Enabling Policy Innovation in Interdomain Routing: A Software-Defined Approach

Anduo Wang*  Zhijia Chen*  Tony Yang†  Minlan Yu‡
*Temple University  †Johns Hopkins University  ‡Harvard University
interdomain routing

determining data path connecting communicating hosts
interdomain routing

border gateway protocol (BGP) — the only de-facto interdomain routing system
BGP and AS policy

BGP supports AS policies
- extending path-vector system
- overriding the shortest AS-path behavior
BGP and AS policy

(example) hot potato policy of AS 3

- select a path that minimizes internal cost
influence policy in the downstream

can AS 4 / AS 5 influence routes in AS3?
influence the downstream

can AS 4 / AS 5 influence routes in AS3?
- AS 5 requests AS 3 to bypass AS 2?
can AS 4 / AS 5 influence routes in AS3?

- AS 5 requests AS 3 to bypass AS 2?
- AS 4 demands AS 3 for joint traffic engineering?
influence the downstream

but the flow of policy attribute in BGP is unidirectional

− policy carried by the routes
BGP policy is restricted new extensions to path-vector?
influence the downstream

can AS 4 / AS 5 influence routes in AS3?

- AS 5 … — BGP + negotiation [MIRO, sigcomm’08]
- AS 4 … — BGP + new attribute [Wiser, sigcomm’07]
coordination

simultaneously, AS 5 wants to avoid AS 2, AS 4 demands joint TE?
coordination

still impossible with (BGP + MIRO + Wiser)
coordination

(AS 5 avoids AS 2)+(AS 4 demands joint TE)

- AS 3 needs to *properly* combine the sub-routes: simple concatenation does not work!
coordination

(AS 5 avoids AS 2)+(AS 4 demands joint TE)

- AS 3 needs to resolve conflicts — modify the subpath by the less important policy (AS 4 demands joint TE)
coupling routes and policies — including any path vector based policy (BGP, MIRO, Wiser …) — is inherently flawed
separate policies from routes

a new policy system with SDN

- SDN controller: route discovery and dissemination
- policy system: express and process high-level intention
separate policies from routes

a new policy system with SDN

- SDN controller: route discovery and dissemination
- policy system: express and process high-level intention

key idea: making policies logic statements that — like routes — freely flow and interact
exchangeable logic policy

more flexible

- logic unifies disparate policies

![Diagram showing AS 1 and AS 2 with route compliance and policy flow]

policy L:
admin(r) \Rightarrow safe(r)
cust(r) \Rightarrow safe(r)

route flow → policy flow (L, X, Y, ...) →
exchangeable logic policy

more flexible

- logic unifies disparate policies
- immediately allows control of the downstream

route flow ➔ policy flow (L,X,Y,...) ➔
exchangeable logic policy

more flexible

- logic unifies disparate policies
- immediately allows control of the downstream

route flow ➔ policy flow (L, X, Y, …) ➔
exchangeable logic policy

more flexible

- logic unifies disparate policies
- immediately allows control of the downstream

route flow ➔ policy flow (L,X,Y,...) ➔
exchangeable logic policy

- compute the “impact” of a policy on another policy

policy Y:
cust(r) ⇒ fast(r)

policy L:
admin(r) ⇒ safe(r)
cust(r) ⇒ safe(r)

route flow ➔ policy flow (L, X, Y, …) ➔
exchangeable logic policy

co�dination

- compute the “impact” of a policy on another policy

but some routes cannot be simultaneously safe and fast

route flow → policy flow (L,X,Y,...) →
exchangeable logic policy

coordination

- compute the “impact” of a policy on another policy

route flow → policy flow (L,X,Y,...) →

cust(r) ⇒ fast(r)

cust(r) ⇒ safe(r)
cust(r) ∧ safe(r) ⇒ fast(r)

cust(r) ⇒ safe(r)

cust(r) ⇒ safe(r)

cust(r) ⇒ fast(r)

X-compliant route

policy L: admin(r) ⇒ safe(r)
cust(r) ⇒ safe(r)

AS 1

AS 2

AS 3
exchangeable logic policy

coordination
- compute the “impact” of a policy on another policy

route flow → policy flow (L,X,Y,...) →

- policy Y: cust(r) ⇒ fast(r)
- policy L: admin(r) ⇒ safe(r)
- cust(r) ⇒ safe(r)
exchangeable logic policy

coordination

- compute the “impact” of a policy on another policy

route flow → policy flow (L,X,Y,...)
exchangeable logic policy

coordination
- compute the “impact” of a policy on another policy

cust(r) ⇒ safe(r)
cust(r) ∧ safe(r) ⇒ fast(r)

policy Y:
cust(r) ⇒ fast(r)

policy L:
admin(r) ⇒ safe(r)
cust(r) ⇒ safe(r)

route flow policy flow (L,X,Y,...)
exchangeable logic policy

coordination
- compute the “impact” of a policy on another policy

route flow → policy flow (L,X,Y,...) →
Boléro — a realization with Ravel
Boléro — a realization with Ravel

Ravel
- uses a high-performance database as the controller

Boléro uses a high-performance database as the controller.
Boléro — a realization with Ravel

Ravel
- uses a high-performance database as the controller
- discovers and disseminates routes (maintain database tables)
Boléro — a realization with Ravel

**Ravel controller**
- uses a high-performance database as the controller
- discovers and disseminates BGP routes

**Boléro policy system**
- exchanges and processes logic policies
**Boléro — a realization with Ravel**

**Ravel controller**
- uses a high-performance database as the controller
- discovers and disseminates BGP routes

**Boléro policy system**
- exchanges and processes logic policies

**Key idea:** process policy — semantic knowledge — by semantic transformation

---

**Diagram:**
- **AS** nodes: AS 2, AS 3, AS 4
- **Routes:** X, Y, X&Y'-compliant route
- **Policy:** Y', OK
- **Compliant routes:** Y'-compliant route
- **Network:** Ravel controller — uses a high-performance database as the controller
- **Logic policies:** exchanges and processes

**Notes:**
- **Boléro — a realization with Ravel**
- **Ravel controller**
- **Boléro policy system**
- **Key idea:** process policy — semantic knowledge — by semantic transformation
Boléro overview

- flexible policy routing
- policy coordination

**Transformation**

- incoming routes from neighbors
- neighbor policy
- policy suggestion (orchestrated policy set)
- local policy
- outgoing policy compliant routes to neighbors
Boléro overview

- Transformation
  - Policy routing
  - Transform routes using the policies

- Incoming routes from neighbors
- Neighbor policy
- Policy suggestion (orchestrated policy set)
- Local policy
- Outgoing policy compliant routes to neighbors
Boléro overview

transformation
- policy routing
  - transform routes using the policies
- policy coordination
  - transform one by another policy

incoming routes from neighbors

neighbor policy

policy suggestion (orchestrated policy set)

local policy

outgoing policy compliant routes to neighbors
representation

<table>
<thead>
<tr>
<th>prefix</th>
<th>next hop</th>
<th>AS path</th>
<th>...</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- routing state as queryable tables
- factual data
representation

| prefix | next hop | AS path | ...
|--------|----------|---------|-----
|        |          |         |     |
|        |          |         |     |

--- MIRO-like policy
\[-\text{route}(D,N,P), \ ('AS2' \text{ in } P).\]

- routing state as queryable tables
- factual data
- policy as data integrity constraint (IC)
- logical statement about what (must be avoided) are the valid route data
- semantic data
### Routing State as Queryable Tables

- **Factual Data**

### Policy as Data Integrity Constraint (IC)

- Logical statement about what (must be avoided) are the valid route data

### Semantic Data

--- MIRO-like policy

```prolog
:- route(D,N,P), ('AS2' in P).
```

<table>
<thead>
<tr>
<th>prefix</th>
<th>next hop</th>
<th>AS path</th>
<th>...</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Meaning**

```
False <- route(I,R,P)
\{('AS2' in P)}
```
--- MIRO-like policy
:- route(D,N,P), ('AS2' in P).

--- Wiser-like policy
:- route(D,N,P),
   Wiser(D,R,C),Advertise(R,C₂),
   Wiser(D,R',C'),Advertise(R',C₂'),
   C+C₂>C'+C₂'.

- routing state as queryable tables
- factual data
- policy as data integrity constraint (IC)
- logical statement about what (must be avoided) are the valid route data
- semantic data
route transformation

- compute the “impact” of a policy
route transformation

MIRO policy (avoiding AS 2)

\[-\text{route}(l, R, P), ('AS 2' in P)\]

\[-('AS 2' in P)\]

- compute the “impact” of a policy
- obtain residue by partial subsumption
route transformation

MIRO policy (avoiding AS 2)

route(I,R,P), ('AS 2' in P)

:- ('AS 2' in P)

route(I,R,P)

compute the “impact” of a policy

obtain residue by partial subsumption

attach the residue

MIRO-compliant route

route(I,R,P) {:-('AS2' in P)}
route transformation

MIRO policy (avoiding AS 2)

route(I,R,P), ('AS 2' in P)

route(I,R,P)

:- ('AS 2' in P)

MIRO-compliant route

route(I,R,P) {:-('AS2' in P)}

meaning

{¬('AS2' in P)}

- compute the “impact” of a policy
- obtain residue by partial subsumption
- attach the residue
policy transformation

--- Wiser policy
:- route(D,N,P),
   Wiser(D,R,C), Advertise(R,C₂),
   Wiser(D,R',C'), Advertise(R',C₂'),
   C+C₂>C'+C₂'.

--- MIRO-compliant Wiser policy
:- route(D,N,P),
   Wiser(D,R,C), Advertise(R,C₂),
   Wiser(D,R',C'), Advertise(R',C₂'),
   C+C₂>C'+C₂', \{-('AS2' in P)}.
preliminary result

time the database delay for computing policy-compliant routes

- policies
  - MIRO
  - Wiser
  - MIRO&Wiser

- topology: Pocketful ISP topology embedded in Skitter AS-level topology

- incoming routes
  - Routeview BGP feeds
Preliminary results indicate a small and scalable database delay for 10,000 BGP feeds. The CDF graph shows:

- Delay is small and scales well.
- 95% route insertion < 1.424ms
- 95% MIRO < 0.174ms
- 95% Wiser < 0.64ms
- 95% Wiser & MIRO < 0.844ms
preliminary result

time the database delay for 10,000 BGP feeds

- delay is small and scales well
- delay grows as policy becomes more complex
conclusion

coupling routes and policies is inherently flawed

this talk

- decouple policies from routes by a new policy system with SDN

benefits

- flexible policies
- automated coordination

future work

- implementation
- anonymize policies