BRTP: Border Routing & Transport Protocol

Debish Fesehaye & Klara Nahrstedt
University of Illinois-Urbana Champaign

Routing & Transport Protocols

- Any network *communication* system involves finding a path (*routing*) and controlling the rate (*transport*) of communication.
- Existing routing techniques:
  - Lack an efficient *routing metric* which reflects the up-to-date status of the network.
- Existing transport protocols (TCP)
  - Lack the good and up-to-date knowledge of the *rate* of communication

BRTP Design

- Main Ideas
  - Cleverly *aggregated* information from the *border* routers (ingress and egress) gives enough information about the up-to-date *load* condition of the *core* routers.
  - A BRTP *server* system which has the aggregated information can then compute the best ingress-egress (*in-eg*) *path* and *rate*
  - The *rate* of the *flows* in one *in-eg* path is then obtained by sharing the "capacity" (rate) of the *in-eg* path obtained by the BRTP fairly or proportionally
  - The final rate at which a *flow sends* data is then the *minimum* of the rates (bottleneck rate) it obtains from all *in-eg* pairs in its path to the destination
  - This can be done by adding a *shim layer* to the TCP header of packets which can be overwritten by ingress routers in the path.

BRTP Algorithm

- Each *ingress* router *in* the network (AS) sends to the BRTP server the *total* number $L^{in\text{-}eg}$ of *packets* it transfered to the *egress* router *eg* every control interval d^{in}.
  - If *in* uses more than one path to *eg*, then *eg* can be thought of as a different egress nearby *eg* (duplicate virtual *eg*) and hence another path.
- The BRTP server aggregates these values and *calculates* the rate R^i and per packet price p^i on the behalf of each of its AS router *i* as follows.
  - For each *ingress* router
    - For each *path* of the ingress router to the egress routers
      - For each *link* crossed by the path
        - Obtain A^i for the R^i calculation as the *sum* of all *actual flow* counts $A_k^{\ i}$ of each path of the ingress router *k*
        - A *simple count* of flows in each ingress path can also be used here for simplicity
- For each ingress router path in the network
  - Obtain R^i on the behalf of router *i* in the AS (network)
  - Obtain p^i on the behalf of router i in the AS (network)

BRTP Algorithm … cont'd

- The PAS calculates the *local path*, path *rate* (capacity) C_p^i and cost of each path P_p^i for *each* of its ingress router *i* as
  - The path with the *maximum* of the *minimum* R^i of each path
  - The cost is the *sum* of the p^i of each router in the selected path
- This scheme results in each *in-eg* path as a *virtual link* with capacity C_p^i
- Then the *rate* of *each flow* passing through path *p* of the ingress router can be obtained the same way the R^i of each link *i* is obtained (shown next) or
- A more sophisticated approach which *doesn't* need flow *classification* at the ingress router can be used to obtain the *rate* share of each *flow* passing through the path of the ingress router as shown next.

BRTP Algorithm .... cont'd

- Each BRTP *source j* can then send packets with a *shim layer* attached to the TCP header
  - With *throughput* value R_j set to infinity
  - With the *price* value P_j set to zero
- Each ingress router which receives the packet then *overwrites*
  - the *throughput* in the packet header with the *minimum* of the rate it calculated and the throughput value in the packet.
  - the *price* in the packet header with the *sum* of the price in the packet and the price it calculated
- The *destination* of the packet then copies the *throughput* value and *price* to an *ACK* packet.
- The source *j* then sets its cwnd $w_j = RTT_j \times R_j$ and *knows* the cost of the path flow *j* is crossing
- The source can then apply *policies* based on the price
  - Change a service provider (route) or *prioritize* its flows
- The above scheme can also be used *without* the *price* option just as a *congestion control* scheme.

Computation of fair rate

- *Notations*: C^i, Q^i, N^i, $L_p{}^i$ and d^i are the capacity, the queue length, number of flows, total number of packets of previous interval and control interval at router *i* in the AS.
- The *fair rate* at router *i* is then

$$R^i = \frac{C^i - Q^i / d^i}{N^i}$$

- The fair *packet count* (*cwnd*) at router *i* is $w^i = R^i d^i$ and $w_p{}^i$ is the w^i of the previous round (control interval d^i).
- But some flows *may not* have enough data to send to utilize their share of the bandwidth.
- This may result in link *under-utilization* while other legitimate flows which have more data to send could use the bandwidth.
- So count some flows as *less than one* flow as follows:

Computation of fair ... cont'd

- Then we have

$$n_j^i = \begin{cases} 1 & \text{if } w_j^i > w_p^i \\ \frac{w_j^i}{w_p^i} & \text{otherwise} \end{cases}$$

$$A^i = \sum_{j=1}^{N^i} n_j^i$$

Where n_i^j is a flow indicator and A_i is the *actual flow count*

- The rate is then $R^i = \frac{(C^i - \frac{Q^i}{d^i})}{A^i}$ and $w^i = R^i \cdot d^i$

where

$$Q^i = \begin{cases} 0 & \text{if } L_p^i \leq C^i d^i \\ L_p^i - C^i d^i & \text{otherwise} \end{cases}$$

Computation of packet price

- The *per packet price* p^i is a function of the fair rate R^i.
- If R^i *increases* there is *less demand* and hence cheaper price.
- To capture this we use current rate R^i and price p^i and previous round values $R_p^{\ i}$ and $p_p^{\ i}$ and calculate the current per packet price as

$$p^i = p_p^i \frac{R_p^i}{R^i}$$

- Other more sophisticated *pricing* functions can be used
- The total flow price of flow j at link i in a given round is $w_j p^i$.
- Different R^i values can also be obtained for each flow based on some *priorities* (weights).

Computation of fair rate without flow classification

- *Notations*: C_p^i, Q_p^i, L_p^i, R_j are capacity, queue length, the packet count of path *p* of ingress router *i* and the rate at which flow j is sending to this interface *p*.
- Then the *rate share* of a *flow* at path *p* of ingress router *i* is

$$R_p^i = \frac{C_p^i d^i - Q_p^i}{\sum_j^{L_p^i} \frac{1}{R_j}}$$

Some Basic Considerations

- Our Scheme assumes that each ingress router notifies (the BRTP server) that its link is fully *functional*.
  - Using the usual information routers share to tell that they are alive or
- If each AS router *i* can send the *total packet count* per round (control interval) to the BRTP server then
  - the BRTP server not only knows that router *i* is *alive*, but it also
  - checks the L_p^i it obtains on *behalf* of router *i* using the information it aggregates from the ingress routers against the *actual packet* count it gets *from router i* .

Summary

- We have presented *BRTP*, an efficient *cross layer routing* and *congestion* control protocol.
- BRTP can be implemented in the *current Internet* by leveraging capabilities of the ingress routers or by adding some router-like filter boxes near the ingress routers.
- BRTP also has a *pricing* scheme to help the AS know path costs to charge customers accordingly and to help it choose the less expensive path
- The BRTP server can use *cloud computing* to speed up computation
- The scheme BRTP uses can make *clean-slate* protocols easily deployable in the current Internet with out the need of making changes in the core routers.
- We are working on real *implementation* of BRTP in Linux.