Distributed Route Aggregation on the Global Network (DRAGON)

João Luís Sobrinho

Laurent Vanbever, Franck Le, Jennifer Rexford

ACM CoNEXT 2014, Sydney

1 Instituto de Telecomunicações, 1 IST Universidade de Lisboa
2 ETH Zurich, 3 IBM T. J. Watson Research, 4 Princeton University
Last year in the news (August 2014)

Echoes of Y2K: Engineers Buzz That Internet Is Outgrowing Its Gear
Routers That Send Data Online Could Become Overloaded as Number of Internet Routes Hits '512K'

By DREW FITZGERALD
Updated Aug. 13, 2014 7:38 p.m. ET

Browsing speeds may slow as net hardware bug bites
By Mark Ward
Technology correspondent, BBC News

Some routers could not process the +512 K IPv4 prefixes they were learning about
Not a scalable routing system

Most of the originated prefixes are routed globally (by BGP)
Not a scalable routing system

Most of the originated prefixes are routed globally (by BGP)
Not a scalable routing system

Most of the originated prefixes are routed globally (by BGP)
No scalability: poor performance

- Forwarding tables (FIBs) growth & address look-up time increase
- Routing tables (RIBs) growth
- BGP session set-up time increase
- Churn & convergence time increase
Further scalability concerns

• IPv6 prefixes can be formed in potentially larger numbers than IPv4 prefixes

• Secure BGP adds computational overhead to routing processes
DRAGON

Distributed solution to scale the Internet routing system

**Basic DRAGON**: 49% savings on routing state

**Full DRAGON**: 79% savings on routing state

*No changes to the BGP protocol*

*No changes to the forwarding plane*

**Readily implemented with updated router software**
Outline

• Scalability: global view
• DRAGON: filtering strategy
• DRAGON: aggregation strategy
• DRAGON: performance evaluation
• Conclusions
Outline

• Scalability: global view
  • DRAGON: filtering strategy
  • DRAGON: aggregation strategy
  • DRAGON: performance evaluation
• Conclusions
Scalability: global view (routing)

Specificity
Prefix q is more specific than prefix p if the bits of p are the first bits of q

Propagation of more specific prefixes only in a small vicinity of their origin ASs
Scalability: global view (forwarding)

Most ASs forward data-packets on the (aggregated) less specific prefixes
Scalability: global view (forwarding)

AS 1  1.0.0.0/16

AS 2
  1.0.0.0/16 origin
  1.0.1.0/24

dest. addr.  data-packet

1.0.1.1

AS 3
  1.0.0.0/16
  1.0.1.0/24 origin
Hope for scalability? Hierarchies

AS hierarchy aligned with prefix hierarchy

- Provider: AS 1 (1.0.0.0/16)
  - Customer: AS 2 (1.0.1.0/24)
Hope for scalability? Clustering

Routing Information Registry (RIR)

AS 1
1.0.0.0/24

AS 2
1.0.1.0/24

AS 3
1.0.0.0/24 + 1.0.1.0/24 + 1.0.2.0/23 = 1.0.0.0/22

1.0.1.0/24

1.0.2.0/23

AS 4

AS 5

AS 6

1.0.0.0/24

1.0.1.0/24

1.0.2.0/23

Geography *roughly* clusters together ASs with aggregatable address space
Challenge: global vs. local

How to realize the global view through automated local routing decisions?

especially, given that the Internet routing system is as decentralized as it can be:

• each AS decides where to connect
• each AS decides where to acquire address space
• each AS sets its own routing policies
Outline

• Scalability: global view

• DRAGON: filtering strategy

• DRAGON: aggregation strategy

• DRAGON: performance evaluation

• Conclusions
Filtering strategy

• Locally filter the more specific prefixes when possible
  – no black holes
  – respect routing policies

• Use built-in incentives to filter locally
  – save on forwarding state
  – forward along best route (dictated by routing policies)

• Exchange routing information with standard BGP
Providers, customers, and peers

![Diagram showing relationships between providers, customers, and peers]
Prefixes

#6 originates q (1.0.0.0/24); #4 originates p (1.0.0.0/16)

q more specific than p
Routes

Route
Association between a prefix and an attribute, from a totally ordered set of attributes

$p$: origin

$q$: origin

$q$-route (route pertaining to $q$)
Gao-Rexford routing policies

preferences: customer then peer then provider

exportations: all routes from customers; all routes to customers

route attributes:
“customer”
“peer”
“provider”

q-route
Gao-Rexford routing policies

preferences: customer then peer then provider

exportations: all routes from customers; all routes to customers
Gao-Rexford routing policies

preferences: customer then peer then provider

exportations: all routes from customers; all routes to customers
Gao-Rexford routing policies

preferences: customer then peer then provider

exportations: all routes from customers; all routes to customers

route attributes:
“customer”
“peer”
“provider”

$q$-route
Final state for prefix $q$

route attributes:
“customer”
“peer”
“provider”
Final state for prefixes $q$ and $p$

<table>
<thead>
<tr>
<th>Node</th>
<th>$p$</th>
<th>$q$</th>
</tr>
</thead>
<tbody>
<tr>
<td>#1</td>
<td>peer</td>
<td>peer</td>
</tr>
<tr>
<td>#2</td>
<td>cust.</td>
<td>cust.</td>
</tr>
<tr>
<td>#3</td>
<td>prov.</td>
<td>cust.</td>
</tr>
<tr>
<td>#4</td>
<td>origin</td>
<td>cust.</td>
</tr>
<tr>
<td>#5</td>
<td>prov.</td>
<td>prov.</td>
</tr>
<tr>
<td>#6</td>
<td>prov.</td>
<td>origin</td>
</tr>
</tbody>
</table>

**route attributes:**
- “customer”
- “peer”
- “provider”

**forwarding:** longest prefix match rule
Filtering code (FC)

Other than origin of $p$, in the presence of $p$, filter $q$ if only if:

- attribute of $p$-route same or preferred to attribute of $q$-route

$\sqrt{\text{ASs that filter } q \text{ upon execution of FC}}$
AS 2 applies FC

- AS 2 filters $q$
  - AS 2 saves on forwarding state
  - AS 1 is oblivious of $q$; it saves on forwarding and routing state
All ASs apply FC

AS 1, AS 2, and AS 3 forgo q → forwarding to q using less specific p

filtered prefix

AS forgoes q
Global property: correctness

Correctness: no routing anomalies (no black holes)
Global property: route consistency

Route consistency: attribute of route used to forward data-packets is preserved

Optimal route consistency: set of ASs that forgo $q$ is maximal
Partial deployment

forwarding data-packets with destination in q

ASs that filter q upon execution of FC
AS 2 (and AS 3) has a double incentive to apply the FC:
• saves on forwarding state
• improves attribute of route used to forward data-packets
Partial deployment: incentives

AS 2 applies FC

AS 2 reverts to forwarding data-packets with address in \( q \) to AS 4

Forwarding data-packets with destination in \( q \)
Partial deployment: route consistency

- Forwarding data-packets with destination in $q$
First to apply FC are ASs that elect a peer or provider q-route
Partial deployment: route consistency

Next to apply FC are ASs for which providers have already applied FC
Partial deployment: route consistency

Next to apply FC are ASs for which providers have already applied FC
Filtering strategy: general case

• Trees of prefixes learned from BGP
  – FC for a prefix in relation to the parent prefix

• Correctness
  – for the routing policies for which BGP is correct

• Route consistency (optimal and through partial deployment)
  – for isotone routing policies (includes Gao-Rexford)

Optimal route consistency is not synonymous with efficiency (think shortest paths)
Outline

• Scalability: global view
• DRAGON: filtering strategy
• DRAGON: aggregation strategy
• DRAGON: performance evaluation
• Conclusions
Aggregation strategy

• Locally originate aggregation prefixes when beneficial
  – new address space is *not* created
  – allow filtering of provider-independent prefixes
  – self-organization when more than one AS originates the same aggregation prefix

• *Again, exchange routing information with standard BGP*
Aggregation prefix

1. no routable address space is created
2. at least two covered prefixes
3. customer route is elected for each of the covered prefixes

\[ p0 + p10 + p11 = p; \ p \text{ is an aggregation prefix at AS 3} \]
AS 3 originates $p$

AS 1 is oblivious of $p_0$, $p_{10}$, and $p_{11}$

AS 2 filters $p_0$, $p_{10}$, and $p_{11}$

AS 4 filters $p_{10}$ and $p_{11}$

AS 5 filters $p_0$ and $p_{11}$

AS 6 filters $p_0$ and $p_{10}$
Aggregation strategy: general case

• Trees of prefixes learned from BGP
  – aggregation prefixes cover parentless prefixes

• Self-organization
  – for the routing policies for which BGP is correct

• Optimal origins
  – for isotone routing policies (includes Gao-Rexford)
Outline

• Scalability: global view
• DRAGON: filtering strategy
• DRAGON: aggregation strategy
• DRAGON: performance evaluation
• Conclusions
Data-sets

• Annotated topology (CAIDA, Feb. 2015)
  – ~50K ASs; ~42K stub ASs
  – ~94K provider links; ~94K customer links; 180K peer links

• IPv4-prefixes-to-ASs mapping (CAIDA, Feb. 2015)
  – ~530K prefixes
  – ~270K parentless prefixes
  – ~210K prefixes have same origin AS as parent
FIB filtering efficiency: definition

Normalized amount of reduction brought by DRAGON to the forwarding tables of an AS

\[
\text{FilterEff} = \frac{\# \text{(FIB entries BGP)} - \# \text{(FIB entries DRAGON)}}{\# \text{(FIB entries BGP)}}
\]
### FIB filtering efficiency: results

<table>
<thead>
<tr>
<th></th>
<th>Basic DRAGON (filtering)</th>
<th>Full DRAGON (filtering &amp; aggregation)</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Min. FilterEff</strong></td>
<td>47%</td>
<td></td>
</tr>
<tr>
<td><strong>% of ASs with at least Min. FilterEff</strong></td>
<td>100%</td>
<td></td>
</tr>
<tr>
<td><strong>Max. FilterEff</strong></td>
<td>49%</td>
<td></td>
</tr>
<tr>
<td><strong>% of ASs attaining Max. FilterEff</strong></td>
<td>87%</td>
<td></td>
</tr>
</tbody>
</table>
# FIB filtering efficiency: results

<table>
<thead>
<tr>
<th></th>
<th>Basic DRAGON</th>
<th>Full DRAGON</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>filtering</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Min. FilterEff</td>
<td>47%</td>
<td>69%</td>
</tr>
<tr>
<td>% of ASs with at least Min. FilterEff</td>
<td>100%</td>
<td>100%</td>
</tr>
<tr>
<td>Max. FilterEff</td>
<td>49%</td>
<td>79%</td>
</tr>
<tr>
<td>% of ASs attaining Max. FilterEff</td>
<td>87%</td>
<td>87%</td>
</tr>
</tbody>
</table>
Outline

• Scalability: global view
• DRAGON: filtering strategy
• DRAGON: aggregation strategy
• DRAGON: performance evaluation
• Conclusions
Conclusions

• DRAGON is a BGP add-on to scale the Internet routing system

• DRAGON can be deployed incrementally

• DRAGON reduces the amount of forwarding state by approximately 80%

• DRAGON is – more fundamentally – a solid framework to reason about route aggregation
Thank you!

Visit us at

www.route-aggregation.net

Thank you!