

# Measuring BGP pass-through times

Anja Feldmann  
TU Munich  
anja@net.in.tum.de

Hongwei Kong  
Agilent Labs, Beijing  
hongwei@agilent.com

Olaf Maennel  
TU Munich  
olafm@net.in.tum.de

Alexander Tudor  
Agilent Labs, Palo Alto  
alex\_tudor@agilent.com

## 1. INTRODUCTION

Even though BGP has been studied extensively within the last few years, e.g., [1], we still do not have a good understanding on how and where the delays occur. Unexplainably large convergence times have been reported. It is unclear whether they are attributable to protocol interactions, implementation idiosyncracies, hardware limitations or other factors.

While one may attribute convergence times greater than 30 minutes to route flap damping [2] and those that are roughly multiples of 30 seconds and are less than 2 minutes to the MRAI timer [3] the wide variability of in-between times is puzzling [4]. An unknown component of the delay is the contribution of each router along the path of the BGP update. A detailed exploration on the delay of each router and how it relates to factors, such as CPU load, number of BGP peers, etc., can be very helpful for understanding such larger convergence time. In this paper, a methodology for studying the dependency of BGP pass-through times on such factors is presented together with some initial results.

## 2. TEST APPROACH

We propose to explore pass-through times of BGP updates using a black-box testing approach in a controlled environment with appropriate instrumentation. To this effect we have setup a test-bed that consists of:

- Device under test (DUT): the device to be tested, a router.
- Load framework: that can be used to impose a specific, pre-defined load on the DUT. In our case it consists of several PCs and an Agilent router tester. BGP workloads can be generated by the PCs as well as the router tester. We utilize the router tester to generate controlled rate data traffic. Usually tests are repeated with both PC as well as router tester generated BGP workloads.
- Instrumentation framework: that allows us to measure not just the pass-through times of BGP updates but also the load imposed by the load framework. It consists of several packet-level monitors as well as the router tester.

Note that this kind of test-bed can also be used to explore many other router specific measures, such as line card FIB convergence.

Our methodology goes beyond RFC compliance tests and benchmarks, e.g. IETF's BMWG, in that we do not consider pass-through times in isolation. Rather, we investigate the correlation between router load and pass-through time using a variety of stressors. The most critical situations arise once the router load reaches its limits. Among the many variables, that affect pass-through times, are: (1)

Number of peers, (2) Routing table size, (3) BGP update rate, and (4) CPU load. Additional parameters include the input/output queue length of BGP processes, the ACLs' complexity and hit ratio, BGP policy settings such as route-maps, peer-groups, filter-lists and/or communities, as well as AS prepending, etc.

In general there are three approaches for measuring in the Internet: passively observing regular traffic, actively evaluating injecting traffic at end points within the injecting application, and passively measuring actively injected traffic. Since we operate in a test bed the first option is not applicable. Accordingly we use active probes with specifically designed updates. These are passively measured by using synchronized packet monitors. Using special updates provides us with the ability to impose patterns with certain characteristics. The alternative approach is to replay measured BGP updated traces. While this provides useful background traffic it suffers several shortcomings. First of all the pass-through time of a router depends on the settings of several BGP specific parameters such as the value of the MRAI timer. In order to distinguish what delay is due to this timer from the delay due to the router we need certain update patterns which may or may not be present in regular BGP traces. Second, not all incoming BGP updates trigger an outgoing BGP update. Therefore it is hard to tell which BGP updates are discarded or are combined into a single update. Furthermore the amount of work associated with each BGP update will vary depending on its content with respect to the router's configuration.

## 3. TEST METHODOLOGY

There are three major parts that we need to investigate further. How to separate router processing delay from MRAI timer delay, how to impose a controllable load on the DUT, and how to measure pass-through time.

To consider the effect of the MRAI timer assume that it takes place at time  $t - 28$  seconds and  $t$ . All updates received and processed by the router within this interval are batched and sent via the outgoing BGP session(s) shortly after time  $t$ . A lower bound for the pass-through time estimation can be derived by using only those updates with a minimal MRAI timer delay. An upper bound is derivable from the updates with a maximum MRAI timer delay. In addition it is possible to disable the MRAI timer, which is a Juniper router's default configuration. Therefore it is possible to compare bounds using different MRAI values including zero. While it is desirable that the upper and lower bounds enclose the delay estimations without MRAI timer, that may not be the case due to the specific path through the BGP processing code.

To control the amount of resources a router has for processing BGP updates we start by studying the effects of

number of BGP peers, routing table size, BGP update rate and background CPU load. Note that the background CPU load is independent from the BGP update rate and the number of BGP peers while the routing table size and the number of BGP peers are correlated.

It is difficult to generate a constant background router CPU load as it implies generating a packet stream that is uniformly served by a task running at a uniform priority. This is rarely the case. In IOS the BGP processes (and any routing tasks) have higher scheduling priority than most anything else targeted at the CPU. We load the CPU by sending a controlled rate packet stream to an interface of the DUT. The payload is chosen such that it mainly raises the CPU consumption of the Cisco IOS IP input process.

Routing table size and number of peers tests currently consist mainly of measurement update patterns. We plan to also include BGP sessions that generate additional updates. They contain synthetic or trace collected updates of different sizes, profiles, and characteristics: “customer or peering” session vs. “full BGP feeds”. These differ in the overlap of their address space, sizes and often update rate.

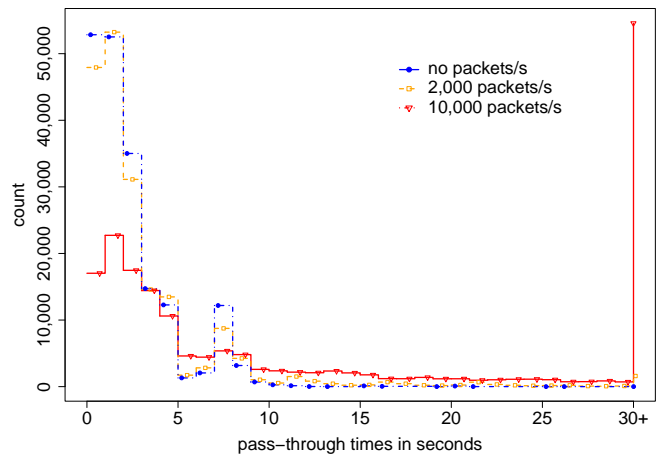
The simplest form of a BGP update probe pattern is comprised of a single new prefix. Since the prefix is new, the update has to be propagated to all neighbors, policy permitting. The drawback of this method is that the routing table size grows ad infinitum and available memory becomes an unintended co-variable. This can be avoided by sending BGP withdrawals after most updates have passed-through. This interval is experimentally established a priori. The next simplest form of probes are those that impose a FIB change. One way to cause a FIB change is to change the next hop. For this it is essential to send multiple updates from different peers, preferable with the same AS path length. The BGP update rate for both probes and traces can be controlled.

Pass-through time is not a constant. It varies with the makeup of BGP updates and other load causing factors. Accordingly, we obtain sample values by measuring the time difference between in-bound (into the router) probe injections and out-bound passing (to monitoring BGP session) of the probe update. In order to avoid synchronization errors, all PC based BGP sessions use different interfaces on the same machine; probing and monitoring sessions when using the router tester are synchronized by design. Router tester experiments use DAG cards for traffic capture. These cards are installed in the same machine and synchronized with each other. Packet monitors are used on the PC for both in-bound (probing) and out-bound (monitoring) BGP sessions.

## 4. INITIAL RESULTS

Due to the large number of parameters we cannot test all combinations. Rather, we perform a number of tests to determine the variables to which pass-through time is most sensitive.

The simplest test consists of 500 BGP sessions with update probes and no background BGP traffic. The probes are sent at equally spaced 5 second intervals resulting in an average update load of 100 updates per second. Due to limited space we present only the results that show the sensitivity of the pass-through time to a data traffic stream directed to the CPU and thereby, indirectly, to the CPU load. To load the CPU we send packets to a router interface using the Agilent router tester at a rate of 0, 2,000, and 10,000 pack-



**Figure 1: Histogram of pass-through times with 0, 2,000, 10,000 packets of background traffic.**

ets a second. This in itself introduces an average CPU load that is centered around 0.36%, 20%, and 83%. Note that measuring router CPU load is problematic since we depend on the “show processes cpu” command of the router which since it is performed via the command line has a rather low priority. The same is true for the corresponding SNMP request. Adding BGP updates increases the average CPU load to 15%, 47%, 69%. These numbers are only approximations. Deterministic setting of CPU load is router OS dependent and remains a topic for future work.

Figure 1 shows a histogram of the resulting BGP pass-through times without active MRAI timer. As long as the CPU load is not artificially increased the pass-through times are reasonable small with an average of 1.92 and 50th, 90th, 95th quantiles of 1, 6, 7. As the CPU load increases the pass-through times increase as well. The mean increases from 1.92 over 2.86 to 104.82, the 95th quantile from 7 over 9 to 1126 seconds.

## 5. SUMMARY

Our results show that large BGP convergence times may not just stem from protocol related parameters, such as MRAI, and route flap damping. The delay imposed by the router, the pass-through delay, may play a big role as well. We observe that under extreme conditions, this delay can cause the drop of a BGP session. For the evolution of interdomain routing further analysis regarding the limitations imposed by current and future routers regarding how many updates can be processed how fast is needed. Furthermore additional work is needed to safeguard routers from DDoS attacks, since an attack can easily impose a similar packet stream as the one used in our experiments.

## 6. REFERENCES

- [1] T. Griffin, “Interdomain Routing Links.” <http://www.cambridge.intel-research.net/~tgriffin/interdomain/>.
- [2] Z. M. Mao, G. Varghese, R. Govindan, and R. Katz, “Route flap damping exacerbates Internet routing convergence,” in *Proc. ACM SIGCOMM*, 2002.
- [3] T. G. Griffin and B. J. Premore, “An experimental analysis of BGP convergence time,” in *Proc. International Conference on Network Protocols*, 2001.
- [4] Z. M. Mao, R. Bush, T. Griffin, and M. Roughan, “BGP beacons,” in *Proc. Internet Measurement Conference*, 2003.