Implementation and Preliminary Evaluation of an ISP-Driven Informed Path Selection

Damien Saucez, Benoit Donnet, Olivier Bonaventure
Universite catholique de Louvain Belgium*

1. MOTIVATIONS

During the past ten years, we have seen the emergence of a set of applications requiring more and more quality of service (QoS). For instance, IPTV needs large bandwidth and delays as lows as possible. Further, while previously a content was located in a single place, it is, nowadays, frequent that the content is replicated among a set of servers located anywhere on five continents or even among users themselves. Perfect examples of this are peer-to-peer (P2P) applications and FTP mirrors. In addition, *multihoming*, i.e., the ability of having different connections to Internet potentially through different providers, is becoming more and more popular [1, 2]. Finally, network level protocols such as SHIM6 or LISP must often choose the best path among a list of highly disparate paths according to traffic engineering or policies considerations (see discussions on IETF mailing lists).

This situation leads to more complexity in ensuring QoS to the user. We have now to maximize the QoS perceived by the user based on a set of highly disparate paths. In addition, in many new applications, path selection and QoS are ensured at the application layer resulting in an ineffective network level resources use.

Today, although many of measurements techniques have been developed within the IPPM working group of the IETF, an application that needs to select a path or a server must implement its own measurement system to obtain data to perform its selection. Thus, several applications running on the same host or in the same campus will probably perform almost the same kind of measurements. Duplicating those measurements is not the appropriate solution.

*This work was partially supported by the European-founded 034819 OneLab project and the European-founded 027609 Agave project.

CoNEXT'07, December 10-13, 2007, New York, NY, U.S.A.

A better solution to this problems would be to develop a service that runs continuously and could be queried by applications requiring QoS and path information. This service, named *IDIPS* (for ISP-Driven Informed Path Selection) [3], is described in Sec. 2. We further present an evaluation of our current implementation in Sec. 3. Finally, Sec. 4 concludes this extended abstract.

2. THE IDIPS SERVICE

The idea behind IDIPS is to provide ordered paths according to pre-defined criteria. IDIPS is composed of three entities: the *client*, the *destination* and the *IDIPS server* itself. The client is the entity sending request for QoS and path information towards a destination. The IDIPS server aims at providing the requested information to the client. IDIPS is operated by ISPs or ASes.

A client sends a list of all the possible source and destination addresses to the IDIPS server. In addition, the client gives its criteria for evaluating paths. When an IDIPS server receives such a request, it creates a set of candidates (source, destination) pairs, where each source and destination belongs to the initial list provided by the client. The IDIPS server then orders the pairs according to the client's criteria so that the first pair in the list is the best choice while the last one is the least preferred. The IDIPS server responds to the client by sending back this ordered list. Note that the addresses list provided by the client might be composed of prefixes, not necessarily of complete addresses.

Estimating the quality of a potentially large set of paths is a difficult task that can also be quite network-intrusive, even if an ISP provides a single IDIPS server. This is why IDIPS works in a two levels granularity in the fashion of PlanetSeer [4]. The first stage provides a high-level view of a path quality while the second gives a finer-grain point of view.

Achieving the first level is done using passive information such as BGP feed, IGP costs, SNMP statistics, or network policies. The main advantage of this first level is that the response can be quickly sent to the client at a very low cost (see Sec. 3 for details). The sec-

ond level is achieved by performing a set of active and passive measurements, such as RTT, bandwidth, TCP flows analysis, etc. For scalability reasons, measurements cannot be performed for every request. Only the prefixes (or addresses belonging to a particular prefix) that often appears in various clients requests are eligible for active measurements. Hence, measurements are performed only when the IDIPS server decide it without duplication. It is worth to notice that measurements must be performed independently of the request.

IDIPS is somewhat similar to the *oracle* recently proposed by Aggarwal et al. [5].The main difference stands in the fact that IDIPS is a more general solution than the oracle to this problem. For instance, IDIPS is able to manage not only a set of destinations but also a set of sources. This allows IDIPS to solve a larger range of problems, from choosing the best neighbors in a P2P system as done by the oracle to detecting the best addresses pair for multihomed systems.

3. PRELIMINARY EVALUATION

In this section, we study the processing time of the IDIPS server for the first level of granularity.

Our implementation is based on two concepts: the *knowledge base* (KB) and the *cost function* (CF). The KB is a Patricia tree of IP prefixes with metric attributes. The CFs get some attributes for a source and a destination prefix in the KB and combine them to form the cost of the pair. The cost of the pairs gives the order in the returned list: the pair with the lowest cost is the most attractive. For more details about the implementation, see [3].

In our testbed [3], the IDIPS server is connected to a XORP BGP router. XORP is fed with 4 different RouteViews RIBs where the number of routes in the 2001, 2003, 2005 and 2007 RIB is 107k, 140k, 183k, and 244k respectively. The server ran on a FreeBSD 5.5 Pentium 4 2.60 GHz computer with 1GB of memory.

We consider a set of 100 clients contacting IDIPS. Each client periodically generates a list of pseudo-uniform random source and destination prefixes and requests the server for a classification. The interval time between two requests of a particular client is given by a Poisson distribution with a mean of one second. The number of source and destination addresses in the requests follows a uniform random distribution of between 0 and 16.

Fig. 3 shows the evolution of the server processing time (in ms, vertical axis) with the number of returned pairs (horizontal axis). The possible pairs are ordered according to the BGP decision process. Neither clients nor server maintain caches and the time in buffers is considered as a part of the processing time.

We choose to consider the BGP decision process because its implementation is representative of the implementation of other cost functions. Moreover, for randomly chosen prefixes, the BGP attributes can be very different from prefix to prefix which gives good evaluation of the sorting algorithm performances.

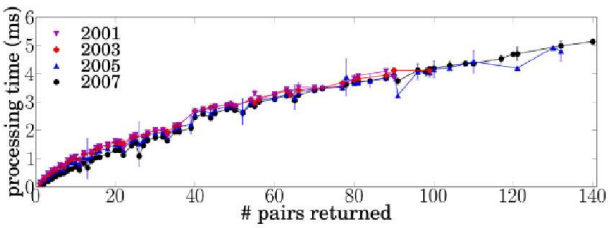

Figure 1: Evolution of the processing time with the number of returned pairs

From Fig. 3, we see that the IDIPS server processing time oscillates between 0.1 ms (when a single pair is returned) and 4.16 ms (when 100 pairs are returned). We further observe that the size of the returned list never reaches 256 (maximum is 140 for the 2007 curve), i.e., the maximum that can be returned in our tests (all the pairs formed by the 16 sources and 16 destinations). The IDIPS server removes some invalid pairs from the returned list. From our tests, we notice that invalid paths are more frequent for small RIBs than for bigger.

It is worth to notice that a list of 100 pairs has no real interest. Indeed, if the client must use the 100^{th} entry in the list, it means that the previous 99 pairs are invalid, which should never occurs. We would recommend to limit the number of returned pairs to 16. In such a situation, the processing time is 1.28 ms on average. Therefore, from a client point of view, the cost associated to IDIPS is negligible, i.e., no more expensive than a DNS request.

4. CONCLUSION

In this extended abstract, we introduced IDIPS, a scalable solution allowing ISPs to qualify paths between a set of sources and a set of destinations.

The evaluation of our IDIPS implementation indicated that the cost of using IDIPS is negligible for clients compared to the current selection techniques.

We are currently implementing active measurements metrics. In the next future, heuristics must be found to efficiently limit the number of returned pairs.

5. REFERENCES

[1] A. Akella, S. A., and R. Sitaraman, "A measurement-based analysis of multihoming," in *Proc. ACM SIGCOMM*, 2003.

[2] G. Huston, "BGP routing table analysis reports," 2004, see http://bgp.potaroo.net.

[3] D. Saucez, B. Donnet, and O. Bonaventure, "The IDIPS project," Université catholique de Louvain," Draft, 2007, see http://inl.info.ucl.ac.be/projects/idips.

[4] M. Zhang, C. Zhang, V. Pai, L. Peterson, and R. Wang, "PlanetSeer: Internet path failure monitoring and characterization in wide-area services," in *Proc. OSDI*, 2004.

[5] V. Aggarwal, A. Feldmann, and C. Scheideler, "Can ISPs and P2P users cooperate for improved performance," *ACM SIGCOMM Computer Communication Review*, vol. 37, no. 3, pp. 29–40, 2007.