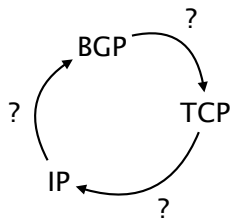
Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Prof. Laurent Vanbever
Networked Systems Group

Communication Networks

Solution: Additional Exam Preparation Questions

Circularity



Chicken or the egg?

Consider two directly-connected routers A and B that have an eBGP session (running over TCP) between them. Explain how BGP, which is used to propagate *IP-based* routing information, can rely on establishing a TCP connection to exchange routes, which itself relies on IP. How is this circularity resolved? How do the two TCP endpoints manage to reach each other?

Solution: As the two routers A and B are directly connected, they do not need a routing protocol to learn about how to reach each other. Both routers are configured with an IP address belonging to the same subnet (say, 10.0.0.1/30 and 10.0.0.2/30).

When a BGP session is configured on A (resp. B) for 10.0.0.2/30 (resp. 10.0.0.1/30), the router uses its direct link to reach the other end-point. The circularity problem highlighted in the question would only happen if the subnet connecting the two routers itself would have to be learned via BGP, which is not the case.

Curious students

Consider that ITET has a local DNS server serving the DNS requests for all students' devices connected in the department. How could you determine if an external website has been visited recently by a fellow colleague of yours? Explain.

Solution: You can simply use dig to issue a query for any external website you're interested in and observe the response time. If the external website has been visited recently, the response time should be close to immediate (as the local DNS server is located close by). If not, the response time will be much slower as the local DNS server would have to initiate a new remote query. Observe that looking at the TTL value returned by the local server would not give you a definite answer as you would not know what was the initial TTL value received by the local DNS server in the first place.

Name it or Route it: pick one

Any clear winner?

In the course, we saw two ways to replicate and load-balance content: *(i)* using Anycast routing; or *(ii)* using DNS.

a) List and briefly justify three pros and cons of each;

Solution: DNS

(i) Advantages
   i. Simplicity: DNS-based load-balancing can be implemented by any CDN.
   ii. (potentially) Fine-grained: Decisions can be particularized on a per-source IP basis.
   iii. Near real-time: Assuming small TTL values (modulo the problem of load, see below), DNS load-balancing enables frequent load adaption.

(ii) Disadvantages
   i. Infrastructure cost: Good load-balancing requires small TTL values which induce a high load at the DNS server level, which in turn requires to dimension the DNS infrastructure accordingly.
   ii. Location: The source IP seen by the server and on which the load-balancing decision is made is the one from the resolver (e.g. Swisscom's one), not the direct client, meaning the resolver and the client can actually be far away from each other (think of Google's open DNS resolver).
   iii. (potentially) Coarse-grained: The DNS resolver source IP can actually serve a huge amount of clients which will therefore share the same load-balancing decisions.

Anycast routing

(i) Advantages
   i. Simplicity: The required infrastructure is simple as the load-balancing is done by the network directly.
   ii. Reliability: Whenever a route fails, the network will simply converge to another replica (route).
   iii. Expandablility: It is easy to add an additional replica, as simply an additional location needs to start advertising the respective prefix.

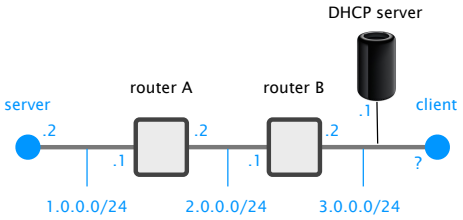
(ii) Disadvantages
   i. Broken Connections: As routing changes can happen at anytime, packets of the same TCP flow might arrive at different replicas effectively breaking the connection.
   ii. Content and performance: The load-balancing is not aware of the utilization of the replicas. Hence, it is not possible to offload work from heavily utilized replicas.
   iii. Location: Packets may not use the nearest replica due to routing policies.

b) We saw that CDNs often rely on DNS for distributing their load. Could they also use Anycast routing instead? Explain why or why not.

Solution: Yes, you can use Anycast routing instead to distribute the load. However, the load-balancing is neither content nor performance aware. You have to make sure that the same content is available on all replicas and none of the replicas is fully utilized.

Putting everything together

Consider the network on the left composed of three Ethernet segments separated by two intermediate routers (A and B). In this network, the server's interface along with the routers' interfaces are configured with static IP addresses. While clients connected to the 3.0.0.0/24 Ethernet segment obtain an IP address via DHCP.



Describe everything that happens to the packets sent between the client and the server

Assuming that the client has just started, with a perfectly empty state, precisely describe all packets that are generated when the command "ping 1.0.0.2" is issued (until the server answers back). Among others, your answer *must* include the content of the Layer 2 and Layer 3 headers.

Solution:

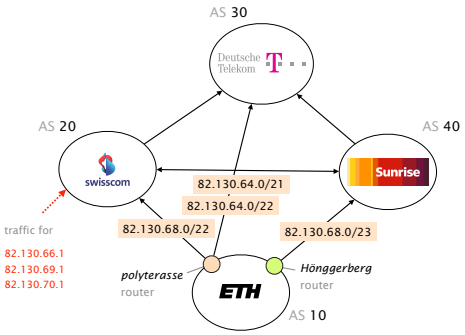
- src_mac: *client_mac*, dst_mac: *broadcast*
  src_ip: *0.0.0.0*, dst_ip: *255.255.255.255*
  payload: *DHCP discovery*
- src_mac: *dhcp_server_mac*, dst_mac: *broadcast*
  src_ip: *3.0.0.1*, dst_ip: *255.255.255.255*
  payload: *DHCP offer for 3.0.0.3*

  Note: In addition to the IP address of the client (arbitrarily picked by the DHCP server), the DHCP offer also contains the gateway (3.0.0.2) to use. The DHCP offer can be sent either in broadcast or unicast (both answers accepted). Broadcast is often used as some network stacks will not accept "unicasted" frames unless an IP address is configured first.
- src_mac: *client_mac*, dst_mac: *broadcast*
  payload: *ARP request: Who has 3.0.0.2? Tell 3.0.0.3.*
- src_mac: *routerB_right_mac*, dst_mac: *client_mac*
  payload: *ARP reply: 3.0.0.2 is at routerB_right_mac*
- src_mac: *client_mac*, dst_mac: *routerB_right_mac*
  src_ip: *3.0.0.3*, dst_ip: *1.0.0.2*
  payload: *ICMP echo request*

... (rest of the packets are omitted)

- src_mac: *server_mac*, dst_mac: *routerA_left_mac*
  src_ip: *1.0.0.2*, dst_ip: *3.0.0.3*
  payload: *ICMP echo reply*

Traffic (not so much) Engineered



Where are my packets going?

After passing the Communication Networks exam with flying colors, ETH hires you as a junior network engineer. *Congrats!*

Your first mission is to analyze their BGP configuration. They indeed suspect that something might be wrong, especially since they installed this box from Sisco Systems that automatically configure BGP announcements according to Traffic Engineering objectives. For the sake of simplicity, assume again that ETH has only one prefix: 82.130.64.0/21 and three providers: Swisscom, Deutsche Telekom and Sunrise. The actual announcements are depicted on the left. Customers are drawn below their providers (Swisscom is a customer of Deutsche Telekom), while peers are drawn next to each other (Swisccom is a peer of Sunrise).

Consider the incoming traffic from Swisscom. What path is taken for packets destined to:

a) 82.130.66.1? **Solution:** [20, 30, 10]

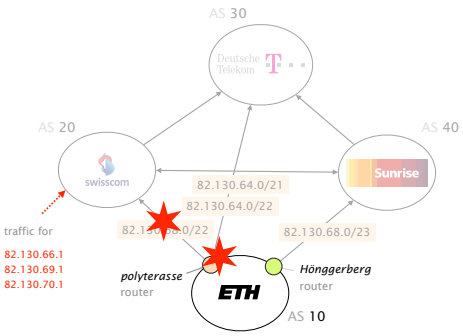
b) 82.130.69.1? **Solution:** [20, 40, 10]

c) 82.130.70.1? **Solution:** [20, 10]

Are Swisscom, Deutsche Telekom and Sunrise happy about these announcements? Can they do anything about that? Explain briefly.

Solution: Swisscom and Sunrise are not happy. Should ETH advertise them its full prefix, they could direct all traffic to it directly and earn money instead of paying money to Deutsche Telekom or use their peering (revenue neutral) link. Deutsche Telekom is happy though as it receives extra traffic (and therefore, revenues) from Swisscom and Sunrise for traffic pertaining to the sub-prefixes.

As routing and forwarding in the Internet is done according to the longest destination prefix, there is nothing that Swisscom or Sunrise can do besides convincing ETH to change its announcements.



Is this network as redundant as it looks?

As the *polyterasse* router is getting older, its reliability starts to suffer. In theory, this should not be a big problem as ETH is *triple*-homed! Yet, in practice, the ETH engineers observe regular network connectivity upon failures.

Assuming the same announcements, what path ends up being taken by the packets destined to the above three IP addresses when:

a) the link between *polyterasse* and Swisscom goes down?
   (i) 82.130.66.1? **Solution:** [20, 30, 10]
   (ii) 82.130.69.1? **Solution:** [20, 40, 10]
   (iii) 82.130.70.1? **Solution:** [20, 30, 10]

b) the *polyterasse* router dies?
   (i) 82.130.66.1? **Solution:** *drop*
   (ii) 82.130.69.1? **Solution:** [20, 40, 10]
   (iii) 82.130.70.1? **Solution:** *drop*

What would you change in the ETH announcements to improve reliability, without disturbing the inbound Traffic Engineering performed by the Sisco box in the steady case (without failures)? Explain briefly.

Solution: A simple solution would be to advertise the full prefix 82.130.64.0/21 to all providers *along* with the more-specific ones. Since traffic is based on the longest-prefix match, announcing additional less-specific prefix will not change the forwarding in the steady state but will enable any of its three providers to continue providing full transit in case of failures.