Longitudinal Study of BGP Monitor Session Failures *

Pei-chun Cheng
Department of Computer Science
University of California, Los Angeles
pccheng@cs.ucla.edu

Xin Zhao
Department of Computer Science
University of Arizona
zhaox@email.arizona.edu

Beichuan Zhang
Department of Computer Science
University of Arizona
bzhang@cs.arizona.edu

Lixia Zhang
Department of Computer Science
University of California, Los Angeles
lixia@cs.ucla.edu

ABSTRACT

BGP routing data collected by RouteViews and RIPE RIS have become an essential asset to both the network research and operation communities. However, it has long been speculated that the BGP monitoring sessions between operational routers and the data collectors fail from time to time. Such session failures lead to missing update messages as well as duplicate updates during session re-establishment, making analysis results derived from such data inaccurate. Since there is no complete record of these monitoring session failures, data users either have to sanitize the data discretionarily with respect to their specific needs or, more commonly, assume that session failures are infrequent enough and simply ignore them. In this paper, we present the first systematic assessment and documentary on BGP session failures of RouteViews and RIPE data collectors over the past eight years. Our results show that monitoring session failures are rather frequent, more than 30% of BGP monitoring sessions experienced at least one failure every month. Furthermore, we observed failures that happen to multiple peer sessions on the same collector around the same time, suggesting that the collector's local problems are a major factor in the session instability. We also developed a web site as a community resource to publish all session failures detected for RouteViews and RIPE RIS data collectors to help users select and clean up BGP data before performing their analysis.

Categories and Subject Descriptors

C.2.2 [**Computer Communication Networks**]: Routing Protocols

General Terms

Measurement, Experimentation, Reliability

Keywords

BGP Monitoring, BGP Session Reset

1. INTRODUCTION

RouteViews [4] and RIPE RIS [3] have been collecting BGP [14] routing data from the global Internet over a decade. The original purpose was to provide network operators "looking glasses" on the routing system from other networks' point of view. Over time, this data source has also become indispensable to the research community to help understand various aspects of the global routing system, such as Internet topology [16], BGP convergence [13], ISP peering policies [10], and prefix hijack monitoring [12], to name just a few.

Unfortunately, the quality of the collected BGP data is known to be far from perfect. BGP sessions between the data collector and operational routers fail now and then, and when such a session failure occurs, the collector misses BGP updates during the session downtime and receives superfluous updates due to the table transfer after each session re-establishment [14]. Yet there has been no systematic measurement on the monitoring session failures or assessment of their impact on the quality of the collected BGP data.

The importance of this data cleanup step has been highlighted in several prior works, such as the analysis of BGP stability during worm attacks [18], comparison of routing stability among different prefixes [15], and correlations of routing events in a network [7]. Of course the exact impact of the data deficiency depends on the nature of each specific purpose. For example, missing updates during the session downtime may not affect the results of collecting Internet topology over a long period of time, but can affect the results of analyzing routing dynamics, and can even be critical if the downtime is correlated with routing dynamics. Similarly, the extra updates from table transfers may also affect different work in different ways. As an example, based on the large update surges at BGP collectors during worm attacks, Cowie *et al.* [8] conjectured that worm attacks caused BGP routing instability. However Wang *et al.* [18] showed later that the update surge was largely due to the monitoring session resets and the worm did not lead to significant instability in the global routing system. Had the session failure information been available, the misinterpretation would have been avoided.

In this paper, we report our findings from the first longitudinal study of BGP monitoring session failures for six RouteViews and RIPE collectors over the last 8 years. We use the Minimum Collection Time [19] algorithm as the main tool to identify BGP session resets between operational routers and the data collectors. We also analyze the impacts of collector instability and BGP timer on session failures. Our results confirm the speculation that the raw BGP data collected by RouteViews and RIPE contain noises caused by measurement artifacts. Our main findings can be summarized as follows:

- The monitoring session failures are relatively frequent, averaging a few times a month. Most failures have a session downtime within tens of minutes.
- A significant number of failures are caused by the collectors local problems, resulting in multiple peer sessions reset at the same time.

*This work is partially supported by US National Science Foundation under Contract No CNS-0551736.

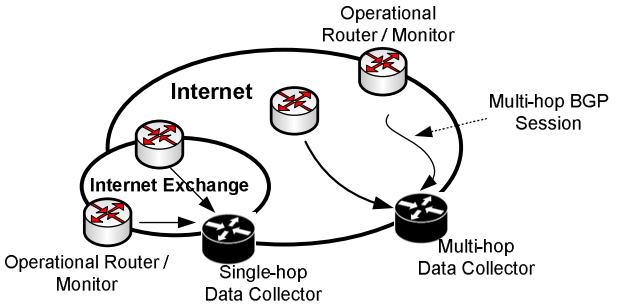


Figure 1: BGP Monitoring

- Although disabling BGP Keepalive and Holddown timers, as RIPE did from 2002 to 2006, may make a BGP session more robust against packet losses, it can also lead to unnoticed session failures and extremely long session downtime.

As the main outcome of this study, we have developed a web site, `http://bgpreset.cs.arizona.edu`, to publish the detected session failures with the occurring times and durations for historical RouteViews and RIPE data; the web page is also updated periodically to include the latest data. Given this information, users of RouteViews and RIPE data can choose which period of data to use and which part of the data to sanitize for accurate analysis.

The rest of the paper is organized as follows. Section 2 gives brief background on BGP monitoring projects and BGP sessions. Section 3 describes the data source and the technique we use to detect session failures. Section 4 presents the overall statistic results and observations for RouteViews and RIPE monitoring session failures, Section 5 correlates session failures to infer the failures due to collectors' local problems. Section 6 investigates the impact of the historical decision on turning off BGP Keepalive/Holddown timers. Section 7 briefly reviews related work, and Section 8 summarizes the paper.

2. BACKGROUND

RouteViews and RIPE RIS, the two best known BGP data collection projects, operate a number of *collectors* that establish BGP peering sessions with routers in many operational networks. We call each operational router connected to a collector a *monitor* or a *peer*, and the BGP session between the monitor and the collector a *monitoring session*. A monitoring session can be either *single-hop* or *multi-hop* depending on whether the session is across a single or multiple router hops. As shown in Figure 1, single-hop monitoring sessions are usually deployed at an Internet Exchange, while multi-hop monitoring sessions are established over wide-area networks. The data collectors receive BGP routing updates from its peers and write the collected BGP updates into files every 15 minutes (RouteViews) or every 5 minutes (RIPE) in the Multi-threaded Routing Toolkit (MRT) [6] format. These files are then made publicly available. The collectors also dump snapshots of the BGP routing table, the RIB, for each of its peers every two hours in the MRT format.

BGP uses TCP for reliable communication. After successfully setting up a TCP connection, two BGP peers negotiate BGP timer settings and capabilities [14] to establish a BGP session in between. They then exchange with each other the full routing table, which is called *table transfer*. After this initial table exchange, the peers only send to each other new updates when any route changes, which are called *incremental* updates.

A BGP session may fail due to a variety of causes, such as (1) malformed updates which may in turn be caused by hardware or software defects, (2) TCP connection failures due to link or interface failures, (3) data traffic congestion which results in the loss of three consecutive BGP Keepalive messages, or (4) either end (the host or its routing daemon) fails. BGP employs two timers, Keepalive and Holddown, whose default values are 60 seconds and 180 seconds respectively, to maintain its session. BGP peers send to each other Keepalive messages at every Keepalive timer interval. If no Keepalive message is received before the Holddown timer expires, a BGP router will tear down the existing session and initiate a new one, which is called a *session reset*.

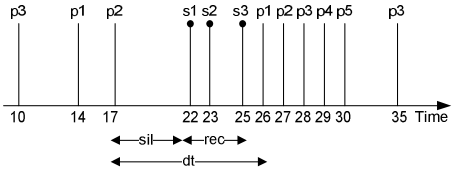


Figure 2: BGP Update Stream (sil: silence period; rec: session reconnection; dt: downtime)

Table 1: BGP Data Sources

Collector	Type	Start Date	Location
RRC00	Multi-hop	2001 Jan	Amsterdam
RRC01	Single-hop	2001 Jan	London
RRC02	Single-hop	2001 Mar	Paris
OREG	Multi-hop	2001 Oct	Oregon
LINX	Single-hop	2004 Mar	London
EQIX	Single-hop	2004 May	Ashburn

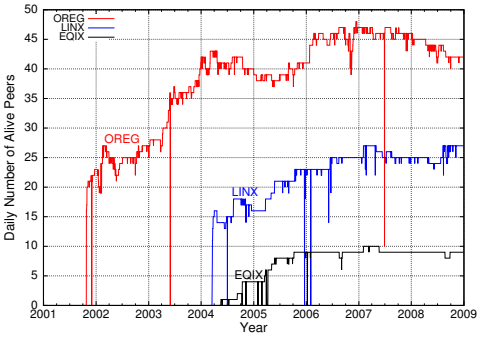
Let us use a simple example to illustrate the impact of BGP session reset on the data collection. Assuming that a monitor has a routing table of 5 prefixes, Figure 2 shows a BGP message stream arrived at the collector. The first three messages are regular BGP updates (for prefixes $p1$, $p2$, $p3$) received at time 10, 14, and 17, respectively. Then the session fails at time 17 and restarts at time 22. The session re-establishment takes time from 22 to 25, during which three BGP *state messages* are recorded. The state message $s1$ marks the time when a router initiates a BGP session, while $s3$ marks the time when the session is fully established. We show three state messages here for illustration purpose; in reality establishing a BGP session may require more state changes [14]. Following state messages are the table transfer updates during time period [26, 30], which include the entire routing table entries ($p1$ to $p5$), followed by incremental updates afterwards.

The above example shows that, if BGP updates arrive after time 17 and before time 25, they will be missed by the collector. In addition, 5 extra table transfer updates are introduced by the session reset.

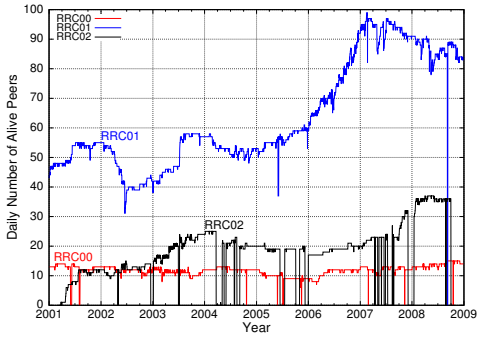
3. DATA SOURCE AND METHODOLOGY

3.1 Data Sources

RouteViews and RIPE started collecting BGP data in the late 1990's, but they went through a learning period in the first few years before the data collection process stabilized. Thus this paper uses the data from January 2001 onward. We take data from six collectors whose information is summarized in Table 1. Figure 3 shows how the number of peers at each of these six collectors



(a) RouteViews



(b) RIPE

Figure 3: Number of monitors over time.

have changed over the last eight years. For each day we count the number of unique peers that logged any BGP data. The downward spikes in the figures mean that a large number of peers did not log any data on those days, which could be caused by collector outage or maintenance, and we will investigate the collector's local problems in more details later.

3.2 Detecting BGP Session Resets

As Figure 2 shows, session state messages ($s1$, $s2$, $s3$) mark when a new BGP session is attempted and when it is fully established. With this information we can identify all session resets accurately. Unfortunately state messages are only logged by RIPE, but not by RouteViews. State messages also do not help identify the end of the table transfer.

Zhang *et al.* [19] developed an algorithm called Minimum Collection Time (MCT) that can identify the start and the duration of table transfers from BGP data in the absence of state messages. Based on the fact that all prefixes in the routing table are announced during a table transfer, MCT searches for the smallest time window during which the full table is announced. Using three months of data from 14 different monitored peers, this method successfully detected over 94% of session resets[1]. We have developed an enhanced MCT algorithm that further improves the detection accuracy.

In this paper, we use MCT as the main tool to detect BGP session failures, and a combination of MCT with state messages when handling RIPE data. Since MCT accuracy improves with large routing table sizes, in this study we only consider monitors whose exported routing tables have more than 500 entries. Due to space limitation, we refer interested readers to [19] and [1] for the detail algorithms.

In [17] Wang *et al.* used syslog messages to detect failures of

[1]The false positive in [19] is lower than 5%.

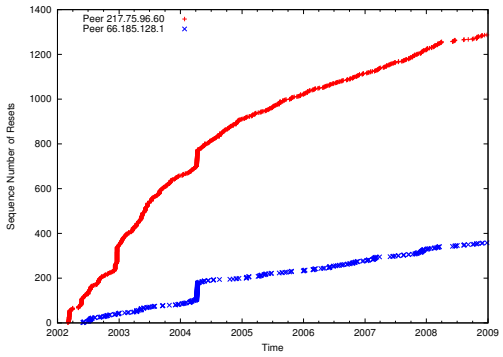


Figure 7: Resets of two example sessions

BGP sessions in a tier-1 ISP. Unfortunately syslog information is not available from RIPE or RouteViews collectors. Currently, RIPE makes available the BGP log files from Quagga [2], the routing software running on its collectors, but Quagga log does not explicitly record BGP session resets. RouteViews maintains logs from Rancid, a tool that monitors the changes of router configuration. However, Rancid log is only generated once every hour. We use these logs to cross check our results, but cannot rely on them as the primary method of detecting session resets.

In the following three sections, we characterize failures of RIPE and RouteViews monitoring sessions identified by MCT and BGP state messages.

4. CHARACTERISTICS OF SESSION RESETS

We present the overall BGP session statistics in this section, and then investigate the stability of the collectors and the impact of disabling BGP Keepalive and Holddown timers in the next two sections.

Since data collectors only passively receive BGP updates from their peering monitors and are not involved in forwarding data traffic, the monitoring sessions between data collectors and monitors have simple configuration, low workload, and requires little maintenance. Thus the monitoring sessions are expected to be stable and long lived, and users of BGP data usually do not pay much attention to possible session resets during their measurement periods.

Our results, however, show that monitoring session resets are relatively frequent. Figure 7 shows the cumulative number of resets for two monitoring sessions at the OREG collector, 66.185.128.1 and 217.75.96.60, over the past eight years. The session with 66.185.128.1 has 4.5 resets per month on average, a typical case among the sessions at OREG. The session with 217.75.96.60 is the worst case at OREG, averaging 15.8 resets per month. Although some months have more BGP session resets than others, overall the resets occur persistently over time.

Frequent session resets are also observed across all the collectors, regardless of the type of the session (single-hop or multi-hop), the age of the collector, or its location. Figure 4 shows the cumulative distribution of the number of resets per peer per month for all the 6 monitors we measured. For the two multi-hop collectors, OREG and RRC00, 10-20% session-months do not have any reset, while the 50-percentile is 3 resets, and the 90-percentile is 12 to 15 resets per session-month. The worst case at OREG is a monitor that had 117 resets in one month, while one of the RRC00 peers had 4205 resets in one month. The single-hop collectors have fewer resets, but the numbers are still alarming. RRC01 and RRC02 also

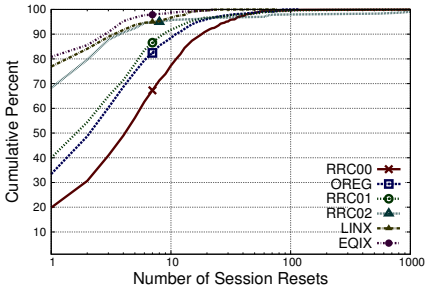


Figure 4: Number of Resets per session-month.

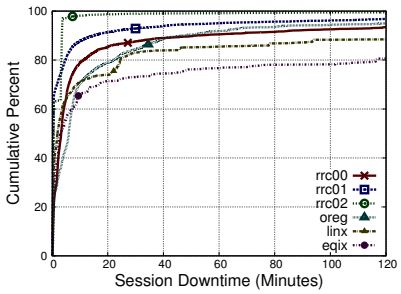


Figure 5: Session Downtime

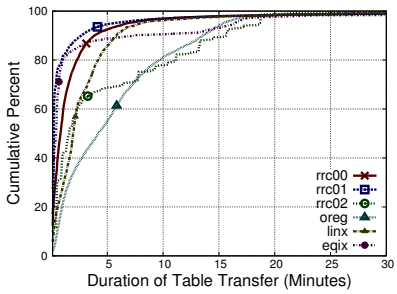


Figure 6: Duration of Table Transfers

have some sessions that had thousands of resets in a month. These cases were likely caused by hardware problems or misconfigurations that made the sessions up and down constantly before they were fixed.

When a monitoring session fails, the observed session downtime usually ranges from one or a few minutes to a few tens of minutes, during which routing updates will not be received from the peer. Figure 5 shows the cumulative distribution of session downtimes. Here, *session downtime* is defined as the time between when the failure is detected and when the BGP session is fully re-established. Since the failure itself is not logged in the BGP data, we measure session downtime from the last BGP update preceding a reset and the first BGP update after the session re-establishment, as illustrated in Figure 2 from time 17 to 26 which represent the reasonable upper bound on the *real* session downtime. In Figure 5, we observe that the majority of session downtimes are within ten minutes, but some cases are much longer. For example, at OREG the session downtime has a 25-percentiles at 1 minute, 50-percentiles at 6 minutes, and 90-percentiles at 48 minutes. All collectors have cases in which the session downtimes are more than 10 days. Users of BGP data can easily spot very long session downtimes (*e.g.*, days) and take precautions accordingly in their data processing. However, given that majority of the session downtimes are within 10 minutes, without knowing the existence of session resets, it is difficult for the BGP data users to identify these short durations of quiet periods as data missing and take proper measures accordingly.

Figure 6 shows the cumulative distribution of table transfer duration after each session reset. Over 90% of all table transfers finish within around 5 minutes, while table transfers at OREG tend to take longer time to finish, with 50-percentile at 4.5 minutes and 90-percentile at 14 minutes. We have calculated and found that the table transfer time is not significantly correlated with the routing table size, which indicates that the link bandwidth is not the limiting factor. As Houidi *et al.* [11] has discovered, slow table transfers are largely caused by router's timer-driven processing in sending BGP updates.

The main point to take away from this section is that the BGP monitoring session resets occur frequently, averaging a few times per peer per month across all the 8 years and 6 collectors that we have examined. Majority of session downtimes last within 10 minutes and the following table transfers usually complete within another few more minutes, during this time period actual BGP updates are missing and superfluous table transfer updates are introduced. There exist extreme cases with thousands of resets in a month, or downtime for multiple days, or tens of minutes or longer to finish a table transfer. It is imperative for users to be aware of these events and take them into account when using the BGP data.

5. COLLECTOR STABILITY

Maintaining a stable data collecting service is critical to the quality of logged BGP data. Collecting services may be disrupted by hardware defects, software bugs, network problems, or planned maintenance. For example, RouteViews has reported sporadic collector outages owing to interface malfunctions, memory problems, fiber cuts, software upgrades, and other problems [5]. RIPE also occasionally announces degraded service for maintenance [3]. Unfortunately, neither RIPE nor RouteViews maintains complete information about collector outages. In this section, we identify collector problems by correlating session resets on the same collector.

5.1 Correlating Session Resets

From the session resets identified in the previous section, we find that a collector's session resets across different peers are sometimes clustered within a short time window. For example, Figure 8 shows the session resets for RRC00 during August, 2003. On August 19th, almost all peers had session resets. This implies that the collector itself might have experienced a problem.

We define *synchronized session resets* of a collector as a group of resets occurring within a time window w, *synchronized peers* as the peers appearing in synchronized resets, and *synchronization ratio* as the ratio of the number of synchronized peers to the number of total alive peers of the collector at that time. For example, if five out of ten peers have resets within w, these five resets are synchronized resets associated with five synchronized peers, and the synchronization ratio is 0.5.

Figure 9 shows the cumulative distribution for the number of synchronized peers for four collectors [2]. For RRC00, about half of the session resets are standalone (*i.e.*, the number of synchronized peer is 1), and the rest of resets are synchronized to some extent. For other collectors, synchronized resets contribute to more than 70% of all the resets. There is a sharp increase near the tail of the curve, indicating that a significant number of session resets involves most or all peers.

Figure 10 shows the cumulative distribution of the synchronization ratio. There is a sharp increase among all the four collectors between 0% to 10%. This is because the collectors usually have 10 to 20 concurrently alive peers, which leads to a lower bound on the synchronization ratio of approximately 5% to 10%. After the synchronization ratio passes 90%, there is another sharp increase, which accounts for 10% to 30% of the total session resets.

[2]We use four example collectors to demonstrate the distributions of synchronized resets.

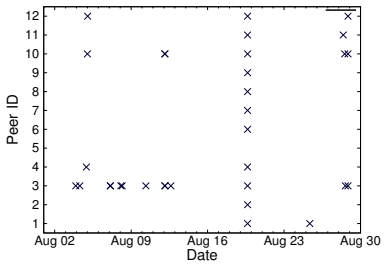


Figure 8: Session Resets in August 2003

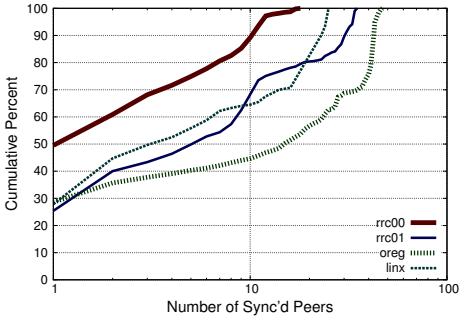


Figure 9: CDF of Sync'd Peers

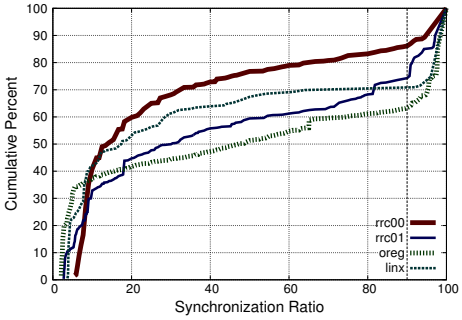


Figure 10: CDF of Sync Ratio

5.2 Identifying Collector Problems

We assume that if all or most peers have session resets at the same time, the cause is likely to be a local problem at or near the collector. We name such a problem "collector-restart", even though the session resets can be due to different local problems, such as a collector machine reboot, a BGP daemon restart, or network connectivity problems, and so on.

We use a 90% of synchronization ratio as the threshold to detect collector-restart and require that there must be at least five alive peers. As a result, we detected 72 collector-restarts at RRC00 from August 2002 to December 2008. August 2002 is used as the starting time because RIPE started to archive the process log of the collector daemon at that time. The process log records the termination and startup of the collector process, and thus can be used to verify our detection results. After matching the *observed* restarts against those *recorded* in collector process logs, we find 7 observed collector restarts that are detected by our method but not recorded in collector process logs. Further inspection finds that 5 of the 7 cases are due to errors in the collector log and 2 cases are due to a large number of BGP re-connections in a short time, which might be caused by network instability. There are also 22 collector restarts that are recorded in collector process logs but our scheme failed to detect. Among these cases, 2 are due to two consecutive collector restarts, so that there is no BGP session successfully established in between. The other 20 cases are due to some peers that disconnected or failed but are still counted as active, so that a collector could not successfully re-establish sessions to these peers after collector restart, and results in a synchronization ratio which is lower than our 90% threshold. We are modifying our algorithm to capture such cases to reduce false negatives. Overall this simple algorithm yields over 95% correctness and detects 80% of collector restarts.

Note that without using this inference algorithm, we may still directly identify collector-restarts solely based on collector logs. However, as we show in the previous comparison, the collector log itself is incomplete. In addition, collector logs are not even available for RouteViews. We have contacted RouteViews operators, and they plan to provide collector logs in the near future. Still, for historical data, detecting synchronized session resets provides a practical way to identify RouteViews collector problems.

Table 2 shows the number of collector-restarts detected at each collector along with the number of session resets triggered by these restarts. We can see that 14% to 37% of session resets are caused by collector-restarts [3]. The problem is more pronounced for collectors that have many peers, such as OREG, for which 37% of session resets are due to local problems at the collector. Since collectors' local problems are a major contributor to session failures, it is important to improve the stability of the collector, including its network connectivity, software and hardware, in order to reduce monitoring session failures.

[3]Since RRC02 sessions are quite stable in general, the number of session resets is not large enough to conclude a collector restart by using synchronization ratio.

Table 2: Session Resets on Collector Restarts

collector	no. restarts	no. session resets (%)
RRC00	105	1154 (14%)
RRC01	112	1999 (26%)
RRC02	-	-
OREG	178	6370 (37%)
LINX	29	673 (30%)
EQIX	9	69 (14%)

Table 3: RIPE BGP Timers Settings

Time Period	Keepalive	Holddown
Before 2002 Oct 17	60 sec	180 sec
After 2002 Oct 17 Before 2006 Nov 23	0 sec	0 sec
After 2006 Nov 23	60 sec	180 sec

6. KEEPALIVE AND HOLDDOWN TIMERS

In October 2002 RIPE disables all its collectors' BGP Keepalive/Holddown timers. This was due to the observation that, during periodic RIB archiving, some old collectors stopped sending BGP messages, causing BGP sessions to timeout and triggering a surge of session resets. To alleviate this problem, RIPE disabled BGP timers. However afterwards it was noticed that disabling Keepalive/Holddown timers caused BGP to lose the ability to detect connectivity problems such as link failures, and thus introduced long, unexpected session downtimes. Since later collector software fixed the BGP message blocking problem during RIB archiving, RIPE restored the BGP timers on all its collectors in November 2006. Table 3 summarizes the timer settings for RIPE; note that a value of 0 disables a timer. In this section, we document and quantify the impacts of changing BGP Keepalive/Holddown timers on the stability of RIPE monitoring sessions.

One issue we observed is that, while RIPE's plan was to disable the timers for all the BGP monitoring sessions, the Keepalive/Holddown timers for some peers were never turned off. This could be due to the fact that a zero timer value was not allowed

Table 4: KAE / KAD Peers

Collector	Total Peers	KAE	KAD
RRC00	42	9	33
RRC01	57	5	52
RRC02	15	2	13

on some Juniper routers as of 2002, or due to misconfigurations, which we will discuss later.

No matter what may be the cause, to measure the impact of disabling BGP timers, we need to differentiate between BGP sessions that have the timers disabled, and those that have the timers enabled. We define *Keepalive-enabled (KAE)* sessions as BGP sessions that have the Keepalive timer enabled, and *Keepalive-disabled (KAD)* sessions as the sessions that have the timer turned off.

6.1 Identifying KAE/KAD Sessions

Differentiating between KAD and KAE sessions poses a challenge since RIPE does not keep historical records for collector configurations. In this section, we proposed a heuristic method to distinguish these two kinds of sessions.

The basic idea is to infer the BGP Holddown timer value based on the distribution of session downtime. More specifically, we divide a session's downtime into a *silence period* followed by a *recovery period*. We define the *silence period* preceding a session reestablishment as the duration when a failed session remains silent. Figure 2 shows an example silence period, sil, between times 17 and 22. In general, silence periods indicate how long it takes for a data collector to detect failures. For session resets triggered by Holddown Timer expiration, the duration of silence period should be close to the length of the Holddown Timer. Figure 11 shows the distribution of silence time for session resets from an example RRC00 session with 90 second Holddown Timer, which shows that a significant number of session resets are associated with a 90 second silence period. We also define the *recovery period* as the length of time taken to re-establish a BGP session. Figure 2 shows an example recovery period, rec, between times 22 and 25.

Based on these definitions, we identify KAE sessions as those with a single silence period duration length which is associated with more than 10% of session resets. This 10% threshold is chosen conservatively based on the measurement result in [17], which observed that more than 20% of session resets are triggered by the expiration of BGP Holddown Timers.

Applying this algorithm on RRC00 data, we identified 9 KAE sessions out of total 42 BGP sessions. Figure 12 and Figure 13 show the distribution of silence time for one identified KAE session and one KAD session, respectively. The vertical lines mark the dates when RIPE disabled and enabled BGP timers. These two figures verify that, after RIPE disabled timers on Oct 17, 2002, the identified KAE session continued to trigger session resets after a 90 second silent period, but the KAD session did not. Table 4 summarizes the inference results for three RIPE collectors. In the remaining of this section we only consider session resets from the KAD sessions.

6.2 Number of Session Resets

We first measure the number of session resets before and after disabling timers. Figure 14(a) shows the cumulative distribution of the number of session resets per month for KAD sessions. We group session resets into three periods based on the dates RIPE disabled and enabled timers: *Before 2002.11*, *2002.11 to 2006.11*, and *After 2006.11*. After disabling BGP timers in 2002.11, we can observe a left shift of the distribution, which indicates a drop in the number of session resets. The median number of session resets of "*Before 2002.11*" is about 4 times of that of "*2002.11 to 2006.11*". This shows that disabling BGP timers did reduce the number of session resets.

After 2006.11, when RIPE restored the timers, the distribution shifts right, but with a smaller magnitude. This is because the newer version of the collector software fixed the BGP message blocking problem during RIB archiving. Thus there should not be as many resets as before Nov, 2002. We observed a similar distribution of the number of session resets for other RIPE collectors.

6.3 Session Downtime

In this section, we measure the *silence period* and *recovery period* for the unnoticed side effect of disabling BGP timers.

Figure 14(b) shows the CDF of the silence period for KAD sessions. Before disabling BGP timers, there are two consecutive sharp jumps at around 90 and 180 seconds silence time, which represent session resets trigger by Holddown timers with 90-second and 180-second values. After disabling Keepalive timers, these two jumps basically disappeared and the CDF of the silence period began to follow a long-tail distribution. This is because, with KeepAlive timers disabled, BGP sessions could no longer detect failures by the timeout interval. These failures either went on unnoticed, or were eventually detected by external signals such as TCP errors, at much later time.

Figure 14(c) shows the cumulative percentage of recovery time for session resets. We observed that disabling BGP timers did change the distribution of recovery time. This seems counterintuitive because Keepalive/ Holddown timers are expected to only affect the silence time but not the recovery time. One possible explanation is that, though disabling timers does not change the recovery time for a given session failure, it could potentially change the *visibility* of some session failures.

More specifically, [17] observed that session failures can mainly be categorized into 4 groups. The first and second groups contains failures such as *admin resets* and *peer closed sessions*, these types of resets can recover fast. The third group contains *local holddown timer expired*, which results in moderate downtime. The fourth group contains *local router shutdown* and *peer de-configured*, which have very long recovery times. As a result, disabling Keepalive timers would make a BGP session *blind* to the third group of failures, and skew the distribution of recovery time towards the other three groups, which have either much shorter or longer recovery times. This explains the increase in percentage of both short recovery times and long recovery times in Figure 14(c).

In this section, we analyzed RIPE BGP data to show that disabling Keepalive timers indeed reduced the number of session resets. At the same time, it also led to a long-tail distribution of session silence time, during which session failures went unnoticed and real BGP updates were lost. Thus we recommend not to disable Keepalive and Holddown timers, even though this is allowed in the BGP specification[14]. In addition, when interpreting historical RIPE data, users need to be aware that long silence times might be the result of unnoticed BGP session failures, rather than live BGP sessions suddenly became quiet.

7. RELATED WORK

The quality of BGP data collected by RouteViews and RIPE is far from perfect because of measurement artifacts and missing data. A number of previous works have recognized the need to identify updates due to table transfers following monitoring session resets. Wang *et al.* [18] uses BGP session state message to identify the start

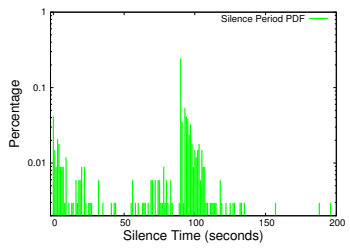


Figure 11: Sample Silence Period Distribution

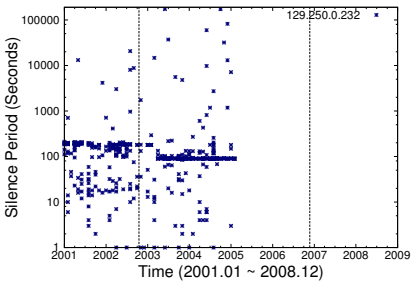


Figure 12: KAE Silence Period

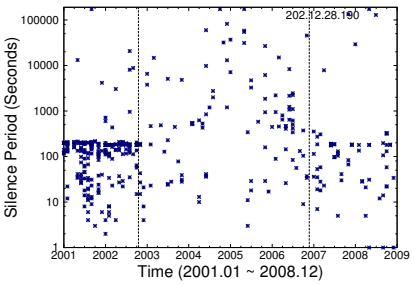
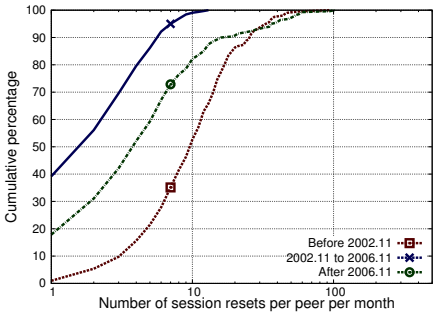
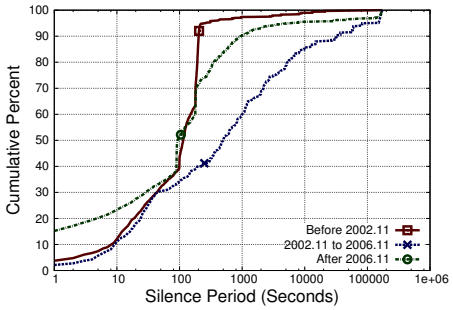


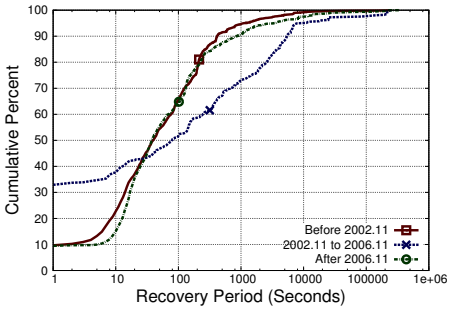
Figure 13: KAD Silence Period



(a) Number of Session Resets



(b) Silence Period



(c) Recovery Period

Figure 14: Impact of Disabling Keepalive Timer, RRC00.

of a BGP session re-establishment but this scheme is only available for RIPE data. Rexford *et al.* [15] removes all duplicate BGP announcements from the update stream which is an effective means to remove updates due to table transfers, though it also removed real duplicates. Anderson *et al.* [7] identify table transfers using a rough estimate, it splits the BGP update stream into 30-second bins and discards any bin that contains more than 1000 prefixes. Zhang *et al.* [19] developed MCT to accurately detect the occurring and duration of table transfers from BGP update messages. All these efforts focus on cleaning up BGP data by removing table transfer updates, rather than quantifying BGP resets of monitoring sessions which is the goal of this paper. Furthermore, as we showed in this paper, there were significant amount of session downtimes, during which actual BGP update messages are not recorded. Unlike table transfers which can be filtered out, there is no way to recover missing historical data. For all users of historical BGP data, it is critical to know when BGP data may be missing, and we have made such information available.

Wang *et al.* [17] infers the root cause of session failures in one large ISP. By using syslog event, router configurations, and SNMP traffic data, their scheme provides a practical way to identify the direct cause of operational session failures. However, such information is unavailable from RV/RIPE to understand the failures between a data collector and its peering monitors. Also, in [17], Wang *et al.* report the normalized results for one ISP which might not fully represent the characteristics and the impact of session failures in all other ISPs. The work reported in [9] may be considered most relevant to ours, in which Flayel *et al.* checks the consistency of BGP data. However our focus differs significantly from [9] in that we focus on a longitudinal quantification of monitoring session resets and their impact on BGP data quality.

Most recently Houidi *et al.* [11] found that, for routers from three particular vendors, the long table transfer durations are caused by routers process timers that regulate the processing of updates, which explains the lack of observed correlation between the routing table size and the transfer time.

8. SUMMARY

In this paper we reported the first systematic assessment on the BGP session failures of RouteViews and RIPE data collectors over the last eight years. Our results show that failures of the BGP monitoring sessions are relatively frequent, averaging a few session resets per monitor per month. How to make BGP sessions robust against transient packet losses remains an open problem both in BGP monitoring projects and in operational networks. Our measurement also show that failures local to the data collectors contributed between 14% to 37% of the total session resets. Although some cases could be due to intended administrative maintenance, they nevertheless affect the quality of the data being collected.

In the process of analyzing BGP session resets using the historical data, we also found that disabling BGP's Keepalive timer leads to negative consequence of unnoticed session failures. We proposed an efficient algorithm to detect ISP peers that turned off BGP timers. Users of historical RIPE BGP data should take into account the potential long downtime and missing updates for the affected peers in order to achieve reliable results.

To help users avoid the negative impact caused by BGP monitoring session failures, we have developed a website, *BGPReset*, which reports monitoring session failures, together with their occurring time and duration, for three RouteViews collectors (OREG, LINX, EQIX) and three RIPE collectors (RRC00, RRC01, RRC02). The URL is `http://bgpreset.cs.arizona.edu/`. Two types of failure information are reported:

- Session Resets
  The occurring time of session resets, together with the proceeding session downtime and duration of the table transfer when the session is re-established.
- Collector Restarts
  The occurring time of each collector's outage/restarts, identified by synchronized session resets of all sessions on the same collector, including the number of monitor peers affected.

Users can either use the exported query interface to lookup session resets of particular collector, monitor, time period, etc., or download raw result files for offline processing.

9. REFERENCES

[1] BGPReset website - algorithm. "http://bgpreset.cs.arizona.edu/#methodology".

[2] Quagga Software Routing Suite. http://www.quagga.net/.

[3] RIPE Routing Information Service. http://www.ripe.net/projects/ris/.

[4] The RouteViews project. http://www.routeviews.org/.

[5] The RouteViews project - Data Archives. http://www.routeviews.org/update.html.

[6] MRT routing information export format. http://www.ietf.org/internet-drafts/draft-ietf-grow-mrt-07.txt, 2007.

[7] D. G. Andersen, N. Feamster, S. Bauer, and H. Balakrishnan. Topology inference from bgp routing dynamics. In *IMW '02: Proceedings of the 2nd ACM SIGCOMM Workshop on Internet measurment*, pages 243–248, New York, NY, USA, 2002. ACM.

[8] J. Cowie, A. Ogielski, B. Premore, and Y. Yuan. Global routing instabilities triggered by Code Red II and Nimda worm attacks. *Renesys Corporation. Hanover, New Hampshire, USA.[Online]. Available: http://www. renesys. com/projects/papers/renesys_bgp_instabilities2001. pdf*, 2001.

[9] A. Flavel, O. Maennel, B. Chiera, M. Roughan, and N. Bean. CleanBGP: Verifying the Consistency of BGP Data. In *Internet Network Management Workshop 2008*, Oct. 2008.

[10] L. Gao. On inferring autonomous system relationships in the Internet. *ACM/IEEE Transactions on Networking*, 9(6):733–745, 2001.

[11] Z. B. Houidi, M. Meulle, and R. Teixeira. Understanding Slow BGP Routing Table Transfers. In *IMC 2009*, Nov. 2009.

[12] M. Lad, D. Massey, D. Pei, Y. Wu, B. Zhang, and L. Zhang. Phas: A prefix hijack alert system. In *USENIX Security Symposium*, 2006.

[13] R. Oliveira, B. Zhang, D. Pei, R. Izhak-Ratzin, and L. Zhang. Quantifying path exploration in the internet. In *IMC '06: Proceedings of the 6th ACM SIGCOMM conference on Internet measurement*, pages 269–282, New York, NY, USA, 2006. ACM.

[14] Y. Rekhter, T. Li, and S. Hares. A Border Gateway Protocol 4 (BGP-4). RFC 4271 (Draft Standard), Jan. 2006.

[15] J. Rexford, J. Wang, Z. Xiao, and Y. Zhang. Bgp routing stability of popular destinations. In *IMW '02: Proceedings of the 2nd ACM SIGCOMM Workshop on Internet measurment*, pages 197–202, New York, NY, USA, 2002. ACM.

[16] G. Siganos, M. Faloutsos, P. Faloutsos, and C. Faloutsos. Power-laws and the as-level internet topology. In *ACM/IEEE Transactions on Networking*, August 2003.

[17] L. Wang, M. Saranu, J. Gottlieb, and D. Pei. Understanding BGP Session Failures in a Large ISP. In *INFOCOM 2007. 26th IEEE International Conference on Computer Communications. IEEE*, pages 348–356, May 2007.

[18] L. Wang, X. Zhao, D. Pei, R. Bush, D. Massey, A. Mankin, S. F. Wu, and L. Zhang. Observation and analysis of bgp behavior under stress. In *IMW '02: Proceedings of the 2nd ACM SIGCOMM Workshop on Internet measurment*, pages 183–195, New York, NY, USA, 2002. ACM.

[19] B. Zhang, V. Kambhampati, M. Lad, D. Massey, and L. Zhang. Identifying bgp routing table transfers. In *MineNet '05: Proceedings of the 2005 ACM SIGCOMM workshop on Mining network data*, pages 213–218, New York, NY, USA, 2005. ACM.