Making the Internet more scalable and manageable

Laurent Vanbever
Princeton University

ETH Zürich
March, 17 2014
“Human factors are responsible for 50% to 80% of network outages”
“Cost per network outage can be as high as 750,000$”
At 12:47 AM PDT on April 21st 2011, a network change was performed as part of our normal scaling activities…

During the change, one of the steps is to shift traffic off of one of the redundant routers…
At 12:47 AM PDT on April 21st 2011, a network change was performed as part of our normal scaling activities…

During the change, one of the steps is to shift traffic off of one of the redundant routers…

The traffic shift was executed incorrectly and the traffic was routed onto the lower capacity redundant network.

This change disconnected both the primary and secondary network simultaneously…
Amazon is currently experiencing a degradation. They are **working on it**. We are still waiting on them to get to our volumes. Sorry.

*reddit is down.*
Amazon is currently experiencing a degradation. They are working on it. We are still waiting on them to get to our volumes. Sorry.

reddit is down.

Service Unavailable

We encountered an error on your last request. Our service is new, and we are just working out the kinks. We apologize for the inconvenience.
Amazon is currently experiencing a degradation. They are **working on it**. We are still waiting on them to get to our volumes. Sorry.

**reddit is down.**

**friendfeed**

We're currently having an unexpected outage, and are working to get the site back up as soon as possible. Thanks for your patience.
Amazon is currently experiencing a degradation. They are working on it. We are still waiting on them to get to our volumes. Sorry.

reddit is down.

friendfeed

We're currently having technical difficulties. We encourage you to report any issues you encounter.

Quora

A continually improving collection of questions and answers

Foursquare

Sorry! We're having technical difficulties

Latest post from status.foursquare.com:

Thu Apr 21 2011
This morning's downtime and slowness

Hi all,

Our usually-amazing datacenter hosts, Amazon EC2, are having a few hiccups this morning, which affected us and a bunch of other services that use them. Everything looks to be getting back to normal now. We'll update this when we have the all clear. Thanks for your patience.
Amazon is currently experiencing a degradation. They are working on it. We are still waiting on them to get to our volumes. Sorry.

Reddit is down.

Friendfeed
Service unavailable
We encourage you to try our new site back up at friendfeed.com

Quora
A continually improving collection of questions and answers
We encourage you to try our new site back up at quora.com

Foursquare
Sorry! We're down.
Latest post from a disconnected location
Thu Apr 21 2011
This morning's down location update.
Hi all,
Our usually-amazing datacenter is currently affected and a bunch of sites are down, but we're working on getting things back to normal now. We'll update you here when we know things are working.

Owls need a break sometimes too
We'll be back in action shortly -- in the meantime go outside and flap your arms around, you may find that flying ain't very easy.
In the meantime, if you can't wait to send a Tweet, head over to Twitter web to share your 140 character musings.

フクロウはときどき休まないとけないのです。
疲れた中でそれでも長い時間はかからないと思います。その頃、ちょっと外に出掛けてみて、翅をふっと休め、そして空高く羽ばたくことは実際に結構難しいのです。なので考えてしまうのが結構苦しい。
ツイートするのが待たれないのは、直接Twitterを開き、あなたの感想を140字で投稿してみましょう。
During the change, one of the standard steps is to shift traffic off of one of the redundant routers in the primary EBS network to allow the upgrade to happen. The trigger for this event was a poorly executed network update.
Internet
Internet
Internet

routing system
Internet

↓

Border Gateway Protocol (BGP)
2 fundamental properties of a good routing system

- Scalability: tolerate growth
- Flexibility: routing policies
2 fundamental properties… not met by BGP

- **scalability**
  - tolerate growth
  - keep track of too much state

- **flexibility**
  - routing policies
  - low-level management device-by-device
The Internet is a network of networks, referred to as Autonomous Systems (AS)
BGP is the routing protocol “glueing” the Internet together
ASes exchange information about the IP prefixes they can reach
ASes exchange information about the IP prefixes they can reach
Reachability information is propagated hop-by-hop
Reachability information is propagated hop-by-hop
Life of a BGP router is made of three consecutive steps

while true:

- receives routes from my neighbors
- select one best route for each prefix
- export the best route to my neighbors
Each AS can apply local routing policies

Each AS is free to

- select and use any path
  preferably, the cheapest one
always prefer Deutsche Telekom routes over AT&T
always prefer Deutsche Telekom routes over AT&T

IP traffic
Each AS can apply local routing policies

Each AS is free to

- select and use any path
  preferably, the cheapest one

- decide which path to export (if any) to which neighbor
  preferably, none to minimize carried traffic
do not export ETH routes to AT&T
do not export ETH routes to AT&T
2 fundamental properties of a good routing system

- scalability
  - tolerate growth
- flexibility
  - routing policies
  - keep track of too much state
Scalable routing systems maintain

- detailed information about nearby destination
- coarse-grained information about far-away destination
BGP maintains detailed information about every destination (i.e., network)
The problem is that the number of devices connected to the Internet increases rapidly.

mobile  Internet of things  sensors  virtual machines
BGP routers must also maintain routes for IPv6 networks in addition of IPv4 networks.

IPv6 ramping up could easily double the size of the Internet routing table.
The growth of the number of destinations has serious consequences for the Internet

- memory
- routing and forwarding table size
- time
- convergence time after a failure
- boot time for a router, session, ...
- security
- cost of signing & verifying BGP route
DRAGON: Distributed Route AGgregatiON

Joint work with: João Luís Sobrinoh, Franck Le and Jennifer Rexford

1  Background
   Route aggregation 101

2  Distributed filtering
   preserving consistency

3  Performance
   up to 80% of filtering efficiency
DRAGON: Distributed Route AGgregatiON

1 Background
Route aggregation 101

Distributed filtering
preserving consistency

Performance
up to 80% of filtering efficiency
How do you maintain less routing and/or forwarding information?
You make use of the IP prefix hierarchy to remove redundant information

Routing Table

<table>
<thead>
<tr>
<th>IP prefix</th>
<th>Output Interface</th>
</tr>
</thead>
<tbody>
<tr>
<td>...</td>
<td></td>
</tr>
<tr>
<td>129.0.0.0/8</td>
<td>IF#2</td>
</tr>
<tr>
<td>129.132.1.0/24</td>
<td>IF#2</td>
</tr>
<tr>
<td>129.132.2.0/24</td>
<td>IF#2</td>
</tr>
<tr>
<td>129.133.0.0/16</td>
<td>IF#3</td>
</tr>
<tr>
<td>...</td>
<td></td>
</tr>
</tbody>
</table>
An IP prefix identifies a set of IP addresses

Routing Table

<table>
<thead>
<tr>
<th>IP prefix</th>
<th>Output Interface</th>
</tr>
</thead>
<tbody>
<tr>
<td>129.0.0.0/8</td>
<td>IF#2</td>
</tr>
<tr>
<td>129.132.1.0/24</td>
<td>IF#2</td>
</tr>
<tr>
<td>129.132.2.0/24</td>
<td>IF#2</td>
</tr>
<tr>
<td>129.133.0.0/16</td>
<td>IF#3</td>
</tr>
<tr>
<td>...</td>
<td></td>
</tr>
</tbody>
</table>

129.0.0.0/8

$2^{(32-8)}$ IP addresses

prefix length
An IP prefix identifies a set of IP addresses which can be included into another one.

Routing Table

<table>
<thead>
<tr>
<th>IP prefix</th>
<th>Output Interface</th>
</tr>
</thead>
<tbody>
<tr>
<td>...</td>
<td></td>
</tr>
<tr>
<td>129.0.0.0/8</td>
<td>IF#2</td>
</tr>
<tr>
<td>129.132.1.0/24</td>
<td>IF#2</td>
</tr>
<tr>
<td>129.132.2.0/24</td>
<td>IF#2</td>
</tr>
<tr>
<td>129.133.0.0/16</td>
<td>IF#3</td>
</tr>
<tr>
<td>129.133.0.0/16</td>
<td></td>
</tr>
</tbody>
</table>

![Diagram of network routing]
Forwarding is done along the most specific prefix, i.e., the smallest set containing the IP address

Routing Table

<table>
<thead>
<tr>
<th>IP prefix</th>
<th>Output Interface</th>
</tr>
</thead>
<tbody>
<tr>
<td>...</td>
<td></td>
</tr>
<tr>
<td>129.0.0.0/8</td>
<td>IF#2</td>
</tr>
<tr>
<td><strong>129.132.1.0/24</strong></td>
<td>IF#2</td>
</tr>
<tr>
<td>129.132.2.0/24</td>
<td>IF#2</td>
</tr>
<tr>
<td>129.133.0.0/16</td>
<td>IF#3</td>
</tr>
<tr>
<td>...</td>
<td></td>
</tr>
</tbody>
</table>

Input packet: 129.132.1.1
A child prefix can be filtered whenever it shares the same output interface as its parent.

Routing Table

<table>
<thead>
<tr>
<th>IP prefix</th>
<th>Output Interface</th>
</tr>
</thead>
<tbody>
<tr>
<td>...</td>
<td></td>
</tr>
<tr>
<td>129.0.0.0/8</td>
<td>IF#2</td>
</tr>
<tr>
<td>129.132.1.0/24</td>
<td>IF#2</td>
</tr>
<tr>
<td>129.132.2.0/24</td>
<td>IF#2</td>
</tr>
<tr>
<td>129.133.0.0/16</td>
<td>IF#3</td>
</tr>
<tr>
<td>...</td>
<td></td>
</tr>
</tbody>
</table>

In the diagram, the parent prefix is 129.0.0.0/8, and the child prefixes are 129.132.1.0/24, 129.132.2.0/24, and 129.133.0.0/16. Each child prefix shares the same output interface as its parent.
A child prefix can be filtered whenever it shares the same output interface as its parent.

<table>
<thead>
<tr>
<th>IP prefix</th>
<th>Output Interface</th>
</tr>
</thead>
<tbody>
<tr>
<td>...</td>
<td></td>
</tr>
<tr>
<td>129.0.0.0/8</td>
<td>IF#2</td>
</tr>
<tr>
<td>129.132.1.0/24</td>
<td>IF#2</td>
</tr>
<tr>
<td>129.132.2.0/24</td>
<td>IF#2</td>
</tr>
<tr>
<td>129.133.0.0/16</td>
<td>IF#3</td>
</tr>
<tr>
<td>...</td>
<td></td>
</tr>
</tbody>
</table>

Routing Table

```
129.0.0.0/8
129.132.1.0/24
129.132.2.0/24
129.133.0.0/16
```

- **Parent**: 129.0.0.0/8
- **Child**: 129.132.1.0/24, 129.132.2.0/24, 129.133.0.0/16
A child prefix can be filtered whenever it shares the same output interface as its parent

Exactly the same forwarding as before
A child prefix can be filtered whenever it shares the same output interface as its parent.

Routing Table

<table>
<thead>
<tr>
<th>IP prefix</th>
<th>Output Interface</th>
</tr>
</thead>
<tbody>
<tr>
<td>...</td>
<td></td>
</tr>
<tr>
<td>129.0.0.0/8</td>
<td>IF#2</td>
</tr>
<tr>
<td>129.133.0.0/16</td>
<td>IF#3</td>
</tr>
<tr>
<td>...</td>
<td></td>
</tr>
</tbody>
</table>

Input packet: 129.132.1.1

Exactly the same forwarding as before
 Numerous previous works have studied this problem

<table>
<thead>
<tr>
<th>Year</th>
<th>Authors and Conferences</th>
</tr>
</thead>
<tbody>
<tr>
<td>2013</td>
<td>(Rétvári, SIGCOMM); (Rottenstreicht, INFOCOM)</td>
</tr>
<tr>
<td>2012</td>
<td>(Karpilovsky, IEEE TNSM)</td>
</tr>
<tr>
<td>2011</td>
<td>(Li, INFOCOM); (Uzmi, CoNEXT)</td>
</tr>
<tr>
<td>2010</td>
<td>(Zhao, INFOCOM); (Liu, GLOBECOM)</td>
</tr>
<tr>
<td>2009</td>
<td>(Ballani, NDSI)</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>1999</td>
<td>(Draves, INFOCOM)</td>
</tr>
</tbody>
</table>
The problem is that they only provide local gain

- local gain
- router or network

(Rétvári, SIGCOMM); (Rottenstreich, INFOCOM)
(Karpilovsky, IEEE TNSM)
(Li, INFOCOM); (Uzmi, CoNEXT)
(Zhao, INFOCOM); (Liu, GLOBECOM)
(Ballani, NDSI)
(...)
(Draves, INFOCOM)
Others proposed clean-slate approach to improve scalability, but none of them is incrementally deployable.
DRAGON provides both Internet-wide gain and incremental deployability

<table>
<thead>
<tr>
<th>Existing</th>
<th>DRAGON</th>
</tr>
</thead>
<tbody>
<tr>
<td>Local gain</td>
<td>Global gain</td>
</tr>
<tr>
<td>Router or network</td>
<td>Internet-wide</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Clean-slate</th>
<th>Works with BGP</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hard to deploy</td>
<td>Incrementally deployable</td>
</tr>
</tbody>
</table>
DRAGON: Distributed Route AGgregatiON

Background
Route aggregation 101

2 Distributed filtering
preserving consistency

Performance
up to 80% of filtering efficiency
DRAGON is a distributed route-aggregation technique where routers “think globally, but act locally.”

**Main result**

By comparing routes for different prefixes, a router can locally compute which routes it can filter and not export while preserving routing & forwarding decisions globally.
DRAGON is distributed route-aggregation technique where routers “think globally, but act locally”

Main result

By comparing routes for different prefixes, a router can locally compute which routes it can filter and not export while preserving routing & forwarding decisions globally.
When a router filters q, it does not create any forwarding entry for q and does not export q to any neighbor.
DRAGON is a distributed route-aggregation technique where routers “think globally, but act locally.”

**Main result**

By comparing routes for different prefixes, a router can locally compute which routes it can filter and not export while preserving routing & forwarding decisions globally.
DRAGON filters routing information, preserving the flow of data traffic
DRAGON guarantees network-wide routing and/or forwarding consistency post-filtering.

Routing consistency

Forwarding consistency

preserved property at every node for each data packet

route attribute

forwarding neighbors
DRAGON guarantees network-wide routing and/or forwarding consistency post-filtering.

This talk

preserved property at every node for each data packet

Routing consistency

route attribute

Forwarding consistency

forwarding neighbors
Let’s consider a mini–Internet using simplified routing policies
Solid lines join a provider and a customer, with the provider drawn above the customer.
advertises $p$ (parent)

advertises $q$ (child)
advertises p  (10.0.0.0/16)

advertises q  (10.0.0.0/24)
2 route attributes

- learned from customer
- learned from provider

2 exportation rules

- customer routes to every neighbor
- provider routes to customers
Current routing state for \( q \)

2 route attributes

- learned from customer
- learned from provider

2 exportation rules

- customer routes to every neighbor
- provider routes to customers
Current routing state for q

2 route attributes
- learned from customer
- learned from provider

2 exportation rules
- customer routes to every neighbor
- provider routes to customers
Current routing state for $q$

2 route attributes
- learned from customer
- learned from provider

2 exportation rules
- customer routes to every neighbor
- provider routes to customers
Final routing state for \( q \)

2 route attributes
- learned from customer
- learned from provider

2 exportation rules
- customer routes to every neighbor
- provider routes to customers
2 route attributes

- learned from customer
- learned from provider

2 exportation rules
- customer routes to every neighbor
- provider routes to customers
Current routing state for p

2 route attributes
- learned from customer
- learned from provider

2 exportation rules
- customer routes to every neighbor
- provider routes to customers
2 route attributes

- learned from customer
- learned from provider

2 exportation rules

- customer routes to every neighbor
- provider routes to customers
2 route attributes

- learned from customer
- learned from provider

2 exportation rules

- customer routes to every neighbor
- provider routes to customers
2 route attributes
- learned from customer
- learned from provider

2 exportation rules
- customer routes to every neighbor
- provider routes to customers

Current routing state for p
Current routing state for p

2 route attributes
- learned from customer
- learned from provider

2 exportation rules
- customer routes to every neighbor
- provider routes to customers
2 route attributes
- learned from customer
- learned from provider

2 exportation rules
- customer routes to every neighbor
- provider routes to customers
These three node elect different attribute for both q and p. They cannot filter.
These node elect the same attribute for $q$ and $p$. They are of type PR.
What if PR nodes filter?
Combined routing state

Legend

- Blue: customer
- Red: provider
u₄ filters q and stops propagating it to u₃
$u_4$ filters $q$ and stops propagating it to $u_3$
$u_3$ looses its only customer route to $q$
u₃ starts using a provider route for q

Legend

- customer
- provider
But what if $u_3$ filters?
if $u_3$ filters, it uses a customer route again for forwarding $q$ ...

... and it saves space!
All PR nodes filtering is a Nash Equilibrium

Any node has two incentives to filter q-routes:

- retrieve a better route to forward traffic
- gain space in its routing and forwarding tables

with no node having an unilateral incentive to move away
Simple route consistent algorithm

Considering a node $u$, a child prefix $q$, its parent prefix $p$, 
Simple route consistent algorithm

Considering a node $u$,
a child prefix $q$,
its parent prefix $p$,

Algorithm

If $u$ is not the destination for $q$ and
If elected $q$-route $\geq$ elected $p$-route
then $u$ filters $q$-routes
The algorithm is provably correct

Theorem 3: No matter the order in which node runs the algorithm, a route consistent state is eventually reached.
The algorithm is provably correct

**Theorem 1**
For every node $u$, the elected $q$-route can only worsen when an arbitrary set of nodes filter $q$-routes.

**Theorem 3**
No matter the order in which node runs the algorithm, a route consistent state is eventually reached.
The algorithm is provably correct

**Theorem 1** For every node $u$, the elected $q$-route can only worsen when an arbitrary set of nodes filter $q$-routes

**Theorem 2** The elected $q$-route at a node $u$ for which the elected $q$-route $< \text{elected } p$-route is not affected if an arbitrary set of nodes filters

**Theorem 3** No matter the order in which node runs the algorithm, a route consistent state is eventually reached
The algorithm is provably correct

**Theorem 1**
For every node $u$, the elected $q$-route can only worsen when an arbitrary set of nodes filter $q$-routes.

**Theorem 2**
The elected $q$-route at a node $u$ for which the elected $q$-route $< \text{elected } p$-route is not affected if an arbitrary set of nodes filters.

**Theorem 3**
No matter the order in which node runs the algorithm, a route consistent state is eventually reached.
DRAGON relies on isotonicity, a property which characterizes the combined policies of two neighbors.

<table>
<thead>
<tr>
<th>Isotonicity</th>
<th>If an AS u prefers one route over another, a neighboring AS does not have the opposite preference</th>
</tr>
</thead>
<tbody>
<tr>
<td>Observation</td>
<td>required for optimality, not correctness verified in a lot of actual routing policies</td>
</tr>
</tbody>
</table>
DRAGON: Distributed Route AGgregation

Background
Route aggregation 101

Distributed filtering
preserving consistency

3 Performance
up to 80% of filtering efficiency
Cummulated % of ASes

Filtering efficiency (%)
In today’s Internet, optimal filtering is ~50% as half of the Internet prefixes are parentless.
~80% of the ASes reaches optimal filtering efficiency
DRAGON node can automatically introduce aggregation prefix to filter prefixes without parent

Node can autonomously announce aggregation prefixes based on local computation and preserving consistency

Routing system self-organizes itself in case of conflict when more than one node announce the same parent prefix

Number of aggregation prefixes introduced can be tuned e.g., maximum prefix length or minimum # covered children
Introducing <10% of parent prefixes boosts the optimal efficiency to 79%
Again, ~80% of the ASes reach optimal filtering efficiency.
DRAGON: Distributed Route AGgregation

Background
Route aggregation 101

Distributed filtering
preserving consistency

Performance
up to 80% of filtering efficiency
DRAGON is a distributed route-aggregation algorithm which automatically harnesses any aggregation potential

DRAGON works on today’s routers
only require a software update and offers incentives to do it

DRAGON preserves routing and forwarding decision
leveraging the isotonicity properties of Internet policies

DRAGON is more general than BGP
shortest-path, ad-hoc networks, etc.
2 fundamental properties of a good routing system

- **scalability**
  - tolerate growth

- **flexibility**
  - routing policies
  - low-level management
    - device-by-device
A network is a distributed system which requires each element to be configured properly.
Configuring a distributed system is error-prone & time consuming (especially if done manually!)
In contrast, SDN simplifies network management...
... by removing the intelligence from the routers
... by removing the intelligence from the routers
...and program forwarding entries, from logically-centralized controller
So far, SDN has mostly been applied within a network…
... but managing BGP between networks is notoriously difficult and inflexible
How do you deploy SDN in a network composed of 50,000 subnetworks?
How do you deploy SDN in a network composed of 50,000 subnetworks?

Well, you don’t …
Instead, you aim at finding locations where deploying SDN can have the most impact.
Instead, you aim at finding locations where deploying SDN can have the most impact

Deploy SDN in locations that

- connect a large number of networks
- carry a large amount of traffic
- are opened to innovation
Internet eXchange Points (IXP) meet all the criteria

Deploy SDN in locations that

- connect a large number of networks
- carry a large amount of traffic
- are opened to innovation

AMS-IX

- 650 networks
- 2.7 Tbps (peak)
- BGP Route Server
- Mobile peering
- Open peering...

https://www.ams-ix.net
A single deployment can have a large impact

Deploy SDN in locations that

- connect a large number of networks
- carry a large amount of traffic
- are opened to innovation

AMS-IX

- 650 networks
- 2.7 Tb/s (peak)
- BGP Route Server
- Mobile peering
- Open peering...

https://www.ams-ix.net
SDX = SDN + IXP

Joint work with: Arpit Gupta, Muhammad Shahbaz, Russ Clark, E. Katz-Bassett, Nick Feamster, Jennifer Rexford and Scott Shenker
SDX = SDN + IXP

Augment the IXP data-plane with SDN capabilities keeping default forwarding and routing behavior

Enable fine-grained inter domain policies bringing new features while simplifying operations
**SDX = SDN + IXP**

- **Augment** the IXP data-plane with SDN capabilities, keeping default forwarding and routing behavior.

- **Enable** fine-grained inter domain policies, bringing new features while simplifying operations.

  ... with **scalability** and **correctness** in mind, supporting the load of a large IXP and resolving conflicts.
In a SDX, each participant connects its edge router(s) to a shared SDN-enabled network.
Each participant writes policies independently in a high-level language and transmits them to the controller.

Participant #1’s policy:
- \text{match}(\text{dstip}=\text{Google}), \text{fwd}(1.1)
- \text{match}(\text{dstip}=\text{Yahoo}), \text{fwd}(1.2)

Participant #2’s policy:
- \text{match}(\text{dstip}=\text{ip1}), \text{fwd}(1)
- \text{match}(\text{dstip}=\text{ip2}), \text{fwd}(3)
- \text{match}(\text{dstip}=\text{ip3}), \text{fwd}(5)

Participant #n’s policy:
- \text{match}(\text{dstip}=\text{ipX}), \text{fwd}(n.1)
The SDX controller compiles policies to forwarding entries ensuring isolation, scalability and avoiding conflicts.
SDX enables a wide range of novel applications

- **security**
  - Prevent/block policy violation
  - Prevent participants communication
  - Upstream blocking of DoS attacks

- **forwarding optimization**
  - Middlebox traffic steering
  - Traffic offloading
  - Inbound Traffic Engineering
  - Fast convergence

- **peering**
  - Application-specific peering

- **remote-control**
  - Influence BGP path selection
  - Wide-area load balancing
SDX works today!

We have running code (*)
controller and BGP daemon

We have a first deployment
@Telx Internet Exchange in Atlanta

Many interested parties
including AMS-IX, LINX, Amazon, Facebook & Google

(*) https://github.com/agupta13/sdx-optimized
2 fundamental properties of a good routing system

scalability
tolerate growth

manageability
enable flexibility
This talk

DRAGON
distributed filtering

SDX
flexible policies
What’s next?
Part I: A SDX-mediated Internet
SDX currently consider a single deployment
What about interconnecting SDX platforms?
What about replacing BGP completely with a SDX-mediated Internet?
“Let’s take over the world”
Towards a SDX–mediated Internet

New endpoint peering paradigm
more flexible, tailored to the traffic exchanged

Simple, scalable & policy neutral Internet core
SDX–to–SDX only, just carry bits

In–synch with the current Internet ecosystem
content consumer vs content provider vs transit network
Many novel research questions!

<table>
<thead>
<tr>
<th>policy analysis?</th>
<th>New endpoint peering paradigm more flexible, tailored to the traffic exchanged</th>
</tr>
</thead>
<tbody>
<tr>
<td>routing mechanism?</td>
<td>Simple, scalable &amp; policy neutral Internet core SDX-to-SDX only, just carry bits</td>
</tr>
<tr>
<td>new provider type?</td>
<td>In-synch with the current Internet ecosystem content consumer vs content provider vs transit network</td>
</tr>
</tbody>
</table>
Internet ❤️ SDN

Part II: Rethinking inter-domain routing
SDN controllers sitting in different domains will have to exchange reachability information...
What protocol shall they use?
hint: not BGP!
Instead of just exchanging destination, what about transmitting abstract annotated graphs?

$\text{capacity}=1\text{Mbps}, \text{ delay}=1\text{ms}$

$\text{capacity}=100\text{Mbps}, \text{ delay}=100\text{ms}$
Annotated graphs reveal more information about paths while still letting each AS implements local policies

Announcing network can hide information using abstraction e.g., hide internal topology, more costly exit points…

Receiving network composes the graph with its own topology then use its own objective function to compute path

BGP is just a special case in which each graph is a “node” support partial deployment in the Internet
Many novel research questions!

**abstraction operator?**
Announcing network can hide information using abstraction e.g., hide internal topology, more costly exit points…

**composition mechanism?**
Receiving network composes the graph with its own topology then use its own objective function to compute path

**correctness?**

**data-plane realization?**
BGP is just a special case in which each graph is a “node” support partial deployment in the Internet
Making the Internet more scalable and manageable

Laurent Vanbever
www.vanbever.eu

ETH Zürich
March, 17 2014