Active BGP Probing

Lorenzo Colitti

Roma Tre University – RIPE NCC

Agenda

- Our techniques
  - Primitives
  - Applications
  - Results
- Operational impact
  - Why it is safe
  - Why it is low-impact
  - Why it doesn't hamper debugging
- Tests over IPv4?

Our Techniques

The Problem

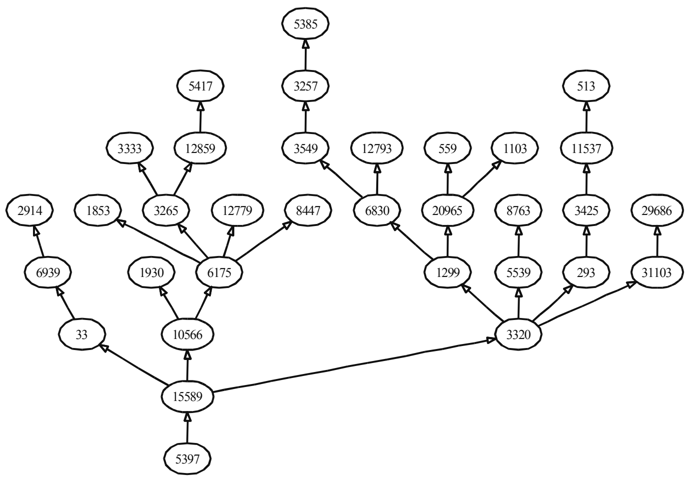
- Point of view: an ISP
  - We want to know how other ASes treat our prefixes
  - Why?
    - Predict the effect of network faults
    - Perform effective traffic engineering
    - Develop peering strategies
    - Evaluate quality of upstreams
    - ...
- Existing BGP discovery methods are good at discovering topology but bad at discovering policy
  - We can look at RIS or ORV...
  - ... but we can't find out how the world treats **our** prefixes

Can we do better?

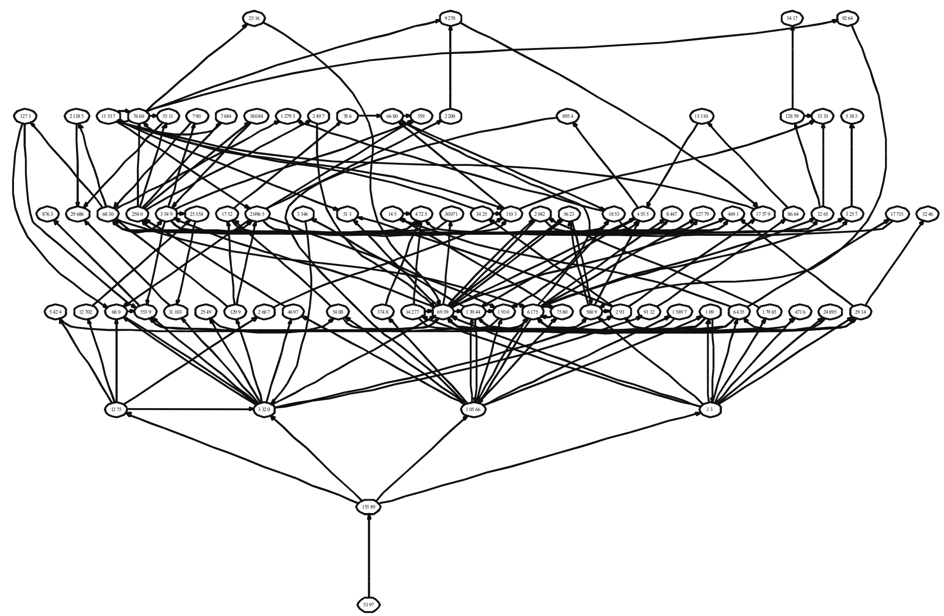
- We would like to know:
  - Where our announcements go
    - Trivial: just look at RIS or ORV
  - Where our announcements **could** go: “feasibility”
    - What happens if a link fails and backups come up?
    - What are the margins for traffic engineering?
  - How other ASes treat our prefixes
    - Do other ASes have preferences about how to reach us?
- How can we obtain this information?

Just to get an idea

Standard RIS query



Using our techniques



Feasibility

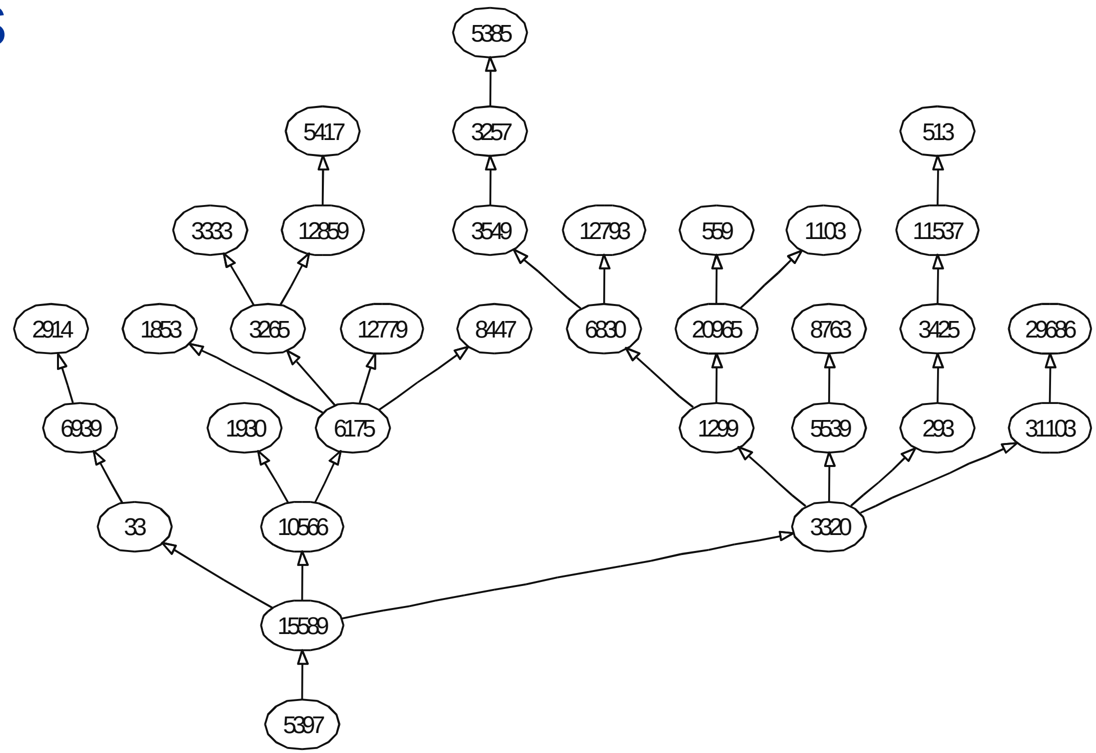
- "Where can our announcements go?"

- An AS-path is **feasible** for a prefix *p* if "the policies of the ASes in the Internet allow it to be announced"
  - Active ("best") paths, backup paths, alternate paths

- A BGP peering is feasible for *p* if it's part of a feasible AS-path
  - That is, if it is possible, in some state of the Internet, for the announcements for *p* to traverse it

Feasibility graph

- Directed graph: nodes = ASes, arcs = feasible peerings



- Shows us only [a subset of] the portion of the Internet where our announcements can go

Active BGP probing

- Basic idea: inject updates into the network and observe results
  - Use a test prefix *p* to avoid disrupting production traffic
  - Use RIS or ORV to see (and react to) results in real-time
  - Use looking glasses and route servers to see steady state results

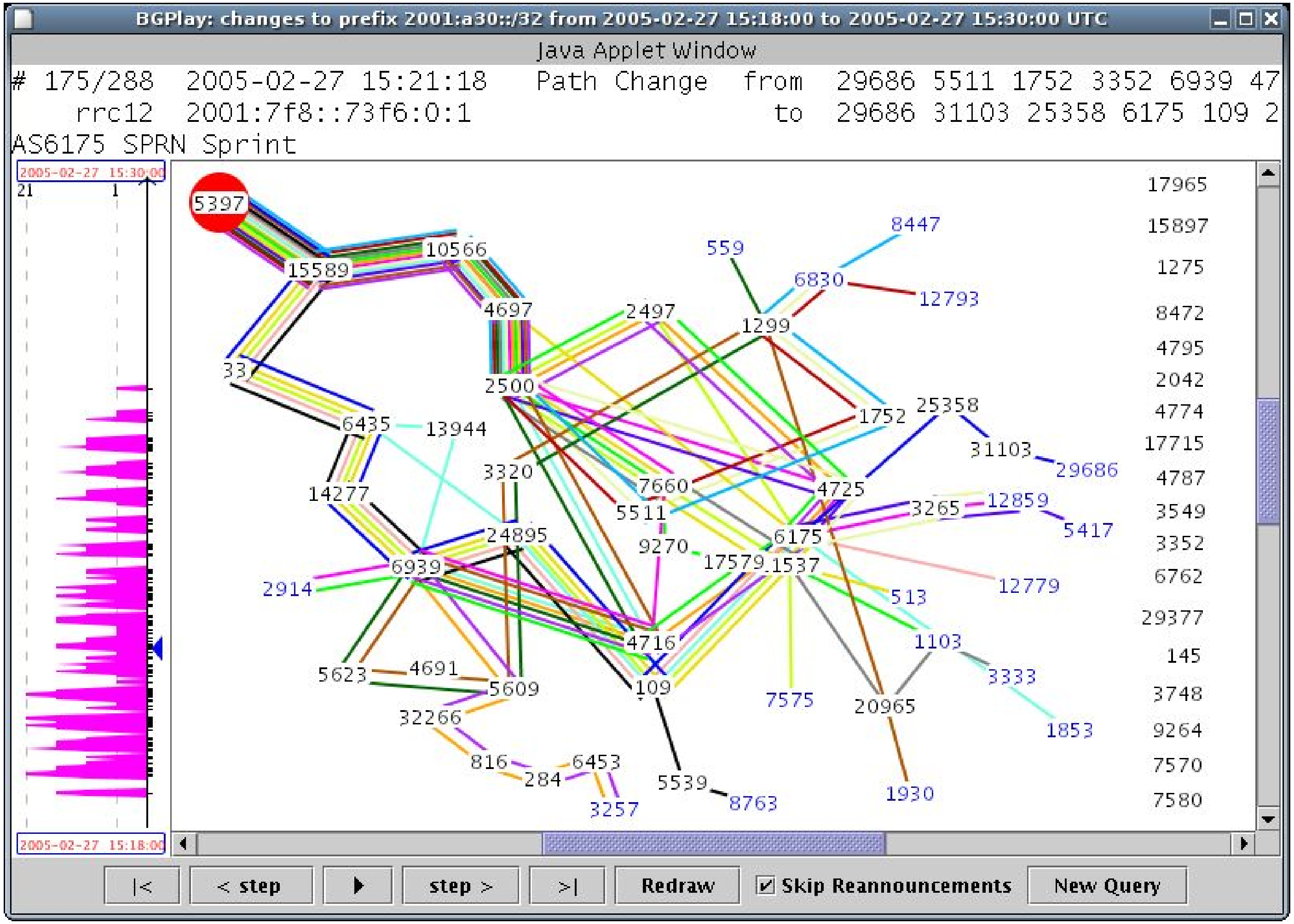
- Two primitives:
  - Withdrawal Observation
    - Let BGP explore alternate paths
  - AS-set Stuffing
    - Force BGP to take alternate paths by “prohibiting” certain ASes

Withdrawal Observation

- BGP explores many alternate paths before realizing a route has been withdrawn
  - An AS sends a withdrawal only if all its alternate paths have been withdrawn
  - Else it sends out an update for one of the alternate paths
- We can use this to discover alternate paths
  - Withdraw the test prefix *p*
  - Record BGP paths seen during convergence process
  - Merge paths to get a feasibility graph
- BGP does a lot of the work for us

Lorenzo Colitti . RIPE 50, 5 May 2005 . colitti@dia.uniroma3.it – lorenzo@ripe.net

Withdrawal observation: BGPlay



http://www.ris.ripe.net/cgi-bin/bgplay.cgi?prefix=84.205.89.0/24&start=2005-03-01+00:00&end=2005-03-01+00:10

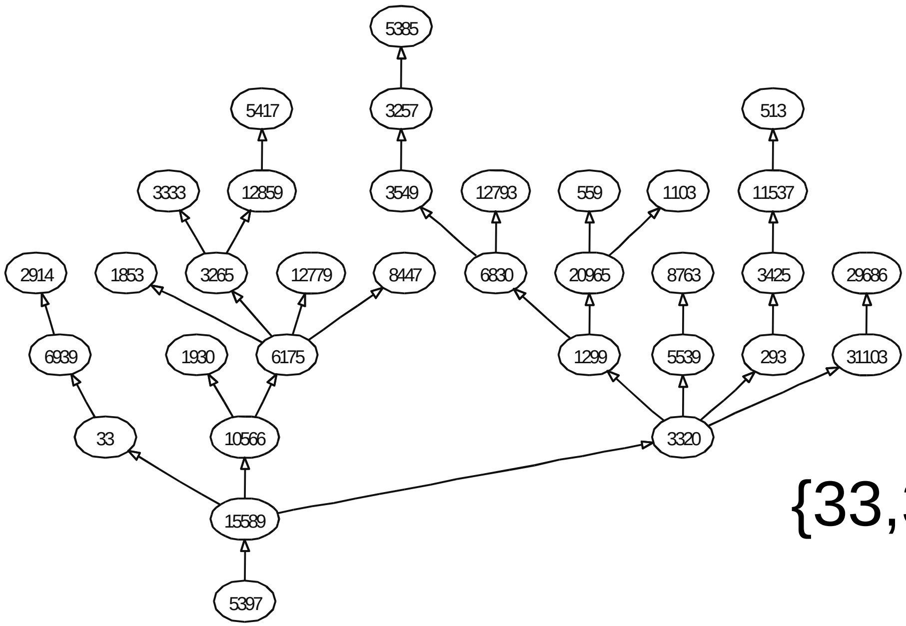
AS-set Stuffing

- Prepend an AS-set containing arbitrary ASes A_i
  - The AS-paths seen by the Internet end in Z $\{A_1, A_2, \ldots, A_i\}$ where Z is our AS number
- We say the ASes A_i are “prohibited”
  - They will not receive or process the announcements
  - They disappear from the Internet as far as p is concerned
- What this allows:
  - Topology discovery
  - Path feasibility and policy discovery
  - Measurements in “altered network state”

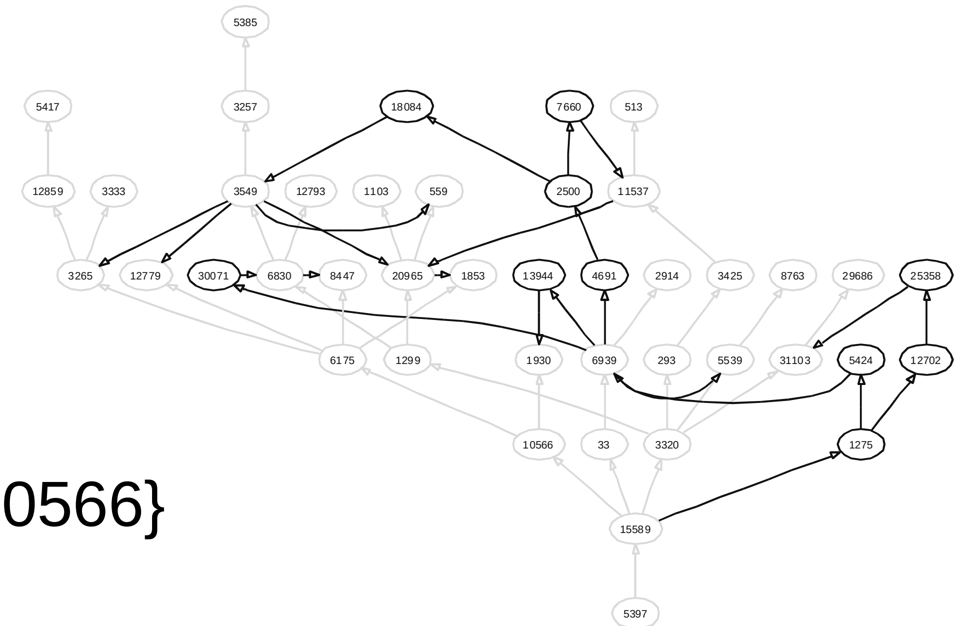
Topology discovery

- Announcing an AS-set containing ASes in active paths causes alternate paths to appear
  - So we find new ASes and peerings
- Simple algorithm to find out out a larger topology: “Level-by-level” exploration:
- Proceed by increasing topological distance:
  - Prohibit all ASes at certain distance
    - Observe paths seen during convergence and after convergence
    - Add all ASes and peerings found to feasibility graph
  - If new ASes appear at this distance, turn them off too
  - When no new ASes appear, increase distance by one

Example: prohibit level 2



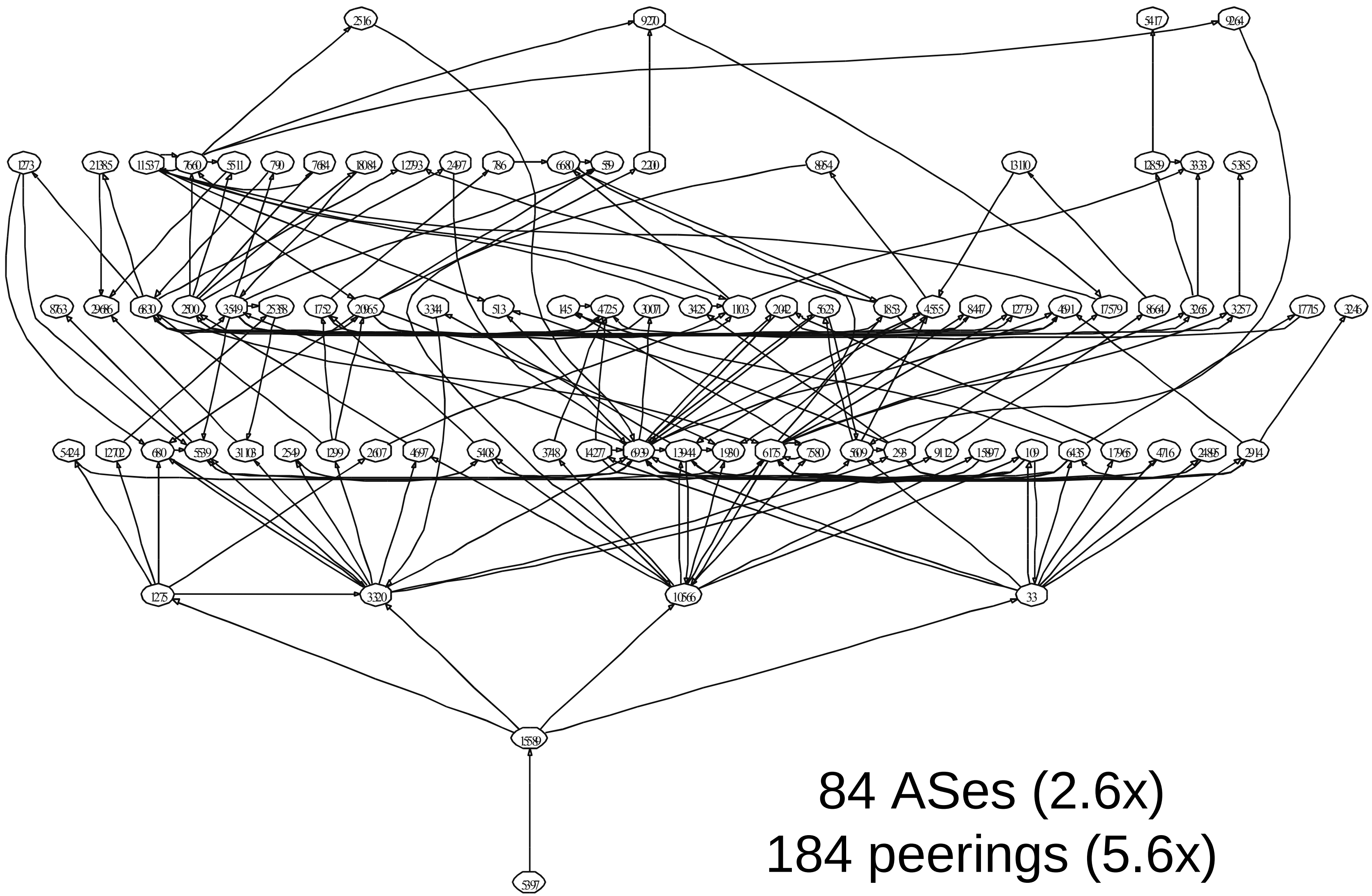
{33,3320,10566}



32 ASes
33 peerings

42 ASes
57 peerings

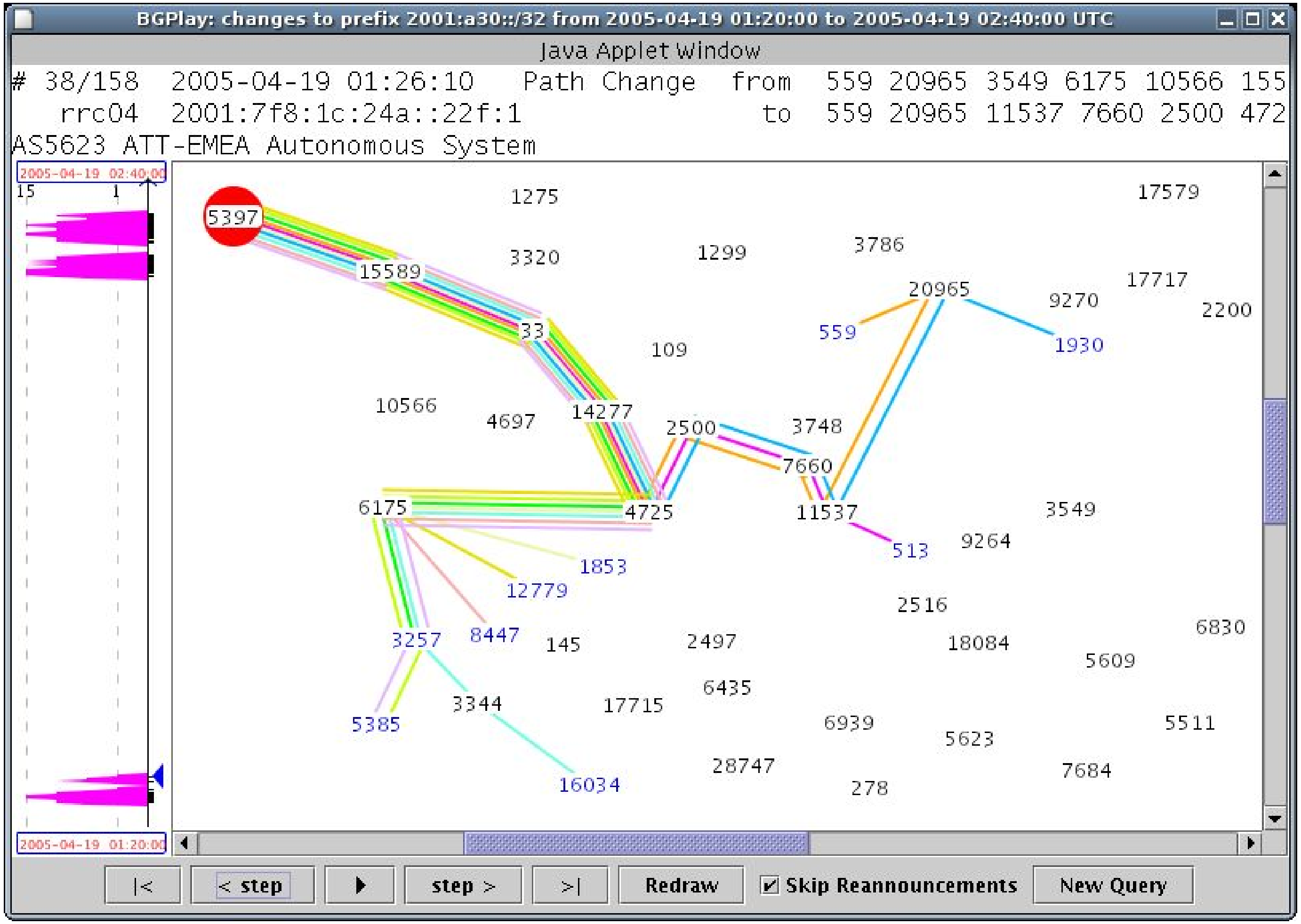
After 4 levels...



84 ASes (2.6x)

184 peerings (5.6x)

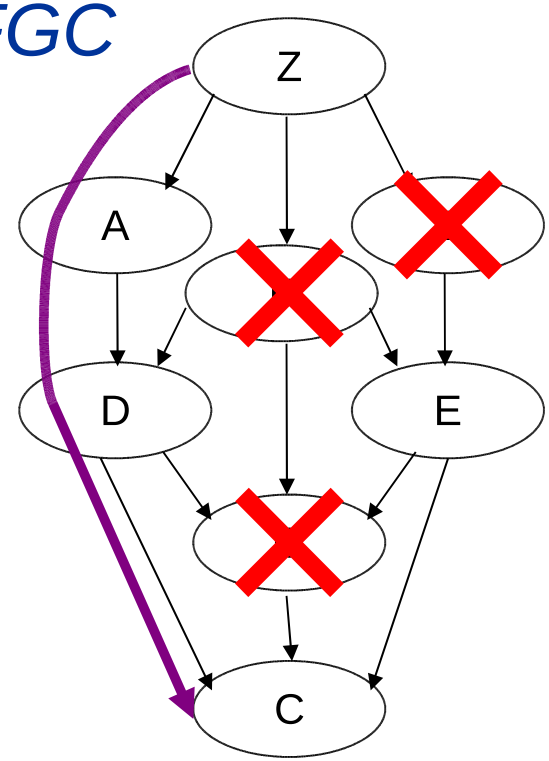
Level-by-level exploration: BGPlay



http://www.ris.ripe.net/cgi-bin/bgplay.cgi?prefix=2001:a30::/32&start=2005-04-19+01:20&end=2005-04-19+02:40

Path Feasibility determination

- Suppose the route collector C sees *ZFGC*
- Is the path *ZADC* feasible?



- Announce {*B*,*F*,*G*}

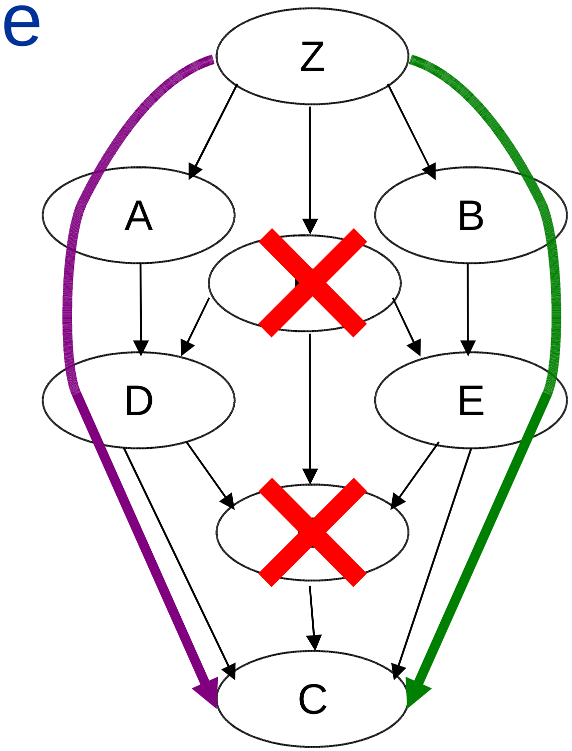
- If C sees *ZADC*, *ZADC* is feasible (obviously)
- If C does not see anything, *ZADC* is not feasible

Lorenzo Colitti

colitti@dia.uniroma3.it – lorenzo@ripe.net

Path Preference discovery

- Suppose *ZADC* and *ZBEC* are feasible
- Which does *C* prefer?



- Announce {*F*,*G*}

- The path *C* prefers is the one it chooses as best

“Altered state” measurements

- Use AS-set stuffing to put network into altered state
  - e.g. “turn off” one of our upstreams' upstreams
- Then measure network performance
  - Look at looking glasses in other ASes
  - Or use RTT measurements
    - Forward path stays the same!

Testing and Results

- We tested on the IPv6 backbone:
  - Fewer legacy devices
  - Fewer mission-critical services
  - Much smaller size
- Announcements were for 2001:a30::/32 and originated in AS5397
- For results, see our technical report:

http://www.dia.uniroma3.it/~compunet/bgp-probing/bgp-probing-tr.pdf

Operational Impact

This is safe

- Equipment tests
  - Juniper, old Cisco: reset session at 125 ASes
    - This is not specific to our techniques!
  - New Cisco: ignore path at 75 ASes
  - We never needed more than ~50
- IPv6 tests
  - 11/2004 – 2/2005 (reprise in April); no problems reported
  - AS-sets noticed only twice (first time after 3 months)
- Observation in the wild (IPv4)
  - Jan 2001: 123-element AS-set; Jan 2002, 124-element
  - Nobody complained of problems due to these events

This is low impact

- Dampening limits us to ~ 1 update per hour
  - A typical Tier-1 router might receive 15k updates per hour
- A 100-element AS-set should require about 200 bytes of memory
  - Core routers are already using tens of megabytes of memory for BGP

This doesn't hamper debugging

- People already prepend other people's AS numbers
- Our techniques are more transparent
  - Our AS is the first AS before the AS-set
  - Apart from the AS-set, the rest of the path is the path the announcement took
  - Such large AS-sets are obviously unlikely to result from route aggregation
- The routes can be tagged with communities
  - Thanks to Tim Griffin for suggesting this
- A whois on the prefix immediately reveals the origin

Ethical Issues

- We're using BGP for stuff it was not designed to do
  - This happens frequently!
    - e.g.: NAT, IP-in-IP tunneling, dupacks for congestion control, ...
- We're using people's AS numbers without their permission
  - People already do it, if not in such an obvious way
  - The announcements should not cause confusion
    - A whois query on the prefix immediately reveals the origin
    - The announcements are immediately recognizable
  - We believe the usefulness of our techniques for ISPs makes it worthwhile

Testing in the IPv4 backbone

Testing over IPv4

- We believe these techniques can be useful for ISPs
  - There are no good technical reasons not to do this
- We would like to discover how effective they are in the IPv4 Internet
  - We have tested in the lab
  - We have tested on the IPv6 backbone, with good results
    - See the technical report for details
  - We would like to test on the IPv4 backbone
    - Applying our techniques to the IPv4 Internet might also provide new insights on the structure of the network

Questions?