2. replica failures have no impact on forwarding
Fibbing shows the *benefits* of central control over distributed protocols

- Realizes SDN management model
  network-wide automated control

- Simplifies controllers and improves robustness
  heavy work is still done by routers

- Simplifies network design
  IGPs are in charge of all intra-domain paths
SDN with Link-State Routing Protocols

Olivier Tilmans

olivier.tilmans@uclouvain.be

Tell me lies, tell me sweet little lies
— Fleetwood Mac

fibbing.net
SDN with Link-State Routing Protocols

Backup slides
MPLS+RSVP-TE/SR can solve all the presented use-cases

- Need to provision one tunnel per ingress point
- Hard to add/remove equal-paths for elephant flows
- Fibbing also controls path cost seen by other protocols
Fibbing manipulates IGP topology, so does MTR, what’s the difference?

- MTR is CLI-driven (configuration changes to do on every router vs flooding)
- Cannot do uneven load-balancing
Why shouldn’t I use Policy-Based Routing?

- PBR is CLI-driven (configuration changes to do on every router vs flooding)
- PBR decisions are local to a single router
- CPU fallback
How am I supposed to troubleshoot a network with fake elements?

- The controller is the primary source of information
- Fake elements can be quickly identified in LSDBs
Openflow solves everything

- The controller has to setup flow entries on every switch
- Switches rely on the controller to handle failures
- IGPs are getting extensions to support Flowspec, …
Experiments on real routers show that Fibbing has very limited impact on routers.

<table>
<thead>
<tr>
<th># fake nodes</th>
<th>router memory (MB)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 000</td>
<td>0.7</td>
</tr>
<tr>
<td>5 000</td>
<td>6.8</td>
</tr>
<tr>
<td>10 000</td>
<td>14.5</td>
</tr>
<tr>
<td>50 000</td>
<td>76.0</td>
</tr>
<tr>
<td>100 000</td>
<td>153</td>
</tr>
</tbody>
</table>

CPU utilization always under 4%
The controller can choose between a (very) fast algorithm or one that minimize the augmented topology Rocketfuel topology of AS1239 (300+ routers)
We study which messages to inject for controlling intra-domain routing protocols.
The output of the controlled protocol is specified by operators’ requirements.
To control IGP output, the Fibbing controller inverts the shortest-path function
SDN achieves high manageability by relying on a centralized controller.
Fibbing is as manageable as SDN, but centralizes only high-level decisions.
Fibbing keeps installation distributed, relying on distributed protocols.
Distributed installation is controlled by injecting carefully-computed information.
Fibbing combines advantages of SDN and traditional networking.

<table>
<thead>
<tr>
<th>Feature</th>
<th>Traditional</th>
<th>Fibbing</th>
<th>SDN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Manageability</td>
<td>low</td>
<td>high</td>
<td>high</td>
</tr>
<tr>
<td>Flexibility</td>
<td>low</td>
<td>high</td>
<td>highest</td>
</tr>
<tr>
<td>Scalability</td>
<td>by design</td>
<td>by design</td>
<td>ad hoc</td>
</tr>
<tr>
<td>Robustness</td>
<td>high</td>
<td>high</td>
<td>low</td>
</tr>
</tbody>
</table>
Fibbing *combines* advantages of SDN and traditional networking

- **Manageability**: high
- **Flexibility**: high
- **Scalability**: by design
- **Robustness**: high

- **centralized controller**
- **per-destination full control**
- **some functions are distributed**
Our prototype includes algorithms to compute augmented topologies of limited size.
The controller listens through an OSPF adjacency to keep an up-to-date view of the topology

Compilation  Augmentation  Optimization

+ network topology → per-destination forwarding DAGs → augmented topology → reduced topology

Injection/Monitoring

running network

path reqs.